# Handling Payoff Uncertainty with Adversary Bounded Rationality in Green Security Domains

**A. Yadav*, T. H. Nguyen*, F. Delle Fave[1], M. Tambe, N. Agmon[2], M. Jain, W. Ramono[3], T. Batubara[3]**
University of Southern California, Los Angeles, CA, 90089, [1]Disney Research, Boston, MA, 02142
[2]Bar-Ilan University, Israel, [3]YABI, Indonesia
{amulyaya,thanhhng,tambe,manishja}@usc.edu,
[1]francesco.dellefave@disneyresearch.com,[2]agmon@cs.biu.ac.il,
[3]widodoramono@yahoo.com, timbulbatubara@gmail.com

## Abstract

Research on Stackelberg Security Games (SSG) has recently shifted to green security domains, for example, protecting wildlife from illegal poaching. Previous research on this topic has advocated the use of behavioral (bounded rationality) models of adversaries in SSG. As its first contribution, this paper, for the first time, provides validation of these behavioral models based on real-world data from a wildlife park. The paper's next contribution is the first algorithm to handle payoff uncertainty – an important concern in green security domains – in the presence of such adversarial behavioral models.

## 1 Introduction

Given the successful deployments of Stackelberg Security Games (SSG) for infrastructure protection [Shieh *et al.*, 2012; Basilico *et al.*, 2009; Letchford and Vorobeychik, 2011], research on SSG has shifted to green security domains. This research focuses on optimally allocating limited security resources in a vast geographical area against environmental crime, for example, improving the effectiveness of protection of wildlife or fisheries [Yang *et al.*, 2014; Brown *et al.*, 2014].

These green security domains exhibit at least two unique challenges. First, adversaries attack without spending as much time/effort on each attack as in terrorist attacks on infrastructure; it thus becomes more important to model the adversaries' bounded rationality in these domains. Second, there is a significant need to handle uncertainty in both players' payoffs since key domain features like animal density, that contribute to the payoffs are difficult to precisely estimate. Unfortunately, previous work has failed to address these challenges and leverage these opportunities. First, although there are a number of behavioral models proposed to handle adversaries' bounded rationality, none of these have yet been evaluated based on real-world data. Second, previous work applied several robust optimization methods to handle payoff uncertainty; but they have failed to address such uncertainty in the context of aforementioned behavioral models [Kiekintveld *et al.*, 2013].

In this paper, as our first contribution, we provide the first results on the usefulness of behavioral models using real-world data in a wildlife protection domain. Our second contribution is CONQUER (***CO**nstraint ge**N**eration for comput**i**ng **QU**antal Response based minimax r**E**g**R**et*), the first security game algorithm that can solve the *behavioral minimax regret problem*. MiniMax Regret (MMR), to minimize maximum regret from a solution [French, 1986], is a robust solution approach to handle payoff uncertainty; a key advantage of using MMR is that it is less conservative than the standard maximin approach [Nguyen *et al.*, 2014]. CONQUER is the first algorithm to compute MMR in the presence of a behavioral (bounded rationality) model, rather than assuming a perfectly rational adversary; it is also the first to handle payoff uncertainty in both the adversary and the defenders' payoffs in SSG. However, handling of adversary bounded rationality and uncertainty in both players' payoffs creates the challenge of solving a non-convex optimization problem; CONQUER provides an efficient solution to such problems. Lastly, we conduct extensive experiments to evaluate CONQUER on both artificial and real-world data.

## 2 Background & Related Work

**Stackelberg Security Games:** In SSG, the defender attempts to protect a set of $T$ targets from an attack by an adversary by optimally allocating a set of $R$ resources ($R < T$) [Korzhyk *et al.*, 2010; Tambe, 2011]. The key assumption in SSG is that the defender commits to a (*mixed*) strategy first and the adversary can observe that strategy and attack 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 $\mathbf{X} = \{\mathbf{x} : 0 \leq x_t \leq 1, \sum_t x_t \leq R\}$. If the adversary attacks $t$ when the defender is not protecting it, he receives a reward $R_t^a$, otherwise, he 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 defender ($U_t^d$) and adversary's ($U_t^a$) expected utilities at $t$ is computed as $U_t^d(\mathbf{x}, \mathbf{R^d}, \mathbf{P^d}) = x_t R_t^d + (1-x_t)P_t^d$ and $U_t^a(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = x_t P_t^a + (1-x_t)R_t^a$.

**Boundedly rational attacker:** In SSG, attacker bounded rationality is often modeled via behavior models such Quantal Response (QR) [McFadden, 1972; McKelvey and Palfrey, 1995]. The recent SUQR model (Subjective Utility Quantal Response) builds on QR by integrating the subjective utility

*A.Yadav and T.H. Nguyen are joint first-authors for this paper

function $\hat{U}^a_t(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = w_1 x_t + w_2 R^a_t + w_3 P^a_t$ into QR ($w_1$, $w_2$ and $w_3$ are parameters indicating the importance of corresponding features for the adversary), and it was shown to provide a better prediction accuracy than QR [Nguyen *et al.*, 2013]. SUQR predicts the adversary's probability of attacking $t$, $q_t(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})$, as:

$$q_t(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = \frac{e^{\hat{U}^a_t(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})}}{\sum_{t'} e^{\hat{U}^a_{t'}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})}} \quad (1)$$

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 considered as a normalization of attacking frequency.

**Payoff uncertainty:** One key approach to modeling payoff uncertainty is to express the adversary's payoffs as lying within specific intervals [Kiekintveld *et al.*, 2013]: for each $t$, $R^a_t \in [R^a_{min}(t), R^a_{max}(t)]$ 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. MMR-based solution was introduced in previous work to address payoff uncertainty in SSG while assuming a *perfectly rational* adversary [Nguyen *et al.*, 2014]. Unfortunately, they only address uncertainty in the adversary's payoff.

**Green security domains:** These domains include challenges such as protecting wildlife from poaching or protecting fisheries from illegal fishing. We focus on wildlife protection, which is an international problem — many species such as rhinos are in danger of extinction from illegal poaching [Montesh, 2013]. In previous work, game-theoretic approaches have been advocated to generate rangers' patrols [Yang *et al.*, 2014] wherein the forest area is divided into a grid where each cell represents a target. These ranger patrols are designed to counter poachers (whose behaviors are modeled using SUQR) that attempt to capture animals by setting traps such as wire snares. A similar game-theoretic system has also been developed for protecting fisheries from illegal fishing [Brown *et al.*, 2014].

Unfortunately, this previous work in green security domains has two weaknesses [Yang *et al.*, 2014]. First, models like SUQR/QR have not been compared on available real world data such as poaching signs observed by rangers [Stokes, 2010]. Second, this work addresses adversary bounded rationality but fails to simultaneously address payoff uncertainty — an important issue because of the difficulty of precisely estimating payoffs in green security domains (for example, animal densities are hard to estimate within the national park).

## 3 Behavioral Modeling Validation

Our first contribution is to use World Wildlife Fund's (WWF) real-world patrol/poaching data from a wildlife reserve in Indonesia (name of the park is withheld intentionally) to analyze the effectiveness of SUQR/QR in predicting attacks by real-world poachers. Our dataset consisted of information about patrols conducted over a period of 5 months in Indonesia. For creating this dataset, rangers at WWF divided the entire wildlife park area into 244 2x2 km grid cells (total park area $\sim$1000 $km^2$). For each patrol, they collected information about which grid cells the patrollers covered, along with various features of those cells that they observed (e.g., animal density). We also had information about how many poaching signs (e.g., snares) were observed in each cell.

**Description of raw dataset**: In this section, we describe details about the raw dataset that we got from WWF in Indonesia. As mentioned before, the park rangers discretize the entire park area into 244 $2 \times 2$ km grid cells. Further, each grid cell was subdivided into several segments. Each segment represents a significant area that the park rangers need to spend some time at, while on patrol. Whenever the rangers visit a segment for patrol, they note down observations about all kinds of activities going in that area. For example, they note down the segment ID, the date when the segment was visited, number of rangers assigned to this particular patrol, etc. More importantly, they qualitatively measured features of the segment such as amount of overhead tree cover (canopy), amount of underlying vegetation (understory), amount of rain in that segment, amount of hunting signs observed in that area, etc. They do this qualitative measurement by assigning a number ranging from 0 to 5 which indicates the abundance of that feature in that segment. So, a hunting measure of 5 would indicate severe hunting, whereas a hunting measure of 0 would indicate no observed hunting at all.

Our raw dataset consisted of many such segment records made by the patrolling rangers. Each record corresponds to an observation made by the rangers at a particular segment. In order to prepare this data for our learning algorithms (for predicting the poachers' attack locations), we first preprocessed the dataset, which we describe next.

**Dataset preprocessing.** The raw dataset consisted of 6 features with qualitative entries (range=1-5). These include animal density, area canopy, area understory, area litter, area habitat and area slope. Our first aim is to aggregate information from the segment based records to get qualitative measures for the 6 feature values for each grid cell (as opposed to raw qualitative measures for each segment). To that end, we average the 6 qualitative feature values across various segments in a grid cell. We also average the qualitative values for the hunting frequency across segments in a grid cell. Also, in order to evaluate the patrol frequency in a grid cell (approximately), we add the number of segment records in our dataset which have segment IDs lying within that grid cell. These quantities are then normalized over all grid cells to get an approximate patrol frequency for each grid cell.

After this averaging process, our processed dataset consists of 244 records (one for each grid cell). Each record contains 7 feature values: patrol frequency, animal density, area canopy, area understory, area litter, area habitat and area slope. Moreover, each record has a label that gives the average hunting frequency in that grid cell. We are merely interested in predicting whether each target gets attacked or not, as opposed to predicting number of attacks by poachers on each grid cell. Therefore, we convert our dataset's label (average number of poaching signs observed in that cell) into a binary label (whether that cell was attacked or not). In order to do that, we use a threshold $\beta$ which can take values ranging from 0 to 5. As long as the hunting frequency observed in a grid cell is greater than $\beta$, we considered that cell to be attacked (label 1), and otherwise, we consider the cell unattacked (label 0).

Each value of $\beta$ gives us a different set of labels, and therefore, we have a different dataset for each value of the threshold $\beta$. Thus, we get 4 different datasets by using $\beta = 1 - 4$. Note that $\beta = 0$ corresponds to the case when every target gets attacked and $\beta = 5$ corresponds to the case when no target is attacked. As a deterministic prediction strategy would yield 100% prediction accuracy for these $\beta$ values, we don't consider them in our experiments.

**Gaussian Smoothing**: Most of the features that were collected by the patrollers are unlikely to change abruptly in between grid cells. This is because most features (such as animal density, area slope etc.) are dependent on geographical features and terrain which changes gradually. In order to ensure that our dataset reflects this gradual gradation of feature values across grid cells, we use a gaussian kernel of size $3 \times 3$ ($\sigma = 4$) to smooth out all feature values (except patrol and attack frequencies).

**Attack Frequency Correction**: Some grid cells in our dataset are patrolled with greater frequency and some grid cells with lesser frequency. A key insight is that the more time the rangers spend in patrolling a grid cell, the more attack signs they will observe in that grid cell. This leads us to conclude that the our attack labels are skewed for grid cells with low patrol frequencies. Essentially, there would be less number of attacks on targets with less patrol frequency because we did not patrol it well enough to observe more attacks, not because the poachers did not attack there [Lemieux, 2014]. Therefore, in order to resolve this issue, we correct the attack frequencies in two different ways. **Linear Normalization** corresponds to a linearly proportional increase in the number of attacked cells. We also try **Log normalization** which corresponds to a log proportional increase in the number of attacked cells.

**Behavioral Models**: To use models like SUQR/QR to model the poacher's behavior, we make the general Stackelberg assumption that the poachers were able to observe our entire dataset before planning their attacks. Thus, for each grid cell, they observed all the 7 features in our dataset. Armed with this knowledge, the adversary chooses which targets to attack using their behavioral models. We also assume that the poachers pay a constant penalty on all grid cells, if the rangers catch them (this assumption is reasonable since any ranger goes to jail when they get caught). We tested 5 different behavioral models in our experiments: **QR** (expected utility is calculated using the constant penalty term and animal density as reward), **SUQR-3** (subjective utility is calculated by a weighted combination of patrol frequency, animal density and the constant penalty term), **SUQR-7** (subjective utility is calculated by weighted combination of all 7 features present in our dataset), Cognitive Hierarchy Level-0 or **COG-0** (subjective utility is calculated by a weighted combination of all features except patrol frequency, thus implying that the poacher does not reason about the rangers' patrol patterns), and **Perfect Rational Model** (poacher only attacks the grid cell with highest utility).

**Learning Method**: We use Maximum Likelihood Estimation (MLE) to learn the parameters of the 3 different models that we test in our experiments. Thus, for **QR** model, we learn the $\lambda$ parameter, while for **SUQR-3** and **SUQR-7**, we learn the $\omega_i$ parameters ($i = 1..3$ for **SUQR-3** and $i = 1..7$ for **SUQR-7**). For exposition, we explain the process of learning **SUQR-3** parameters here (a similar process applies for the other models). Given N datapoints which form our dataset $\mathbf{D}$, the log-likelihood of $(\omega_1, \omega_2, \omega_3)$ is given by:

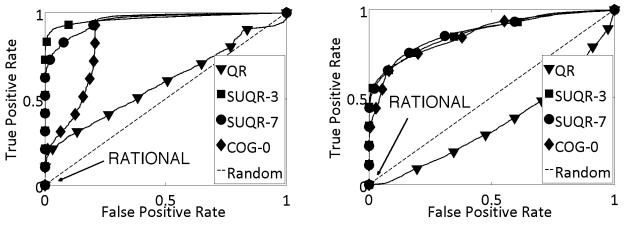
$$logL(\omega_1, \omega_2, \omega_3 | \mathbf{D}) = \sum_{t=1}^{T} N_t log[q_t(\omega_1, \omega_2, \omega_3 | \mathbf{D})] \quad (2)$$

Here, $T$ is the total number of targets that we have, $N_t$ is the number of times target $t$ has been attacked and $q_t(\omega_1, \omega_2, \omega_3 | \mathbf{D})$ is the probability with which adversary having weights $(\omega_1, \omega_2, \omega_3)$ attacks target $t$. Ultimately, our goal is to find values of $(\omega_1, \omega_2, \omega_3)$ which maximizes $logL(\omega_1, \omega_2, \omega_3 | \mathbf{D})$. This log-likelihood function can be shown to be concave: we can show that the Hessian matrix of $logL(\omega_1, \omega_2, \omega_3 | \mathbf{D})$ is negative semi-definite. Thus, this function has a unique local maximum point and therefore, we can use a convex optimization solver (e.g., fmincon in MATLAB) to compute the optimal weights $(\omega_1, \omega_2, \omega_3)$.

For each created dataset (remember that we had 4 different datasets with different $\beta$ values), we do random subsampling validation to create 1000 random 90:10 training/test splits. For each split, we train our behavioral models using MLE to learn their parameters. These learnt parameters are then used to get probabilities of attack on each grid cell in the test set. Thus, for each grid cell in the test set (generated via 90:10 random split), we get the actual label (whether the target was attacked or not) along with our predicted probability of attack on the cell. Thus, for each of the 1000 randomly generated 90:10 splits, we end up with a set of actual test labels and a corresponding *predicted* probability of attack. By combining these 1000 different sets of labels and predicted probabilities, we plot Receiver Operating Characteristic (ROC) curves to analyze the performance of the various models.

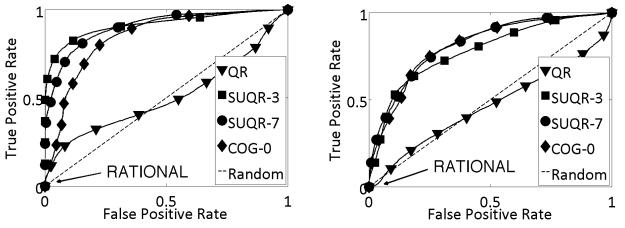
**Learning Results**: Figures 1a, 2a, 3a, 4a show ROC curves with linear attack frequency normalization for $\beta$ ranging from 1 to 4. Also, Figures 1b, 2b, 3b, 4b show ROC curves with logarithmic attack frequency normalization for $\beta$ ranging from 1 to 4. These figures show that Rational Model is very bad at predicting responses of real world poachers. Clearly, this shows that poachers in the real world are not perfectly rational. With linear normalization, SUQR-3 and SUQR-7 perform extremely well for $\beta = 1$ and $\beta = 2$ and COG-0 performs worse comparatively. For higher values of $\beta$, the prediction accuracy goes down in general but is still much better than a random classifier. We observe similar effects with logarithmic normalization which shows the robustness of our learning method. These figures also show that QR model does only slightly better than a random classifier on most occassions and does even worse rarely (Figure 1b).

These figures show that SUQR-7 and SUQR-3 do extremely well across different values of $\beta$ and both kinds of normalization. Since SUQR-7 contains features specific to the WWF dataset, we use SUQR-3 as our model of choice in the rest of the paper (as it is generalizable to standard security games). This experiment is the first validation of any behavioral models on real-world data in the context of SSG.



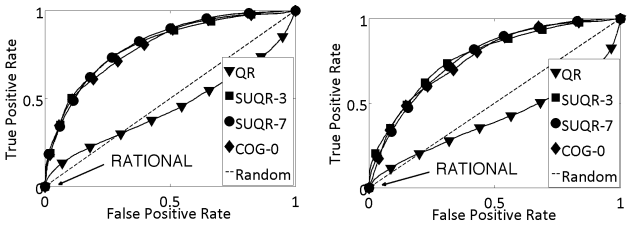

(a) Linear Normalization (b) Log Normalization

Figure 1: $\beta = 1$



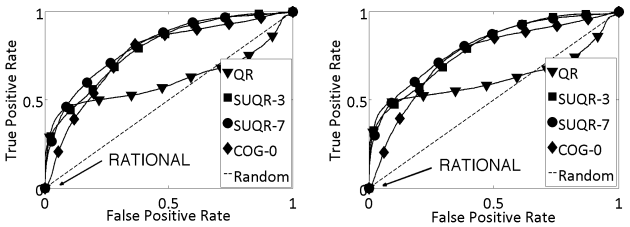

(a) Linear Normalization (b) Log Normalization

Figure 2: $\beta = 2$




(a) Linear Normalization (b) Log Normalization

Figure 3: $\beta = 3$




(a) Linear Normalization (b) Log Normalization

Figure 4: $\beta = 4$

# 4 Behavioral Minimax Regret (MMR)

While we can learn a behavioral model from real-world data, we still face the challenge of significant payoff uncertainty. Hence, we present our new algorithm, CONQUER. Here, we primarily focus on zero-sum games as motivated by recent work in green security domains [Haskell *et al.*, 2014; Brown *et al.*, 2014], and earlier major SSG applications that use zero-sum games [Shieh *et al.*, 2012]). In addition, we use SUQR as the adversary's behavioral model, given its high prediction accuracy as shown in Section 3. Yet, our methods can be generalized to non-zero-sum games with a general class of QR as described in Online Appendix C.[1]

**Algorithm 1:** CONQUER Outline

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

## 4.1 Problem Definition

We now formulate MMR 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:*

$$\mathrm{R_b}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = \max_{\mathbf{x'} \in \mathbf{X}} F(\mathbf{x'}, \mathbf{R^a}, \mathbf{P^a}) - F(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) \quad (3)$$

$$\text{where } F(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) = \sum\nolimits_t q_t(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) U_t^d(\mathbf{x}, \mathbf{R^d}, \mathbf{P^d}) \quad (4)$$

Here, $F(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})$ is the defender's utility (which is non-convex fractional in $\mathbf{x}$) for playing $\mathbf{x}$ where the payoff of the adversary, whose response follows SUQR, is $(\mathbf{R^a}, \mathbf{P^a})$ and $\mathbf{R^d} = -\mathbf{P^a}$ and $\mathbf{P^d} = -\mathbf{R^a}$. In addition, the attacking probability, $q_t(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a})$, is given by Equation 1.

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

$$\mathrm{MR_b}(\mathbf{x}, \mathbf{I}) = \max_{(\mathbf{R^a}, \mathbf{P^a}) \in \mathbf{I}} \mathrm{R_b}(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}) \quad (5)$$

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

$$\mathrm{MMR_b}(\mathbf{I}) = \min_{\mathbf{x} \in \mathbf{X}} \mathrm{MR_b}(\mathbf{x}, \mathbf{I}) \quad (6)$$

As our experiments show later, if the defender uses MMR for a perfectly rational attacker instead of $\mathrm{MMR_b}$, she may suffer a significant utility loss.

## 4.2 CONQUER: Algorithm Description

We now present CONQUER (Algorithm 1) to solve $\mathrm{MMR_b}$ in (6). CONQUER's two novelties compared to previous work [Nguyen *et al.*, 2014] — addressing uncertainty in both players' payoffs and a boundedly rational attacker — lead to two new computational challenges: 1) uncertainy in defender payoffs makes the defender's expected utility at every target $t$ non-convex in $\mathbf{x}$ and $(\mathbf{R^d}, \mathbf{P^d})$ ; and 2) SUQR represents the attacker's bounded rationality 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 4) — which is then used in computing $\mathrm{MMR_b}$ (Equation 6),

[1] https://www.dropbox.com/s/vrii1mt32is34d1/Appendix.pdf?dl=0

making it computationally expensive. Overall, $\text{MMR}_\text{b}$ can be reformulated as follows:

$$\min_{\mathbf{x}\in\mathbf{X},r\in\mathbb{R}} r \tag{7}$$
$$\text{s.t. } r \geq F(\mathbf{x}', \mathbf{R^a}, \mathbf{P^a}) - F(\mathbf{x}, \mathbf{R^a}, \mathbf{P^a}), \forall (\mathbf{R^a}, \mathbf{P^a}) \in \mathbf{I}, \mathbf{x}' \in \mathbf{X}$$

Unfortunately, since $\mathbf{X}$ and $\mathbf{I}$ are continuous, the set of constraints is infinite. One practical approach to optimization with large constraint sets is *constraint sampling* [De Farias and Van Roy, 2004], coupled with *constraint generation* [Boutilier *et al.*, 2006]. Following this approach, CONQUER samples a subset of constraints in Problem (7) and gradually expands this set by adding violated constraints to the relaxed problem until convergence to the optimal $\text{MMR}_\text{b}$ solution. Specifically, CONQUER 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 algorithm PASAQ [Yang *et al.*, 2012] to obtain a finite set $S$ of sampled constraints (Line 2). These sampled constraints are then used to solve the corresponding *relaxed* $\text{MMR}_\text{b}$ program (line 4) using the PALMS algorithm (described in Section 4.3) — we call this problem as *relaxed* $\text{MMR}_\text{b}$ as it only has samples of constraints in (7). We thus obtain the optimal solution $(lb, \mathbf{x}^*)$ which provides a lower bound ($lb$) on the true $\text{MMR}_\text{b}$. Then constraint generation is applied to determine violated constraints (if any). This uses the REALMS algorithm (described in Section 4.4) which computes $\text{MR}_\text{b}(\mathbf{x}^*, \mathbf{I})$ — the optimal regret of $\mathbf{x}^*$ which is an upper bound ($ub$) on the true $\text{MMR}_\text{b}$. If $ub > lb$, the optimal solution of REALMS, $\{\mathbf{x}'^{,*}, \mathbf{R^{a,*}}, \mathbf{P^{a,*}}\}$, provides the maximally violated constraint (line 5), which is added to $S$. Otherwise, $\mathbf{x}^*$ is the minimax optimal strategy and $lb = ub = \text{MMR}_\text{b}(\mathbf{I})$.

### 4.3 PALMS: Compute Relaxed Behavioral MMR

The first step of CONQUER is to solve the relaxed $\text{MMR}_\text{b}$ problem using PALMS (***P**iece-wise linear & bin**A**ry search for re**L**axed **M**inimax against **S**UQR adversary*). As we show later, this relaxed $\text{MMR}_\text{b}$ problem is non-convex and fractional. Thus, PALMS presents two key ideas to efficiently solve it: 1) binary search (which iteratively searches the defender's utility space to find the optimal solution) to remove the fractional terms in the relaxed $\text{MMR}_\text{b}$; and 2) it then applies piecewise-linear approximation to linearize the remaining non-convex terms. Overall, the relaxed $\text{MMR}_\text{b}$ problem can be represented as follows:

$$\min_{\mathbf{x}\in\mathbf{X},r\in\mathbb{R}} r \tag{8}$$
$$\text{s.t. } r \geq F(\mathbf{x}'^{,\mathbf{k}}, \mathbf{R^{a,k}}, \mathbf{P^{a,k}}) - F(\mathbf{x}, \mathbf{R^{a,k}}, \mathbf{P^{a,k}}), \forall k = \overline{1, K}$$

where $(\mathbf{x}'^{,\mathbf{k}}, \mathbf{R^{a,k}}, \mathbf{P^{a,k}})$ is the $k^{th}$ sample in $S$ where $k = \overline{1, K}$ and $K$ is the total number of samples in $S$ and $r$ is the defender's max regret for playing $\mathbf{x}$ against sample set $S$. Finally, $F(\mathbf{x}'^{,\mathbf{k}}, \mathbf{R^{a,k}}, \mathbf{P^{a,k}})$ is the defender's optimal utility for every sample of attacker payoffs $(\mathbf{R^{a,k}}, \mathbf{P^{a,k}})$ where $\mathbf{x}'^{,\mathbf{k}}$ is the corresponding defender's optimal strategy (which may be obtained via PASAQ [Yang *et al.*, 2012]. The term $F(\mathbf{x}, \mathbf{R^{a,k}}, \mathbf{P^{a,k}})$ which is included in relaxed $\text{MMR}_\text{b}$'s constraints is non-convex and fractional in $\mathbf{x}$ (Equation 4), making (8) non-convex and fractional. We can use any non-convex solver with multiple starting points, e.g., Fmincon of MATLAB to solve it; however, these solvers are time consuming. We now detail the two key ideas of PALMS.

**Binary search.** In each binary search step, given a value of $r$, PALMS tries to solve the following decision problem (**P1**):

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

Based on Theorem 1, we can convert (**P1**) into the following non-convex and *non-fractional* optimization problem (**P2**) of which optimal solution determines the feasibility of (**P1**):

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

$$\text{s.t. } v \geq \sum_t \left[r_k - U_t^{d,k}(\mathbf{x})\right] e^{w_1 x_t + w_2 R_t^{a,k} + w_3 P_t^{a,k}}, \forall k = \overline{1, K}$$

where $U_t^{d,k}(\mathbf{x}) = -\left[x_t P_t^{a,k} + (1 - x_t) R_t^{a,k}\right]$ is the defender's utility and $r_k = F(\mathbf{x}'^{,\mathbf{k}}, \mathbf{R^{a,k}}, \mathbf{P^{a,k}}) - r$ given the $k^{th}$ sample.

**Theorem 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.*[2]

**Piecewise linear approximation.** We now describe our algorithm for solving (**P2**). Overall, (**P2**) is a non-convex optimization problem as 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 $x_t$. 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 $i = \overline{1, M}$ as follows:

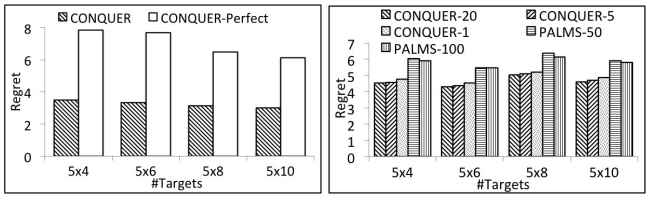
$$f_t^k(x_t) \approx f_t^k(0) + \sum_{i=1}^{M} \alpha_{t,i}^k x_{t,i} \tag{9}$$

where $\alpha_{t,i}^k$ is the slope of the $i^{th}$ 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., $x_t = \sum_i x_{t,i}$. For example, suppose that $M = 5$ and $x_t = 0.3$, as $\frac{1}{5} < x_t < \frac{2}{5}$, we obtain $x_{t,1} = \frac{1}{5}$, $x_{t,2} = 0.1$, and $x_{t,3} = x_{t,4} = x_{t,5} = 0$. 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 CPLEX:

$$\textbf{(P2')}: \min_{x_{t,i}, z_{t,i}, v} v \tag{10}$$
$$\text{s.t. } v \geq \sum_t f_t^k(0) + \sum_t \sum_i \alpha_{t,i}^k x_{t,i}, \forall k = \overline{1, K} \tag{11}$$
$$\sum_{t,i} x_{t,i} \leq R \tag{12}$$
$$z_{t,i} \frac{1}{M} \leq x_{t,i}, \forall t, i = \overline{1, M-1} \tag{13}$$
$$x_{t,i+1} \leq z_{t,i}, \forall t, i = \overline{1, M-1} \tag{14}$$
$$z_{t,i} \in \{0, 1\}, 0 \leq x_{t,i} \leq \frac{1}{M}, \forall t, i = \overline{1, M-1} \tag{15}$$

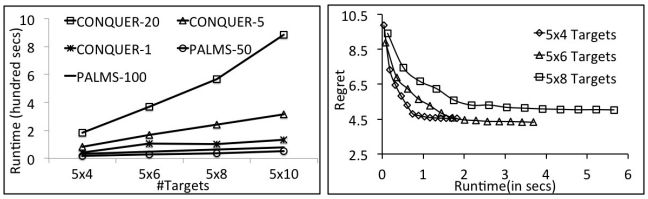
where $z_{t,i}$ is an auxiliary integer variable which ensures that the portions of $x_t$ satisfies $x_{t,i} = \frac{1}{M}$ if $x_t \geq \frac{i}{M}$ ($z_{t,i} = 1$) or $x_{t,i+1} = 0$ if $x_t < \frac{i}{M}$ ($z_{t,i} = 0$) (constraints (13 – 15)). Constraints (11) are equivalent to constraints of **(P2)** after the approximation. In addition, constraint (12) guarantees that the resource allocation condition, $\sum_t x_t \leq R$, holds true.

[2] All proofs appear in the Online Appendix.




(a) CONQUER regret (b) Influence of parameters

Figure 5: Solution quality of CONQUER




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

Figure 6: Runtime performance of CONQUER

### 4.4 REALMS: Compute Behavioral Max Regret

Given the optimal solution $\mathbf{x}^*$ returned by PALMS, the second step of CONQUER computes the $\mathrm{MR_b}$ of $\mathbf{x}^*$ using REALMS (*computing max **RE**gret using loc**AL** search with **M**ultiple re**S**tarts*) (line 5 in Algorithm 1). Overall, computing $\mathrm{MR_b}$ can be represented as follows:

$$\max_{\mathbf{x}' \in \mathbf{X}, (\mathbf{R^a}, \mathbf{P^a}) \in \mathbf{I}} F(\mathbf{x}', \mathbf{R^a}, \mathbf{P^a}) - F(\mathbf{x}^*, \mathbf{R^a}, \mathbf{P^a}) \qquad (16)$$

This optimization problem is also non-convex. It is difficult to apply binary search and piecewise linear approximation (like PALMS) in REALMS since it is a subtraction of two non-convex fractional functions, $F(\mathbf{x}', \mathbf{R^a}, \mathbf{P^a})$ and $F(\mathbf{x}^*, \mathbf{R^a}, \mathbf{P^a})$. Thus, we use local search with multiple starting points to solve $\mathrm{MR_b}$.

## 5 Experimental Results

We evaluate solution quality and runtime of CONQUER and PE heuristics in zero-sum games, assuming an SUQR attacker. This section presents key experimental results (more results are in Online Appendix D). 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 ($\alpha = 0.05$).*

### 5.1 Synthetic Data

We first conduct experiments using synthetic data to simulate a wildlife protection area. We assume area is divided into a grid where each cell represents a target, and we create different payoff structures using these grid cells. Each data point in our results is averaged over *40 payoff structures* randomly generated by GAMUT [Nudelman *et al.*, 2004]. 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 estimated to be confiscated by the defender [Yang *et al.*, 2014]. Here, the defender's regret indicates the animal loss, 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 the interval size is 4.0.

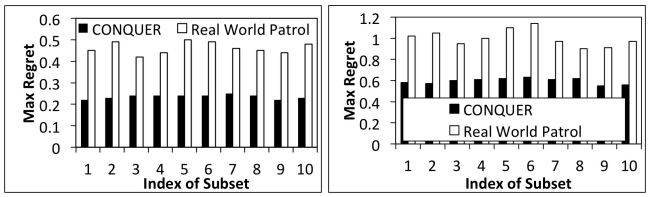
**Solution Quality of CONQUER.** The results are shown in Figure 5 where the x-axis is the grid size (number of targets) and the y-axis is the defender's max regret. First, we demonstrate the importance of handling the attacker's bounded rationality in CONQUER by comparing solution quality of CONQUER with CONQUER-Perfect (an extension of the MMR algorithm for a perfectly rational attacker [Nguyen *et al.*, 2014] that addresses uncertainty in *both* players' payoffs (described in Online Appendix B)). The defender's regret obtained by playing CONQUER and CONQUER-Perfect against the SUQR attacker is shown in Figure 5(a). The figure shows that the defender's regret significantly increases when playing CONQUER-Perfect's strategies, which shows the importance of addressing the attacker's bounded rationality.

Second, we examine how CONQUER's parameters influence the MMR solution quality; a factor we show later affects its key runtime-solution quality tradeoff. We examine whether the defender's regret significantly increases if (i) the number of starting points in REALMS decreases (i.e., CONQUER with 20 (CONQUER-20), 5 (CONQUER-5) and 1 (CONQUER-1) starting points for REALMS and 40 iterations to iteratively add 40 payoff samples into the set $S$), or (ii) when CONQUER only uses PALMS (without REALMS) to solve relaxed $\mathrm{MMR_b}$ (i.e., PALMS with 50 (PALMS-50) and 100(PALMS-100) uniformly random payoff samples). Figure 5(b) shows that the number of starting points in REALMS does not have an impact on solution quality. In particular, CONQUER-1's solution quality is approximately the same as CONQUER-20 after 40 iterations. This result shows that the shortcoming of local search in REALMS (where solution quality depends on the number of starting points) is compensated by a sufficient number (e.g., 40) of iterations in CONQUER; hence number of REALMS starting points have low impact. Further, as PALMS-50 and PALMS-100 only solve relaxed $\mathrm{MMR_b}$, they both lead to much higher regret. Thus, the presence of REALMS has a great impact in improving CONQUER's performance in reducing $\mathrm{MR_b}$.

**Runtime Performance of CONQUER.** Figure 6(a) shows the runtime of CONQUER with different parameter settings. In all settings, CONQUER's runtime linearly increases in the number of targets. Further, Figure 5(a) shows that CONQUER-1 obtains approximately the same solution quality as CONQUER-20 while running significantly faster (Figure 6(a)). This result shows that one starting point of REALMS might be adequate for solving $\mathrm{MMR_b}$ in considering the trade-off between runtime performance and solution quality. Figure 6(b) plots the trade-off between runtime and the defender's regret in 40 iterations of CONQUER-20 for 20-40 targets which shows that the defender's regret reduces quickly as the runtime of CONQUER increases.

### 5.2 Real-world Data

Lastly, we use our WWF dataset (Section 3) to analyze the difference between patrols conducted by rangers (in the Indonesian wildlife park) and the patrol strategies generated by CONQUER. Out of 244 grid cells (targets), we pick 25 cells (chosen randomly). Before these wildlife areas were patrolled, there was uncertainty in the features values at those




(a) Small interval (b) Larger interval

Figure 7: Real world max regret comparison

areas. We simulate these conditions faced by real world patrollers by introducing uncertainty intervals in the real-world rewards and penalties on each target in two cases: a small and a larger interval of sizes 0.5 and 1 respectively. For both cases, we compared the max regret achieved by the real world patrols with the max regret of CONQUER's patrols. Figures 7(a) and 7(b) compares the max regret achieved by CONQUER and real world patrols for 10 different randomly generated subsets of 25 targets when the uncertainty interval size is 0.5 and 1 respectively. The x-axis refers to 10 different subsets and the y-axis is the corresponding max regret. These figures clearly show that CONQUER generates patrols having significantly less regret as compared to real-world patrols.

## 6 Conclusion

In summary, this paper focuses on solving these security problems while providing the following main contributions: 1) we for the first time test key behavioral models such as SUQR on real-world wildlife protection data and show their usefulness in predicting adversary decisions; 2) we propose a novel algorithm, CONQUER, to solve the behavioral MMR problem which addresses both the attacker's bounded rationality and uncertainty in both players' payoffs; and 3) we provide the evaluation based on a real-world wildlife domain.

## References

[Basilico *et al.*, 2009] Nicola Basilico, Nicola Gatti, and Francesco Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *AAMAS*, 2009.

[Boutilier *et al.*, 2006] Craig Boutilier, Relu Patrascu, Pascal Poupart, and Dale Schuurmans. Constraint-based optimization and utility elicitation using the minimax decision criterion. *Artificial Intelligence*, 2006.

[Brown *et al.*, 2014] Matthew Brown, William B. Haskell, and Milind Tambe. Addressing scalability and robustness in security games with multiple boundedly rational adversaries. In *GameSec*, 2014.

[De Farias and Van Roy, 2004] Daniela Pucci De Farias and Benjamin Van Roy. On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of operations research*, 2004.

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

[Haskell *et al.*, 2014] William B Haskell, Debarun Kar, Fei Fang, Milind Tambe, Sam Cheung, and Lt Elizabeth Denicola. Robust protection of fisheries with compass. In *IAAI*, 2014.

[Kiekintveld *et al.*, 2013] Christopher Kiekintveld, Towhidul Islam, and Vladik Kreinovich. Security games with interval uncertainty. In *AAMAS*, 2013.

[Korzhyk *et al.*, 2010] Dmytro Korzhyk, Vincent Conitzer, and Ronald Parr. Complexity of computing optimal stackelberg strategies in security resource allocation games. In *AAAI*, 2010.

[Lemieux, 2014] A M Lemieux. Situational Prevention of Poaching. *Routledge Press*, pages 108–109, 2014.

[Letchford and Vorobeychik, 2011] Joshua Letchford and Yevgeniy Vorobeychik. Computing randomized security strategies in networked domains. In *AARM*, 2011.

[McFadden, 1972] Daniel McFadden. Conditional logit analysis of qualitative choice behavior. Technical report, 1972.

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

[Montesh, 2013] Moses Montesh. Rhino poaching: A new form of organised crime1. Technical report, University of South Africa, 2013.

[Nguyen *et al.*, 2013] Thanh Hong Nguyen, Rong Yang, Amos Azaria, Sarit Kraus, and Milind Tambe. Analyzing the effectiveness of adversary modeling in security games. In *AAAI*, 2013.

[Nguyen *et al.*, 2014] Thanh H Nguyen, Amulya Yadav, Bo An, Milind Tambe, and Craig Boutilier. Regret-based optimization and preference elicitation for stackelberg security games with uncertainty. In *AAAI*, 2014.

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

[Shieh *et al.*, 2012] E. Shieh, B. An, R. Yang, M. Tambe, C. Baldwin, J. DiRenzo, B. Maule, and G. Meyer. Protect: A deployed game theoretic system to protect the ports of the united states. In *AAMAS*, 2012.

[Stokes, 2010] Emma J Stokes. Improving effectiveness of protection efforts in tiger source sites: developing a framework for law enforcement monitoring using mist. *Integrative Zoology*, 2010.

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

[Yang *et al.*, 2012] Rong Yang, Fernando Ordonez, and Milind Tambe. Computing optimal strategy against quantal response in security games. AAMAS, 2012.

[Yang *et al.*, 2014] Rong Yang, Benjamin Ford, Milind Tambe, and Andrew Lemieux. Adaptive resource allocation for wildlife protection against illegal poachers. In *AAMAS*, 2014.