Stop the Compartmentalization: Unified Robust Algorithms for Handling Uncertainties in Security Games

Thanh H. Nguyen, Albert Jiang, Milind Tambe
University of Southern California, Los Angeles, CA 90089
{thanhhng, jiangx, tambe} @usc.edu

ABSTRACT

Given the real-world applications of Stackelberg security games (SSGs), addressing uncertainties in these games is a major challenge. Unfortunately, we lack any unified computational framework for handling uncertainties in SSGs. Current state-of-the-art has provided only compartmentalized robust algorithms that handle uncertainty exclusively either in the defender's strategy or in adversary's payoff or in the adversary's rationality, leading to potential failures in real-world scenarios where a defender often faces multiple types of uncertainties. Furthermore, insights for improving performance are not leveraged across the compartments, leading to significant losses in quality or efficiency.

In this paper, we provide the following main contributions: 1) we present the first unified framework for handling the uncertainties explored in SSGs; 2) based on this unified framework, we propose the first set of "unified" robust algorithms to address combinations of these uncertainties; 3) we introduce approximate scalable robust algorithms for handling these uncertainties that leverage insights across compartments; 4) we present experiments demonstrating solution quality and runtime advantages of our algorithms.

Categories and Subject Descriptors

H.4 [**Computing Methodology**]: Game Theory

General Terms

Algorithms, Security

Keywords

Game Theory, Robust Optimization, Security, Uncertainty

1. INTRODUCTION

Real-world counter-terrorism applications of defender-attacker Stackelberg Security Games (SSGs) [3, 9, 13] have led to significant research emphasis on handling uncertainty that naturally arises in these games. Two different approaches have been pursued to handle this uncertainty. The first approach models this uncertainty using probability distributions and solves the resulting Bayesian Stackelberg game models [7, 16]; the second takes a robust optimization approach of maximizing defender expected utility under the worst case resulting from such uncertainty [5, 6, 12, 15]. While the first approach assumes a known distribution of uncertainties beforehand, the second does not assume such prior knowledge. Since in many real world domains, including applications in counter-terrorism, we may lack data to generate a prior distribution, in this paper, we focus on the second approach.

Unfortunately, all previous work in robust optimization in SSGs compartmentalizes the uncertainties. For example, while some research has focused exclusively on uncertainty over defender's assessment of adversary's payoff [6], other work has focused exclusively on uncertainty over defender's execution of provided strategy and adversary's surveillance of this strategy [15], and yet other exclusively on the uncertainty given adversary's bounded rationality [5, 12]. The lack of a unified framework implies that existing algorithms suffer losses in solution quality in handling uncertainties in real-world security situations – where multiple types of uncertainties may exist simultaneously. In addition, insights for improving performance are not leveraged across these compartments; again leading to losses in solution quality or efficiency.

This paper remedies these weaknesses of state-of-the-art algorithms when addressing uncertainties in SSGs by providing the following key contributions. First, we are the first to present a unified computational framework – a single core problem representation – for handling the different types of uncertainties, as addressed so far in SSGs, and their combinations. Second, based on this unified framework, we present a unified algorithmic framework from which we can derive different "unified" robust algorithms which address any combination of uncertainties in our framework, avoiding the compartmentalization mentioned above – no other previous algorithm can handle these combinations of uncertainties. Third, exploiting new insights from our unified framework, we present fast approximate algorithms for handling different subsets of uncertainties in large-scale security games. Finally, our experiments show the solution quality and runtime advantages of our algorithms in highly uncertain domains.

The key insight in our unified robust algorithms is that under any type of uncertainty, the space of the defender's strategies can be partitioned into different sets; for any defender strategy within a set, the adversary's feasible strategies are identical. Thus, we can solve any robust optimization problem for addressing uncertainties as the maxima of all corresponding simpler sub-optimization problems created by this partition.

2. STACKELBERG SECURITY GAMES

In SSGs, there is a defender who attempts to optimally allocate her limited resources for protecting a set of T targets, labeled $\{1, \ldots, T\}$, against an adversary who tries to attack one of the targets. The key assumption of SSGs is that the defender commits to a mixed strategy first and the adversary can observe that strategy

Appears in: *Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2014), Lomuscio, Scerri, Bazzan, Huhns (eds.), May, 5–9, 2014, Paris, France.*

and make his target choice based on that observation [4, 8].

We denote by $\mathbf{x}$ the defender's strategy. Specifically, x_i refers to the marginal probability that the defender protects target i. In this paper, we will focus on the case where the defender can assign R resources to targets arbitrarily, as long as at most one resource is on each target [6, 15]. The resulting set of feasible marginal probabilities is $\mathbf{X} = \{\mathbf{x} : \ 0 \leq x_i \leq 1, \sum_i x_i \leq R\}$. In SSGs, if the adversary attacks target i, he will receive a reward R_i^a if the defender is not protecting that target, otherwise, he will receive a penalty P_i^a. Conversely, the defender will receive a penalty P_i^d in former case and a reward R_i^d in latter case. Given that the defender chooses strategy $\mathbf{x}$ and the adversary chooses to attack target i, the expected utility of the adversary, $U_i^a(\mathbf{x})$, and the defender, $U_i^d(\mathbf{x})$, are then equal to $U_i^a(\mathbf{x}) = x_i P_i^a + (1 - x_i)R_i^a$ and $U_i^d(\mathbf{x}) = x_i R_i^d + (1 - x_i)P_i^d$ respectively. Finally, we denote by $\mathbf{y} \in \mathbf{Y}$ the adversary's strategy where $\mathbf{Y} = \{\mathbf{y} \in \mathbb{R}^T : y_i \geq 0, \sum_i y_i = 1\}$ is the feasible region of the adversary's strategy, i.e., y_i is the probability that the adversary attacks target i. The expected utility of the defender can be computed as $\sum_i y_i U_i^d(\mathbf{x})$.

3. RELATED WORK

There are two leading approaches for addressing uncertainties in SSGs: 1) modeling uncertainties based on Bayesian Stackelberg game models; and 2) applying robust optimization techniques.

Bayesian Stackelberg approach. Bayesian game is a model of probabilistic uncertainty in games, and has been applied to SSGs for modeling payoff, observation, and execution uncertainties[7, 16]. Furthermore, [16] handles a combination of such uncertainties in SSGs by discretizing this continuous uncertainty space and solving the resulting Bayesian Stackelberg games with discrete follower types. Thus, its solution quality depends on the number of samples. Additionally, [16] *does not integrate* uncertainty due to adversary's bounded rationality. More importantly, the assumption of a known distribution of uncertainties may be inapplicable for many real-world security domains. Therefore, in this work, we focus on applying the second approach.

Robust optimization approach. All previous works following this approach attempt to compartmentalize uncertainties and apply different algorithms to only address a particular type of uncertainty. One simple robust algorithm for dealing with uncertainty in the adversary's bounded rationality is Maximin, which assumes that the adversary can choose any arbitrary strategy given the defender's strategy $\mathbf{x}$. Given Maximin can generate extremely conservative strategies, BRASS [12] provided an advance to handle uncertainty due to adversary bounded rationality: BRASS assumes that the adversary can attack any targets which provide within ϵ of the maximum expected utility for the adversary where ϵ is a given constant.

A more recent robust algorithm, RECON [15], shifted focus away from bounded rationality. RECON only deals with uncertainty in the defender's execution and the adversary's observation. In particular, given the defender's planned strategy $\mathbf{x}$, the real strategy that is executed lies within the range $H(\mathbf{x}) = \{\hat{\mathbf{x}} : \hat{x}_i \in [x_i - \gamma_i, x_i + \gamma_i] \cap [0, 1]\}$. Moreover, given that strategy, $\hat{\mathbf{x}}$, the defender's strategy perceived by the adversary according to his observations lies within the range $[\hat{x}_i - \eta_i, \hat{x}_i + \eta_i] \cap [0, 1]$. As a result, the defender's final strategy at target i perceived by the adversary can be any value within the range $[x_i - \gamma_i - \eta_i, x_i + \gamma_i + \eta_i] \cap [0, 1]$ where (γ_i, η_i) are given constants.

Kiekintveld et al,, on the other hand, proposed a new robust algorithm called ISG which only focused on uncertainty in the adversary's payoffs [6]. In particular, given an assumed adversary's payoff at target i, (R_i^a, P_i^a), the adversary's actual reward and penalty will lie within the ranges $[R_i^a - \alpha_i, R_i^a + \alpha_i]$ and $[P_i^a - \beta_i, P_i^a + \beta_i]$ respectively, where (α_i, β_i) are given constants.

The most recent algorithm, monotonic maximin [5], attempts to deal with uncertainty exclusively in the adversary' bounded rationality. In particular, it attempts to generalize the Quantal Response (QR) [10] model of bounded rationality which was proposed in behavioral economics and has been applied to SSGs. In many cases such as counter-terrorism domains, the defender does not have sufficient data on the adversary's behaviors to accurately estimate the parameters of QR models. Instead, monotonic maximin assumes that the adversary can choose any strategy $\mathbf{y} \in \mathbf{Y}$ that has the following monotonicity property (which is satisfied by all known variants of QR models): $U_i^a(\mathbf{x}) \leq U_j^a(\mathbf{x}) \implies y_i \leq y_j$. In other words, the higher the expected utility of a target, the more likely the adversary will attack that target.

The above discussion summarizes *major* thrusts to handle robustness in SSGs as reported in the literature that we unify in our work. Recent research has explored addressing other types of uncertainties [1, 2], but these have yet to provide robust algorithms. Furthermore, our paper is exactly focused on trying to remedy such salami-slicing of handling of uncertainty.

4. A UNIFIED ROBUST FRAMEWORK

4.1 The space of uncertainties in SSGs

We first summarize the major types of uncertainties that have been studied in previous works as a 3-dimensional *uncertainty space*, shown in Figure 1. As shown in Figure 1, the three dimensions of the uncertainty space are: 1) uncertainty in the adversary's payoff; 2) uncertainty related to the defender's strategy; and 3) uncertainty in the adversary's rationality. These dimensions refer to three key aspects which directly affect both the defender and the adversary's utilities. The origin point of the uncertainty space corresponds to the case with perfectly rational adversary and no uncertainty in the adversary's payoff or related to the defender's strategy.

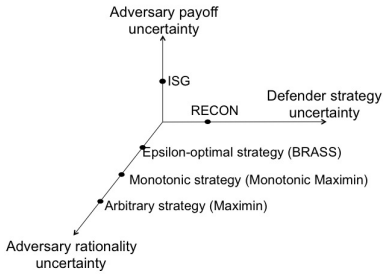


Figure 1: The uncertainty space

Uncertainty in the adversary's payoff has been addressed by ISG [6]. Uncertainty in the defender's strategy can be classified into 2 cases, both addressed by RECON [15]: 1) uncertainty in the defender's execution in which the executed strategy is different from the planned strategy of the defender; and 2) uncertainty in the adversary's observation of the defender's executed strategy. In the dimension of uncertainty in the adversary's rationality, the existing methods can be classified into 3 cases: 1) the adversary can choose any arbitrary strategy (Maximin); 2) the adversary can choose any strategy that satisfies the monotonic property (monotonic maximin [5]); and 3) the adversary can choose any $\epsilon-$optimal strategy (BRASS [12]). It is known that computing the defender's strategy with the first and third cases is equivalent to a special case of uncertainty in adversary's payoff [6, 12]. Therefore, when dealing with rationality uncertainty, we will focus only on the case of monotonic adversary, i.e., monotonic maximin.

As can be seen, the existing robust solution concepts attempt to address only a specific type of uncertainty and thus lie on axes of the space. Thus, we can identify combinations of different uncertainties which correspond to points not on any of the axes, that have not been addressed by previous works.

4.2 A general formulation of uncertainty sets

The existing robust solution concepts for SSGs all follow a standard robust-optimization approach: first represent the uncertainty in the system as an *uncertainty set* of possible models, then choose the decision variables (defender strategy $\mathbf{x}$ in our case) such that the objective (defender utility) is optimized given the worst-case model from the uncertainty set. The main difference among the solution concepts is the way uncertainty sets are defined for different types of uncertainties. For example, in ISG, the uncertainty sets are intervals of adversary payoffs; in RECON, there is an hyper-rectangular uncertainty set around $\mathbf{x}$ representing the strategy executed by the defender, and another hyper-rectangular uncertainty set representing the defender strategy perceived by the adversary.

The first key component of our unified framework is a unified formulation of uncertainty sets for SSGs that captures all major existing approaches. Consider an SSG where all or any subset of the aforementioned uncertainties may be present. We begin by examining our objective function, which is the defender's expected utility $\sum_i y_i U_i^d(\mathbf{x})$. In general, $U_i^d(\mathbf{x})$ is affected by the uncertainty about the execution of defender strategies. The adversary strategy $\mathbf{y}$ will generally depend on his expected utilities $U_i^a(\mathbf{x})$ for all actions i, as well as how he makes decisions based on these expected utilities. Naturally, $\mathbf{y}$ is affected by the uncertainty about adversary rationality. Also, the uncertainties about adversary payoffs and adversary's observation of defender's strategy both affect $U_i^a(\mathbf{x})$, which in turn affects $\mathbf{y}$; furthermore, since the uncertainty about the defender's executed strategy will affect the adversary's observation of it, that in turn also affects $\mathbf{y}$.

Based on the above observations, we build an uncertainty set that captures all uncertainties that affect $U_i^d(\mathbf{x})$, and another for uncertainties that affect $\mathbf{y}$. The former task is simpler since only the execution uncertainty affects $U_i^d(\mathbf{x})$. Given $\mathbf{x} \in \mathbf{X}$, let $H(\mathbf{x}) \subseteq \mathbf{X}$ be the set of possible executed defender strategies, assumed to be convex and non-empty. For example, in RECON the execution uncertainty is represented as intervals around each x_i, as described in Section 3. Other forms of convex sets are possible, but for concreteness in this paper we will focus on the hyper-rectangular form defined by RECON. Given an executed strategy $\hat{\mathbf{x}} \in H(\mathbf{x})$, the defender's expected utility when adversary chooses i is $U_i^d(\hat{\mathbf{x}})$.

The task of defining an uncertainty set for $\mathbf{y}$ is more complex because multiple types of uncertainties are involved. First of all, recall that execution uncertainty indirectly affects $\mathbf{y}$; but since we have already represented execution uncertainty using $H(\mathbf{x})$ above, we take an executed defender strategy $\hat{\mathbf{x}} \in H(\mathbf{x})$ as the input of our definition for the uncertainty set for $\mathbf{y}$. Specifically, given an executed defender strategy $\hat{\mathbf{x}} \in H(\mathbf{x})$, we define $\Psi(\hat{\mathbf{x}}) \subseteq \mathbf{Y}$ as the set of possible adversary strategies, resulting from all or any subset of uncertainties about adversary payoffs, adversary's observation, and adversary's rationality. In this paper we will consider the case with all uncertainties (Sections 4.3 and 5) as well as special cases with subsets of uncertainties (Sections 6 and 7), so it is important to have a definition of the uncertainty set $\Psi(\hat{\mathbf{x}})$ that is general and versatile. With that motivation in mind, we define the general form of $\Psi(\hat{\mathbf{x}})$ as follows.

DEFINITION 4.1. *Given $\hat{\mathbf{x}} \in H(\mathbf{x})$, the uncertainty set $\Psi(\hat{\mathbf{x}}) \subseteq \mathbf{Y}$ is represented as a set of linear constraints on the adversary strategy $\mathbf{y}$. Specifically, there are K potential linear constraints on $\mathbf{y}$, each of which is activated or not depending on $\hat{\mathbf{x}}$. Formally,*

$$\Psi(\hat{\mathbf{x}}) = \{\mathbf{y} \in Y : \ D_k(\hat{\mathbf{x}}) \implies A_k(\mathbf{y}) \geq 0, k = \overline{1,K}\},$$

where $D_k(\hat{\mathbf{x}})$ is a disjunction (logical OR) of a set of conditions: $D_k(\hat{\mathbf{x}}) = \vee_s (D_{ks}(\hat{\mathbf{x}}) \geq 0)$[1] where $D_{ks} : \mathbf{X} \to \mathbb{R}$ are known scalar piecewise linear functions of $\mathbf{x}$. Finally, $A_k : \mathbf{Y} \to \mathbb{R}$ are known scalar linear functions of $\mathbf{y}$, i.e., $A_k(\mathbf{y}) = \sum_i \sigma_{ik} y_i$.

Then, the robust optimization problem of maximizing defender utility given worst-case uncertainty can be formulated as follows:

$$\textbf{P1}: \ \max_{\mathbf{x}} \min_{\hat{\mathbf{x}} \in H(\mathbf{x})} \min_{\mathbf{y} \in \Psi(\hat{\mathbf{x}})} \sum_i y_i U_i^d(\hat{\mathbf{x}})$$

Next, we show that P1 is sufficiently general, i.e., it can capture combinations of the previously-studied types of uncertainties.

4.3 Representation of combined uncertainties

We will focus on two cases in which specific formulations of the uncertainty set are different: 1) combinations of uncertainties with a rational adversary, i.e., uncertainty in the adversary's payoff and the defender's strategy; and 2) combinations of all other uncertainties with a monotonic adversary. Other points in the uncertainty space can then be separated into these two cases.

Consider an SSG where there is uncertainty in the adversary's payoff, i.e., for each target i, the adversary's reward and penalty lie within the range $[R_i^a - \alpha_i, R_i^a + \alpha_i]$ and $[P_i^a - \beta_i, P_i^a + \beta_i]$ respectively. Furthermore there is uncertainty about execution and adversary's observation of the defender strategy as in RECON, with the former represented by $H(\mathbf{x})$ and the latter represented by intervals $[\hat{x}_i - \eta_i, \hat{x}_i + \eta_i] \cap [0,1], \ i = \overline{1,T}$.

Given the defender's executed strategy $\hat{\mathbf{x}}$, the adversary's utility at target i will vary within the range $[\hat{U}^a_{min}(\hat{\mathbf{x}}, i), \hat{U}^a_{max}(\hat{\mathbf{x}}, i)]$ where $\hat{U}^a_{min}(\hat{\mathbf{x}}, i)$ and $\hat{U}^a_{max}(\hat{\mathbf{x}}, i)$ are computed as the following:

$$\hat{U}^a_{min}(\hat{\mathbf{x}}, i) = (R_i^a - \alpha_i)(1 - \hat{x}_i^{max}) + (P_i^a - \beta_i)\hat{x}_i^{max} \quad (1)$$

$$\hat{U}^a_{max}(\hat{\mathbf{x}}, i) = (R_i^a + \alpha_i)(1 - \hat{x}_i^{min}) + (P_i^a + \beta_i)\hat{x}_i^{min} \quad (2)$$

where $\hat{x}_i^{max} = \min\{1, \hat{x}_i + \eta_i\}$ and $\hat{x}_i^{min} = \max\{0, \hat{x}_i - \eta_i\}$.

Rational Adversary. Overall, target i could be potentially attacked by the adversary only when $\hat{U}^a_{max}(\hat{\mathbf{x}}, i) \geq \max_j\{\hat{U}^a_{min}(\hat{\mathbf{x}}, j)\}$. Otherwise, there always exists a target j even with all the uncertainties such that the adversary's utility at target j is greater than at target i, which means that the adversary will never attack target i. Therefore, in the uncertainty set, we have $K = T$ potential linear constraints with $A_k(\mathbf{y}) = -y_k$ and $D_k(\hat{\mathbf{x}}) = \vee_{s=\overline{1,T}}(\hat{U}^a_{min}(\hat{\mathbf{x}}, s) - \hat{U}^a_{max}(\hat{\mathbf{x}}, k) > 0)$ where $k = \overline{1,T}$. Then $(D_k(\hat{\mathbf{x}}) \implies A_k(\mathbf{y}) \geq 0)$ means that given target k, if there is a target $s \in \{1, 2, \ldots, T\}$ such that $\hat{U}^a_{min}(\hat{\mathbf{x}}, s) - \hat{U}^a_{max}(\hat{\mathbf{x}}, k) > 0$, then $y_k = 0$.

Monotonic Adversary. Overall, because the adversary is monotonic, for any pair of targets (i, j), the following constraint must hold: $\hat{U}^a_{min}(\hat{\mathbf{x}}, i) \geq \hat{U}^a_{max}(\hat{\mathbf{x}}, j) \implies y_i \geq y_j$. Conversely, there is no constraint on the attacking probability between target i and j if $\hat{U}^a_{min}(\hat{\mathbf{x}}, i) < \hat{U}^a_{max}(\hat{\mathbf{x}}, j)$ and $\hat{U}^a_{min}(\hat{\mathbf{x}}, j) < \hat{U}^a_{max}(\hat{\mathbf{x}}, i)$. Therefore, in the uncertainty set, we have $\frac{T(T-1)}{2}$ potential linear constraints indexed by $k = (i, j), i \neq j, i, j = \overline{1,T}$ with $A_k(\mathbf{y}) = y_i - y_j$ and $D_k(\hat{\mathbf{x}}) = (\hat{U}^a_{min}(\hat{\mathbf{x}}, i) - \hat{U}^a_{max}(\hat{\mathbf{x}}, j) \geq 0)$. In this case, there is only one condition in $D_k(\hat{\mathbf{x}})$, i.e., $s = 1$.

For example, in a 3-target game, we suppose that $\hat{U}^a_{max}(\hat{\mathbf{x}}, 3) > \hat{U}^a_{min}(\hat{\mathbf{x}}, 1) > \hat{U}^a_{max}(\hat{\mathbf{x}}, 2) > \hat{U}^a_{min}(\hat{\mathbf{x}}, 3)$. In the case of a rational adversary, target 2 will be never attacked by the adversary (the adversary's expected utility at this target is always smaller than at target 1; $\hat{U}^a_{min}(\hat{\mathbf{x}}, 1) > \hat{U}^a_{max}(\hat{\mathbf{x}}, 2)$). In contrast, as both target 1 and 3 satisfy $\hat{U}^a_{max}(\hat{\mathbf{x}}, 3), \hat{U}^a_{max}(\hat{\mathbf{x}}, 1) > \max_i \hat{U}^a_{min}(\hat{\mathbf{x}}, i)$, these two targets could be potentially attacked by the adversary. Therefore, we have $\Psi(\hat{\mathbf{x}}) = \{\mathbf{y} \in Y : \ -y_2 \geq 0\}$.

[1] As we will see later in the paper, for certain types of uncertainties we will use strict inequalities $D_{ks}(\hat{\mathbf{x}}) > 0$ instead of weak inequalities $D_{ks}(\hat{\mathbf{x}}) \geq 0$.

On the other hand, in the case of a monotonic adversary, as $\hat{U}^a_{min}(\hat{\mathbf{x}}, 1) > \hat{U}^a_{max}(\hat{\mathbf{x}}, 2)$ – target 1 will be attacked with higher probability than target 2, we have $\Psi(\hat{\mathbf{x}}) = \{\mathbf{y} \in Y : y_1 - y_2 \geq 0\}$.

4.4 Simplifying the formulation

In general, it is not straightforward to simplify the max-min-min problem **P1** as a single maximin problem because both the defender's executed strategy $\hat{\mathbf{x}}$ and the adversary's strategy $\mathbf{y}$ directly involve in the objective function $\sum_i y_i U^d_i(\hat{\mathbf{x}})$ while the feasible region $\Psi(\hat{\mathbf{x}})$ of $\mathbf{y}$ depends on $\hat{\mathbf{x}}$. In the following, we show that **P1** can be reformulated as a *single maximin* problem based on which we propose a unified robust algorithmic framework described in Section 5.

Given the defender's original strategy, $\mathbf{x}$, the adversary's utility at target i lies within the range $[U^a_{min}(\mathbf{x}, i), U^a_{max}(\mathbf{x}, i)]$ where $U^a_{min}(\mathbf{x}, i)$ and $U^a_{max}(\mathbf{x}, i)$ can be represented as the following:

$$U^a_{min}(\mathbf{x}, i) = (R^a_i - \alpha_i)(1 - x^+_i) + (P^a_i - \beta_i)x^+_i \quad (3)$$

$$U^a_{max}(\mathbf{x}, i) = (R^a_i + \alpha_i)(1 - x^-_i) + (P^a_i + \beta_i)x^-_i \quad (4)$$

where $x^+_i = \min\{1, x_i + \gamma_i + \eta_i\}$ and $x^-_i = \max\{0, x_i - \gamma_i - \eta_i\}$. We define the set of the adversary's strategies, $L(\mathbf{x})$, as the following: 1) in the case of combined uncertainties with a monotonic adversary, $L(\mathbf{x}) = \{\mathbf{y} \in \mathbf{Y} : U^a_{min}(\mathbf{x}, i) \geq U^a_{max}(\mathbf{x}, j) \implies y_i - y_j \geq 0\}$; 2) in the case of combined uncertainties with a rational adversary, $L(\mathbf{x}) = \{\mathbf{y} \in \mathbf{Y} : \vee_i (U^a_{min}(\mathbf{x}, i) - U^a_{max}(\mathbf{x}, j) > 0) \implies -y_j \geq 0\}$. In addition, we define $U^d_{min}(\mathbf{x}, i)$ as follows:

$$U^d_{min}(\mathbf{x}, i) = \max\{0, x_i - \gamma_i\}(R^d_i - P^d_i) + P^d_i \quad (5)$$

In other words, $U^d_{min}(\mathbf{x}, i)$ is the lowest possible value of defender expected utility given that defender chose $\mathbf{x}$ and attacker chose i.

THEOREM 4.2. ***P1** is equivalent to the following:*

$$\textbf{\textit{P2}}: \max_{\mathbf{x} \in \mathbf{X}} \min_{\mathbf{y} \in L(\mathbf{x})} \sum_i y_i U^d_{min}(\mathbf{x}, i)$$

PROOF. See Appendix A of the Online Appendix.[2] □

Finally, as $L(\mathbf{x})$ exhibits the same structure as $\Psi(\hat{\mathbf{x}})$, we can represent $L(\mathbf{x})$ as $\{\mathbf{y} \in Y : D_k(\mathbf{x}) \implies A_k(\mathbf{y}) \geq 0, k = \overline{1, K}\}$. In particular, in the case of combined uncertainties with a rational adversary, for all $k = \overline{1, T}$, we obtain the following: $A_k(\mathbf{y}) = -y_k$ and $D_k(\mathbf{x}) = \vee_s (U^a_{min}(\mathbf{x}, s) - U^a_{max}(\mathbf{x}, k) > 0)$.

Similarly, in the case of combined uncertainties with a monotonic adversary, for all $k = (i, j), i, j = \overline{1, T}$, we have $A_{(i,j)}(\mathbf{y}) = y_i - y_j$ and $D_{(i,j)}(\mathbf{x}) = (U^a_{min}(\mathbf{x}, i) - U^a_{max}(\mathbf{x}, j) \geq 0)$. For example, in a 3-target game, given $\mathbf{x}$, suppose that $U^a_{min}(\mathbf{x}, 1) \geq U^a_{max}(\mathbf{x}, 2)$ and $U^a_{min}(\mathbf{x}, 1) \geq U^a_{max}(\mathbf{x}, 3)$ which mean that the conditions $D_{(1,2)}(\mathbf{x})$ and $D_{(1,3)}(\mathbf{x})$ are true; then the constraints $A_{(1,2)}(\mathbf{y}) \geq 0$ and $A_{(1,3)}(\mathbf{y}) \geq 0$ must hold on $\mathbf{y}$.

5. UNIFIED ROBUST ALGORITHM

5.1 Divide-and-Conquer

In order to solve **P2**, we first need to define $L(\mathbf{x})$ which depends on the defender's strategy $\mathbf{x}$ – this is not a standard maximin problem. We define a subset $C(\mathbf{x})$ as $\{k \in \{1, 2, \ldots, K\} : D_k(\mathbf{x})\}$ which refers to the set of activated linear constraints on $\mathbf{y}$, i.e., $\forall k \in C(\mathbf{x}), A_k(\mathbf{y}) \geq 0$. We define a $T \times K$-constraint matrix $M(C(\mathbf{x}))$ as the following: for all $i = \overline{1, T}$ and $k = \overline{1, K}$:

$$M(C(\mathbf{x}))_{ik} = \begin{cases} \sigma_{ik} & \text{, if } k \in C(\mathbf{x}) \\ 0 & \text{, otherwise} \end{cases}$$

[2]Link: http://aamas2014-paper65.webs.com/Appendix.pdf

where σ_{ik} is introduced in Definition 4.1.

PROPOSITION 5.1. *The set of the adversary's strategies, $L(\mathbf{x})$, can be formulated as $L(\mathbf{x}) = \{\mathbf{y} \in \mathbf{Y} : [M(C(\mathbf{x}))]' \mathbf{y} \geq 0\}$ where $[M(C(\mathbf{x}))]'$ is the transposed matrix of $M(C(\mathbf{x}))$.*

PROOF. See Appendix B of the Online Appendix. □

In the 3-target example described in Section 4.4, we have $C(\mathbf{x}) = \{(1, 2), (1, 3)\}$ and $A_{(i,j)}(\mathbf{y}) = y_i - y_j, \ \forall i, j = \overline{1, T}$ which means that $\sigma_{ik} = 1$ and $\sigma_{jk} = -1$ with $k = (i, j)$, thus $M(C(\mathbf{x}))$ is represented as the following:

$$\begin{pmatrix} (1,2) & (1,3) & (2,1) & (2,3) & (3,1) & (3,2) \\ 1 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Although the constraint matrix depends on the defender's strategy $\mathbf{x}$, the set of $M(C(\mathbf{x}))$ is finite. Specifically, $M(C(\mathbf{x}))$ is a function of the subset $C(\mathbf{x}) \subseteq \{1, 2, ..., K\}$. In other words, each subset of $\{1, 2, ..., K\}$ corresponds to a unique constraint matrix. Because the set $\{1, 2, ..., K\}$ has 2^K subsets, there are at most 2^K constraint matrices. Our key idea of the unified algorithm is to conceptually divide the original optimization problem into multiple sub-optimization problems according to the constraint matrix $M(C(\mathbf{x}))$.

Given a constraint matrix $M(C)$ where $C \subseteq \{1, 2, ..., K\}$, the corresponding sub-optimization problem is defined as

$$\max_{\mathbf{x} \in S^d_C} \min_{\mathbf{y} \in S^a_C} \sum_i y_i U^d_{min}(\mathbf{x}, i) \quad (6)$$

where S^d_C and S^a_C are in turn the sets of the defender's and adversary's strategies corresponding to C. In particular, $S^d_C = \{\mathbf{x} \in \mathbf{X} : C(\mathbf{x}) = C\} = \{\mathbf{x} \in \mathbf{X} : \vee_s (D_{ks}(\mathbf{x}) \geq 0) \ \forall k \in C, \ \wedge_s (D_{ks}(\mathbf{x}) < 0) \ \forall k \notin C\}$ and $S^a_C = \{\mathbf{y} \in \mathbf{Y} : [M(C)]' \mathbf{y} \geq 0\}$. In other words, for all $\mathbf{x} \in S^d_C$, $L(\mathbf{x}) = S^a_C$. As a result, the inner minimization of (6) can be represented as the following LP:

$$\min_{\mathbf{y}} \sum\nolimits_i y_i U^d_{min}(\mathbf{x}, i) \quad (7)$$

$$[M(C)]' \mathbf{y} \geq 0 \quad (8)$$

$$\sum\nolimits_i y_i = 1, 0 \leq y_i \leq 1. \quad (9)$$

Replacing this LP with its dual, the sub-optimization problem (6) can be formulated as the following single maximization problem:

$$\max_{\mathbf{x}, \boldsymbol{\theta}, t} t \quad (10)$$

$$(M(C)\boldsymbol{\theta})_i + t \leq U^d_{min}(\mathbf{x}, i), \ \forall i = \overline{1, T} \quad (11)$$

$$\mathbf{x} \in S^d_C \quad (12)$$

$$\boldsymbol{\theta} \geq 0. \quad (13)$$

where $\boldsymbol{\theta}$ is dually associated with the constraint (8).

Finally, by introducing integer variables to encode the OR operators in S^d_C, each sub-optimization problem (10-13) can be solved as a MILP. The final optimization solution of **P2** can be computed as the maximum of all these sub-optimization problems. However, there is an exponential number, i.e., 2^K, of such sub-optimization problems that we need to solve. In the next section, we introduce a single MILP representation for more efficiently solving **P2**.

5.2 URAC: unified algorithmic framework

Overall, we define an integer vector $\mathbf{z} \in \{0, 1\}^K$ which encodes the set C. Specifically, given a subset $C \subseteq \{1, 2, ..., K\}$, then $z_k = 1$ if $k \in C$; otherwise, $z_k = 0$. As a result, we

have: $M(C)_{ik} = z_k\sigma_{ik}$. Therefore, by using z to refer to 2^K sub-optimization problems, we obtain the general MILP (14-22).

Overall, this MILP with an instantiation of $\mathbf{z}$ is equivalent to a specific sub-optimization problem (10-13). Since the MILP optimizes over all possible values of $\mathbf{z}$, it computes the maximum of all these sub-optimization problems, which is the optimal solution of **P2**. In particular, constraints (11) and (13) correspond to constraints (15-16). In constraint (11), we can rewrite the first term $(M(C)\boldsymbol{\theta})_i$ as $\sum_k z_k\sigma_{ik}\theta_k$ which is a quadratic expression. We apply a standard technique to transform it to a linear expression. Specifically, we define a new variable $\phi_k = z_k\theta_k$. We have $z_k = 0 \implies \phi_k = 0$. On the other hand, $z_k = 1 \implies \phi_k = \theta_k$ where $0 \leq \theta_k$, which means that we only need constraints $0 \leq \phi_k$. As a result, we have the following constraints on ϕ_k: $0 \leq \phi_k \leq Nz_k$ where N is a sufficiently large constant.

$$\max_{\mathbf{x},\mathbf{z},\boldsymbol{\phi},\mathbf{q},t} \; t \tag{14}$$

$$\sum\nolimits_k \sigma_{ik}\phi_k + t \leq U^d_{min}(\mathbf{x}, i),\ \forall i \tag{15}$$

$$0 \leq \phi_k \leq Nz_k,\ \forall k \tag{16}$$

$$r_k + (1 - z_k)N \geq 0,\ \forall k \tag{17}$$

$$r_k - z_kN < 0,\ \forall k \tag{18}$$

$$r_k \geq D_{ks}(\mathbf{x}),\ \forall k, s \tag{19}$$

$$r_k \leq D_{ks}(\mathbf{x}) + (1 - q_{ks})N,\ \forall k, s \tag{20}$$

$$\sum\nolimits_i x_i \leq R,\ x_i \in [0, 1] \tag{21}$$

$$\sum\nolimits_s q_{ks} = 1,\ \forall k, z_i, q_{ks} \in \{0, 1\}. \tag{22}$$

Furthermore, the feasible set of the defender's strategies, S^d_C, in constraint (12) is computed according to the constraints (17-21). Specifically, the constraint $\{\vee_s(D_{ks}(\mathbf{x}) \geq 0)\ \forall k \in C\}$ of S^d_C can be replaced as constraint (17) where $r_k = \max_s D_{ks}(\mathbf{x})$ can be determined by constraints (19-20). In addition, the constraint $\{\wedge_s(D_{ks}(\mathbf{x}) < 0)\ \forall k \notin C\}$ of S^d_C corresponds to constraint (18). MILP solvers generally can not directly deal with strict inequality constraints like (18). In our implementation, we replace (18) with $r_k + \epsilon - z_kN \leq 0,\ \forall k, s$, where ϵ is a small positive constant. This usage of ϵ is consistent with previous formulations such as RECON and ISG [6, 15].

Finally, in order to express $D_{ks}(\mathbf{x})$ and $U^d_{min}(\mathbf{x}, i)$ which are piecewise linear functions of $\mathbf{x}$, we need extra integer variables. For example, we can compute $U^d_{min}(\mathbf{x}, i)$ (Equation (5)) using integer variable $h_i \in \{0, 1\}$ as the following:

$$P^d_i \leq U^d_{min}(\mathbf{x}, i) \leq P^d_i + h_iN \tag{23}$$

$$b_i \leq U^d_{min}(\mathbf{x}, i) \leq b_i + (1 - h_i)N \tag{24}$$

where $b_i = (x_i - \gamma_i)(R^d_i - P^d_i) + P^d_i$. We can determine $U^a_{min}(\mathbf{x}, i)$ and $U^a_{max}(\mathbf{x}, i)$ in $D_{ks}(\mathbf{x})$ in a similar way.

We refer to the MILP (14-22) as the Unified Robust Algorithmic framework for addressing unCertainties (URAC). By replacing $D_k(\mathbf{x})$ and $A_k(\mathbf{x})$ with specific formulations, we obtain a version of URAC for addressing a particular type of uncertainty, i.e., when $A_k(\mathbf{y}) = -y_k$ and $D_k(\mathbf{x}) = \vee_s(U^a_{min}(\mathbf{x}, s) - U^a_{max}(\mathbf{x}, k) > 0)$, the corresponding version of URAC addresses a combined uncertainty with a rational adversary. In fact, no previous robust algorithm could handle all these combinations of uncertainties.

MILP relaxation: We can approximate the piecewise linear functions $U^a_{min}(\mathbf{x}, i), U^a_{max}(\mathbf{x}, i)$ and $U^d_{min}(\mathbf{x}, i)$ with linear functions to reduce the computational complexity of URAC, e.g., we can replace $U^d_{min}(\mathbf{x}, i)$ as $U^d_{min}(\mathbf{x}, i) = (x_i - \gamma_i)(R^d_i - P^d_i) + P^d_i$. In addition, $U^a_{min}(\mathbf{x}, i)$ and $U^a_{max}(\mathbf{x}, i)$ can be approximated in a similar way. We refer to this approximate algorithm as a-URAC.

6. A SCALABLE ROBUST ALGORITHM I

Although (a-)URAC can handle any type of uncertainty in the uncertainty space, these algorithms struggle to scale up to larger problems, due to having potentially large numbers of integer variables. Nevertheless, having the general formulation of uncertainty sets allows us to make the following observation: the constraint functions $A_k(\mathbf{y})$ exhibit two different important properties depending on the types of uncertainties under consideration: 1) in the case of combined uncertainties with a rational adversary, $A_k(\mathbf{y})$ imposes constraints on the targets separately, i.e., $A_k(\mathbf{y}) = -y_k$; 2) in the case of combined uncertainties with a monotonic adversary, $A_k(\mathbf{y})$ involves multiple targets into constraints, i.e., $A_{(i,j)}(\mathbf{y}) = y_i - y_j$. By using these properties, in the next two sections we present two scalable algorithms.

In the case of **combined uncertainties with a rational adversary** (Group 1), overall, we want to apply the binary search method to iteratively search through the space of the defender's utility. At each iteration of the binary search, we need to determine if there exists a feasible solution of the defender's strategy, $\mathbf{x}$, such that $\min_{\mathbf{y}\in L(\mathbf{x})} \sum_i y_iU^d_{min}(\mathbf{x}, i) \geq t$ where t is a given value. This corresponds to a feasibility version of the MILP (14-22), where given t we are asked to find a feasible $\mathbf{x}$. At a high level, because $A_k(\mathbf{y})$ includes only a single target k, constraints (15-20) of $\mathbf{x}$ can be decomposed into separate constraints of x_i. Then by examining the conditions on the defender's coverage at every target independently, we can determine if a utility value t is feasible.

In particular, constraints (15-20) provide conditions by which the linear constraint $A_j(\mathbf{y})$ at target j for all $j = \overline{1, T}$ is (not) activated. Denote by $r = \max_j U^a_{min}(\mathbf{x}, j)$ the maximum value of the adversary's lowest utilities over all targets. We call r the adversary's "cut-off" utility. In addition, we define $i = \text{argmax}_j\, U^a_{min}(\mathbf{x}, j)$ as the "cut-off" target. In fact, if t and i are known in advance, constraints (15-20) reduce to $x_j \geq x^{min}_j$, where x^{min}_j is the required minimum coverage probability of the defender at target j (which we will explain in detail later). As a result, we can determine the defender's minimum coverage, $\{x^{min}_j\}_j$, such that the lowest utility of the defender is t and the "cut-off" target is i. Therefore, given t, minimum amount of resources required is $R^{min} = \min_i\{\sum_j x^{min}_j | i \leftarrow \text{"cut-off" target}\}$.

Hence, t is a feasible utility for the defender only when $R^{min} \leq R$ (Constraint 21). By following the binary search approach, we obtain a scalable algorithm called $\delta-$Optimal Robust Algorithm for Addressing unCertainties ($\delta-$ORAC). This algorithm guarantees an $\delta-$optimal solution for addressing uncertainties in this group where δ is a given positive small value. $\delta-$ORAC is a generalization of ISG which only deals with uncertainty in the adversary's payoff [6]. $\delta-$ORAC arises out of our unified framework and then with $\gamma = \eta = 0$, it becomes equivalent to ISG, whereas with $\alpha = \beta = 0$, it becomes a robust algorithm for dealing with uncertainty in the defender's strategy.

Finally, given a feasible utility t and the "cut-off" target i, x^{min}_j for all $j = \overline{1, T}$ can be determined as follows. As r_k can be computed as $r_k = \max_s D_{ks}(\mathbf{x}) = r - U^a_{max}(\mathbf{x}, k)$, the MILP constraints (15-20) reduce to the following conditions:

Defender utility condition for targets at which linear constraints are not activated: For all such targets j, we have $z_j = 0$. As $\sigma_{jk} = -1$ if $j = k$; otherwise, $\sigma_{jk} = 0$, constraint (15) can be reduced to the following:

$$-\phi_j + t \leq U^d_{min}(\mathbf{x}, j),\ \forall j = \overline{1, T} \tag{25}$$

which implies that if $z_j = 0$, $U^d_{min}(\mathbf{x}, j) \geq t$ or equivalently,

$$x_j^{min} \geq m_j^d \tag{26}$$

where m_j^d is the minimum coverage of the defender on target j ensuring that $U^d_{min}(\mathbf{x}, j)$ is at least t. In particular, if $P_j^d \geq t$, then $m_j^d = 0$. Otherwise, $m_j^d = \frac{t - P_j^d}{R_j^d - P_j^d} + \gamma_j$.

In addition, constraint (18) can be formulated as the following:[3]

$$r - U^a_{max}(\mathbf{x}, j) - z_j N \leq 0, \ \forall j = \overline{1,T} \tag{27}$$

which implies that $U^a_{max}(\mathbf{x}, j) \geq r$ if $z_j = 0$.

As the linear constraint at the "cut-off" target i is always not activated, we have: $x_i^{min} = m_i^d$. Thus, the adversary's "cut-off" utility can be computed as $r = U^a_{min}(\mathbf{x}, i) = \max\{P_i^a - \beta_i, (R_i^a - \alpha_i)(1 - m_i^d - \gamma_i - \eta_i) + (P_i^a - \beta_i)(m_i^d + \gamma_i + \eta_i)\}$.

Adversary utility condition for targets at which linear constraints are activated: For all such targets j, we have $z_j = 1$. Constraint (17) can be simplified as the following:

$$r - U^a_{max}(\mathbf{x}, j) + (1 - z_j)N > 0, \ \forall j = \overline{1,T} \tag{28}$$

which implies that $U^a_{max}(\mathbf{x}, j) < r$ when $z_j = 1$. We approximate this constraint by $U^a_{max}(\mathbf{x}, j) \leq r - \epsilon$, where ϵ is a small positive constant.

Thus, if $z_j = 1$, then $x_j^{min} \geq m_j^a$ where $m_j^a = 0$ if $R_j^a + \alpha_j \leq r - \epsilon$. Otherwise, $m_j^a = \frac{R_j^a + \alpha_j - r + \epsilon}{R_j^a + \alpha_j - (P_j^a + \beta_j)} + \gamma_j + \eta_j$.

As z_j is either 0 or 1, we obtain: $x_j^{min} \geq \min\{m_j^d, m_j^a\}$.

Cut-off utility condition for all targets: Finally, we have:

$$U^a_{min}(\mathbf{x}, j) \leq r, \ \forall j \tag{29}$$

This constraint implies: $x_j^{min} \geq m_j^k$ for all target j where $m_j^k = \max\{0, \frac{R_j^a - \alpha_j - r}{R_j^a - \alpha_j - (P_j^a - \beta_j)} - \gamma_j - \eta_j\}$. In fact, the constraint (29) is equivalent to constraint (19).

As a result, we can determine the smallest necessary amount of defender's resources at every target j as follows:

$$x_j^{min} = \max\{m_j^k, \min\{m_j^d, m_j^a\}\} \tag{30}$$

Moreover, if $\{x_j^{min}\}_j$ satisfy constraint (21), the defender's strategy $\mathbf{x}$ with $x_j = x_j^{min}$ for all j is a feasible solution given t.

7. A SCALABLE ROBUST ALGORITHM II

We now turn to providing an approximate algorithm for the second group, which combines monotonic adversaries with other uncertainties. To that end, we first focus on approximate algorithm for the monotonic maximin problem without other uncertainties, i.e., $\max_{\mathbf{x} \in \mathbf{X}} \min_{\mathbf{y} \in L^m(\mathbf{x})} \sum_i y_i U^d(\mathbf{x}, i)$, by exploiting the structure of the feasible region which corresponds to $A_k(\mathbf{y})$, $L^m(\mathbf{x}) = \{\mathbf{y} \in \mathbf{Y} : U_i^a(\mathbf{x}) \geq U_j^a(\mathbf{x}) \implies y_i \geq y_j\}$.

The main computational difficulty of URAC for this case of uncertainty is due to its T^2 integer variables $z_k, k = (i,j), i, j = \overline{1,T}$. At a high level, our approach builds an alternative formulation with fewer integer variables. Given a defender strategy $\mathbf{x}$, the optimal solution of the inner minimization problem, $\mathbf{y}^*$, will be one of the extreme points of the polytope of $L^m(\mathbf{x})$ which means that $\forall i, j$ such that $y_i^*, y_j^* > 0$, then $y_i^* = y_j^*$. This implies that $L^m(\mathbf{x})$ has at most T extreme points [5]. In practice, we observe that the number of extreme points of $L^m(\mathbf{x}^*)$ where $\mathbf{x}^*$ is the optimal solution of the monotonic maximin problem is often much smaller than T. To exploit this observation, our idea is to find the optimal $\mathbf{x}^*$ within a subset $S_p^d \subseteq \mathbf{X}$ such that for each $\mathbf{x} \in S_p^d$, there are only p extreme points of $L^m(\mathbf{x})$ where $p \ll T$. Intuitively, having to consider fewer extreme points should make the computation simpler. Indeed, this optimization problem can be formulated as a MILP with only pT integer variables.

Specifically, we define S_p^d to be the set of $\mathbf{x}$ such that the targets can be clustered into p groups, each group having the same attacker expected utility. Formally, given $\mathbf{x} \in S_p^d$, define $G_1, G_2, ..., G_p$ as a partition of the T targets such that $\forall \ k = \overline{1,p}$, we have $U_i^a(\mathbf{x}) = U_j^a(\mathbf{x}), \forall i, j \in G_k$, and $\forall \ k < k'$, we have $U_i^a(\mathbf{x}) > U_j^a(\mathbf{x}), \forall i \in G_k, j \in G_{k'}$. Since the monotonic property implies that $U_i^a(\mathbf{x}) = U_j^a(\mathbf{x}) \implies y_i = y_j$, therefore $\forall i, j \in G_k$, we have $y_i = y_j$. We can then write the set of extreme points of $L(\mathbf{x})$ as $S_a(\mathbf{x}) = \{\mathbf{y}^k : y_i^k = \frac{1}{\sum_{r=1}^k |G_r|}, \forall i \in G_s, s \leq k; \ y_i^k = 0, \forall i \in G_s, s > k\}_{k=\overline{1,p}}$. Intuitively, each extreme point $\mathbf{y}^k$ corresponds to the case that the adversary only attacks targets belonging to group $G_1, G_2, \ldots, G_k$ with the same probability. As the optimal strategy of the adversary is an extreme point of $L(\mathbf{x})$, then it belongs to $S_a(\mathbf{x})$. In fact, $\mathbf{y}^* = \text{argmin}_{\mathbf{y}^k} \sum_i y_i^k U_i^d(\mathbf{x})$.

Denote by $B_p^d = \cup_{k=\overline{1,p}} S_k^d$ the set of the defender's strategy such that each $\mathbf{x} \in B_p^d$ will categorize the targets into no more than p groups. Finally, given that $\mathbf{x}^* \in B_p^d$, the monotonic maximin problem becomes $\max_{\mathbf{x} \in B_p^d} \min_{\mathbf{y} \in S_a(\mathbf{x})} \sum_i y_i U_i^d(\mathbf{x})$, which can be encoded as the MILP formulated in (31-40), referred to as the Grouping Monotonic Maximin-p (GMM-p) where p indicates the maximum number of groups.

$$\max_{\mathbf{x}, \mathbf{h}, \mathbf{s}, t, \mathbf{m}} \ t \tag{31}$$

$$U_i^a(\mathbf{x}) + (1 - h_{k,i})N \geq m_k, \ \forall k, i \tag{32}$$

$$m_k \geq U_i^a(\mathbf{x}) - h_{k,i}N + \epsilon, \ \forall k, i \tag{33}$$

$$m_1 \geq U_i^a(\mathbf{x}), \ \forall i \tag{34}$$

$$m_k \geq U_i^a(\mathbf{x}) - h_{k-1,i}N, \ \forall k, i \tag{35}$$

$$h_{k,i} \geq h_{k-1,i}, \ \forall k, i \tag{36}$$

$$s_{k,i} + t \leq U_i^d(\mathbf{x}) + (1 - h_{k,i})N, \ \forall k, i \tag{37}$$

$$-h_{k,i}N \leq s_{k,i} \leq h_{k,i}N, \ \sum_i s_{k,i} = 0, \ \forall k, i \tag{38}$$

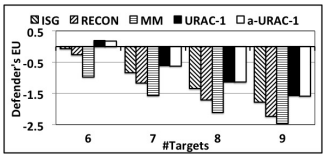
$$h_{p,i} = 1, \ \forall i, \sum_i h_{1,i} \geq 1 \tag{39}$$

$$\sum_i x_i \leq R, 0 \leq x_i \leq 1, h_{k,i} \in \{0,1\}, \ \forall k, i. \tag{40}$$

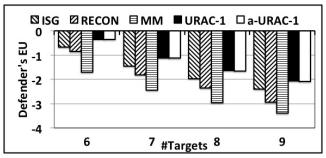
In this MILP, $\mathbf{h}_k$ ($k = \overline{1,p}$) is an integer vector which indicates targets belonging to individual groups. Specifically, if $h_{k,i} = 0$ and $h_{k+1,i} = 1$, target i must belong to group $k+1$ and $h_{k',i} = 1, \ \forall k' \geq k+1$ and $h_{k',i} = 0, \ \forall k' \leq k$. The variable $\mathbf{m}$ represents the adversary's expected utility for each group, i.e., all targets belonging to group k must have the adversary's expected utility equal to m_k. Variable t is the maximum utility of the defender and also the objective value for us to optimize. Finally, $\mathbf{s_k}$ is an auxiliary variable used for computing the defender's utility which is corresponding to a potential optimal strategy of the adversary.

Overall, constraints (32-36) guarantee that all targets in the same group will have the same adversary's expected utility and $\forall i \in G_k, j \in G_{k'}, k < k'$, we have: $U_i^a(\mathbf{x}) > U_j^a(\mathbf{x})$. Furthermore, constraints (37-38) guarantee that if t^* is the optimal objective value, then $t^* = \min_k\{\overline{U}^d(\mathbf{x}, k)\}$ where $\overline{U}^d(\mathbf{x}, k)$ is corresponding defender's utility to a potential optimal strategy $\mathbf{y}^k$ of

[3]As $D_k(\mathbf{x})$ refers to strict inequalities in this case of uncertainty (Section 4.4) which differs from Definition 4.1, constraint (27) is not a strict inequality as (18).

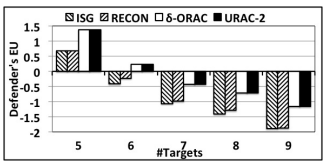


(a) Uncertainty setting: $\alpha = \beta = .1, \gamma = \eta = .01$

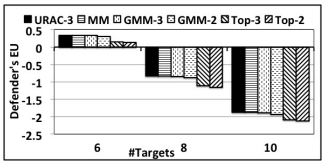


(b) Uncertainty setting: $\alpha = \beta = .5, \gamma = \eta = .05$

Figure 2: Solution quality, all uncertainties



(a) Uncertainty setting: $\alpha = \beta = .5, \gamma = \eta = .05$



(b) Uncertainty setting: monotonic adversary

Figure 3: Solution quality, approximate algorithms

the adversary in $S^a(\mathbf{x})$. Details of the MILP are provided in the proof of Theorem 7.1, in Appendix C of the online appendix.

THEOREM 7.1. *Denote by v_p^* the maximum utility of the defender returned by GMM-p. For all $p > p'$, we have: $v_p^* \geq v_{p'}^*$. Morever, there exists $1 \leq p \leq T$ such that $v_p^* = v^*$ where v^* is the maximum utility of the defender computed by monotonic maximin.*

In the case of **combination of monotonic adversary with other uncertainties**, the grouping idea can still be applied but a somewhat different approach is needed.

8. EXPERIMENTAL RESULTS

We systematically generated payoff structures based on covariance games in GAMUT [11]. We adjust the covariance value $r \in [-1.0, 0.0]$ with step size $\lambda = 0.1$ to control the correlation between rewards of players. The rewards and penalties of both the defender and the adversary are chosen within the ranges $[1, 10]$ and $[-10, -1]$ respectively. The experimental results are obtained using CPLEX on a 2.3 GHz machine with 4GB main memory. All comparison results except where noted with our algorithms are statistically significant under bootstrap-t ($\alpha = .05$) [14].

8.1 Solution quality

We show that our robust algorithms outperform other existing robust algorithms discussed in Section 3 in both small-scale and large-scale game scenarios, under conditions of low or high uncertainties, and given any combinations of uncertainties.

Small-scale domains: In our first set of experiments, we examine the performance of our algorithms in the case of small-scale games which are motivated by real-world domains such as LAX or Boston harbor [13]. We first examine the game settings in which uncertainties exist in all 3 dimensions of the uncertainty space (Figure 1): the adversary's payoff, the defender's strategy, and the adversary's rationality. Specifically, we examine two cases: 1) low uncertainty: $(\alpha = \beta = 0.1, \gamma = \eta = 0.01)$; and 2) high uncertainty: $(\alpha = \beta = 0.5, \gamma = \eta = 0.05)$. In both cases, the adversary responds monotonically. We evaluate the performance of URAC-1 and a-URAC-1 – versions addressing a combination of all uncertainties including monotonic adversary against ISG, RECON, and monotonic maximin (MM). For ISG and RECON, we search over the range of [0.1, 5.0] with step size $\lambda_1 = 0.2$ and the range of [0.01, 0.5] with step size $\lambda_2 = 0.02$ to find the parameter settings for $(\alpha, \beta, \gamma, \eta)$ that provide the highest defender's expected utility in our settings. In fact, when the sampled values of parameters are sufficiently large, i.e., $\alpha = \beta = 5.0$, ISG's optimal solution corresponds to Maximin's. In these figures, the results are averaged over 500 payoff structures.

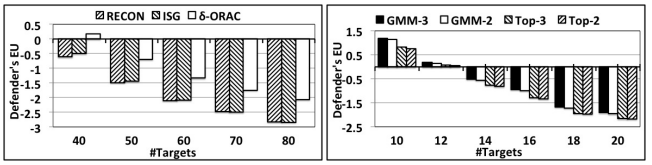
Figure 2 shows the defender's worst-case expected utility (y-axis) while varying the number of targets (x-axis). As shown in Figure 2, even though the parameters of both ISG and RECON are optimally tuned over the sampled values, both URAC-1 and a-URAC-1 still obtain significantly higher defender's expected utility. For example, in Figure 2(a), in the 6-target case, while ISG, RECON, and MM achieve a utility of -0.064, -0.2631, and -0.9672, respectively, the defender's utilities obtained by URAC-1 and a-URAC-1 are, in turn, 0.1892 and 0.1822.

Now we switch to the case of a combination of a subset of or individual uncertainties. In this case, in addition to URAC, we evaluate the performance of our approximate algorithms. Figure 3(a) shows the solution quality of URAC-2 and $\delta-$ORAC where $\delta = 1e-8$ in comparison with RECON and ISG in the case of combined uncertainties with *a rational adversary*. URAC-2 is a version of URAC to address this combination of uncertainties. The parameter values of both RECON and ISG are selected similarly to the previous experiment. In addition, we compare the solution quality of URAC-3, GMM-3, and GMM-2 with monotonic maximin (MM) and the top-K algorithms with K = 2, 3 when addressing the monotonic adversary without any additional uncertainty (Figure 3(b)). URAC-3 is a version of URAC corresponding to this case of uncertainty. Specifically, the top-K algorithms approximate the monotonic maximin solution; higher K achieves higher solution quality but runs more slowly. The Top-3 and Top-2 algorithms are chosen as they are the top performers [5]. In these figures, the results are averaged over 100 payoff structures.

As shown in Figure 3, $\delta-$ORAC, GMM-2, and GMM-3 significantly outperform the other existing robust algorithms. Their solution quality is approximately the same as URAC-2 and URAC-3, respectively. For example, in Figure 3(a), in the case of 9-target games, while the defender's utility obtained by both URAC-2 and $\delta-$ORAC is -1.16, ISG and RECON only achieve utilities of -1.89 and -1.87, respectively. These results show that our robust algorithms outperform other existing robust algorithms in terms of solution quality in small-scale games. The only exception is the isolated uncertainty of monotonic adversary where MM provides the exact optimal solution.

Large-scale domains: Here, we show that our approximate algorithms significantly outperform other robust algorithms for addressing a combination of a subset of or individual uncertainties. We examine 2 game settings: 1) combined uncertainties with a rational adversary (Figure 4(a)); and 2) monotonic adversary (Figure 4(b)). In these figures, the results are averaged over 100 payoff structures. Given the limited scalability of URAC to large games, we do not include its result. Figure 4 shows that our approximate algorithms obtain a significantly higher utility than other robust algorithms in large-scale games. For example, in Figure 4(a), in the case of 80-target games, $\delta-$ORAC achieves a utility of -2.06 while RECON and ISG obtain only -2.82 and -2.83, respectvely.

Furthermore, even when $\delta-$ORAC only attempts to address a specific type of uncertainty, i.e., uncertainty in the defender's strategy ($\alpha = \beta = 0.0$), it provides a higher solution quality than the fastest robust algorithm, i-RECON [15], for dealing with this type of uncertainty. In this experiment, $\delta-$ORAC guarantees to obtain an $\delta-$optimal solution with $\delta = 1e-8$ while i-RECON does not ensure any solution bound. As shown in Figure 5, while both algorithms achieve the similar expected utility when $\gamma = \eta = .01$, when the uncertainty increases, i.e., $\gamma = \eta = .05$, i-RECON ob-

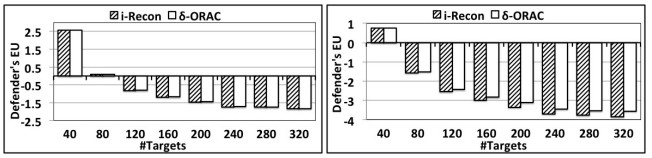


(a) Uncertainty setting: $\alpha = \beta = .5, \gamma = \eta = .05$

(b) Uncertainty setting: monotonic adversary

Figure 4: Solution quality, approximate algorithms

tains lower defender's utility than $\delta-$ORAC. For example, in Figure 5(b), in the case of 320-target games, $\delta-$ORAC obtains a defender's utility of -3.54 on average while i-RECON achieves only -3.84. Overall, our robust algorithms significantly outperform the existing robust algorithms for addressing uncertainties.



(a) $\gamma = \eta = .01$ (b) $\gamma = \eta = .05$

Figure 5: Solution quality, uncertainty in defender's strategy

8.2 Runtime performance

In addition to solution quality, we show that our approximate algorithms obtain an efficient runtime performance in comparison with other robust algorithms in large-scale games. The results are average over 100 payoff structures. In Figure 6, the y-axis indicates the runtime in seconds and the x-axis shows the number of targets. Figure 6(a) shows that $\delta-$ORAC runs significantly faster than i-RECON and its runtime is approximately the same as ISG; i-RECON's runtime grows quickly while $\delta-$ORAC's runtime is consistently fast as the number of targets increases. For example, i-RECON's runtime reaches 144.25 seconds while $\delta-$ORAC and ISG take only 0.05 and 0.046 seconds on average in 320-target games, respectively.

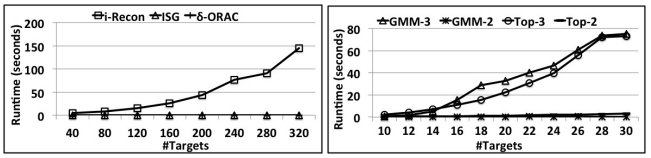
Furthermore, our GMM-p algorithm is shown to have approximately the same runtime as the Top-K algorithm (Figure 6(b)). While our approximate algorithms achieve higher quality without sacrificing runtime, some URAC versions are unable to scale-up. For example, when addressing a combination of all uncertainties (i.e., $\alpha = \beta = 0.5, \gamma = \eta = 0.05$, monotonic adversary), URAC-1's runtime is 85.17 seconds for 9-target games while our approximate algorithms take less than 1 second; of course, URAC-1 addresses a combination of uncertainties that no algorithm can.

9. CONCLUSION

In this paper, we provide the following main contributions: 1) we present the first unified framework to handle all the uncertainties where robust algorithms have been defined in security games; 2) we provide a unified algorithmic framework from which we can derive different "unified" robust algorithms to address combinations of these uncertainties; 3) we introduce approximate robust scalable algorithms; 4) we show through our experiments that our algorithms improve runtime performance and/or solution quality.

10. ACKNOWLEDGEMENT

This research was supported by MURI Grant W911NF-11-1-0332 and by the United States Department of Homeland Security through the Center for Risk and Economic Analysis of Terrorism Events (CREATE) under grant number 2010-ST-061-RE0001.



(a) Uncertainty setting: $\alpha = \beta = .5, \gamma = \eta = .05$

(b) Uncertainty setting: monotonic adversary

Figure 6: Runtime performance, approximate algorithms

11. REFERENCES

[1] B. An, M. Brown, Y. Vorobeychik, and M. Tambe. Security games with surveillance cost and optimal timing of attack execution. In *AAMAS*, 2013.

[2] B. An, M. Tambe, F. Ordonez, E. Shieh, and C. Kiekintveld. Refinement of strong stackelberg equilibria in security games. In *AAAI*, 2011.

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

[4] V. Conitzer and T. Sandholm. Computing the optimal strategy to commit to. In *ACM*, 2006.

[5] A. X. Jiang, T. H. Nguyen, M. Tambe, and A. D. Procaccia. Monotonic maximin: A robust stackelberg solution against boundedly rational followers. In *GameSec*, 2013.

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

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

[8] D. Korzhyk, V. Conitzer, and R. Parr. Complexity of computing optimal stackelberg strategies in security resource allocation games. In *AAAI*, 2010.

[9] J. Letchford and Y. Vorobeychik. Computing randomized security strategies in networked domains. In *AARM Workshop In AAAI*, 2011.

[10] R. McKelvey and T. Palfrey. Quantal response equilibria for normal form games. *Games and economic behavior*, 10(1):6–38, 1995.

[11] E. Nudelman, J. Wortman, Y. Shoham, and K. Leyton-Brown. Run the gamut: A comprehensive approach to evaluating game-theoretic algorithms. In *AAMAS*, 2004.

[12] J. Pita, M. Jain, F. Ordonez, M. Tambe, S. Kraus, and R. Magori-Cohen. Effective solutions for real-world stackelberg games: When agents must deal with human uncertainties. In *AAMAS*, 2009.

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

[14] R. Wilcox. *Applying contemporary statistical techniques*. Academic Press, 2002.

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

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