Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence

Regret-Based Optimization and Preference Elicitation for Stackelberg Security Games with Uncertainty

Thanh H. Nguyen[1], **Amulya Yadav**[1], **Bo An**[2], **Milind Tambe**[1], **Craig Boutilier**[3]
[1]University of Southern California, Los Angeles, CA 90089
{thanhhng, amulyaya, tambe}@usc.edu
[2]Nanyang Technological University, Singapore 639798
boan@ntu.edu.sg
[3]University of Toronto, Canada M5S 3H5
cebly@cs.toronto.edu

Abstract

Stackelberg security games (SSGs) have been deployed in a number of real-world domains. One key challenge in these applications is the assessment of attacker payoffs, which may not be perfectly known. Previous work has studied SSGs with uncertain payoffs modeled by interval uncertainty and provided *maximin-based* robust solutions. In contrast, in this work we propose the use of the less conservative *minimax regret* decision criterion for such payoff-uncertain SSGs and present the first algorithms for computing minimax regret for SSGs. We also address the challenge of preference elicitation, using minimax regret to develop the first elicitation strategies for SSGs. Experimental results validate the effectiveness of our approaches.

Introduction

Defender-attacker Stackelberg security games (SSGs) provide a practical game-theoretic model for security resource allocation (Conitzer 2012; Basilico et al. 2009; Yin and Tambe 2012) and have found application in a number of real-world domains (Tambe 2011; Shieh et al. 2012). However, one recognized area of weakness and potential impediment to their future deployment is *attacker payoff uncertainty* (Jain et al. 2013).

Equilibrium strategies for defenders can be extremely sensitive to these payoffs, yet they can be extremely difficult to assess, requiring security experts to evaluate the potential impact on lives, property, and economic activity associated with specific attacks on particular targets. Even with this effort, this assessment is generally characterized by significant uncertainty.

Recently, optimization techniques have been developed for solving these *payoff-uncertain SSGs*. One key approach uses *Bayesian game models*, which require a prior distribution over possible attacker payoffs (Kiekintveld et al. 2011; Yin and Tambe 2012). Unfortunately, in many cases, particularly in counter-terrorism—the major application area of SSGs—assessing a probabilistic prior can be nearly as difficult as assessing payoffs themselves. Therefore, a second major approach is to use *strict uncertainty*, where defender payoffs are known but attacker payoffs are assumed only to lie within some interval with no distributional information. For these *Strict Uncertainty Payoff games (SPACs)*, robust optimization methods have been developed using the *maximin* decision criterion, in which a defender chooses a strategy that maximizes her worst-case utility over possible payoff realizations (Kiekintveld et al. 2013). In this paper, we will use this SPAC model of SSGs.

Our first contribution is the development of robust optimization methods for SPACs that rely on the *minimax regret* decision criterion (Savage 1972; Kouvelis and Yu 1997; Boutilier et al. 2006). Minimax regret focuses on the loss with respect to *decision quality* over possible payoff realizations, making decisions with the tightest possible optimality guarantees. Maximin, on the other hand, tends to be overly conservative by trying to protect against worst-case payoff realizations without considering the degree to which the defender's actions actually *impact her utility* relative to the optimal action under full information (Kouvelis and Yu 1997). Minimax regret is arguably a more appropriate robustness measure for dealing with uncertainty, especially for guaranteeing that *decisions* have high quality (i.e., are close to optimal). Indeed, minimax regret has not yet been available to (security) policy makers—this work makes it a viable criterion for generating new, less conservative, candidate defensive strategies. Unfortunately, operationalizing minimax regret involves complex, non-convex optimization for which efficient algorithms do not exist. We thus develop novel, efficient algorithms for approximating minimax regret, and offer experimental results showing that high solution quality can be attained quickly.

Our second contribution is a *payoff elicitation* procedure that can be used to optimize the defender's efforts in assessing payoffs, allowing reduction in the uncertainty of those parameters that most improve decision quality. This is yet another reason to use minimax regret as our robustness criterion—it has been proven to be a very effective

driver of elicitation in several domains (Boutilier et al. 2004; Regan and Boutilier 2009). We propose and evaluate several payoff elicitation strategies.

Background and Related Work

Stackelberg Security Games. In SSGs, a *defender* allocates m resources to protect a set of T *targets* from an *attacker* who will attack one of the targets. A defender *mixed* strategy is a vector $\mathbf{x} = (x)_{1\leq t\leq T}$, with $0 \leq x_t \leq 1$ and $\sum_t x_t \leq m$, where x_t denotes the probability of protecting t (Kiekintveld et al. 2009). Let $\mathbf{X} = \{\mathbf{x} : 0 \leq x_t \leq 1, \sum_t x_t \leq m\}$ be the set of feasible defender strategies. The defender first commits to a mixed strategy, after which the attacker observes the defender strategy and then attacks a target.

If the attacker attacks t, he receives a reward R_t^a if the target is unprotected, and a penalty $P_t^a < R_t^a$ if it is protected. Conversely, the defender receives a penalty P_t^d in the former case and a reward $R_t^d > P_t^d$ in the latter. Given a defender strategy $\mathbf{x}$ and the attacked target t, the *expected utilities* of the adversary and the defender, respectively, are:

$$U_t^a(x_t, R_t^a, P_t^a) = R_t^a(1 - x_t) + P_t^a x_t; \quad (1)$$

$$U_t^d(x_t, R_t^d, P_t^d) = R_t^d x_t + P_t^d(1 - x_t). \quad (2)$$

Let $R^a = \{R_t^a\}_t$ and $P^a = \{P_t^a\}_t$ be the the set of attacker rewards/penalties at all targets. The attacker's *attack set* $A(\mathbf{x}, R^a, P^a)$, given defender strategy $\mathbf{x}$, contains the targets that give him the highest utility:

$$A(\mathbf{x}, R^a, P^a) = \{t : U_t^a(x_t, R_t^a, P_t^a) \geq U_{t'}^a(x_{t'}, R_{t'}^a, P_{t'}^a), \forall t'\} \quad (3)$$

The defender's aim is to choose a strategy $\mathbf{x}$ that maximizes her own payoff $v(\mathbf{x}, R^a, P^a)$:

$$v(\mathbf{x}, R^a, P^a) = \max_{t\in A(\mathbf{x},R^a,P^a)} U_t^d(x_t, R_t^d, P_t^d). \quad (4)$$

As in other work in the literature and justified elsewhere, we assume that the attacker breaks ties in favor of the defender (Von Stengel and Zamir 2004). Given a defender's strategy $\mathbf{x}$ and an adversary's payoff (R^a, P^a), we call target t a *key target* if t is the attacked target within the attack set, i.e., $t = \text{argmax}_{t'\in A(\mathbf{x},R^a,P^a)} U_{t'}^d(x_{t'}, R_{t'}^d, P_{t'}^d)$. Thus, if t is the key target, then $v(\mathbf{x}, R^a, P^a) = U_t^d(x_t, R_t^d, P_t^d)$.

Attacker Payoff Uncertainty. We focus on SPACs (Kiekintveld et al. 2013; Yin et al. 2011) where the defender lacks the data to precisely estimate attacker payoffs. We assume that attacker payoffs are known only to lie within specific intervals: for each t, we have $R_t^a \in I_t^r = [R_{min}^a(t), R_{max}^a(t)]$ and $P_t^a \in I_t^p = [P_{min}^a(t), P_{max}^a(t)]$. Let $\mathbf{I} = \{(I_t^r, I_t^p)\}_t$ denote the set of payoff intervals.

Robust Optimization. Robust optimization in SPACs to date has focused on *maximin*: the defender chooses a strategy to maximize her worst case utility, over all possible realizations of attacker payoffs, assuming the attacker responds optimally under that realization (Kiekintveld et al. 2013; Aghassi and Bertsimas 2006). While minimax regret has been successfully used in various domains (Salo and Hamalainen 2001; Boutilier et al. 2006; Hyafil and Boutilier 2004; Renou and Schlag 2010) and has proven very effective for preference elicitation (Braziunas and Boutilier 2010; Boutilier 2013), it has not been applied in SPACs.

Targets	Def. rew.	Def. pen.	Adv. rew.	Adv. pen.
1	-6	-7	[0, 10]	[-4,0]
2	5	-6	[0, 10]	[-4,0]
3	3	-5	[0, 10]	[-4,0]

Table 1: A 3-target, 1-resource SPAC.

Solving SPACs using Minimax Regret

We now formulate the minimax regret solution for SPACs and discuss several methods for its computation.

Minimax Regret

Definition 1. *Given uncertainty interval* $\mathbf{I}$*, the* max regret *of defender strategy* $\mathbf{x} \in \mathbf{X}$ *is:*

$$MR(\mathbf{x}, \mathbf{I}) = \max_{(R^a,P^a)\in\mathbf{I}} \max_{\mathbf{x}'\in\mathbf{X}} \left(v(\mathbf{x}', R^a, P^a) - v(\mathbf{x}, R^a, P^a)\right) \quad (5)$$

Given payoff uncertainty $\mathbf{I}$, max regret evaluates the quality of strategy $\mathbf{x}$, under the worst-case realization of the payoff in $\mathbf{I}$, by measuring the worst-case loss in defender utility of using $\mathbf{x}$ rather than the optimal strategy $\mathbf{x}'$ given that realization. To guard against this loss in utility, the defender can adopt the strategy that minimizes this max regret:

Definition 2. *The* minimax regret (MMR) *of interval* $\mathbf{I}$ *is:*

$$MMR(\mathbf{I}) = \min_{\mathbf{x}\in\mathbf{X}} MR(\mathbf{x}, \mathbf{I}). \quad (6)$$

A *minimax optimal strategy* is any $\mathbf{x}$ that minimizes Eq. 6. Such a strategy has the strongest optimality guarantee given uncertainty $\mathbf{I}$. Table 1 shows why MMR might be more valuable than maximin. The optimal defender strategy under maximin is $[1.0, 0.0, 0.0]$: it allocates all resources to the most vulnerable target 1, simply because it has the lowest reward/penalty, despite the fact that defending target 1 provides little benefit. Maximin also *leaves targets* 2 *and* 3 *unprotected*, and has a max regret of 11. By contrast, MMR diversifies the defender strategy to minimize utility loss over all payoff realizations. The minimax optimal strategy is $[0.34, 0.44, 0.22]$ which has max regret 6.2.

Computing Minimax Regret

We note that MMR can be formulated as follows:

$$\min_{\mathbf{x}\in\mathbf{X}} \theta \quad (7)$$
$$\text{s.t. } \theta \geq v(\mathbf{x}', R^a, P^a) - v(\mathbf{x}, R^a, P^a), \forall \mathbf{x}' \in \mathbf{X}, (R^a, P^a) \in \mathbf{I}$$

Unfortunately, since $\mathbf{X}$ and $\mathbf{I}$ are continuous, the set of constraints is infinite; and the problem is non-convex, making MMR computation difficult. One practical approach to optimization with large constraint sets is *constraint sampling* (De Farias and Van Roy 2004), coupled with *constraint generation* (Boutilier et al. 2006). The key idea is to sample a subset of constraints and gradually expand this set by adding violated constraints to the relaxed problem until convergence to the optimal solution. *MIRAGE (MInimax Regret Algorithm using constraint GEneration)*, see Alg. 1, begins by sampling pairs $(\mathbf{x}, (R^a, P^a))$ of defender strategies and attacker payoffs uniformly from $\mathbf{X}$ and $\mathbf{I}$ to obtain a finite set S of *sampled constraints*. It then solves the corresponding relaxed optimization program Eq. 7—using

Algorithm 1: Constraint-generation (MIRAGE)

1 Initialize $S=\phi, ub=\infty, lb=0$;
2 Randomly generate $(\mathbf{x}', R^a, P^a)$, $S=S\cup\{\mathbf{x}', (R^a, P^a)\}$;
3 **while** $ub > lb$ **do**
4 Call bCISM to compute relaxed MMR w.r.t S. Let $\mathbf{x}^*$ be its optimal solution with objective value lb;
5 Call ALARM to compute $MR(\mathbf{x}^*, \mathbf{I})$. Let $(\mathbf{x}'^*, R^{a,*}, P^{a,*})$ be its optimal solution with objective value ub;
6 $S = S \cup \{\mathbf{x}'^{,*}, R^{a,*}, P^{a,*}\}$;
7 return $(lb, \mathbf{x}^*)$;

the *bCISM* algorithm—whose optimal solution $(lb, \mathbf{x}^*)$ provides a lower bound (lb) on true MMR. Then constraint generation is applied to determine violated constraints (if any). This uses the *ALARM* algorithm, which computes $MR(\mathbf{x}^*, \mathbf{I})$. This provides an upper bound (ub) on true MMR. If $ub > lb$, ALARM's solution provides us with the maximally violated constraint (see line 5), which is added to S. If $ub = lb$, then $\mathbf{x}^*$ is the minimax optimal strategy and $lb = ub = MMR(\mathbf{I})$.[1]

We now detail the two main steps of MIRAGE. The first step, corresponding to line (4) of MIRAGE, solves a relaxed version of Eq. 7. This relaxed problem can be formulated as a mixed integer linear program (MILP) as follows:

$$\min_{\mathbf{x},\mathbf{q},\mathbf{r},\theta} \theta \tag{8}$$

$$\theta \geq v(\mathbf{x}'^j, R^{a,j}, P^{a,j}) - U_t^d(x_t, R_t^d, P_t^d) - (1-q_t^j)M,\ \forall j,t \tag{9}$$

$$U_t^a(x_t, R_t^{a,j}, P_t^{a,j}) + (1-q_t^j)M \geq r^j,\ \forall j,t \tag{10}$$

$$r^j \geq U_t^a(x_t, R_t^{a,j}, P_t^{a,j}),\ \forall j,t \tag{11}$$

$$\sum\nolimits_t x_t \leq m,\ x_t \in [0,1],\ \forall t \tag{12}$$

$$\sum\nolimits_t q_t^j = 1, q_t^j \in \{0,1\},\ \forall j,t \tag{13}$$

$$r^j \in \mathbf{R},\ \forall j. \tag{14}$$

Here the index j ranges over sampled/generated constraints $(\mathbf{x}'^j, (R^{a,j}, P^{a,j}))$ in S, index t ranges over targets and M is a large positive constant. Constraints (10-11) ensure that if t is the key target (i.e., $q_j^t = 1$) for the j^{th} instance of $R^{a,j}$ and $P^{a,j}$, then $U_t^a(x_t, R_t^{a,j}, P_t^{a,j}) \geq U_{t'}^a(x_{t'}, R_{t'}^{a,j}, P_{t'}^{a,j})\ \forall t'$. Constraint (9) requires that if $q_t^j = 1$, then $\theta \geq v(\mathbf{x}'^j, R^{a,j}, P^{a,j}) - U_t^d(x_t, R_t^d, P_t^d)$.

We dub this *mCISM (MILP for Computing dIScretized MMR)*, Unfortunately, mCISM becomes quite slow as S grows. As an alternative, we introduce a *binary-search based algorithm (bCISM)* which searches defender utility space to find the optimal solution in polynomial time. Intuitively, we search for a strategy $\mathbf{x}$ that satisfies constraints (9-14), where θ is computed using binary search. Specifically, given θ, we compute the defender's *minimum coverage* at each target s.t. $\theta \geq v(\mathbf{x}'^j, R^{a,j}, P^{a,j}) - v(\mathbf{x}, R^{a,j}, P^{a,j})$ is satisfied for all $(\mathbf{x}'^j, (R^{a,j}, P^{a,j})) \in S$, and then test if $\mathbf{x} \in \mathbf{X}$ (i.e., is the strategy feasible).

[1]While constraint generation can be used by itself, generating violated constraints is computationally intensive. Initializing with some randomly sampled constraints offers better performance.

Algorithm 2: Compute minimum coverage given θ

1 Initialize $lb_t = 0, x_t^j = +\infty\ \forall j,t$;
2 **while** *true* **do**
3 **for** $j = 1$ *to* $|S|$ **do**
4 **for** $t = 1$ *to* T **do**
5 $c_t^{j,t} = \max\{lb_t, \frac{\theta^j - P_t^d}{R_t^d - P_t^d}\}$;
6 **if** $c_t^{j,t} > 1$ **then** continue;
7 **foreach** $k \neq t$ **do**
8 $c_k^{j,t} = \max\{lb_t, \frac{U_t^a(c_t^j, R_t^{a,j}, P_t^{a,j}) - R_k^{a,j}}{P_k^{a,j} - R_k^{a,j}}\}$;
9 $x_k^j = \min\{x_k^j, c_k^{j,t}\}$;
10 $x_t^j = \min\{x_t^j, c_t^{j,t}\}$;
11 Set $\mathbf{x} = \{x_t :\ x_t = \max_j x_t^j\}_t$;
12 **if** $\mathbf{x} \notin \mathbf{X}$ **then** return false;
13 **else if** $v(\mathbf{x}, R^{a,j}, P^{a,j}) \geq \theta^j\ \forall j$ **then** return $\mathbf{x}$;
14 **else** $lb_t = x_t$;

Let $\theta^j = -\theta + v(\mathbf{x}'^j, R^{a,j}, P^{a,j})$ for the j^{th} constraint in S. We require $\theta^j \leq v(\mathbf{x}, R^{a,j}, P^{a,j})$ for all $j \in S$, with $\mathbf{x}$ is computed by Alg. 2. Alg. 2 iterates through two levels of problem decomposition to find $\mathbf{x}$. First, it finds $c_k^{j,t}$, the minimum defender coverage at each target k for the j^{th} instance such that t is the key target and $\theta^j \leq v(\mathbf{x}, R^{a,j}, P^{a,j})$. We iterate over all possible key targets t to find corresponding $c_k^{j,t}$. Based on these $c_k^{j,t}$, we then compute the minimum defender coverage at target k, denoted x_k^j ($k \leq T$), which satisfies the constraint $\theta^j \leq v(\mathbf{x}^j, R^{a,j}, P^{a,j})$ w.r.t. the j^{th} instance only (lines (4-10)). The resulting $\{x_k^j\}_k$ for all $j \leq |S|$ are then combined to compute $\mathbf{x}$ (line 11), as we elaborate below.

We now explain lines (4-10). Note that if t is the key target of the j^{th} instance, then $v(\mathbf{x}^j, R^{a,j}, P^{a,j}) = U_t^d(x_t^j, R_t^d, P_t^d)$. Here the algorithm goes through all targets t that could potentially be the key target for the j^{th} instance to compute $c_k^{j,t}$. Specifically, if t is the key target, as $U_t^d(c_t^{j,t}, R_t^d, P_t^d) \geq \theta^j$, we have $c_t^{j,t} \geq \max\{lb_t, \frac{\theta^j - P_t^d}{R_t^d - P_t^d}\}$, where lb_t is the lower bound for the defender's coverage at t (line 5). This lower bound is updated in each iteration of the **while** loop. In addition, for any other target k, it follows that $U_k^a(c_k^{j,t}, R_k^{a,j}, P_k^{a,j}) \leq U_t^a(c_t^{j,t}, R_t^{a,j}, P_t^{a,j})$ which means $c_k^{j,t} \geq \max\{lb_k, \frac{U_t^a(c_t^{j,t}, R_t^{a,j}, P_t^{a,j}) - R_k^{a,j}}{P_k^{a,j} - R_k^{a,j}}\}$. Thus, the higher $c_t^{j,t}$, the smaller $U_t^a(c_t^{j,t}, R_t^{a,j}, P_t^{a,j})$ and therefore, the higher $c_k^{j,t}$ for all k. The minimum coverage for target t is then $c_t^{j,t} = \max\{lb_t, \frac{\theta^j - P_t^d}{R_t^d - P_t^d}\}$ and for any other target k is $c_k^{j,t} = \max\{lb_k, \frac{U_t^a(c_t^{j,t}, R_t^{a,j}, P_t^{a,j}) - R_k^{a,j}}{P_k^{a,j} - R_k^{a,j}}\}$.

Proposition 3. *Given constraint j, suppose $\mathbf{x}$ is a feasible strategy s.t. $\theta^j \leq v(\mathbf{x}, R^{a,j}, P^{a,j})$ with key target t and lower bound $x_k \geq lb_k, \forall k$. Then $x_k \geq c_k^{j,t}, \forall k$.* [2]

[2]Proofs of all results and additional algorithm details can be found in an online appendix:

Proposition 4. *For any constraint j, if $c_k^{j,t} > c_k^{j,t'}$ for some target k, then $c_i^{j,t} \geq c_i^{j,t'}$ for* all *targets i.*

Prop. 3 implies that if $x_k < c_k^{j,t}$ for some k, $\mathbf{x}$ is not feasible given that t is the key target and $\theta^j \leq v(\mathbf{x}, R^{a,j}, P^{a,j})$. In other words, $c_k^{j,t}$ is a lower bound on all x_k satisfying this condition. Prop. 4 implies that for any pair of key targets t and t', $c_k^{j,t} \geq c_k^{j,t'}$ for all targets k, or vice versa. Taken together, the minimum defender coverages are $x_k^j = \min_t c_k^{j,t}, \ \forall k$. As strategy $\mathbf{x}$ must satisfy $\theta^j \leq v(\mathbf{x}, R^{a,j}, P^{a,j})$ for all $j \in S$, $x_t \geq x_t^j, \forall t$. Therefore, $x_t \geq \max_j x_t^j, \forall t$ (line 11).

Since this minimum coverage x_t may be infeasible (i.e., $\mathbf{x} \notin \mathbf{X}$), we test feasibility (line (12)). Furthermore, it may not be feasible w.r.t. θ as it may now violate the constraints for the chosen key targets in S. Thus, we check (line (13)) for violations that give an objective value less than θ. If so, we update the lower bound on x_t (line (14)). We repeat until the constraint $\mathbf{x} \in \mathbf{X}$ is violated or a feasible solution is found. bCISM thus determines if some feasible $\mathbf{x}$ satisfies $\theta \geq v(\mathbf{x}', R^a, P^a) - v(\mathbf{x}, R^a, P^a)$ for all $(\mathbf{x}', (R^a, P^a)) \in S$. bCISM is guaranteed to provide a δ-optimal solution to MMR, where δ is the binary-search termination threshold.

Computing Max Regret

The second sub-problem of MIRAGE is computation of $MR(\mathbf{x}, \mathbf{I})$. This is accomplished with *ALARM (Approximate Linearization Algorithm for Reckoning Max regret)*. Given Eq. 5, $MR(\mathbf{x}, \mathbf{I})$ requires computing the strategy $\mathbf{x}'$ and attacker payoff (R^a, P^a) that maximize the loss of $\mathbf{x}$. However, value of $v(\mathbf{x}', R^a, P^a) = \max_{t \in A(\mathbf{x}', R^a, P^a)} U_t^d(x_t', R_t^d, P_t^d)$ depends on the attack set $A(\mathbf{x}', R^a, P^a)$, which in turn is determined by the attacker's utility at each target t—$U_t^a(x_t', R_t^a, P_t^a) = R_t^a(1 - x_t') + P_t^a x_t'$. This is a non-convex function of variables x_t' and (R_t^a, P_t^a), making max regret a non-convex optimization problem. To speed up computation, we now develop a linearization.

Both $v(\mathbf{x}', R^a, P^a)$ and $v(\mathbf{x}, R^a, P^a)$ are dictated by the key targets, which depend on $(R^a, P^a) \in \mathbf{I}$ and $\mathbf{x}' \in \mathbf{X}$. We partition $(\mathbf{I}, \mathbf{X})$ into T^2 subsets such that each pair of targets t, t' are the key targets within a particular subset. We then search over all pairs of possible key targets (t', t) to compute max regret. Specifically, given key targets (t', t), max regret can be reformulated as follows:

$$\max_{(R^a, P^a) \in \mathbf{I}, \mathbf{x}' \in \mathbf{X}} U_{t'}^d(x_{t'}', R_{t'}^d, P_{t'}^d) - U_t^d(x_t, R_t^d, P_t^d) \quad (15)$$

$$\text{s.t. } U_{t'}^a(x_{t'}', R_{t'}^a, P_{t'}^a) \geq U_k^a(x_k', R_k^a, P_k^a), \forall k \neq t, t' \quad (16)$$

$$U_t^a(x_t, R_t^a, P_t^a) \geq U_k^a(x_k, R_k^a, P_k^a), k \in N(t) \setminus \{t, t'\} \quad (17)$$

$$U_t^a(x_t, R_t^a, P_t^a) \geq U_k^a(x_k, R_k^a, P_k^a) + \epsilon, k \notin N(t) \cup \{t, t'\} \quad (18)$$

$$U_{t'}^a(x_{t'}', R_{t'}^a, P_{t'}^a) \geq U_t^a(x_t', R_t^a, P_t^a) \quad (19)$$

$$U_t^a(x_t, R_t^a, P_t^a) \geq U_{t'}^a(x_{t'}, R_{t'}^a, P_{t'}^a) \text{ if } t' \in N(t) \quad (20)$$

$$U_t^a(x_t, R_t^a, P_t^a) \geq U_{t'}^a(x_{t'}, R_{t'}^a, P_{t'}^a) + \epsilon \text{ if } t' \notin N(t) \quad (21)$$

where $N(t) = \{k : U_t^d(x_t, R_t^d, P_t^d) \geq U_k^d(x_k, R_k^d, P_k^d)\}$ is the set of targets at which the defender utility is lower than the utility at t, and ϵ is a small positive constant. We separate constraints for key and non-key targets for expository purposes. Constraints (17, 18, 20, 21) ensure that t is the key target w.r.t. $\mathbf{x}$, while ϵ ensures the strict inequality $U_t^a(x_t, R_t^a, P_t^a) > U_k^a(x_k, R_k^a, P_k^a)$ when $U_t^d(x_t, R_t^d, P_t^d) < U_k^d(x_k, R_k^d, P_k^d)$ (or when $k \notin N(t)$); otherwise, the attacker will attack k instead of t (by tie-breaking). We do allow violation of the tie-breaking constraint for the key target t' w.r.t. $\mathbf{x}'$ for each sub-problem (15-21). Searching over all possible key targets t' guarantees the final solution will satisfy the tie-breaking assumption.

Note that (R_k^a, P_k^a) and $\mathbf{x}'$ are involved in computing $U_k^a(x_k', R_k^a, P_k^a)$ which makes this utility function non-convex (constraints (16–19)). We could solve the MR problem (Eq. 15) using existing commercial solvers for non-convex optimization (e.g., Knitro): (a) solve a non-convex program for every combination of key targets (Multi-NLP); or (b) formulate it as a mixed integer non-linear program (MINLP, see Appendix). However, these perform poorly, as we show below. Hence we use binary search to *linearize* these non-convex constraints to approximate MR (noting that they can be linearized if $x_{t'}'$ and x_t' are known).

Problem 5. ***(ALARM binary search decision problem)*** *Given a value θ, and t', t as key targets: are there $(R^a, P^a) \in \mathbf{I}$ and $\mathbf{x}' \in \mathbf{X}$ satisfying (16–21), such that $U_{t'}^d(x_{t'}', R_{t'}^d, P_{t'}^d) - U_t^d(x_t, R_t^d, P_t^d) \geq \theta$?*

Proposition 6. *If Problem 5 has a feasible solution, then the following solution is feasible: $R_k^a = R_{min}^a(k)$ and $P_k^a = P_{min}^a(k)$, $\forall k \neq t, t'$; and $x_{t'}' = x_{min}'(t') = \max\{0, \frac{\theta + U_t^d(x_t, R_t^d, P_t^d) - P_{t'}^d}{R_{t'}^d - P_{t'}^d}\}$.*

Thus, by replacing (R_k^a, P_k^a) with $(R_{min}^a(k), P_{min}^a(k))$, for all $k \neq t, t'$ and $x_{t'}'$ with $x_{min}'(t')$, we are left with one non-convex constraint (19). We circumvent this non-convex constraint by converting Problem 5 into the following optimization with a non-convex objective:

$$\max_{\{x_k'\}_{k \neq t'}, R_t^a, P_t^a, R_{t'}^a, P_{t'}^a} U_{t'}^a(x_{t'}', R_{t'}^a, P_{t'}^a) - U_t^a(x_t', R_t^a, P_t^a) \quad (22)$$

$$\text{s.t. updated (16–18, 20–21)} \quad (23)$$

Intuitively, constraint (19) is translated into the objective (22), maintaining other constraints with their updates $(R_{min}^a(k), P_{min}^a(k))$ and $x_{min}'(t')$.

Proposition 7. *If the optimum of (22) is no less than zero, Problem 5 has a feasible solution; otherwise it is infeasible.*

As the objective (22) remains non-convex, and direct non-convex methods are inefficient, we apply the following linearization:

Proposition 8. *If x_t' is fixed, then the $\{x_k'\}_{k \neq t', t}$ which minimizes $\max_{k \neq t, t'} U_k^a(x_k', R_k^a, P_k^a)$ are optimal for (22).*

Recall that our variables are $R_t^a, P_t^a, R_{t'}^a, P_{t'}^a$ and x_k' $(k \neq t')$. Given a value of x_t', by Prop. 8 we can apply ORIGAMI (Kiekintveld et al. 2009) to compute x_k' for all $k \neq t, t'$ to minimize $\max_{k \neq t, t'} U_k^a(x_k', R_k^a, P_k^a)$. The remaining variables are $R_t^a, P_t^a, R_{t'}^a, P_{t'}^a$. Starting with an initial value x_t' and the corresponding x_k' for all $k \neq t, t'$ computed

http://teamcore.usc.edu/people/thanhhng/Papers/appendix2.pdf.

Algorithm 3: Local search

Initialize x'_t, $\delta = \inf$;
Compute $x'_{k\neq t,t'}$ which minimize $\max_{k\neq t,t'} U^a_k(x'_k, R^a_k, P^a_k)$ using ORIGAMI;
while $\delta > 0$ **do**
  Given $x'_{k\neq t'}$, solve (22-23) to obtain obj^* and $(R^a_t, P^a_t, R^a_{t'}, P^a_{t'})^*$;
  Given $(R^a_t, P^a_t, R^a_{t'}, P^a_{t'})^*$, solve (22-23) to obtain obj^{**} and $x'_{k\neq t'}$;
  $\delta = obj^{**} - obj^*$;
return $(obj^{**}, (R^a_t, P^a_t, R^a_{t'}, P^a_{t'})^*, x'_{k\neq t'})$;

by ORIGAMI, our local search, see Alg. 3, gradually updates $(R^a_t, P^a_t, R^a_{t'}, P^a_{t'})$ and $x'_{k\neq t'}$ to find a local optimum of Eq. (22). In Alg. 3, obj^* and obj^{**} are the optimal objective values of Eq. (22) given $x'_{k\neq t'}$ and $(R^a_t, P^a_t, R^a_{t'}, P^a_{t'})^*$, respectively. By fixing either $x'_{k\neq t'}$ or $(R^a_t, P^a_t, R^a_{t'}, P^a_{t'})$, Eq. (22) becomes a linear program, allowing one to readily obtain a locally optimal solution. Our experiments show that randomly initializing x'_t even just five times typically suffices to find a *global* optimum.

Payoff Elicitation

Typically, defenders employ the services of expert risk analysts to assess the attacker payoffs at specific targets (Shieh et al. 2012). While our MMR methods offer robust decisions in the face of payoff uncertainty, the resulting max regret level may be too large in certain cases. Thus we develop an interactive process whereby the defender can reduce her uncertainty w.r.t. attacker payoffs by querying the expert, at some cost, for additional information about these payoffs. Note however that reducing uncertainty for its own sake often fails to improve max regret (Boutilier et al. 2006; Braziunas and Boutilier 2010); attention must be focused on those payoffs that actually influence *decisions*. We develop preference elicitation strategies driven by MMR that focus on "relevant" uncertainty. Generally, queries and payoff assessment continue until MMR reaches an acceptable level or some budget limit is met.

We consider *bound queries*, widely used in preference elicitation (Boutilier et al. 2006; French 1986), in which an expert is asked whether the attacker reward/penalty at some target t is greater than some value p. For example, if the reward interval at target t is $[0, 10]$, we can ask if the reward is greater than 5. A positive (negative) response increases the lower bound to 5 (decreases the upper bound to 5). We assume each query q_t is associated with a single target t, and queries both the reward R^a_t and penalty P^a_t terms at the midpoints of their corresponding intervals I^r_t and I^p_t (since assessment of a target generally provides information about both). Thus each query has four possible responses. Query costs may vary with the target.

Since reducing uncertainty at different targets impacts max regret differently, and we may have a fixed budget, queries must be chosen effectively. Alg. 4 outlines our elicitation procedure. We now describe three heuristic *query selection strategies*.

Algorithm 4: Preference elicitation

Initialize $totalcost = 0$, $regret = \infty$;
while $totalcost < budget$ ***or*** $regret > thres$ **do**
  Find the best target t^* using an elicitation strategy;
  Ask a bound query regarding payoff of target t^*;
  Update uncertainty interval $\mathbf{I}_{new}$;
  Update $totalcost$, recompute MMR;

Myopic strategy. We compute the average regret reduction over the four possible responses to each query (target) and query the target with the greatest average reduction.

Approximate strategy. The Myopic strategy may be computationally inefficient as it relies on exact computation of MMR for all $4T$ query responses. This strategy uses bCISM to approximate MMR, but otherwise mimics Myopic.

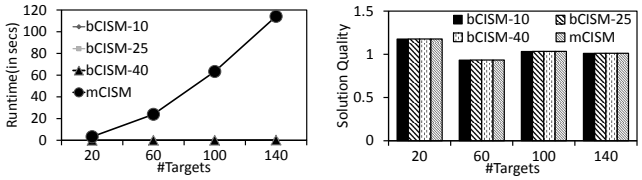
Optimistic/pessimistic strategy. To maximize regret, attacker payoffs are set within uncertainty intervals to induce the attacker to select targets that cause greatest defender loss: let the MMR solution set attacker payoffs to be $(\overline{R}^a, \overline{P}^a)$. If a query at target t obtains a response that makes the attacker's (say) reward $\overline{R}^a_t$ infeasible (by responding with the half interval of I^r_t *not* containing $\overline{R}^a_t$) max regret may be reduced significantly. Our optimistic heuristic evaluates queries by assuming optimistically that each query response will rule out the current "regret-inducing" payoffs at that target, and selects the query that, assuming this response, offers the greatest MMR reduction. Our pessimistic strategy is analogous. We combine the Optimistic and Pessimistic strategies with the Approximate strategy by replacing the exact MMR computation with the approximation bCISM, thereby increasing their computational efficiency. We call these strategies **Opt-Approx** and **Pess-Approx**.

Experimental Evaluation

We evaluate the runtime and solution quality of our algorithms on games generated using GAMUT.[3] All experiments were run on a 2.83GHz Intel processor with 4GB of RAM, using CPLEX 12.3 for LP/MILPs and KNITRO 8.0.0.z for nonlinear optimization. We set the covariance value $r \in [0.0, 1.0]$ with step size $\lambda = 0.2$ to control the correlation of attacker and defender rewards. Upper and lower bounds for payoff intervals are generated randomly from $[-14, -1]$ for penalties and $[1, 14]$ for rewards, with the difference between the upper and lower bound (i.e., interval size) exactly 2 (this gives payoff uncertainty of roughly 30%). All results are averaged over 120 instances (20 games per covariance value) and use eight defender resources unless otherwise specified. All comparison results with our algorithms are statistically significant under bootstrap-t ($\alpha = 0.05$).

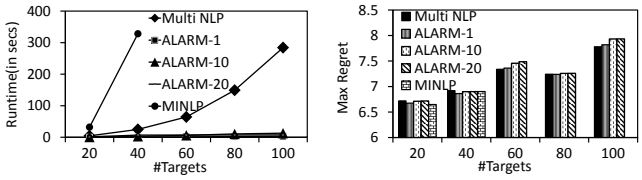
Evaluating MMR Algorithms We first evaluate our algorithms for discretized MMR. Fig. 1(a) shows runtimes of bCISM with 10, 25 and 40 initial samples from S, and mCISM with 10 samples. bCISM is roughly 68 times faster than mCISM (60 targets), and is rather insensitive to the number of initial samples. Fig. 1(b) plots solution quality;

[3] See http://gamut.stanford.edu/.



(a) Runtime (b) Solution quality

Figure 1: Evaluating discretized MMR algorithms

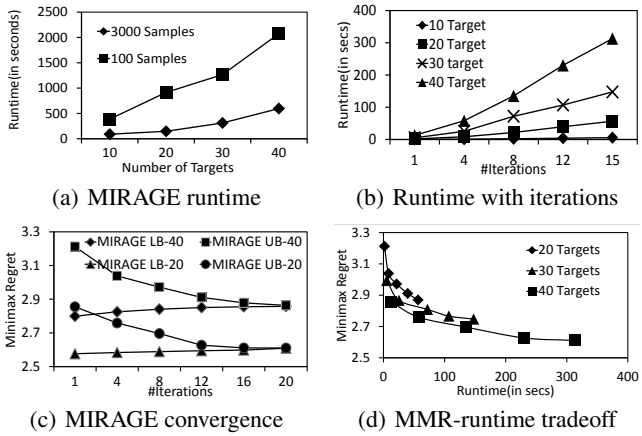


(a) Runtime (b) Solution quality

Figure 2: Evaluating algorithms for computing MR

bCISM's solutions are within 0.01% of the mCISM's optima. Thus, the computational efficiency of bCISM comes at low cost in terms of solution quality.

We next evaluate our three MR algorithms, ALARM (with 1, 10 and 20 samples of x'_t), MINLP, and Multi-NLP. Fig. 2(a) shows that MINLP and Multi-NLP's runtimes increase exponentially with the number of targets. All instances of ALARM run approximately 100 (resp., 30) times faster than MINLP (resp., Multi-NLP) with 100 targets. Fig. 2(b) plots their relative solution quality and shows that ALARM, despite its exponential speedup, has at most a 1% (avg.) loss in solution quality vs. MINLP and Multi-NLP.

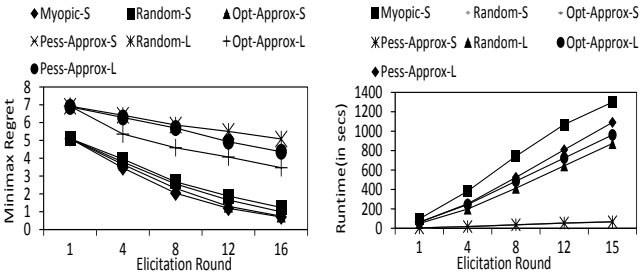


(a) MIRAGE runtime (b) Runtime with iterations

(c) MIRAGE convergence (d) MMR-runtime tradeoff

Figure 3: Evaluating MIRAGE properties

Since bCISM and ALARM handily outperform the other algorithms, we now evaluate MIRAGE only using bCISM and ALARM as its relaxed-MMR and MR subroutines. Fig. 3(a) plots MIRAGE runtimes, as number of targets varies, with either 100 and 3000 samples (constraints) added at each iteration. With 50 targets (not shown), MIRAGE takes about 10mins., and converges after 15 iterations. With 100 samples, MIRAGE takes longer to converge (about 6 times slower than with 3000); and runtime increases roughly by a factor of 14 with every 5 targets. Fig. 3(b) shows MI-



(a) Solution quality (b) Runtime

Figure 4: Evaluating payoff elicitation strategies

RAGE's runtime with the number of iterations (10 to 40 targets), while Fig. 3(c) shows progress of its MMR upper and lower bounds (20 and 40 targets). This demonstrates a promising anytime profile, allowing early termination with high quality solutions. For instance, Fig. 3(c) shows, with 20 targets, that 9 iterations offers a solution within 5% of minimax optimality. Fig. 3(d) shows the tradeoff between runtime and (upper bound on) MMR (20 and 40 targets) up to convergence, confirming the positive anytime profile.

Evaluating Payoff Elicitation Strategies We analyze the performance of the three elicitation strategies described above—Myopic, Opt-Approx and Pess-Approx—as well as a Random strategy (with target queries chosen at random, but eliciting at midpoints as above). We test on small problems with five targets (Random-S, etc.) and large problems with 20 targets (Random-L, etc.). Fig. 4(a) shows how MMR decreases as we ask queries using the different strategies. We see that Myopic is the most effective strategy, followed by Opt-Approx, Pess-Approx, and Random, which performs worst. Fig. 4(b) plots cumulative runtime of the elicitation process using our different strategies. Fig. 4(b) shows that Myopic is about 20 times slower than the other strategies. Opt-Approx is not much slower (1.08 times) than Random, while Pess-Approx is 1.24 times slower than Random. We use five-target games primarily to compare Myopic to the other methods; with 20 targets, Myopic is intractable and cannot be evaluated. With five targets, the strategies (unsurprisingly) do not vary much in elicitation performance. However, with 20 targets, Opt-Approx reduces MMR significantly faster than Random or Pess-Approx. Given its computational effectiveness, it seems to be a reasonable choice for payoff elicitation. However, all strategies point to tradeoffs that, depending on the task at hand and available budget, may make any of them viable.

Summary

Despite significant applications of SSGs for protecting major critical infrastructure, research on *robustness* in SSGs has, to date, focused only on one concept, maximin over interval uncertainty of payoffs. This approach unfortunately leads to extremely conservative security allocations. To remedy this shortcoming, we have proposed the use of MMR as a decision criterion for payoff-uncertain SSGs and presents efficient algorithms for computing MMR for such games. Furthermore, we have addressed, for the first time, the chal-

lenge of preference elicitation in SSGs, providing novel regret-based solution strategies. Experimental results validate the effectiveness of our approaches w.r.t. both computational and informational efficiency.

Acknowledgements: This research was supported by MURI Grant W911NF-11-1-0332, by CREATE under grant number 2010-ST-061-RE0001. Boutilier was supported by NSERC and An was supported by MOE AcRF Tier 1 grant RG33/13.

References

Aghassi, M., and Bertsimas, D. 2006. Robust game theory. *Mathematical Programming* 107(1-2):231–273.

Basilico, N.; Gatti, N.; and Amigoni, F. 2009. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *AAMAS*.

Boutilier, C.; Patrascu, R.; Poupart, P.; and Schuurmans, D. 2006. Constraint-based optimization and utility elicitation using the minimax decision criterion. *Artificial Intelligence* 170(8):686–713.

Boutilier, C.; Sandholm, T.; and Shields, R. 2004. Eliciting bid taker non-price preferences in (combinatorial) auctions. In *AAAI*, 204–211.

Boutilier, C. 2013. Computational decision support: Regret-based models for optimization and preference elicitation.

Braziunas, D., and Boutilier, C. 2010. Assessing regret-based preference elicitation with the utpref recommendation system. In *Proceedings of the 11th ACM conference on Electronic commerce*, 219–228. ACM.

Conitzer, V. 2012. Computing game-theoretic solutions and applications to security. In *AAAI*.

De Farias, D. P., and Van Roy, B. 2004. On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of operations research* 29(3):462–478.

French, S. 1986. *Decision theory: an introduction to the mathematics of rationality*. Halsted Press.

Hyafil, N., and Boutilier, C. 2004. Regret minimizing equilibria and mechanisms for games with strict type uncertainty. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, 268–277. AUAI Press.

Jain, M.; An, B.; and Tambe, M. 2013. Security games applied to real-world: Research contributions and challenges. In *Moving Target Defense II*. Springer. 15–39.

Kiekintveld, C.; Jain, M.; Tsai, J.; Pita, J.; Ordóñez, F.; and Tambe, M. 2009. Computing optimal randomized resource allocations for massive security games. In *The Eighth International Conference on Autonomous Agents and Multiagent Systems*.

Kiekintveld, C.; Islam, T.; and Kreinovich, V. 2013. Security games with interval uncertainty. In *AAMAS*.

Kiekintveld, C.; Marecki, J.; and Tambe, M. 2011. Approximation methods for infinite bayesian stackelberg games: modeling distributional payoff uncertainty. In *AAMAS*.

Kouvelis, P., and Yu, G. 1997. *Robust discrete optimization and its applications*, volume 14. Springer.

Regan, K., and Boutilier, C. 2009. Regret-based reward elicitation for markov decision processes. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, 444–451. AUAI Press.

Renou, L., and Schlag, K. H. 2010. Minimax regret and strategic uncertainty. *Journal of Economic Theory* 145(1):264–286.

Salo, A. A., and Hamalainen, R. P. 2001. Preference ratios in multiattribute evaluation (prime)-elicitation and decision procedures under incomplete information. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on* 31(6):533–545.

Savage, L. 1972. *The foundations of statistics*. DoverPublications. com.

Shieh, E.; An, B.; Yang, R.; Tambe, M.; Baldwin, C.; DiRenzo, J.; Maule, B.; and Meyer, G. 2012. Protect: A deployed game theoretic system to protect the ports of the united states. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

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

Von Stengel, B., and Zamir, S. 2004. Leadership with commitment to mixed strategies.

Yin, Z., and Tambe, M. 2012. A unified method for handling discrete and continuous uncertainty in bayesian stackelberg games. In *AAMAS*.

Yin, Z.; Jain, M.; Tambe, M.; and Ordóñez, F. 2011. Risk-averse strategies for security games with execution and observational uncertainty. In *AAAI*.