Making the most of Our Regrets: Regret-based Solutions to Handle Payoff Uncertainty and Elicitation in Green Security Games

Thanh H. Nguyen[1], Francesco M. Delle Fave[1], Debarun Kar[1], Aravind S. Lakshminarayanan[2], Amulya Yadav[1], Milind Tambe[1], Noa Agmon[3], Andrew J. Plumptre[4], Margaret Driciru[5], Fred Wanyama[5], and Aggrey Rwetsiba[5]

[1] University of Southern California, Los Angeles, USA
[thanhhng,dellefav,dkar,amulyaya,tambe]@usc.edu
[2] Indian Institute of Technology, Madras, India
aravindsrinivas@gmail.com
[3] Bar-Ilan University, Israel
agmon@cs.biu.ac.il
[4] Wildlife Conservation Society, USA
aplumptre@wcs.org
[5] Uganda Wildlife Authority, Uganda
[margaret.driciru,fred.wanyama,aggrey.rwetsiba]@ugandawildlife.org

Abstract. Recent research on Green Security Games (GSG), i.e., security games for the protection of wildlife, forest and fisheries, relies on the promise of an abundance of available data in these domains to learn adversary behavioral models and determine game payoffs. This research suggests that adversary behavior models (capturing bounded rationality) can be learned from real-world data on where adversaries have attacked, and that game payoffs can be determined precisely from data on animal densities. However, previous work has, as yet, failed to demonstrate the usefulness of these behavioral models in capturing adversary behaviors based on real-world data in GSGs. Previous work has also been unable to address situations where available data is insufficient to accurately estimate behavioral models or to obtain the required precision in the payoff values.

In addressing these limitations, as our first contribution, this paper, for the first time, provides validation of the aforementioned adversary behavioral models based on real-world data from a wildlife park in Uganda. Our second contribution addresses situations where real-world data is not precise enough to determine exact payoffs in GSG, by providing the first algorithm to handle payoff uncertainty in the presence of adversary behavioral models. This algorithm is based on the notion of minimax regret. Furthermore, in scenarios where the data is not even sufficient to learn adversary behaviors, our third contribution is to provide a novel algorithm to address payoff uncertainty assuming a perfectly rational attacker (instead of relying on a behavioral model); this algorithm allows for a significant scaleup for large security games. Finally, to reduce the problems due to paucity of data, given mobile sensors such as Unmanned Aerial Vehicles (UAV), we introduce new payoff elicitation strategies to strategically reduce uncertainty.

1 Introduction

Following the successful deployments of Stackelberg Security Games (SSG) for infrastructure protection [24, 1, 13], recent research on security games has focused on Green Security Games (GSG) [27, 4, 21, 7]. Generally, this research attempts to optimally allocate limited security resources in a vast geographical area against environmental crime, e.g., improving the effectiveness of protection of wildlife or fisheries [27, 4].

Research in GSGs has differentiated itself from work in SSGs (which often focused on counter-terrorism), not only in terms of the domains of application but also in terms of the amounts of data available. In particular, prior research on SSGs could not claim the presence of large amounts of adversary data [24]. In contrast, GSGs are founded on the promise of an abundance of adversary data (about where the adversaries attacked in the past) that can be used to accurately learn adversary behavior models which capture their bounded rationality [27, 4, 7]. Furthermore, GSG research assumes that available domain data such as animal/fish density is sufficient to help determine payoff values precisely. However, there remain four key shortcomings in GSGs related to these assumptions about data. First, despite proposing different adversary behavioral models (e.g., Quantal Response [28]), GSG research has yet to evaluate these models on any real-world data. Second, the amount of real-world data available is not always present in abundance, introducing different types of uncertainties in GSGs. In particular, in some GSG domains, there is a significant need to handle uncertainty in both the defender and the adversary's payoffs since information on key domain features, e.g., animal density, terrain, etc. that contribute to the payoffs is not precisely known. Third, in some GSG domains, we may even lack sufficient attack data to learn an adversary behavior model, and simultaneously must handle the aforementioned payoff uncertainty. Finally, defenders have access to mobile sensors such as UAVs to elicit information over multiple targets at once to reduce payoff uncertainty, yet previous work has not provided efficient techniques to exploit such sensors for payoff elicitation [17].

In this paper, we address these challenges by proposing four key contributions. As our first contribution, we provide the first results demonstrating the usefulness of behavioral models in SSGs using real-world data from a wildlife park. To address the second limitation of uncertainty over payoff values, our second contribution is ARROW (i.e., **A**lgorithm for **R**educing **R**egret to **O**ppose **W**ildlife crime), a novel security game algorithm that can solve the *behavioral minimax regret problem*. MiniMax Regret (MMR) is a robust approach for handling uncertainty that finds the solution which minimizes the maximum regret (i.e., solution quality loss) with respect to a given uncertainty set [8]. A key advantage of using MMR is that it produces less conservative solutions than the standard maximin approach [17]. ARROW is the first algorithm to compute MMR in the presence of an adversary behavioral model; it is also the first to handle payoff uncertainty in both players' payoffs in SSGs. However, jointly handling of adversary bounded rationality and payoff uncertainty creates the challenge of solving a non-convex optimization problem; ARROW provides an efficient solution to this problem. (Note that we primarily assume a zero-sum game as done in some prior GSG research; however as discussed our key techniques generalize to non-zero sum games as well.)

Our third contribution addresses situations where we do not even have data to learn a behavior model. Specifically, we propose ARROW-Perfect, a novel MMR-based al-

gorithm to handle uncertainty in *both* players' payoffs, assuming a perfectly rational adversary without any requirement of data for learning. ARROW-Perfect exploits the adversary's perfect rationality as well as extreme points of payoff uncertainty sets to gain significant additional efficiency over ARROW.

Another significant advantage of MMR is that it is very useful in guiding the preference elicitation process for learning information about the payoffs [3]. We exploit this advantage by presenting two new elicitation heuristics which select *multiple* targets at a time for reducing payoff uncertainty, leveraging the multi-target-elicitation capability of sensors (e.g., UAVs) available in green security domains. Lastly, we conduct extensive experiments, including evaluations of ARROW based on data from a wildlife park.

2 Background & Related Work

Stackelberg Security Games: In SSGs, the defender attempts to protect a set of T targets from an attack by an adversary by optimally allocating a set of R resources ($R < T$) [24]. The key assumption here is that the defender commits to a (*mixed*) strategy first and the adversary can observe that strategy and then attacks a target. Denote by $\mathbf{x} = \{x_t\}$ the defender's strategy where x_t is the *coverage probability* at target t, the set of feasible strategies is $\mathbf{X} = \{\mathbf{x} : 0 \leq x_t \leq 1, \sum_t x_t \leq R\}$.[6] If the adversary attacks t when the defender is not protecting it, the adversary receives a reward R_t^a, otherwise, the adversary gets a penalty P_t^a. Conversely, the defender receives a penalty P_t^d in the former case and a reward R_t^d in the latter case. Let $(\mathbf{R^a}, \mathbf{P^a})$ and $(\mathbf{R^d}, \mathbf{P^d})$ be the payoff vectors. The players' expected utilities at t is computed as:

$$\mathrm{U^a_t}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = x_t P_t^a + (1 - x_t) R_t^a \tag{1}$$

$$\mathrm{U^d_t}(\mathbf{x}, \mathbf{R^d}, \mathbf{P^d}) = x_t R_t^d + (1 - x_t) P_t^d \tag{2}$$

Boundedly rational attacker: In SSGs, attacker bounded rationality is often modeled via behavior models such as Quantal Response (QR) [14, 15]. QR predicts the adversary's probability of attacking t, denoted by $\mathrm{q_t}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})$ (as shown in Equation 3 where the parameter λ governs the adversary's rationality). Intuitively, the higher the expected utility at a target is, the more likely that the adversary will attack that target.

$$\mathrm{q_t}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = \frac{e^{\lambda \mathrm{U^a_t}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})}}{\sum_{t'} e^{\lambda \mathrm{U^a_{t'}}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})}} \tag{3}$$

The recent SUQR model (Subjective Utility Quantal Response) is shown to provide the best performance among behavior models in security games [18]. SUQR builds on the QR model by integrating the following subjective utility function into QR instead of the expected utility:

$$\hat{\mathrm{U}}^{\mathrm{a}}_{\mathrm{t}}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = w_1 x_t + w_2 R_t^a + w_3 P_t^a \tag{4}$$

[6] The true mixed strategy would be a probability assignment to each pure strategy, where a pure strategy is an assignment of R resources to T targets. However, that is equivalent to the set $\mathbf{X}$ described here, which is a more compact representation [12].

where (w_1, w_2, w_3) are parameters indicating the importance of the three target features for the adversary. The adversary's probability of attacking t is then predicted as:

$$\hat{q}_t(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = \frac{e^{\hat{U}^a_t(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})}}{\sum_{t'} e^{\hat{U}^a_t(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})}} \quad (5)$$

In fact, SUQR is motivated by the lens model which suggested that evaluation of adversaries over targets is based on a linear combination of multiple observable features [5]. One key advantage of these behavioral models is that they can be used to predict attack frequency for multiple attacks by the adversary, wherein the attacking probability is a normalization of attack frequency.

Payoff uncertainty: One key approach to modeling payoff uncertainty is to express the adversary's payoffs as lying within specific intervals [10]: for each target t, we have $R^a_t \in [R^a_{min}(t), R^a_{max}(t)]$ and $P^a_t \in [P^a_{min}(t), P^a_{max}(t)]$. Let $\mathbf{I}$ denote the set of payoff intervals at all targets. An MMR-based solution was introduced in previous work to address payoff uncertainty in SSGs; yet it had two weaknesses: (i) this MMR-based solution is unable to handle uncertainty in both players' payoffs since it assumes that the defender's payoffs are exactly known; and (ii) it has failed to address payoff uncertainty in the context of adversary behavioral models [17].

Green security games: This paper focuses on wildlife protection — many species such as rhinos and tigers are in danger of extinction from poaching [16, 22]. To protect wildlife, game-theoretic approaches have been advocated to generate ranger patrols [27] wherein the forest area is divided into a grid where each cell is a target. These ranger patrols are designed to counter poachers (whose behaviors are modeled using SUQR) that attempt to capture animals by setting snares. A similar system has also been developed for protecting fisheries [4]. Unfortunately, this previous work in wildlife protection [27] has four weaknesses as discussed in Section 1.

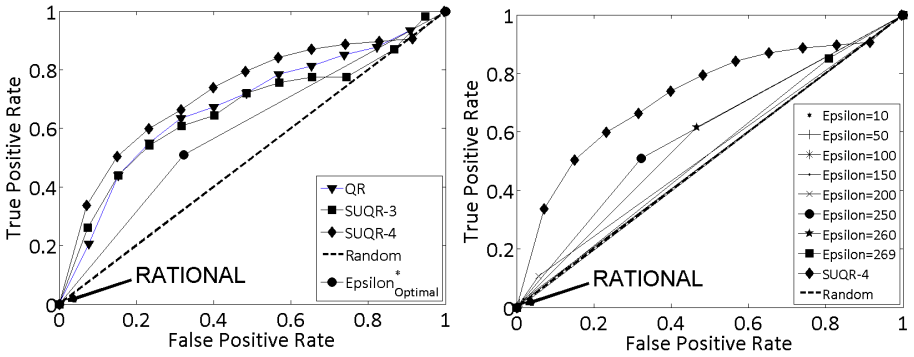
3 Behavioral Modeling Validation

Our first contribution addresses the first limitation of previous work mentioned in Section 1: understanding the extent to which existing behavior models capture real-world behavior data from green security domains. We used a real-world patrol and poaching dataset from Uganda Wildlife Authority supported by Wildlife Conservation Society. This dataset was collected from 1-year patrols in the Queen Elizabeth national park.[7]

3.1 Dataset Description

Our dataset had different types of observations (poacher sighting, animal sighting, etc.) with $40,611$ observations in total recorded by rangers at various locations in the park. The latitude and longitude of the location corresponding to each observation was recorded using a GPS device, thus providing reliable data. Each observation has a feature that specified the total count of the category of observation recorded, for example,

[7] This is the preliminary work on modeling poachers' behaviors. Further study on building more complex behavioral models would be a new interesting research topic for future work.



(a) All Models (b) ϵ-optimal (various values of ϵ) vs SUQR-4

Fig. 1. ROC plots on Uganda dataset

number and type of animals sighted or poaching attacks identified, at a particular location. The date and time for a particular patrol was also present in the dataset. We discretized the park area into 2423 grid cells, with each grid cell corresponding to a $1km \times 1km$ area within the park. After the discretization, each observation fell within one of the 2423 target cells and we therefore aggregated the animal densities and the number of poaching attacks within each target cell. We considered attack data from the year 2012 in our analysis, which has 2352 attacks in total.

Gaussian smoothing of animal densities: Animal density at each target is computed based on the patrols conducted by the rangers and are thus observations at a particular instant of time. Animal density also has a spatial component, meaning that it is unlikely to change abruptly between grid cells. Therefore, to account for movement of animals over a few kilometers in general, we do a blurring of the current recording of animal densities over the cells. To obtain the spatial spread based on recordings of animal sightings, we use Gaussian smoothing; more specifically we use a Gaussian Kernel of size 5×5 with $\sigma = 2.5$ to smoothen out the animal densities over all the grid cells.

Distance as a feature: In addition to animal density, the poachers' payoffs should take into account the distance (or effort) the poacher takes in reaching the grid cell. Therefore, we also use distance as a feature of our SUQR models. Here, the subjective utility function (Equation 4) is extended to include the distance feature: $\hat{U}^a_t(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = w_1 x_t + w_2 R^a_t + w_3 P^a_t + w_4 \Phi_t$ where Φ_t is the distance from the attacker current position to target t. For calculating distance, we took a set of 10 entry points based on geographical considerations. The distance to each target location is computed as the minimum over the distances to this target from the 10 entry points.

3.2 Learning Results

We compare the performance of 13 behavioral models[8] as follows (Figure 1): (i) SUQR-3, which corresponds to SUQR with three features (coverage probability as discussed

[8] Models involving cognitive hierarchies [26] are not applicable in Stackelberg games given that attacker plays knowing the defender's actual strategy.

in Section 2, poacher reward which is considered to be same as the animal density and poacher penalty which is kept uniform over all targets); (ii) SUQR-4, which corresponds to SUQR with four features (coverage probability, animal density, poacher penalty and distance to the target location); (iii) QR; (iv) eight versions of the ϵ-optimal model, a bounded rationality model [20] where the adversary chooses to attack any one of the targets with an utility value which is within ϵ of the optimal target's utility, with equal probability; (v) a random adversary model; and (vi) a perfectly rational model.

From the 2352 total attacks in our dataset, we randomly sampled (10 times) 20% of the attack data for testing and trained the three models: SUQR-3, SUQR-4 and QR on the remaining 80% data. For each train-test split, we trained our behavioral models to learn their parameters, which are used to get probabilities of attack on each grid cell in the test set. Thus, for each grid cell, we get the actual label (whether the target was attacked or not) along with our predicted probability of attack on the cell. Using these labels and the predicted probabilities, we plotted a Receiver Operating Characteristic (ROC) curve (in Figure 1) to analyze the performance of the various models.

The result shows that the perfectly rational model, that deterministically classifies which target gets attacked (unlike SUQR/QR which give probabilities of attack on all targets), achieves an extremely poor prediction accuracy. We also observe that the ϵ^*-optimal model performs worse than QR and SUQR models (Figure 1(a)). Here, by ϵ^*-optimal model, we mean the model corresponding to the ϵ that generates the best prediction (Figure 1(b)). In our case, the best value of ϵ is 250. For the ϵ-optimal model, no matter what ϵ we choose, the curves from the ϵ-optimal method never gets above the SUQR-4 curve, demonstrating that SUQR-4 is a better model than ϵ-optimal. Furthermore, SUQR-4 (Area Under the Curve (AUC) = 0.73) performs better than both QR (AUC = 0.67) and SUQR-3 (AUC = 0.67), thus highlighting the importance of distance as a feature in the adversary's utility. Thus, SUQR-4 provides the highest prediction accuracy and thus will be our model of choice in the rest of the paper.

In summary, comparing many different models shows for the first time support for SUQR from real-world data in the context of GSGs. The SUQR-4 model convincingly beats QR, $\epsilon-$optimal, perfect-rationality and the random model, thus showing the validity of using SUQR in predicting adversary behaviors in GSGs.

4 Behavioral Minimax Regret ($\mathrm{MMR_b}$)

While we can learn a behavioral model from real-world data, we may not always have access to data to precisely compute animal density. For example, given limited numbers of rangers, they may have patrolled and collected wildlife data from only a small portion of a national park, and thus payoffs in other areas of the park may remain uncertain. Also, due to the dynamic changes (e.g., animal migration), players' payoffs may become uncertain in the next season. Hence, this paper introduces our new MMR-based robust algorithm, ARROW, to handle payoff uncertainty in GSGs, taking into account adversary behavioral models. Here, we primarily focus on zero-sum games as motivated by recent work in green security domains [9, 4], and earlier major SSG applications that use zero-sum games [23, 29]). In addition, we use a model inspired by SUQR-4 as the adversary's behavioral model, given its high prediction accuracy presented in

Section 3. More specifically, the subjective utility function in Equation (4) is extended to: $\hat{\mathrm{U}}^{\mathrm{a}}_{\mathrm{t}}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = w_1 x_t + w_2 R^a_t + w_3 P^a_t + w_4 \Phi_t$ where Φ_t is some other feature (e.g., distance) of target t. In fact, our methods generalize to non-zero-sum games with a general class of QR (see Online Appendix A).[9]

We now formulate $\mathrm{MMR_b}$ with uncertain payoffs for both players in zero-sum SSG with a boundedly rational attacker.

Definition 1. *Given* $(\mathbf{R^a}, \mathbf{P^a})$*, the defender's* ***behavioral regret*** *is the loss in her utility for playing a strategy* $\mathbf{x}$ *instead of the optimal strategy, which is represented as follows:*

$$\mathrm{R_b}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = \max_{\mathbf{x'} \in \mathbf{X}} \mathrm{F}(\mathbf{x'}, \mathbf{R^a}, \mathbf{P^a}) - \mathrm{F}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) \tag{6}$$

$$\text{where } \mathrm{F}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = \sum\nolimits_t \hat{\mathrm{q}}_{\mathrm{t}}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) \mathrm{U}^{\mathrm{d}}_{\mathrm{t}}(\mathbf{x}, \mathbf{R^d}, \mathbf{P^d}) \tag{7}$$

Behavioral regret measures the distance in terms of utility loss from the defender strategy $\mathbf{x}$ to the optimal strategy given the attacker payoffs. Here, $\mathrm{F}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})$ is the defender's utility (which is non-convex fractional in $\mathbf{x}$) for playing $\mathbf{x}$ where the attacker payoffs, whose response follows SUQR, are $(\mathbf{R^a}, \mathbf{P^a})$. The defender's payoffs in zero-sum games are $\mathbf{R^d} = -\mathbf{P^a}$ and $\mathbf{P^d} = -\mathbf{R^a}$. In addition, the attacking probability, $\hat{\mathrm{q}}_{\mathrm{t}}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})$, is given by Equation 5. When the payoffs are uncertain, if the defender plays a strategy $\mathbf{x}$, she receives different behavioral regrets w.r.t to different payoff instances within the uncertainty intervals. Thus, she could receive a **behavioral max regret** which is defined as follows:

Definition 2. *Given payoff intervals* $\mathbf{I}$*, the* ***behavioral max regret*** *for the defender to play a strategy* $\mathbf{x}$ *is the maximum behavioral regret over all payoff instances:*

$$\mathrm{MR_b}(\mathbf{x}, \mathbf{I}) = \max_{(\mathbf{R^a}, \mathbf{P^a}) \in \mathbf{I}} \mathrm{R_b}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) \tag{8}$$

Definition 3. *Given payoff intervals* $\mathbf{I}$*, the* ***behavioral minimax regret*** *problem attempts to find the defender optimal strategy that minimizes the* $\mathrm{MR_b}$ *she receives:*

$$\mathrm{MMR_b}(\mathbf{I}) = \min_{\mathbf{x} \in \mathbf{X}} \mathrm{MR_b}(\mathbf{x}, \mathbf{I}) \tag{9}$$

Intuitively, behavorial minimax regret ensures that the defender's strategy minimizes the loss in the solution quality over the uncertainty of all possible payoff realizations.

Example 1. In the 2-target zero-sum game as shown in Table 1, each target is associated with uncertainty intervals of the attacker's reward and penalty. For example, if the adversary successfully attacks Target 1, he obtains a reward which belongs to the interval $[2, 3]$. Otherwise, he receives a penalty which lies within the interval $[-2, 0]$. The attacker's response, assumed to follow SUQR, is defined by the parameters $(w_1 = -10.0, w_2 = 2.0, w_3 = 0.2, w_4 = 0.0)$. Then the defender's optimal mixed strategy generated by behavioral MMR (Equation 9) corresponding to this SUQR model is $\mathbf{x} = \{0.35, 0.65\}$. The attacker payoff values which give the defender the maximum regret w.r.t this behavioral MMR strategy are $(3.0, 0.0)$ and $(5.0, -10.0)$ at Target 1

[9] Online Appendix: https://www.dropbox.com/s/620aqtinqsul8ys/Appendix.pdf?dl=0

Table 1. A 2-target, 1-resource game.

Targets	Attacker reward.	Attacker penalty.
1	[2, 3]	[-2, 0]
2	[5, 7]	[-10, -9]

Algorithm 1: ARROW Outline

```
Initialize S = φ, ub = ∞, lb = 0 ;
Randomly generate sample (x', R^a, P^a), S = S ∪ {x', (R^a, P^a)};
while ub > lb do
    Call R.ARROW to compute relaxed MMR_b w.r.t S. Let x* be its optimal solution
    with objective value lb;
    Call M.ARROW to compute MR_b(x*, I). Let the optimal solution be
    (x'^,*, R^a,*, P^a,*) with objective value ub;
    S = S ∪ {x'^,*, R^a,*, P^a,*};
return (lb, x*);
```

and 2 respectively. In particular, the defender obtains an expected utility of -0.14 for playing $\mathbf{x}$ against this payoff instance. On the other hand, she would receive a utility of 2.06 if playing the optimal strategy $\mathbf{x}' = \{0.48, 0.52\}$ against this payoff instance. As a result, the defender gets a maximum regret of 2.20.

5 ARROW Algorithm: Boundedly Rational Attacker

Algorithm 1 presents the outline of ARROW to solve the $\mathrm{MMR_b}$ problem in Equation 9. Essentially, ARROW's two novelties compared to previous work [17] — addressing uncertainty in both players' payoffs and a boundedly rational attacker — lead to two new computational challenges: 1) uncertainty in defender payoffs makes the defender's expected utility at every target t non-convex in $\mathbf{x}$ and $(\mathbf{R^d}, \mathbf{P^d})$ (Equation 2); and 2) the SUQR model is in the form of a logit function which is non-convex. These two non-convex functions are combined when calculating the defender's utility (Equation 7) — which is then used in computing $\mathrm{MMR_b}$ (Equation 9), making it computationally expensive. Overall, $\mathrm{MMR_b}$ can be reformulated as minimizing the max regret r such that r is no less than the behavioral regrets over all payoff instances within the intervals:

$$\min_{\mathbf{x}\in\mathbf{X}, r\in\mathbb{R}} r \tag{10}$$
$$\text{s.t. } r \geq \mathrm{F}(\mathbf{x}', \mathbf{R^a}, \mathbf{P^a}) - \mathrm{F}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}), \forall (\mathbf{R^a}, \mathbf{P^a}) \in \mathbf{I}, \mathbf{x}' \in \mathbf{X}$$

In (10), the set of (non-convex) constraints is infinite since $\mathbf{X}$ and $\mathbf{I}$ are continuous. One practical approach to optimization with large constraint sets is *constraint sampling* [6], coupled with *constraint generation* [2]. Following this approach, ARROW samples a subset of constraints in Problem (10) and gradually expands this set by adding violated constraints to the relaxed problem until convergence to the optimal $\mathrm{MMR_b}$ solution.

Specifically, ARROW begins by sampling pairs $(\mathbf{R^a}, \mathbf{P^a})$ of the adversary payoffs uniformly from $\mathbf{I}$. The corresponding optimal strategies for the defender given these

payoff samples, denoted $\mathbf{x}'$, are then computed using the PASAQ algorithm [28] to obtain a finite set S of sampled constraints (Line 2). These sampled constraints are then used to solve the corresponding *relaxed* $\mathrm{MMR_b}$ program (line 4) using the R.ARROW algorithm (described in Section 5.1) — we call this problem *relaxed* $\mathrm{MMR_b}$ as it only has samples of constraints in (10). We thus obtain the optimal solution $(lb, \mathbf{x}^*)$ which provides a lower bound (lb) on the true $\mathrm{MMR_b}$. Then constraint generation is applied to determine violated constraints (if any). This uses the M.ARROW algorithm (described in Section 5.2) which computes $\mathrm{MR_b}(\mathbf{x}^*, \mathbf{I})$ — the optimal regret of $\mathbf{x}^*$ which is an upper bound (ub) on the true $\mathrm{MMR_b}$. If $ub > lb$, the optimal solution of M.ARROW, $\{\mathbf{x}'^{,*}, \mathbf{R}^{\mathbf{a},*}, \mathbf{P}^{\mathbf{a},*}\}$, provides the maximally violated constraint (line 5), which is added to S. Otherwise, $\mathbf{x}^*$ is the minimax optimal strategy and $lb = ub = \mathrm{MMR_b}(\mathbf{I})$.

5.1 R.ARROW: Compute Relaxed $\mathrm{MMR_b}$

The first step of ARROW is to solve the relaxed $\mathrm{MMR_b}$ problem using R.ARROW. This relaxed $\mathrm{MMR_b}$ problem is non-convex. Thus, R.ARROW presents two key ideas for efficiency: 1) binary search (which iteratively searches the defender's utility space to find the optimal solution) to remove the fractional terms (i.e., the attacking probabilities in Equation 5) in relaxed $\mathrm{MMR_b}$; and 2) it then applies piecewise-linear approximation to linearize the non-convex terms of the resulting decision problem at each binary search step (as explained below). Overall, relaxed $\mathrm{MMR_b}$ can be represented as follows:

$$\min_{\mathbf{x}\in\mathbf{X}, r\in\mathbb{R}} r \tag{11}$$
$$\text{s.t. } r \geq \mathrm{F}(\mathbf{x}'^{,\mathbf{k}}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}}) - \mathrm{F}(\mathbf{x}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}}), \forall k = 1 \dots K$$

where $(\mathbf{x}'^{,\mathbf{k}}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}})$ is the k^{th} sample in S (i.e., the payoff sample set as described in Algorithm 1) where $k = 1 \dots K$ and K is the total number of samples in S. In addition, r is the defender's max regret for playing $\mathbf{x}$ against sample set S. Finally, $\mathrm{F}(\mathbf{x}'^{,\mathbf{k}}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}})$ is the defender's optimal utility for every sample of attacker payoffs $(\mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}})$ where $\mathbf{x}'^{,\mathbf{k}}$ is the corresponding defender's optimal strategy (which can be obtained via PASAQ [28]). The term $\mathrm{F}(\mathbf{x}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}})$, which is included in relaxed $\mathrm{MMR_b}$'s constraints, is non-convex and fractional in $\mathbf{x}$ (Equation 7), making (11) non-convex and fractional. We now detail the two key ideas of R.ARROW.

Binary search. In each binary search step, given a value of r, R.ARROW tries to solve the decision problem (**P1**) that determines if there exists a defender strategy $\mathbf{x}$ such that the defender's regret for playing $\mathbf{x}$ against any payoff sample in S is no greater than r.

(P1) : $\exists \mathbf{x}$ s.t. $r \geq \mathrm{F}(\mathbf{x}'^{,\mathbf{k}}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}}) - \mathrm{F}(\mathbf{x}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}}), \forall k = 1 \dots K$?

We present the following Proposition 1 showing that (**P1**) can be converted into the *non-fractional* optimization problem (**P2**) (as shown below) of which the optimal solution is used to determine the feasibility of (**P1**):

(P2): $\min_{\mathbf{x}\in\mathbf{X}, v\in\mathbb{R}} v$

$$\text{s.t. } v \geq \sum\nolimits_t \left[\mathrm{F}(\mathbf{x}'^{,\mathbf{k}}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}}) - r - \mathrm{U}_{\mathrm{t}}^{\mathrm{d,k}}(\mathbf{x})\right] e^{\hat{\mathrm{U}}_{\mathrm{t}}^{\mathrm{a}}(\mathbf{x}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}})}, \forall k = 1 \dots K$$

where $\mathrm{U_t^{d,k}}(\mathbf{x}) = -\left[x_t P_t^{a,k} + (1-x_t)R_t^{a,k}\right]$ is the defender's expected utility at target t given $\mathbf{x}$ and the k^{th} payoff sample.

Proposition 1. *Suppose that $(v^*, \mathbf{x}^*)$ is the optimal solution of (**P2**). If $v^* \leq 0$, then $\mathbf{x}^*$ is a feasible solution of the decision problem (**P1**). Otherwise, (**P1**) is infeasible.*

The proof of Proposition 1 is in Online Appendix B. Given that the decision problem **(P1)** is now converted into the optimization problem **(P2)**, as the next step, we attempt to solve **(P2)** using piecewise linear approximation.

Piecewise linear approximation. Although (**P2**) is non-fractional, its constraints are non-convex. We use a piecewise linear approximation for the RHS of the constraints in (**P2**) which is in the form of $\sum_t f_t^k(x_t)$ where the term $f_t^k(x_t)$ is a non-convex function of x_t (recall that x_t is the defender's coverage probability at target t). The feasible region of the defender's coverage x_t for all t, $[0,1]$, is then divided into M equal segments $\left\{\left[0, \frac{1}{M}\right], \left[\frac{1}{M}, \frac{2}{M}\right], \ldots, \left[\frac{M-1}{M}, 1\right]\right\}$ where M is given. The values of $f_t^k(x_t)$ are then approximated by using the segments connecting pairs of consecutive points $\left(\frac{i-1}{M}, f_t^k\left(\frac{i-1}{M}\right)\right)$ and $\left(\frac{i}{M}, f_t^k\left(\frac{i}{M}\right)\right)$ for $i = 1 \ldots M$ as follows:

$$f_t^k(x_t) \approx f_t^k(0) + \sum\nolimits_{i=1}^{M} \alpha_{t,i}^k x_{t,i} \tag{12}$$

where $\alpha_{t,i}^k$ is the slope of the i^{th} segment which can be determined based on the two extreme points of the segment. Also, $x_{t,i}$ refers to the portion of the defender's coverage at target t belonging to the i^{th} segment, i.e., $x_t = \sum_i x_{t,i}$.

Example 2. When the number of segments $M = 5$, it means that we divide $[0,1]$ into 5 segments $\left\{\left[0, \frac{1}{5}\right], \left[\frac{1}{5}, \frac{2}{5}\right], \left[\frac{2}{5}, \frac{3}{5}\right], \left[\frac{3}{5}, \frac{4}{5}\right], \left[\frac{4}{5}, 1\right]\right\}$. Suppose that the defender's coverage at target t is $x_t = 0.3$, since $\frac{1}{5} < x_t < \frac{2}{5}$, we obtain the portions of x_t that belongs to each segment is $x_{t,1} = \frac{1}{5}$, $x_{t,2} = 0.1$, and $x_{t,3} = x_{t,4} = x_{t,5} = 0$ respectively. Then each non-linear term $f_t^k(x_t)$ is approximated as $f_t^k(x_t) \approx f_t^k(0) + \frac{1}{5}\alpha_{t,1}^k + 0.1\alpha_{t,2}^k$ where the slopes of the 1^{st} and 2^{nd} segments are $\alpha_{t,1}^k = 5\left[f_t^k\left(\frac{1}{5}\right) - f_t^k(0)\right]$ and $\alpha_{t,2}^k = 5\left[f_t^k\left(\frac{2}{5}\right) - f_t^k\left(\frac{1}{5}\right)\right]$ respectively.

By using the approximations of $f_t^k(x_t)$ for all k and t, we can reformulate (**P2**) as the MILP **(P2')** which can be solved by the solver CPLEX:

$$\textbf{(P2')}: \min_{x_{t,i}, z_{t,i}, v} v \tag{13}$$

$$\text{s.t. } v \geq \sum\nolimits_t f_t^k(0) + \sum\nolimits_t \sum\nolimits_i \alpha_{t,i}^k x_{t,i}, \forall k = 1 \ldots K \tag{14}$$

$$\sum\nolimits_{t,i} x_{t,i} \leq R, 0 \leq x_{t,i} \leq \frac{1}{M}, \forall t = 1 \ldots T, i = 1 \ldots M \tag{15}$$

$$z_{t,i}\frac{1}{M} \leq x_{t,i}, \forall t = 1 \ldots T, i = 1 \ldots M-1 \tag{16}$$

$$x_{t,i+1} \leq z_{t,i}, \forall t = 1 \ldots T, i = 1 \ldots M-1 \tag{17}$$

$$z_{t,i} \in \{0,1\}, \forall t = 1 \ldots T, i = 1 \ldots M-1 \tag{18}$$

where $z_{t,i}$ is an auxiliary integer variable which ensures that the portions of x_t satisfies $x_{t,i} = \frac{1}{M}$ if $x_t \geq \frac{i}{M}$ ($z_{t,i} = 1$) or $x_{t,i+1} = 0$ if $x_t < \frac{i}{M}$ ($z_{t,i} = 0$) (constraints (15

– 18)). Constraints (14) are piecewise linear approximations of constraints in **(P2)**. In addition, constraint (15) guarantees that the resource allocation condition, $\sum_t x_t \leq R$, holds true and the piecewise segments $0 \leq x_{t,i} \leq \frac{1}{M}$.

Finally, we provide Theorem 1 showing that R.ARROW guarantees a solution bound on computing relaxed $\mathrm{MMR_b}$. The proof of Theorem 1 is in the Online Appendix C.

Theorem 1. *R.ARROW provides an $O\left(\epsilon + \frac{1}{M}\right)$-optimal solution of relaxed $\mathrm{MMR_b}$ where ϵ is the tolerance of binary search and M is the number of piecewise segments.*

5.2 M.ARROW: Compute $\mathbf{MR_b}$

Given the optimal solution $\mathbf{x}^*$ returned by R.ARROW, the second step of ARROW is to compute $\mathrm{MR_b}$ of $\mathbf{x}^*$ using M.ARROW (line 5 in Algorithm 1). The problem of computing $\mathrm{MR_b}$ can be represented as the following non-convex maximization problem:

$$\max_{\mathbf{x}' \in \mathbf{X}, (\mathbf{R^a}, \mathbf{P^a}) \in \mathbf{I}} \mathrm{F}(\mathbf{x}', \mathbf{R^a}, \mathbf{P^a}) - \mathrm{F}(\mathbf{x}^*, \mathbf{R^a}, \mathbf{P^a}) \tag{19}$$

Overall, it is difficult to apply the same techniques used in R.ARROW for M.ARROW since it is a subtraction of two non-convex fractional functions, $\mathrm{F}(\mathbf{x}', \mathbf{R^a}, \mathbf{P^a})$ and $\mathrm{F}(\mathbf{x}^*, \mathbf{R^a}, \mathbf{P^a})$. Therefore, we use local search with multiple starting points which allows us to reach different local optima.

6 ARROW-Perfect Algorithm: Perfectly Rational Attacker

While ARROW incorporates an adversary behavioral model, it may not be applicable for green security domains where there may be a further paucity of data in which not only payoffs are uncertain but also parameters of the behavioral model are difficult to learn accurately. Therefore, we introduce a novel MMR-based algorithm, ARROW-Perfect, to handle uncertainty in both players' payoffs assuming a perfectly rational attacker. In general, ARROW-Perfect follows the same *constraint sampling* and *constraint generation* methodology as ARROW. Yet, by leveraging the property that the attacker's optimal response is a pure strategy (given a perfectly rational attacker) and the game is zero-sum, we obtain the *exact optimal solutions* for computing both relaxed MMR and max regret in *polynomial time* (while we cannot provide such guarantees for a boundedly rational attacker). In this case, we call the new algorithms for computing relaxed MMR and max regret: R.ARROW-Perfect and M.ARROW-Perfect respectively.

6.1 R.ARROW-Perfect: Compute Relaxed MMR

In zero-sum games, when the attacker is perfectly rational, the defender's utility for playing a strategy $\mathbf{x}$ w.r.t the payoff sample $(\mathbf{R^{a,k}}, \mathbf{P^{a,k}})$ is equal to $\mathrm{F}\left(\mathbf{x}, \mathbf{R^{a,k}}, \mathbf{P^{a,k}}\right) = -\mathrm{U_t^a}\left(\mathbf{x}, \mathbf{R^{a,k}}, \mathbf{P^{a,k}}\right)$ if the attacker attacks target t. Since the adversary is perfectly rational, therefore, $\mathrm{F}\left(\mathbf{x}, \mathbf{R^{a,k}}, \mathbf{P^{a,k}}\right) = -\max_t \mathrm{U_t^a}\left(\mathbf{x}, \mathbf{R^{a,k}}, \mathbf{P^{a,k}}\right)$, we can reformulate the relaxed MMR in (11) as the following linear minimization problem:

$$\min_{\mathbf{x} \in \mathbf{X}, r \in \mathbb{R}} r \tag{20}$$

$$\text{s.t. } r \geq \mathrm{F}\left(\mathbf{x}', \mathbf{R^{a,k}}, \mathbf{P^{a,k}}\right) + \mathrm{U_t^a}\left(\mathbf{x}, \mathbf{R^{a,k}}, \mathbf{P^{a,k}}\right), \forall k = 1 \ldots K, \forall t = 1 \ldots T \tag{21}$$

where $\mathrm{F}\left(\mathbf{x}'^{,\mathbf{k}}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}}\right)$ is the defender's optimal utility against a perfectly rational attacker w.r.t payoff sample $\left(\mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}}\right)$ and $\mathbf{x}'^{,\mathbf{k}}$ is the corresponding optimal strategy which is the Maximin solution. In addition, constraint (21) ensures that the regret $r \geq \mathrm{F}\left(\mathbf{x}'^{,\mathbf{k}}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}}\right) + \max_t \mathrm{U}^{\mathrm{a}}_{\mathrm{t}}\left(\mathbf{x}, \mathbf{R}^{\mathbf{a},\mathbf{k}}, \mathbf{P}^{\mathbf{a},\mathbf{k}}\right)$ for all payoff samples. This linear program can be solved exactly in polynomial time using any linear solver, e.g., CPLEX.

6.2 M.ARROW-Perfect: Compute Max Regret

Computing max regret (MR) in zero-sum games presents challenges that previous work [17] can not handle since the defender's payoffs are uncertain while [17] assumes these payoff values are known. In this work, we propose a new exact algorithm, M.ARROW-Perfect, to compute MR in polynomial time by exploiting insights of zero-sum games.

In zero-sum games with a perfectly rational adversary, Strong Stackelberg Equilibrium is equivalent to Maximin solution [30]. Thus, given the strategy $\mathbf{x}^*$ returned by relaxed MMR, max regret in (19) can be reformulated as follows:

$$\max_{\mathbf{x}' \in \mathbf{X}, (\mathbf{R}^{\mathbf{a}}, \mathbf{P}^{\mathbf{a}}) \in \mathbf{I}, v} v - \mathrm{F}(\mathbf{x}^*, \mathbf{R}^{\mathbf{a}}, \mathbf{P}^{\mathbf{a}}) \tag{22}$$

$$\text{s.t. } v \leq -\left[x'_t P^a_t + (1 - x'_t) R^a_t\right], \forall t \tag{23}$$

where v is the Maximin/SSE utility for the defender against the attacker payoff $(\mathbf{R}^{\mathbf{a}}, \mathbf{P}^{\mathbf{a}})$. Moreover, the defender's utility for playing $\mathbf{x}^*$ can be computed as $\mathrm{F}(\mathbf{x}^*, \mathbf{R}^{\mathbf{a}}, \mathbf{P}^{\mathbf{a}}) = -\left[x^*_j P^a_j + (1 - x^*_j) R^a_j\right]$ if the adversary attacks target j. Thus, we divide the attacker payoff space into T subspaces such that within the j^{th} subspace, the adversary always attacks target j against the defender strategy $\mathbf{x}^*$, for all $j = 1 \ldots T$. By solving these T sub-max regret problems corresponding to this division, our final global optimal solution of max regret will be the maximum of all T sub-optimal solutions.

Next, we will explain how to solve these sub-max regret problems. Given the j^{th} attacker payoff sub-space, we obtain the j^{th} sub-max regret problem as:

$$\max_{\mathbf{x}' \in \mathbf{X}, (\mathbf{R}^{\mathbf{a}}, \mathbf{P}^{\mathbf{a}}) \in \mathbf{I}, v} v + (x^*_j P^a_j + (1 - x^*_j) R^a_j) \tag{24}$$

$$\text{s.t. } v \leq -[x'_t P^a_t + (1 - x'_t) R^a_t], \forall t \tag{25}$$

$$x^*_j P^a_j + (1 - x^*_j) R^a_j \geq x^*_t P^a_t + (1 - x^*_t) R^a_t, \forall t \tag{26}$$

where constraints (26) ensures that the adversary attacks target j against the defender strategy $\mathbf{x}^*$. Here, constraints (25) are non-convex for all targets. We provide the following proposition which allows us to linearize constraints (25) for all targets but j.

Proposition 2. *Given target j, the lower bounds of the attacker's payoffs at all targets except j, $\{R^a_{min}(t), P^a_{min}(t)\}_{t \neq j}$, are optimal solutions of $\left\{R^a_j, P^a_j\right\}_{t \neq j}$ for the j^{th} sub-max regret problem.*

The proof of Proposition 2 is in Online Appendix D. Now, only constraint (25) w.r.t target j remains non-convex for which we provide further steps to simplify it. Given the defender strategy $\mathbf{x}'$, we define the attack set as including all targets with the attacker's highest expected utility: $\Gamma(\mathbf{x}') = \{t : \mathrm{U}^{\mathrm{a}}_{\mathrm{t}}(\mathbf{x}', \mathbf{R}^{\mathbf{a}}, \mathbf{P}^{\mathbf{a}}) = \max_{t'} \mathrm{U}^{\mathrm{a}}_{\mathrm{t}'}(\mathbf{x}', \mathbf{R}^{\mathbf{a}}, \mathbf{P}^{\mathbf{a}})\}$.

We provide the following observations based on which we can determine the optimal value of the attacker's reward at target j, R_j^a, for the sub-max regret problem (24–26) (according to the Proposition 3 below):

Observation 1 *If $\mathbf{x}'$ is the optimal solution of computing the j^{th} sub-max regret in (24–26), target j belongs to the attack set $\Gamma(\mathbf{x}')$.*

Since $\mathbf{x}'$ is the Maximin or SSE solution w.r.t attacker payoffs $(\mathbf{R^a}, \mathbf{P^a})$, the corresponding attack set $\Gamma(\mathbf{x}')$ has the maximal size [11]. In other words, if a target t belongs to the attack set of any defender strategy w.r.t $(\mathbf{R^a}, \mathbf{P^a})$, then $t \in \Gamma(\mathbf{x}')$. In (24–26), because target j belongs to the attack set $\Gamma(\mathbf{x}^*)$, we obtain $j \in \Gamma(\mathbf{x}')$.

Observation 2 *If $\mathbf{x}'$ is the optimal solution of computing the j^{th} sub-max regret in (24–26), the defender's coverage at target j: $x'_j \geq x^*_j$.*

Since $j \in \Gamma(\mathbf{x}')$ according to Observation 1, the defender utility for playing $\mathbf{x}'$ is equal to $v = -[x'_j P_j^a + (1 - x'_j) R_j^a]$. Furthermore, the max regret in (24) is always not less than zero, meaning that $v \geq -\left[x_j^* P_j^a + (1 - x_j^*) R_j^a\right]$. Thus, we obtain $x'_j \geq x^*_j$.

Proposition 3. *Given target j, the upper bound of the attacker's reward at j, $R^a_{max}(j)$, is an optimal solution of the attacker reward R_j^a for the j^{th} sub-max regret problem.*

Proof. Suppose that $R_j^a < R^a_{max}(j)$ is optimal in (24–26) and $\mathbf{x}'$ is the corresponding defender optimal strategy, then $v = -[x'_j P_j^a + (1 - x'_j) R_j^a]$ according to the Observation 1. We then replace R_j^a with $R^a_{max}(j)$ while other rewards/penalties and $\mathbf{x}'$ remain the same. Since $R_j^a < R^a_{max}(j)$, this new solution is also feasible for (24–26) and target j still belongs to $\Gamma(\mathbf{x}')$. Therefore, the corresponding utility of the defender for playing $\mathbf{x}'$ will be equal to $-\left[x'_j P_j^a + (1 - x'_j) R^a_{max}(j)\right]$. Since $R_j^a < R^a_{max}(j)$ and $x'_j \geq x^*_j$ (Observation 2), the following inequality holds true:

$$-\left[x'_j P_j^a + (1 - x'_j) R^a_{max}(j)\right] + \left[(x_j^* P_j^a + (1 - x_j^*) R^a_{max}(j)\right] \tag{27}$$

$$= -\left[x'_j P_j^a + (1 - x'_j) R_j^a\right] + \left[(x_j^* P_j^a + (1 - x_j^*) R_j^a\right] + \left[x'_j - x_j^*\right]\left[R^a_{max}(j) - R_j^a\right] \tag{28}$$

$$\geq -\left[x'_j P_j^a + (1 - x'_j) R_j^a\right] + \left[(x_j^* P_j^a + (1 - x_j^*) R_j^a\right]. \tag{29}$$

This inequality indicates that the defender's regret w.r.t $R^a_{max}(j)$ (the LHS of the inequality) is no less than w.r.t R_j^a (the RHS of the inequality). Therefore, $R^a_{max}(j)$ is an optimal solution of the attacker's reward at target j for (24–26). ■

Based on the Proposition 2 & 3 and the Observation 1, the j^{th} sub-max regret (24–26) is simplified to the following optimization problem:

$$\max_{\mathbf{x}' \in \mathbf{X}, P_j^a, v} v + (x_j^* P_j^a + (1 - x_j^*) R^a_{max}(j)) \tag{30}$$

$$\text{s.t. } v = -\left[x'_j P_j^a + (1 - x'_j) R^a_{max}(j)\right] \tag{31}$$

$$v \leq -\left[x'_t P^a_{min}(t) + (1 - x'_t) R^a_{min}(t)\right], \forall t \neq j \tag{32}$$

$$P^a_{max}(j) \geq P_j^a \geq \max\left\{P^a_{min}(j), \frac{C - (1 - x_j^*) R^a_{max}(j)}{x_j^*}\right\} \tag{33}$$

where $C = \max_{t \neq j} x_t^* P_{min}^a(t) + (1 - x_t^*) R_{min}^a(t)$ is a constant. In addition, constraints (31–32) refer to constraint (25) (where constraint (31) is a result of Observation 1) and constraints (33) is equivalent to constraint (26). The only remaining non-convex term is $x'_j P_j^a$ in constraint (31). We then alleviate the computational cost incurred based on Theorem 2 which shows that if the attack set $\Gamma(\mathbf{x}')$ is known beforehand, we can convert (30–33) into a simple optimization problem which is straightforward to solve.

Theorem 2. *Given the attack set $\Gamma(\mathbf{x}')$, the j^{th} sub-max regret problem (30–33) can be represented as the following optimization problem on the variable v only:*

$$\max_{v} v + \frac{av + b}{cv + d} \tag{34}$$

$$\textit{s.t. } v \in [l_v, u_v]. \tag{35}$$

where v is the defender utility for playing $\mathbf{x}'$ in (30–33).

The proof of Theorem 2 is in Online Appendix E. The constants (a, b, c, d, l_v, u_v) are determined based on the attack set $\Gamma(\mathbf{x}')$, the attacker's payoffs $\{R_{min}^a(t), P_{min}^a(t)\}_{t \neq j}$ and $R_{max}^a(j)$, and the number of the defender resources R. Here, the total number of possible attack sets $\Gamma(\mathbf{x}')$ is maximally T sets according to the property that $R_t^a > R_{t'}^a$ for all $t \in \Gamma(\mathbf{x}')$ and $t' \notin \Gamma(\mathbf{x}')$ [11]. Therefore, we can iterate over all these possible attack sets and solve the corresponding optimization problems in (34–35). The optimal solution of each sub-max regret problem (30–33) will be the maximum over optimal solutions of (34–35). The final optimal solution of the max regret problem (22) will be the maximum over optimal solutions of all these sub-max regret problems.

In summary, we provide the M.ARROW-Perfect algorithm to exactly compute max regret of playing the strategy $\mathbf{x}^*$ against a perfectly rational attacker in zero-sum games by exploiting the insight of extreme points of the uncertainty intervals as well as attack sets. Furthermore, we provide Theorem 3 (its proof is in the Online Appendix F) showing that the computational complexity of solving max regret is polynomial.

Theorem 3. *M.ARROW-Perfect provides an optimal solution for computing max regret against a perfectly rational attacker in $O(T^3)$ time.*

7 UAV Planning for Payoff Elicitation (PE)

Our final contribution is to provide PE heuristics to select the best UAV path to reduce uncertainty in payoffs, given any adversary behavioral model. Despite the limited availability of mobile sensors in conservation areas (many of them being in developing countries), these UAVs may still be used to collect accurate imagery of these areas periodically, e.g., every six months to reduce payoff uncertainty. Since the UAV availability is limited, it is important to determine the best UAV paths such that reducing payoff uncertainty at targets on these paths could help reducing the defender's regret the most. While a UAV visits multiple targets to collect data, planning an optimal path (which considers all possible outcomes of reducing uncertainty) is computationally expensive. Thus, we introduce the *current solution*-based algorithm which evaluates a UAV path based solely on the $\mathrm{MMR_b}$ solution given current intervals.[10]

[10] A similar idea was introduced in [2] although in a very different domain without UAV paths.

Algorithm 2: Elicitation process

1 Input: budget: B, regret barrier: δ, uncertainty intervals: $\mathbf{I}$;
2 Initialize regret $r = +\infty$, cost $c = 0$;
3 **while** $c < B$ *and* $r > \delta$ **do**
4 $\quad (r, \mathbf{x}^*, (\mathbf{x}'^{,*}, \mathbf{R}^{\mathbf{a},*}, \mathbf{P}^{\mathbf{a},*})) =$ *ARROW*$(\mathbf{I})$;
5 $\quad \mathbf{P} =$ *calculatePath*$(\mathbf{x}^*, (\mathbf{x}'^{,*}, \mathbf{R}^{\mathbf{a},*}, \mathbf{P}^{\mathbf{a},*}))$;
6 $\quad \mathbf{I} =$ *collectInformationUAV*$(\mathbf{P})$; $c =$ *updateCost*$(\mathbf{P})$;
7 return $(r, \mathbf{x}^*)$;

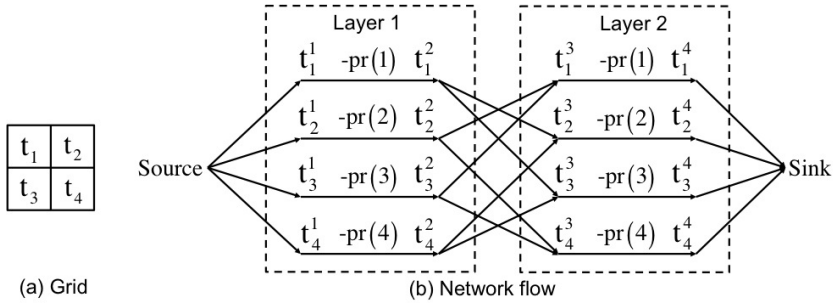
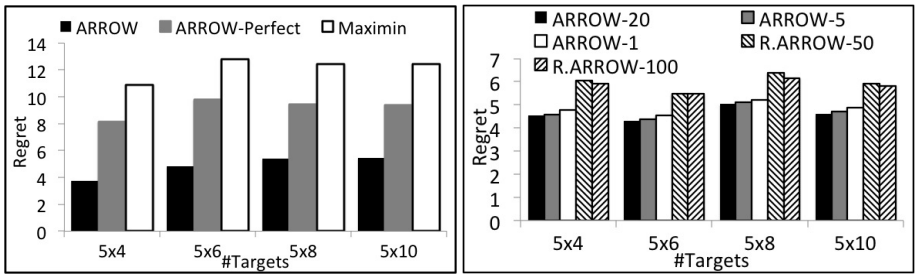


Fig. 2. Min Cost Network Flow

We first present a general elicitation process for UAV planning (Algorithm 2). The input includes the defender's initial budget B (e.g., limited time availability of UAVs), the regret barrier δ which indicates how much regret (utility loss) the defender is willing to sacrifice, and the uncertainty intervals $\mathbf{I}$. The elicitation process consists of multiple rounds of flying a UAV and stops when the UAV cost exceeds B or the defender's regret is less than δ. At each round, ARROW is applied to compute the optimal $\mathrm{MMR_b}$ solution given current $\mathbf{I}$; ARROW then outputs the regret r, the optimal strategy $\mathbf{x}^*$, and the corresponding most unfavorable strategy and payoffs $(\mathbf{x}'^{,*}, \mathbf{R}^{\mathbf{a},*}, \mathbf{P}^{\mathbf{a},*})$ which provide the defender's max regret (line 4). Then the best UAV path is selected based on these outputs (line 5). Finally, the defender controls the UAV to collect data at targets on that path to obtain new intervals and then updates the UAV flying cost (line 6).

The key aspects of Algorithm 2 are in lines 4 and 5 where the $\mathrm{MMR_b}$ solution is computed by ARROW and the *current solution* heuristic is used to determine the best UAV path. In this heuristic, the *preference value* of a target t, denoted $pr(t)$, is measured as the distance in the defender utility between $\mathbf{x}^*$ and the most unfavorable strategy $\mathbf{x}'^{,*}$ against attacker payoffs $(\mathbf{R}^{\mathbf{a},*}, \mathbf{P}^{\mathbf{a},*})$ at that target, which can be computed as follows: $pr(t) = \hat{\mathrm{q}}_{\mathrm{t}}(\mathbf{x}^*, \mathbf{R}^{\mathbf{a},*}, \mathbf{P}^{\mathbf{a},*})\mathrm{U}_{\mathrm{t}}^{\mathrm{d}}(\mathbf{x}^*, \mathbf{R}^{\mathbf{d}}, \mathbf{P}^{\mathbf{d}}) - \hat{\mathrm{q}}_{\mathrm{t}}(\mathbf{x}'^{,*}, \mathbf{R}^{\mathbf{a},*}, \mathbf{P}^{\mathbf{a},*})\mathrm{U}_{\mathrm{t}}^{\mathrm{d}}(\mathbf{x}'^{,*}, \mathbf{R}^{\mathbf{d}}, \mathbf{P}^{\mathbf{d}})$ where $\mathbf{R}^{\mathbf{d}} = -\mathbf{P}^{\mathbf{a},*}$ and $\mathbf{P}^{\mathbf{d}} = -\mathbf{R}^{\mathbf{a},*}$. Intuitively, targets with higher preference values play a more important role in reducing the defender's regret. We use the sum of preference values of targets to determine the best UAV path based on the two heuristics: **Greedy heuristic:** The chosen path consists of targets which are iteratively selected with the maximum pr value and then the best neighboring target.

MCNF heuristic: We represent this problem as a Min Cost Network Flow (MCNF) where the cost of choosing a target t is $-pr(t)$. For example, there is a grid of four



(a) ARROW regret (b) Influence of parameters

Fig. 3. Solution quality of ARROW

cells (t_1, t_2, t_3, t_4) (Figure 2(a)) where each cell is associated with its preference value, namely $(pr(1), pr(2), pr(3), pr(4))$. Suppose that a UAV covers a path of two cells every time it flies and its entry locations (where the UAV takes off or land) can be at any cell. The MCNF for UAV planning is shown in Figure 2(b) which has two layers where each cell t_i has four copies $(t_i^1, t_i^2, t_i^3, t_i^4)$ with edge costs $c(t_i^1, t_i^2) = c(t_i^3, t_i^4) = -pr(i)$. The connectivity between these two layers corresponds to the grid connectivity. There are *Source* and *Sink* nodes which determine the UAV entry locations. The edge costs between the layers and between the *Source* or *Sink* to the layers are set to zero.

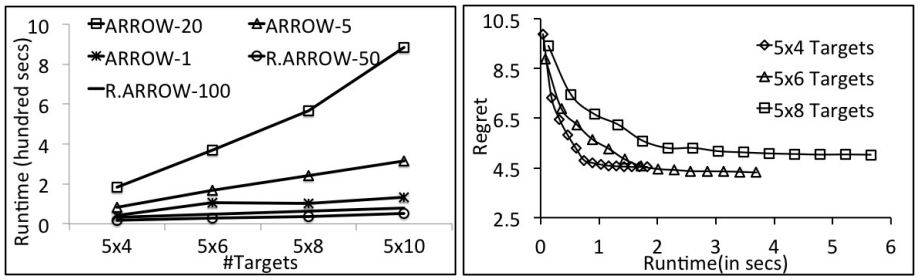
8 Experimental Results

We use CPLEX for our algorithms and Fmincon of MATLAB on a 2.3 GHz/4 GB RAM machine. *Key comparison results are statistically significant under bootstrap-t ($\alpha = 0.05$) [25]*. More results are in the Online Appendix G.

8.1 Synthetic Data

We first conduct experiments using synthetic data to simulate a wildlife protection area. The area is divided into a grid where each cell is a target, and we create different payoff structures for these cells. Each data point in our results is averaged over *40 payoff structures* randomly generated by GAMUT [19]. The attacker reward/defender penalty refers to the animal density while the attacker penalty/defender reward refers to, for example, the amount of snares that are confiscated by the defender [27]. Here, the defender's regret indicates the animal loss and thus can be used as a measure for the defender's patrolling effectiveness. Upper and lower bounds for payoff intervals are generated randomly from [-14, -1] for penalties and [1, 14] for rewards with an interval size of 4.0.

Solution Quality of ARROW. The results are shown in Figure 3 where the x-axis is the grid size (number of targets) and the y-axis is the defender's max regret. First, we demonstrate the importance of handling the attacker's bounded rationality in ARROW by comparing solution quality (in terms of the defender's regret) of ARROW with ARROW-Perfect and Maximin. Figure 3(a) shows that the defender's regret significantly increases when playing ARROW-Perfect and Maximin strategies compared to playing ARROW strategies, which demonstrates the importance of behavioral MMR.



(a) Runtime vs #Targets (b) Runtime vs Regret

Fig. 4. Runtime performance of ARROW

Second, we examine how ARROW's parameters influence the $\mathrm{MMR_b}$ solution quality; which we show later affects its runtime-solution quality tradeoff. We examine if the defender's regret significantly increases if (i) the number of starting points in M.ARROW decreases (i.e., ARROW with 20 (ARROW-20), 5 (ARROW-5) and 1 (ARROW-1) starting points for M.ARROW and 40 iterations to iteratively add 40 payoff samples into the set S), or (ii) when ARROW only uses R.ARROW (without M.ARROW) to solve relaxed $\mathrm{MMR_b}$ (i.e., R.ARROW with 50 (R.ARROW-50) and 100 (R.ARROW-100) uniformly random payoff samples). Figure 3(b) shows that the number of starting points in M.ARROW does not have a significant impact on solution quality. In particular, ARROW-1's solution quality is approximately the same as ARROW-20 after 40 iterations. This result shows that the shortcoming of local search in M.ARROW (where solution quality depends on the number of starting points) is compensated by a sufficient number (e.g., 40) of iterations in ARROW. Furthermore, as R.ARROW-50 and R.ARROW-100 only solve relaxed $\mathrm{MMR_b}$, they both lead to much higher regret. Thus, it is important to include M.ARROW in ARROW.

Runtime Performance of ARROW. Figure 4(a) shows the runtime of ARROW with different parameter settings. In all settings, ARROW's runtime linearly increases in the number of targets. Further, Figure 3(a) shows that ARROW-1 obtains approximately the same solution quality as ARROW-20 while running significantly faster (Figure 4(a)). This result shows that one starting point of M.ARROW might be adequate for solving $\mathrm{MMR_b}$ in considering the trade-off between runtime performance and solution quality. Figure 4(b) plots the trade-off between runtime and the defender's regret in 40 iterations of ARROW-20 for 20-40 targets which shows a useful anytime profile.

Runtime Performance of ARROW-Perfect. Figure 5 shows the runtime performance of ARROW-Perfect compared to ARROW and a non-linear solver (i.e., fmincon of Matlab) to compute MMR of the perfectly rational attacker case. While the runtime of both ARROW and non-linear solver increase quickly w.r.t the number of targets (e.g., it takes them approximately 20 minutes to solve a 200-

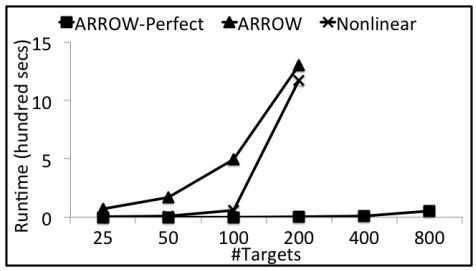
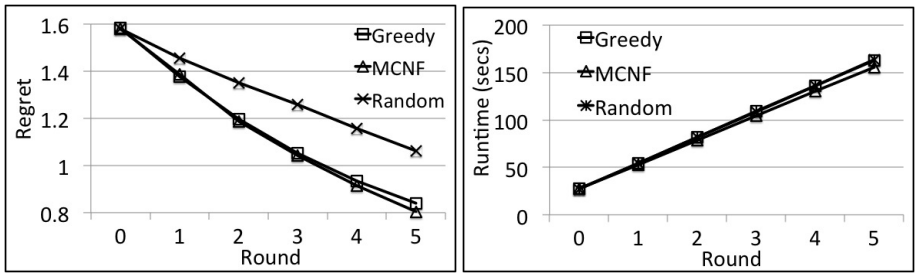
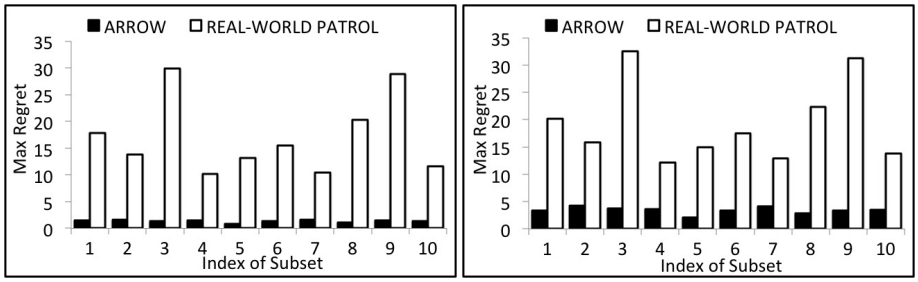


Fig. 5. Runtime Performance of ARROW-Perfect



(a) Solution quality (b) Runtime performance

Fig. 6. UAV planning: uncertainty reduction over rounds



(a) Small interval (b) Larger interval

Fig. 7. Real world max regret comparison

target game on average), ARROW-
Perfect's runtime slightly increases and reaches 53 seconds to solve a 800-target game on average. This result shows that ARROW-Perfect is scalable for large security games.

Payoff Elicitation. We evaluate our PE strategies using synthetic data of 5×5-target (target = 2×2 km cell) games. The UAV path length is 3 cells and the budget for flying a UAV is set to 5 rounds of flying. We assume the uncertainty interval is reduced by half after each round. Our purpose is to examine how the defender's regret is reduced over different rounds. The empirical results are shown in Figure 6 where the x-axis is the number of rounds and the y-axis is the regret obtained after each round (Figure 6(a)) or the accumulative runtime of the elicitation process over rounds (Figure 6(b)). We compare three heuristics: 1) randomly choosing a path (Random) 2) Greedy, and 3) MCNF. Figure 6 shows that the defender's regret is reduced significantly by using Greedy and MCNF in comparison with Random. As mentioned, the difference are statistically significant ($\alpha = 0.05$). Also, both Greedy and MCNF run as quickly as Random.

8.2 Real-world Data

Lastly, we use our wildlife dataset from Uganda (Section 3) to analyze the effectiveness of past patrols conducted by rangers (in the wildlife park) compared with the patrol strategies generated by ARROW. We choose multiple subsets of 50 grid cells each, randomly sampled from the 2423 grid cells for our analysis. Before these wildlife areas were patrolled, there was uncertainty in the features values in those areas. We simulate

these conditions faced by real world patrollers by introducing uncertainty intervals in the real-world payoffs. In our experiments, we impose uncertainty intervals on the animal density for each target, though two cases: a small and a large interval of sizes 5 and 10 respectively. Figures 7(a) and 7(b) show the comparison of the max regret achieved by ARROW and real world patrols for 10 such subsets, under the above mentioned cases of payoff uncertainty intervals. The x-axis refers to 10 different random subsets and the y-axis is the corresponding max regret. These figures clearly show that ARROW generates patrols with significantly less regret as compared to real-world patrols.

9 Summary

Whereas previous work in GSGs had assumed that there was an abundance of data in these domains, such data is not always available. To address such situations, we provide four main contributions: 1) for the first time, we compare key behavioral models, e.g., SUQR/QR on real-world data and show SUQR's usefulness in predicting adversary decisions; 2) we propose a novel algorithm, ARROW, to solve the $\mathrm{MMR_b}$ problem addressing both the attacker's bounded rationality and payoff uncertainty (when there is sufficient data to learn adversary behavioral models); 3) we present a new scalable MMR-based algorithm, ARROW-Perfect, to address payoff uncertainty against a perfectly rational attacker (when learning behavioral models is infeasible), and 4) we introduce new PE strategies for mobile sensors, e.g., UAV to reduce payoff uncertainty.

Acknowledgements: This research was supported by MURI Grant W911NF-11-1-0332 and by CREATE under grant number 2010-ST-061-RE0001. We wish to acknowledge the contribution of all the rangers and wardens in Queen Elizabeth National Park to the collection of law enforcement monitoring data in MIST and the support of Uganda Wildlife Authority, Wildlife Conservation Society and MacArthur Foundation, US State Department and USAID in supporting these data collection financially.

References

1. Basilico, N., Gatti, N., Amigoni, F.: Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In: AAMAS (2009)
2. Boutilier, C., Patrascu, R., Poupart, P., Schuurmans, D.: Constraint-based optimization and utility elicitation using the minimax decision criterion. Artificial Intelligence (2006)
3. Braziunas, D., Boutilier, C.: Assessing regret-based preference elicitation with the utpref recommendation system. In: EC (2010)
4. Brown, M., Haskell, W.B., Tambe, M.: Addressing scalability and robustness in security games with multiple boundedly rational adversaries. In: GameSec (2014)
5. Brunswik, E.: The conceptual framework of psychology, vol. 1. Univ of Chicago Pr (1952)
6. De Farias, D.P., Van Roy, B.: On constraint sampling in the linear programming approach to approximate dynamic programming. Mathematics of operations research (2004)
7. Fang, F., Stone, P., Tambe, M.: When security games go green: Designing defender strategies to prevent poaching and illegal fishing. In: IJCAI (2015)
8. French, S.: Decision theory: an introduction to the mathematics of rationality. Halsted Press (1986)

9. Haskell, W.B., Kar, D., Fang, F., Tambe, M., Cheung, S., Denicola, L.E.: Robust protection of fisheries with compass. In: IAAI (2014)
10. Kiekintveld, C., Islam, T., Kreinovich, V.: Security games with interval uncertainty. In: AAMAS (2013)
11. Kiekintveld, C., Jain, M., Tsai, J., Pita, J., Ordez, F., Tambe, M.: Computing optimal randomized resource allocations for massive security games. In: AAMAS (2009)
12. Korzhyk, D., Conitzer, V., Parr, R.: Complexity of computing optimal stackelberg strategies in security resource allocation games. In: AAAI (2010)
13. Letchford, J., Vorobeychik, Y.: Computing randomized security strategies in networked domains. In: AARM (2011)
14. McFadden, D.: Conditional logit analysis of qualitative choice behavior. Tech. rep. (1972)
15. McKelvey, R., Palfrey, T.: Quantal response equilibria for normal form games. Games and economic behavior 10(1), 6–38 (1995)
16. Montesh, M.: Rhino poaching: A new form of organised crime1. Tech. rep., University of South Africa (2013)
17. Nguyen, T.H., Yadav, A., An, B., Tambe, M., Boutilier, C.: Regret-based optimization and preference elicitation for stackelberg security games with uncertainty. In: AAAI (2014)
18. Nguyen, T.H., Yang, R., Azaria, A., Kraus, S., Tambe, M.: Analyzing the effectiveness of adversary modeling in security games. In: AAAI (2013)
19. Nudelman, E., Wortman, J., Shoham, Y., Leyton-Brown, K.: Run the gamut: A comprehensive approach to evaluating game-theoretic algorithms. In: AAMAS (2004)
20. Pita, J., Jain, M., Tambe, O.M., Kraus, S., Magori-cohen, R.: Effective solutions for real-world stackelberg games: When agents must deal with human uncertainties. In: AAMAS (2009)
21. Qian, Y., Haskell, W.B., Jiang, A.X., Tambe, M.: Online planning for optimal protector strategies in resource conservation games. In: AAMAS (2014)
22. Secretariat, G.: Global tiger recovery program implementation plan: 2013-14. Report, The World Bank, Washington, DC (2013)
23. Shieh, E., An, B., Yang, R., Tambe, M., Baldwin, C., DiRenzo, J., Maule, B., Meyer, G.: Protect: A deployed game theoretic system to protect the ports of the united states. In: AAMAS (2012)
24. Tambe, M.: Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned. Cambridge University Press (2011)
25. Wilcox, R.: Applying contemporary statistical techniques. Academic Press (2002)
26. Wright, J.R., Leyton-Brown, K.: Level-0 meta-models for predicting human behavior in games. In: ACM-EC. pp. 857–874 (2014)
27. Yang, R., Ford, B., Tambe, M., Lemieux, A.: Adaptive resource allocation for wildlife protection against illegal poachers. In: AAMAS (2014)
28. Yang, R., Ordonez, F., Tambe, M.: Computing optimal strategy against quantal response in security games. AAMAS (2012)
29. Yin, Z., Jiang, A.X., Tambe, M., Kiekintveld, C., Leyton-Brown, K., Sandholm, T., Sullivan, J.P.: Trusts: Scheduling randomized patrols for fare inspection in transit systems using game theory. AI Magazine (2012)
30. Yin, Z., Korzhyk, D., Kiekintveld, C., Conitzer, V., Tambe, M.: Stackelberg vs. nash in security games: Interchangeability, equivalence, and uniqueness. AAMAS (2010)