# Partial Adversarial Behavior Deception in Security Games

**Thanh H. Nguyen**[1] , **Arunesh Sinha**[2] , **He He**[1]
[1]University of Oregon
[2]Singapore Management University
{tnguye11, hhe7}@uoregon.edu, aruneshs@smu.edu.sg

## Abstract

Learning attacker behavior is an important research topic in security games as security agencies are often uncertain about attackers' decision making. Previous work has focused on developing various behavioral models of attackers based on historical attack data. However, a clever attacker can manipulate its attacks to *fail* such attack-driven learning, leading to ineffective defense strategies. We study attacker behavior deception with three main contributions. First, we propose a new model, named *partial behavior deception model*, in which there is a deceptive attacker (among multiple attackers) who controls a portion of attacks. Our model captures real-world security scenarios such as wildlife protection in which multiple poachers are present. Second, we introduce a new scalable algorithm, GAMBO, to compute an optimal deception strategy of the deceptive attacker. Our algorithm employs the projected gradient descent and uses the implicit function theorem for the computation of gradient. Third, we conduct a comprehensive set of experiments, showing a significant benefit for the attacker and loss for the defender due to attacker deception.

## 1 Introduction

In many real-world security domains, security agencies are generally uncertain about human attackers' decision making. Therefore, recent research in security games has focused on developing different behavior models of attackers, leveraging historical attack data [Yang *et al.*, 2011; Nguyen *et al.*, 2013; Kar *et al.*, 2015; Sinha *et al.*, 2016]. For example, in wildlife protection, rangers collect poaching signs during their patrols and use such data to learn behavior of poachers. The trained model is then used to generate an optimal defense strategy.

However, since the defender relies on historical attack data, the generated defense strategy is vulnerable to manipulative attacks. The attacker manipulation can influence the learning outcome, leading to ineffective defense strategies. We study the attacker behavior deception in Stackelberg security games (SSGs), a well-known class of games which has been widely applied in security domains [Tambe, 2011]. We aim at characterizing the attacker's deception strategy and analyzing the consequent benefit of the attacker and loss of the defender.

Specifically, we consider the scenario in which there are multiple attackers with different behavior. Among these attackers, there is a rational attacker who intends to play deceptively to mislead the defender. The other attackers always respond honestly. The defender models the whole population of attackers as one adversary and attempts to learn an adversary behavior model based on historical attack data. Previous work in wildlife protection has followed this single-behavior-model approach to predict the behavior of poachers [Fang *et al.*, 2016; Kar *et al.*, 2017]. Our hypothesis is that even if only one of the attackers is deceptive, then this attacker can cause a great harm to the defender by employing deception.

To test our hypothesis, we propose a new model, *partial behavior deception model*, in which all attackers have the same payoff but may behave differently. The defender knows the attacker payoffs but is uncertain about their behavior. In the learning phase, the defender collects attack data and trains an attacker behavior model accordingly. In the planning phase, the defender determines an optimal defense strategy to play based on the learning outcome. While the non-deceptive attackers play honestly in both phases, the rational deceptive attacker intentionally modifies its attack strategy in the learning phase to maximize its gain in the planning phase. The deceptive attacker has no control over attacks by other attackers.

Our second contribution is a new scalable algorithm to solve the problem of finding an optimal strategy of the deceptive attacker in the learning phase, called GAMBO (**G**radient-based method for **A**ttacker **M**anipulation of **B**ehavi**O**r). This computational problem is a tri-level optimization (that is, three levels of nested optimization) as well as non-convex. We employ a projected gradient-based method to solve this optimization. To enable this gradient-based method, we provide a novel *dependent* differentiation technique by exploiting the uniqueness of the optimal defense strategy and making use of the implicit function theorem.

Finally, we conduct extensive experiments to evaluate the impact of the attacker behavior deception. We show that even though the deceptive attacker can only influence a portion of the attacks in the training phase, the attacker's manipulation substantially changes the learning outcome of the defender. Eventually, it leads to a significant benefit for the deceptive attacker and loss for the defender in the planning phase.

## 2 Related Work

**Attacker behavior learning in security games.** Learning bounded rational attacker behavior is an important line of research in security games in which various behavior models of the attacker such as Quantal Response, etc. [Yang *et al.*, 2011; Nguyen *et al.*, 2013; Sinha *et al.*, 2016; Kar *et al.*, 2017] have been explored. The behavior learning outcomes are then used to generate a defender strategy, which is vulnerable to attacker manipulation. Our work is orthogonal to work on learning about a rational attacker [Blum *et al.*, 2014] or a rational attacker with multiple resources [Korzhyk *et al.*, 2011].

**Deception in security games.** There are several works studying deception in security [Carroll and Grosu, 2011; Horák *et al.*, 2017]. A majority of previous work in SSGs, in particular, investigates deception on the defender side [Guo *et al.*, 2017; Rabinovich *et al.*, 2015; Xu *et al.*, 2015; Zhuang *et al.*, 2010]. That is, the defender exploits the information asymmetry to *fool* the attacker. Recently, some research has started to study attacker deception in SSGs [Gan *et al.*, 2019a; Nguyen and Xu, 2019; Nguyen *et al.*, 2019] or on the follower side in general Stackelberg games [Gan *et al.*, 2019b]. These existing work mainly focuses on the scenario the defender is uncertain about the attacker's payoff while the attacker is known to be perfectly rational. In contrast, in our setup, there are multiple attackers whose attack behavior is unknown while the attacker's payoff is known. Also, only the single rational attacker is deceptive.

**Adversarial machine learning (ML).** Adversarial ML has received lot of attention in the ML community [Biggio and Roli, 2018; Huang *et al.*, 2011; Lowd and Meek, 2005; Madry *et al.*, 2017]. The attacker behavior deception in our study can be considered as a *causative* attack in adversarial ML. In adversarial ML, prediction accuracy is the main measure to evaluate attacks against ML algorithms, whereas, in our setting the goal of both players is to maximize their utility, taking into account the learning outcome of the defender.

**Decision-focused learning.** Our work is also related to work on decision-focused learning, where ML models are trained jointly with optimization algorithms to obtain high-quality decisions [Donti *et al.*, 2017; Wilder *et al.*, 2019]. Indeed, our use of implicit function theorem is inspired by the application of this theorem in decision-focused learning, but our novelty is in performing dependent differentiation of a binary search approach to solve an optimization.

## 3 Background

**Stackelberg security games (SSGs).** In SSGs [Tambe, 2011], a defender attempts to protect a set of $T$ targets $\{1, 2, \ldots, T\}$ from an attacker. The defender has a limited number of security resources ($K < T$) to allocate over these targets. A pure strategy of the defender is an allocation of these security resources to $K$ targets. A mixed strategy of the defender is a probability distribution over all of his pure strategies. In this work, we consider the no-scheduling-constraint scenario in which each mixed strategy of the defender can be equivalently represented as a marginal coverage probability vector, denoted by $\mathbf{x} = \{x_1, x_2, \ldots, x_T\}$ where $x_i \in [0, 1]$ is the coverage probability the defender protects target $i$ and $\sum_i x_i \leq K$. In the Stackelberg setting, the attacker is aware of the defender's mixed strategy and chooses one of the targets to attack accordingly.

When the attacker attacks a target $i$, if the defender is not protecting $i$, then the attacker obtains a reward of $R_i^a$ while the defender receives a penalty of $P_i^d$. Conversely, if the defender is protecting $i$, then the attacker has a penalty of $P_i^a < R_i^a$ and the defender gets a reward of $R_i^d > P_i^d$. Given $\mathbf{x}$, the expected utility of the defender and attacker for an attack on target $i$ is formulated as follows:

$$U_i^d(x_i) = x_i R_i^d + (1 - x_i) P_i^d$$
$$U_i^a(x_i) = x_i P_i^a + (1 - x_i) R_i^a$$

**Quantal Response model (QR).** QR is a well known model used to model the attacker's behavior in SSGs [McFadden and others, 1973; McKelvey and Palfrey, 1995; Yang *et al.*, 2011]. Essentially, QR predicts the probability the attacker will attack each target $i$, which is formulated as follows:

$$q_i(\mathbf{x}; \lambda) = \left(e^{\lambda U_i^a(x_i)}\right) \Big/ \left(\sum\nolimits_j e^{\lambda U_j^a(x_j)}\right)$$

Intuitively, the higher expected utility of a target is, the higher probability the attacker will attack that target. The parameter $\lambda$ governs the attacker's rationality. In particular, when $\lambda = 0$, the attacker is predicted to attack each target uniformly at random. When $\lambda = +\infty$, the attacker is perfectly rational (i.e., attacks a target with the highest expected utility).

**Implicit Function Theorem.** We state this theorem informally [Krantz and Parks, 2012; Rudin, 1986]: given $x \in \mathbb{R}^n$, $y \in \mathbb{R}^m$, consider $m$ equations $f_1(x, y) = \ldots = f_m(x, y) = 0$ at a fixed $x = a, y = b$. Then, under some mild conditions, there exists a function $g$ such that $g(x) = y$, in a ball $B$ around $(a, b)$, and $\nabla g(x) = -(A_y(x, g(x)))^{-1} A_x(x, g(x))$ in ball $B$ where $[A_x(x, g(x)) A_y(x, g(x))]$ is the Jacobian of $f = [f_1, \ldots, f_m]$ evaluated at $(x, g(x))$.

Recently, a few papers have used the implicit function theorem on KKT conditions of optimization problems to obtain derivative of the optimal solution with respect to input parameters [Wilder *et al.*, 2019]. For example, consider the following convex optimization problem which depends on some parameter $\theta$: $\min_x f(x, \theta)$ subject to $\mathbf{A}x \leq b$. The implicit function theorem applied on the KKT conditions gives:

$$\begin{bmatrix} \nabla_x^2 f(x, \theta) & \mathbf{A}^T \\ diag(\eta)\mathbf{A} & diag(\mathbf{A}x - b) \end{bmatrix} \begin{bmatrix} \frac{dx}{d\theta} \\ \frac{d\eta}{d\theta} \end{bmatrix} = - \begin{bmatrix} \frac{d \nabla_x f(x, \theta)}{d\theta} \\ 0 \end{bmatrix}$$

Hence, the derivative $\frac{dx}{d\theta}$ can be computed, given the closed form of the optimal decision $x$ as a function of $\theta$ is not known.

## 4 Partial Behavior Deception Model

In our game model, the attackers have the same payoffs but different attack behavior arising from different rationality levels. The defender knows the attacker payoffs but is uncertain about the attackers' behavior. Thus, the defender typically learns a *single* behavior model of the whole attacker population based on historical attack data. This is the prevalent approach in real-world domains such as wildlife protection since park rangers mostly cannot distinguish poaching signs from different poachers [Kar *et al.*, 2017]. We consider the setting in which the defender adopts QR to model the attacker behavior. The overall problem of the defender has two phases: learning and planning. In the learning phase, the defender optimizes the QR parameter based on the training attack dataset, formulated as follows:

$$\lambda^* \in \text{argmax}_\lambda \, \mathcal{L}(\lambda, \mathcal{D})$$

where $\mathcal{D}$ is the training set and $\mathcal{L}(\lambda, \mathcal{D})$ is the log-likelihood function. In particular, $\mathcal{D}$ can be compactly represented as $\{\mathbf{x}^m, \{z_i^m\}_{i=1}^T\}_{m=1}^M$ in which $M$ is the number of defense strategies and $z_i^m$ is the number of times target $i$ is attacked with respect to the defender's mixed strategy $\mathbf{x}^m$.

In the planning phase, the defender computes a strategy $\mathbf{x}^*$ that maximizes the defender's expected utility against a QR attacker with this $\lambda^*$, which is the result of:

$$\mathbf{x}^* \in \underset{\mathbf{x}}{\text{argmax}} \sum\nolimits_{i=1}^{T} q_i(\mathbf{x}, \lambda^*) U_i^d(x_i)$$
$$\text{s.t. } \sum\nolimits_i x_i \leq K, x_i \in [0,1], \forall i$$

While the defender relies on a single behavior model, some among the attackers can change their attack strategy in the learning phase to benefit in the planning phase. In particular, we expect only a perfectly rational attackers to exhibit such clever deceptive behavior. Hence, we focus on the deception situation in which there is a rational deceptive attacker (in addition to other sub-rational and non-deceptive attackers) who can perturb a fixed fraction of the training attack set. The fixed fraction enforces a natural constraint on the limited amount of deception that the deceptive attacker can do.

**Deception mathematical framework.** For each strategy, $\mathbf{x}^m$, in the training set $\mathcal{D}$, we denote by $n_i^m$ the number of attacks from non-deceptive attackers at target $i$ against $\mathbf{x}^m$. The total number of attacks against $\mathbf{x}^m$ of these non-deceptive attackers is $\sum_i n_i^m$. We assume the deceptive attacker can attack at most $f(\sum_i n_i^m)$ attacks against $\mathbf{x}^m$ where $f$ is the attack ratio of the deceptive attacker to the non-deceptive attackers. The problem of finding the best perturbation for the deceptive attacker can be formulated as follows:

$$\max\nolimits_{\{z_i^m\}} U^a(\mathbf{x}^*) \quad (1)$$
$$\text{subject to } \lambda^* \in \text{argmax}_\lambda \, \mathcal{L}(\lambda, \mathcal{D}) \quad (2)$$
$$\mathbf{x}^* \in \underset{\mathbf{x}}{\text{argmax}} \sum\nolimits_i q_i(x_i; \lambda^*) U_i^d(x_i) \quad (3)$$
$$\text{s.t.} \sum\nolimits_i x_i \leq K, x_i \in [0,1], \forall i \quad (4)$$
$$z_i^m \geq n_i^m, z_i^m \in \mathbb{N}, \forall m, i \quad (5)$$
$$\sum\nolimits_i z_i^m \leq (f+1) \sum\nolimits_i n_i^m, \forall m. \quad (6)$$

which maximizes the deceptive attacker's utility in the planning phase of the defender. The variable $z_i^m$ is the perturbed number of attacks at target $i$ against the defender's strategy $\mathbf{x}^m$. Constraints (5–6) guarantee that the deceptive attacker can only manipulate its own attacks. Finding the best perturbation for the attacker involves three nested optimization problems, which is not straightforward to solve. In this work, we propose a new algorithm to tackle this challenge, leveraging the implicit function theorem to apply the projected gradient descent (PGD) method as described in next section.

## 5 GAMBO: A Behavior Deception Algorithm

Our algorithm applies the projected gradient descent (PGD) to find an optimal attacker perturbation $\{z_i^m\}$ given the constraint that the deceptive attacker can only manipulate its own attacks. Since $z_i^m$ is discrete, we relax the feasible region of these variables to be continuous. We denote by $\mathbf{Z} = \{\mathbf{z} : \sum_i z_i^m \leq (f+1)\sum_i n_i^m, \forall m \text{ and } z_i^m \geq n_i^m, \forall m, i\}$ the relaxed region of the attacker's manipulation. Given a starting point $\mathbf{z}_0 \in \mathbf{Z}$, PGD iteratively improves the value of $\mathbf{z}$ based on the gradients of the attacker's utility with respect to the current value of $\mathbf{z}$. In particular, at step $t+1$, given the current value $\mathbf{z}_t$, the new value, $\mathbf{z}_{t+1}$, is updated as follows:

$$\mathbf{z}_{t+1} = \Pi_{\mathbf{z}}(\mathbf{z}_t + \gamma \nabla_{\mathbf{z}_t} U^a)$$

where $\gamma > 0$ is the step size and $\Pi_{\mathbf{z}}$ is the $L_2$ projection. In particular, $\Pi_{\mathbf{z}}(\mathbf{z}_t + \nabla_{\mathbf{z}_t} U^a)$ is defined as the feasible point closest to the gradient ascent update:

$$\Pi_{\mathbf{z}}(\mathbf{z}_t + \gamma \nabla_{\mathbf{z}_t} U^a) \in \text{argmin}_{\mathbf{z} \in \mathbf{Z}} \|\mathbf{z}_t + \gamma \nabla_{\mathbf{z}_t} U^a - \mathbf{z}\|_2$$

The gradient $\nabla_{\mathbf{z}} U^a$ can be decomposed using chain rule:

$$\frac{\partial U^a}{\partial z_i^m} = \sum\nolimits_j \frac{\partial U^a}{\partial x_j^*} \frac{\partial x_j^*}{\partial \lambda^*} \frac{\partial \lambda^*}{\partial z_i^m}, \forall m, i$$

The main challenge of computing $\nabla_{\mathbf{z}} U^a$ lies in the computation of each gradient component $\frac{\partial U^a}{\partial x_j^*}$, $\frac{\partial x_j^*}{\partial \lambda^*}$, and $\frac{\partial \lambda^*}{\partial z_i^m}$ especially when we do not have closed forms for these components. In the following sub-sections, we elaborate the challenges of computing each component and our ideas to address those challenges, in an increasing order of complexity.

### 5.1 Computing the Derivative $\frac{dU^a}{d\mathbf{x}^*} = \{\frac{\partial U^a}{\partial x_j^*}\}$

In the planning phase, the rational deceptive attacker will attack the target which gives it the highest expected utility against $\mathbf{x}^*$. However, this means that $U^a$ is not a differentiable function of $\mathbf{x}^*$. We propose to approximate $U^a$ as the following differentiable function, by assuming that the deceptive attacker follows a QR model with a very large $\lambda$, denoted by $\hat{\lambda}$ (when $\lambda \to \infty$, the attacker is perfectly rational):

$$U^a \approx \sum\nolimits_j q_j(\mathbf{x}^*; \hat{\lambda}) U_j^a(x_j^*)$$

It is now straightforward to compute the derivative $\frac{\partial U^a}{\partial x_j^*}$.

### 5.2 Computing the Derivative $\frac{d\lambda^*}{d\mathbf{z}} = \{\frac{\partial \lambda^*}{\partial z_i^m}\}$

By using Maximum Log-Likelihood Estimation (MLE), finding an optimal $\lambda^*$ with respect to the training set $\{z_i^m\}$ can be formulated as the following optimization problem:

$$\max_\lambda \mathcal{L}(\lambda, \mathcal{D}) = \max_\lambda \sum\nolimits_m \sum\nolimits_i z_i^m \log q_i(\mathbf{x}^m; \lambda)$$
$$= \sum\nolimits_m \Big[\lambda \sum\nolimits_i z_i^m U_i^a(x_i^m) - \Big(\sum\nolimits_i z_i^m\Big) \log \sum\nolimits_i e^{\lambda U_i^a(x_i^m)}\Big]$$

While it is straightforward to compute the optimal $\lambda^*$ with respect to the variable $\{z_i^m\}$, it is unclear how to compute the derivative $\frac{\partial \lambda^*}{\partial z_i^m}$ since we do not have a closed form of

$\lambda^*$ as a function of $\{z_i^m\}$. Therefore, we propose to apply the implicit function theorem which can be directly applied for convex optimization problems. Note that, $-\mathcal{L}(\lambda, \mathcal{D})$ is a convex function (its Hessian matrix is positive semi-definite). Therefore $\lambda^*$ satisfies the following KKT conditions:

$$G = -\frac{\partial \mathcal{L}(\lambda, \mathcal{D})}{\partial \lambda} = 0 \text{ at } \lambda^*$$

By applying the implicit function theorem, we can differentiate the solution to this linear system, which yields: $\forall (m, i)$

$$\frac{\partial \lambda^*}{\partial z_i^m} = -\Big(\frac{\partial G}{\partial z_i^m} \Big/ \frac{\partial G}{\partial \lambda^*}\Big), \forall m, i.$$

The RHSs of these equations are straightforward to compute.

## 5.3 Computing the Derivative $\frac{d\mathbf{x}^*}{d\lambda^*} = \{\frac{\partial x_i^*}{\lambda^*}\}$

Recall that $\mathbf{x}^*$ is the defender's optimal strategy, obtained by maximizing the defender's expected utility with respect to the learnt $\lambda^*$. The optimization problem is formulated as follows:

$$\mathbf{x}^* \in \operatorname{argmax}_{\mathbf{x}} \sum\nolimits_{i=1}^{T} q_i(\mathbf{x}, \lambda^*) U_i^d(x_i) \qquad \texttt{DefOPT}$$
$$\text{s.t. } \sum\nolimits_i x_i \leq K, x_i \in [0,1], \forall i$$

in which the objective is non-convex. In order to compute $\frac{d\mathbf{x}^*}{d\lambda^*}$, we first need to show given a $\lambda^*$ there is a unique $\mathbf{x}^*$, which is not obvious as the objective is non-convex. This is required for $\mathbf{x}^*$ to be a well-defined function of $\lambda^*$. We prove the uniqueness by analyzing the GOSAQ algorithm [Yang *et al.*, 2012] to solve DefOPT, which is briefly explained below.

**The GOSAQ Algorithm**

At a high level, GOSAQ applies binary search and then variable conversion to convert the feasibility problem of binary search into a convex optimization, which can be solved exactly. The binary search idea is simple: given some lower and upper bound on the objective, say $(L, U)$, it starts by solving the feasibility problem: does there exist a strategy, $\mathbf{x}$, such that the defender's expected utility is greater than $r = (L+U)/2$?

$$\sum\nolimits_{i=1}^{T} q_i(\mathbf{x}, \lambda^*) U_i^d(x_i) \geq r$$

If it is feasible, then binary search updates the lower bound as $L = r$. Otherwise, it updates the upper bound as $U = r$. Binary search then solves the feasibility problem again with the updated bounds. This process will continue until it reaches a stopping condition (i.e., $U - L \leq \epsilon$).

Observe that the feasibility problem can be solved by solving the following minimization problem (BinaryOPT):

$$\min\nolimits_{\mathbf{x}} \sum\nolimits_i e^{\lambda^* U_i^a(x_i)} (r - U_i^d(x_i)) \tag{7}$$
$$\text{s.t.} \sum\nolimits_i x_i \leq K, x_i \in [0,1], \forall i \tag{8}$$

Specifically, if the minimum of the objective in (7) is less than zero, then the corresponding optimal solution of $\mathbf{x}$ for (7–8) is a feasible solution of the feasibility problem in binary search. Otherwise, if it is strictly greater than zero, there does not exist a feasible $\mathbf{x}$. Note that (7–8) is a non-convex optimization problem, GOSAQ then uses the variable transformation $y_i = e^{\lambda^*(P_i^a - R_i^a)x_i}$ for (7–8), resulting in the following strictly convex program (BinaryOPTTransformed):

$$\min_{\mathbf{y}} \sum_{i=1}^{T} e^{\lambda^* R_i^a} (r - P_i^d) y_i + \sum_{i=1}^{T} \frac{(R_i^d - P_i^d) e^{\lambda^* R_i^a}}{\lambda^* (R_i^a - P_i^a)} y_i \ln y_i \tag{9}$$
$$\text{s.t.} \sum\nolimits_{i=1}^{T} \frac{1}{\lambda^*(P_i^a - R_i^a)} \ln y_i - K \leq 0 \tag{10}$$
$$y_i - 1 \leq 0, \forall i \tag{11}$$
$$-y_i + e^{\lambda^*(P_i^a - R_i^a)} \leq 0, \forall i \tag{12}$$

where the objective in (9) is equivalent to the objective in (7). Constraints (10–12) correspond to constraint (8).

**Proof of the Uniqueness of $\mathbf{x}^*$**

Based on the strict convexity of BinaryOPTTransformed, our Lemma 1 proves the uniqueness of $\mathbf{x}^*$ for any given $\lambda^*$.

**Lemma 1.** *Let $r_{max}$ be the maximum value of DefOPT. There is a unique $\mathbf{x}^*$ such that $r_{max} = \sum_{i=1}^{T} q_i(x_i^*; \lambda^*) U_i^d(x_i^*)$*

*Proof.* The optimal solution of BinaryOPTTransformed is unique for any $r$, by strict convexity. Thus, there is a unique $\mathbf{y}^*$ as the optimal solution of BinaryOPTTransformed for $r = r_{max}$. As the $\mathbf{x}$-to-$\mathbf{y}$ transform is one-one, there is a corresponding unique optimal solution $\mathbf{x}^*$ of BinaryOPT.

In addition, by definition of $r_{max}$, it must be the case that the objective value of BinaryOPT at $\mathbf{x}^*$ is 0, which is equivalent to $r_{max} = \sum_{i=1}^{T} q_i(x_i^*; \lambda^*) U_i^d(x_i^*)$. This can be checked by contradiction: assume the optimal objective value in (7) is not 0, then let it be $a \neq 0$. First assume $a > 0$, we obtain:

$$\sum\nolimits_i e^{\lambda^* U_i^a(x_i)} (r_{max} - U_i^d(x_i)) \geq a, \forall \mathbf{x}$$
$$\Longrightarrow r_{max} \geq \frac{a}{\sum_j e^{\lambda^* U_j^a(x_j)}} + \sum\nolimits_i q_i(\mathbf{x}, \lambda^*) U_i^d(x_i)$$
$$> \sum\nolimits_i q_i(\mathbf{x}, \lambda^*) U_i^d(x_i), \forall \mathbf{x}$$
$$\Longrightarrow r_{max} > \max_{\mathbf{x}} \sum\nolimits_i q_i(\mathbf{x}, \lambda^*) U_i^d(x_i) = r_{max}$$

which is contradiction. Next, assume $a < 0$, then, we obtain:

$$\sum\nolimits_i e^{\lambda^* U_i^a(x_i^*)} (r_{max} - U_i^d(x_i^*)) = a < 0$$
$$\Longrightarrow r_{max} < \sum\nolimits_i q_i(\mathbf{x}^*, \lambda^*) U_i^d(x_i^*)$$

which directly contradicts the definition of $r_{max}$. □

Since the optimal solution $\mathbf{x}^*$ of DefOPT is unique for each $\lambda^*$ according to Lemma 1, $\mathbf{x}^*$ is a well-defined function of $\lambda^*$. Yet, we do not have a closed-form function of $\lambda^*$ for $\mathbf{x}^*$. Therefore, we propose to apply the implicit function theorem to compute $\frac{d\mathbf{x}^*}{d\lambda^*}$, as explained in next section.

**Applying the Implicit Function Theorem**

In order to apply the implicit function theorem to get the derivative $\frac{d\mathbf{x}^*}{d\lambda^*}$, we propose to use the KKT conditions of BinaryOPTTransformed. Note that we cannot use KKT conditions of BinaryOPT or DefOPT as these are non-convex optimizations and thus KKT conditions are not the sufficient conditions for the optimal solution of these two problems.

First we relate $\mathbf{x}^*$ (the optimal solution of DefOPT) and $\mathbf{x}^+$ (the optimal solution of BinaryOPT) (the uniqueness of $\mathbf{x}^+$ follows the strict convexity of BinaryOPTTransformed). This relation is required to get the correct derivative of $\mathbf{x}^*$ through the dependent $\mathbf{x}^+$.[1] While $\mathbf{x}^*$ and $\mathbf{x}^+$ are dependent variables as $\mathbf{x}^*(\lambda^*)$ and $\mathbf{x}^+(\lambda^*, r)$, for notation ease, we skip writing the explicit dependence in the rest of this paper.

**Lemma 2.** $\mathbf{x}^* = \mathbf{x}^+$ *under* $F(\mathbf{x}^+, \lambda^*, r) = 0$ *where:*

$$F(\mathbf{x}^+, \lambda^*, r) = \left[\sum\nolimits_{i=1}^{T} q_i(\mathbf{x}^+, \lambda^*) U_i^d(x_i^+)\right] - r$$

*Proof.* Given $\lambda^*$, $F(\mathbf{x}^+, \lambda^*, r) = 0$ implies:

$$\sum\nolimits_i e^{\lambda^* U_i^a(x_i^+)}(r - U_i^d(x_i^+)) = 0$$

Since $\mathbf{x}^+$ is an optimal solution of BinaryOPT with respect to $r$, we obtain the following inequality: for all $\mathbf{x}$,

$$\sum\nolimits_i e^{\lambda^* U_i^a(x_i)}(r - U_i^d(x_i)) \geq \sum\nolimits_i e^{\lambda^* U_i^a(x_i^+)}(r - U_i^d(x_i^+)) = 0$$

$$\implies r \geq \sum\nolimits_{i=1}^{T} q_i(\mathbf{x}, \lambda^*) U_i^d(x_i) \qquad (13)$$

Based on (13) and $F(\mathbf{x}^+, \lambda^*, r) = 0$, then $r$ is the maximum objective of DefOpt and $\mathbf{x}^+$ is the corresponding maximizer. Since such a maximizer is unique (Lemma 1), $\mathbf{x}^+ = \mathbf{x}^*$. □

Thus, $\frac{d\mathbf{x}^*}{d\lambda^*}$ is same as $\frac{d\mathbf{x}^+}{d\lambda^*}$ under the dependence $F(\mathbf{x}^+, \lambda^*, r) = 0$; in the sequel we refer to $F(\mathbf{x}^+, \lambda^*, r)$ using the shorthand $F$. Based on this dependency, we invoke the implicit function theorem to get $\frac{d\mathbf{x}^+}{d\lambda^*}$ as follows. We first use the KKT conditions of BinaryOPTTransformed around its optimal solution $\mathbf{y}^*$. We then replace $y_i^* = e^{\lambda^*(P_i^a - R_i^a)x_i^+}$ in these KKT conditions and apply Lemma 2 (from where we get $F$) to obtain the following equality system of $\mathbf{x}^+$:

$$e^{\lambda^* R_i^a}(r - P_i^d) + \frac{(R_i^d - P_i^d)e^{\lambda^* R_i^a}}{\lambda^*(R_i^a - P_i^a)}(1 + \lambda^*(P_i^a - R_i^a)x_i^+) + \frac{\mu_0}{\lambda^*(P_i^a - R_i^a)e^{\lambda^*(P_i^a - R_i^a)x_i^+}} + \mu_i - \mu_{T+i} = 0, \forall i \qquad (F_i)$$

$$\mu_0\left(\sum\nolimits_{i=1}^{T} x_i^+ - K\right) = 0 \qquad (F_{\mu_0})$$

$$\mu_i(e^{\lambda^*(P_i^a - R_i^a)x_i^+} - 1) = 0, \forall i \qquad (F_{\mu_i})$$

$$\mu_{T+i}(-e^{\lambda^*(P_i^a - R_i^a)x_i^+} + e^{\lambda^*(P_i^a - R_i^a)}) = 0, \forall i \qquad (F_{\mu_{T+i}})$$

$$\frac{\sum_{i=1}^{T} e^{\lambda^* U_i^a(x_i^+)} U_i^d(x_i^+)}{\sum_{j=1}^{T} e^{\lambda^* U_j^a(x_j^+)}} - r = 0 \qquad (F)$$

where $\{\mu_0, .., \mu_{2T}\}$ are dual variables. We denote by $G = [F_1, \ldots, F_T, F_{\mu_0}, F_{\mu_1}, \ldots, F_{\mu_T}, F_{\mu_{T+1},\ldots,}, F_{\mu_{2T}}, F]^T$ the LHSs of above *equality* constraints. All of these functions take as arguments $(\mathbf{x}^+, \lambda^*, r, \mu)$. We get a system of $3T+2$ equations, which allows us to invoke the implicit function theorem. The implicit function theorem states that given the function $G(\lambda^*, (\mathbf{x}^+, r, \mu)) = 0$, which is a function from $3T+3$ variables to $3T+2$ dimensional space given by the $3T+2$ equations above, we have the derivative:

$$\frac{\partial(\mathbf{x}^+, r, \mu)}{\partial \lambda^*} = -\mathbf{M}^{-1}\mathbf{X}$$

where $\mathbf{M}$ is the $(3T+2) \times (3T+2)$ matrix, represented as:

$$\begin{pmatrix}
\frac{\partial F_1}{\partial x_1^+} & \cdots & \frac{\partial F_1}{\partial x_T^+} & \frac{\partial F_1}{\partial r} & \frac{\partial F_1}{\partial \mu_0} & \frac{\partial F_1}{\partial \mu_1} & \cdots & \frac{\partial F_1}{\partial \mu_{2T}} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\frac{\partial F_T}{\partial x_1^+} & \cdots & \frac{\partial F_T}{\partial x_T^+} & \frac{\partial F_T}{\partial r} & \frac{\partial F_T}{\partial \mu_0} & \frac{\partial F_T}{\partial \mu_1} & \cdots & \frac{\partial F_T}{\partial \mu_{2T}} \\
\frac{\partial F_{\mu_0}}{\partial x_1^+} & \cdots & \frac{\partial F_{\mu_0}}{\partial x_T^+} & \frac{\partial F_{\mu_0}}{\partial r} & \frac{\partial F_{\mu_0}}{\partial \mu_0} & \frac{\partial F_{\mu_0}}{\partial \mu_1} & \cdots & \frac{\partial F_{\mu_0}}{\partial \mu_{2T}} \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\frac{\partial F_{\mu_{2T}}}{\partial x_1^+} & \cdots & \frac{\partial F_{\mu_{2T}}}{\partial x_T^+} & \frac{\partial F_{\mu_{2T}}}{\partial r} & \frac{\partial F_{\mu_{2T}}}{\partial \mu_0} & \frac{\partial F_{\mu_{2T}}}{\partial \mu_1} & \cdots & \frac{\partial F_{\mu_{2T}}}{\partial \mu_{2T}} \\
\frac{\partial F}{\partial x_1^+} & \cdots & \frac{\partial F}{\partial x_T^+} & \frac{\partial F}{\partial r} & \frac{\partial F}{\partial \mu_0} & \frac{\partial F}{\partial \mu_1} & \cdots & \frac{\partial F}{\partial \mu_{2T}}
\end{pmatrix}$$

and $\mathbf{X}$ is the $(3T+2) \times 1$ column vector, represented as:

$$\mathbf{X} = \left[\frac{\partial F_1}{\partial \lambda^*}, \ldots, \frac{\partial F_T}{\partial \lambda^*}, \frac{\partial F_{\mu_0}}{\partial \lambda^*}, \frac{\partial F_{\mu_1}}{\partial \lambda^*}, \ldots, \frac{\partial F_{\mu_{2T}}}{\partial \lambda^*}, \frac{\partial F}{\partial \lambda^*}\right]^T$$

and all the partial derivatives are evaluated at $\mathbf{x}^+, \lambda^*, r, \mu$. The first $T$ components of $-\mathbf{M}^{-1}\mathbf{X}$ is the derivative $\frac{d\mathbf{x}^+}{d\lambda^*}$ under the dependence $F = 0$, which is same as $\frac{d\mathbf{x}^*}{d\lambda^*}$ (Lemma 2).

## 6 Experiments

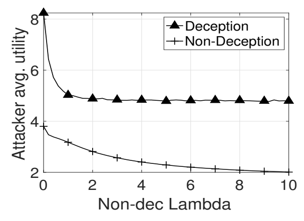
We analyze the impact of the attacker deception on: (i) the deceptive attacker's utility benefit; (ii) the defender's utility loss; and (iii) the defender's learning outcome. For the sake of analysis, besides the rational attacker, we consider the presence of a boundedly rational (non-deceptive) attacker whose responses follow QR with a fixed $\lambda$. We compare two scenarios: (i) Deception: the rational attacker plays deceptively—its attacks in the training phase are computed by our algorithm, GAMBO; (ii) Non-deception: the rational attacker plays non-deceptively. We use the game generator, GAMUT (http://gamut.stanford/edu) to generate player payoffs.

In creating the training dataset, for each game, we generate $M = 5$ different defense strategies uniformly at random. For each generated strategy $\mathbf{x}^m$, we sample 50 attacks (i.e., $\sum_i n_i^m = 50$) for the boundedly rational attacker with respect to its $\lambda$. We plot the experiment results in three cases (the x-axis in the plotted figures): (i) varying the $\lambda$ of the boundedly rational attacker; (ii) varying the percentage of deceptive attacks ($\frac{f}{f+1}$); and (iii) varying resource-target ratio ($\frac{K}{T}$). Each data point is averaged over more than 3000 game instances.

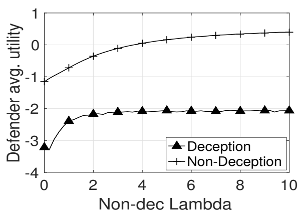
### 6.1 Evaluation on Utility of Players

In the following, we highlight the results of 40-target games. Figure 1 illustrates our evaluation on the utility of the rational (deceptive) attacker and the defender. Note that the defender's utility is computed as an expectation over the deceptive and non-deceptive attackers. Figure 1 shows that the rational attacker obtains a significantly higher expected utility on average when it plays deceptively compared to when it

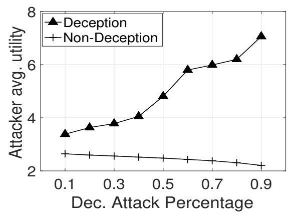
[1]Capturing dependence is critical to get the correct derivative. For example, $\frac{\partial xy}{\partial x} = y$ when $y$ is not dependent on x, but, under the dependence $y = x$ we have $\frac{\partial xy}{\partial x} = 2x$



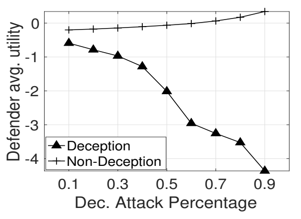
(a) Vary $\lambda$ of non-dec. attacker



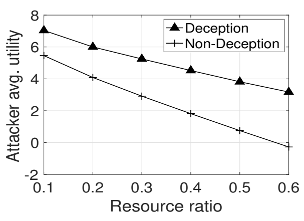
(b) Vary $\lambda$ of non-dec. attacker



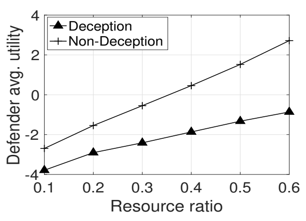
(c) Vary % of dec. attacks



(d) Vary % of dec. attacks

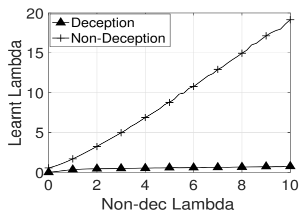


(e) Vary resource-target ratio

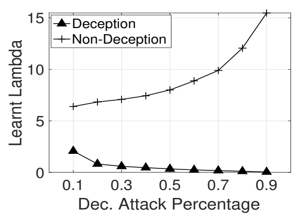


(f) Vary resource-target ratio

Figure 1: Evaluation on the players' utility

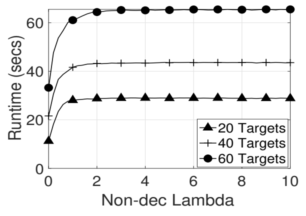


(a) Vary $\lambda$ of non-dec. attacker

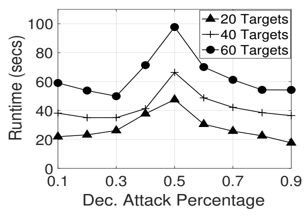


(b) Vary % of dec. attacks

Figure 2: Evaluation on the learnt $\lambda^*$ of the defender



(a) Vary $\lambda$ of non-dec. attacker



(b) Vary % of dec. attacks

Figure 3: Runtime performance of `GAMBO`

plays honestly (`Deception` versus `Non-Deception`). Conversely, the defender receives a significantly lower utility.

In Figures 1(a) and 1(b), the attacker's utility is roughly a decreasing convex function while the defender's utility is an increasing concave function of the `QR`'s $\lambda$ of the non-deceptive attacker. When this $\lambda$ is close to zero, the non-deceptive attacker becomes less strategic and thus has less impact on the learning outcome, reflecting increased chance for successful deception of the rational attacker.

In Figures 1(c) and 1(d), in `Deception`, the utility of the rational attacker increases quickly while the utility of the defender decreases quickly as the percentage of deceptive attacks increases. The increase in the attack percentage reflects the growth in options for deception, leading to increased benefit of the rational attacker for misleading the defender.

In Figures 1(e) and 1(f), the average utility of players is roughly a linearly function (decreasing for the attacker and increasing for the defender) of the resource-target ratio. This makes sense as the players' expected utility at each target is a linearly function (also decreasing for the attacker and increasing for the defender) of the defender's coverage probability.

## 6.2 Evaluation on Learning Outcomes

Figure 2 shows the learning outcome (i.e., the learnt $\lambda^*$) of the defender. When the rational attacker plays honestly, the learnt $\lambda^*$ gradually increases when $\lambda$ of the non-deceptive attacker or the percentage of attacks by the rational attacker increases. Conversely, by playing deceptively, the rational attacker could influence the learning outcome towards a very small learnt $\lambda^*$. Finally, our results show that the learnt $\lambda^*$ is roughly constant when varying the resource-target ratio, showing that this factor does not impact the learning outcome (we do not plot these results due to limited space).

## 6.3 Runtime Performance

Finally, the runtime performance of our algorithm, `GAMBO`, is shown in Figure 3. The y-axis is the runtime in seconds on average. Figure 3(a) shows that the runtime of `GAMBO` increases gradually when the $\lambda$ of the non-deceptive attacker increases and then becomes roughly constant when this $\lambda$ reaches the value of 2. This shows that when the non-deceptive attacker is nearly non-strategic (its $\lambda$ is close to zero), `GAMBO` can quickly find a deception strategy for the rational attacker. In Figure 3(b), the peak of `GAMBO`'s runtime is at the attack percentage of $0.5$, reflecting that the time complexity of `GAMBO` is highest when the deceptive and non-deceptive attackers contribute equally to the attack dataset. Finally, `GAMBO` can solve large games (i.e., 200-target games) in approximately $10$ minutes (we do not plot these results due to limited space).

## 7 Summary

This paper studied the problem of the attacker deception in security games, where the defender relies on historical attack data to learn the attackers' behavior. We proposed a novel partial behavior deception model and introduced a new scalable algorithm, `GAMBO`, to compute an optimal deception strategy of the deceptive attacker. Our algorithm provided a novel usage of the implicit function theorem to solve a tri-level optimization problem. Our thorough experiments reveal a significant benefit for the deceptive attacker while the defender suffers a significant loss in utility.

## References

[Biggio and Roli, 2018] Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

[Blum *et al.*, 2014] Avrim Blum, Nika Haghtalab, and Ariel D Procaccia. Learning optimal commitment to overcome insecurity. In *Neural Information Processing Systems*, 2014.

[Carroll and Grosu, 2011] Thomas E Carroll and Daniel Grosu. A game theoretic investigation of deception in network security. *Security and Communication Networks*, 4(10):1162–1172, 2011.

[Donti *et al.*, 2017] Priya Donti, Brandon Amos, and J Zico Kolter. Task-based end-to-end model learning in stochastic optimization. In *Neural Information Processing Systems*, 2017.

[Fang *et al.*, 2016] Fei Fang, Thanh H Nguyen, Rob Pickles, Wai Y Lam, Gopalasamy R Clements, Bo An, Amandeep Singh, Milind Tambe, and Andrew Lemieux. Deploying paws: Field optimization of the protection assistant for wildlife security. In *IAAI*, 2016.

[Gan *et al.*, 2019a] Jiarui Gan, Qingyu Guo, Long Tran-Thanh, Bo An, and Michael Wooldridge. Manipulating a learning defender and ways to counteract. In *NeurIPS*, 2019.

[Gan *et al.*, 2019b] Jiarui Gan, Haifeng Xu, Qingyu Guo, Long Tran-Thanh, Zinovi Rabinovich, and Michael Wooldridge. Imitative follower deception in stackelberg games. In *Proceedings of the ACM EC*, EC '19, 2019.

[Guo *et al.*, 2017] Qingyu Guo, Bo An, Branislav Bosansky, and C. Kiekintveld. Comparing strategic secrecy and Stackelberg commitment in security games. In *IJCAI*, 2017.

[Horák *et al.*, 2017] Karel Horák, Quanyan Zhu, and Branislav Bošanskỳ. Manipulating adversary's belief: A dynamic game approach to deception by design for proactive network security. In *GameSec*, 2017.

[Huang *et al.*, 2011] Ling Huang, Anthony D Joseph, Blaine Nelson, Benjamin IP Rubinstein, and J Doug Tygar. Adversarial machine learning. In *ACM workshop on Security and Artificial Intelligence*, pages 43–58, 2011.

[Kar *et al.*, 2015] Debarun Kar, Fei Fang, Francesco Delle Fave, Nicole Sintov, and Milind Tambe. A game of thrones: when human behavior models compete in repeated stackelberg security games. In *AAMAS*, 2015.

[Kar *et al.*, 2017] Debarun Kar, Benjamin Ford, Shahrzad Gholami, Fei Fang, Andrew Plumptre, Milind Tambe, Margaret Driciru, Fred Wanyama, Aggrey Rwetsiba, Mustapha Nsubaga, et al. Cloudy with a chance of poaching: Adversary behavior modeling and forecasting with real-world poaching data. In *AAMAS*, 2017.

[Korzhyk *et al.*, 2011] Dmytro Korzhyk, Vincent Conitzer, and Ronald Parr. Security games with multiple attacker resources. In *IJCAI*, 2011.

[Krantz and Parks, 2012] Steven G Krantz and Harold R Parks. *The implicit function theorem: history, theory, and applications*. Springer Science & Business Media, 2012.

[Lowd and Meek, 2005] Daniel Lowd and Christopher Meek. Adversarial learning. In *SIGKDD*, 2005.

[Madry *et al.*, 2017] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

[McFadden and others, 1973] Daniel McFadden et al. Conditional logit analysis of qualitative choice behavior. 1973.

[McKelvey and Palfrey, 1995] Richard D McKelvey and Thomas R Palfrey. Quantal response equilibria for normal form games. *Games and economic behavior*, 10, 1995.

[Nguyen and Xu, 2019] Thanh Nguyen and Haifeng Xu. Imitative attacker deception in stackelberg security games. In *IJCAI*, pages 528–534, 7 2019.

[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.*, 2019] Thanh H Nguyen, Yongzhao Wang, Arunesh Sinha, and Michael P Wellman. Deception in finitely repeated security games. In *AAAI*, 2019.

[Rabinovich *et al.*, 2015] Zinovi Rabinovich, Albert Xin Jiang, Manish Jain, and Haifeng Xu. Information disclosure as a means to security. In *AAMAS*, 2015.

[Rudin, 1986] W. Rudin. *Principles of Mathematical Analysis*. McGraw - Hill Book C., 1986.

[Sinha *et al.*, 2016] Arunesh Sinha, Debarun Kar, and Milind Tambe. Learning adversary behavior in security games: A pac model perspective. In *AAMAS*, 2016.

[Tambe, 2011] Milind Tambe. *Security and game theory: algorithms, deployed systems, lessons learned*. Cambridge university press, 2011.

[Wilder *et al.*, 2019] Bryan Wilder, Bistra Dilkina, and Milind Tambe. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *AAAI*, pages 1658–1665, 2019.

[Xu *et al.*, 2015] Haifeng Xu, Zinovi Rabinovich, Shaddin Dughmi, and Milind Tambe. Exploring information asymmetry in two-stage security games. In *AAAI*, 2015.

[Yang *et al.*, 2011] Rong Yang, Christopher Kiekintveld, Fernando Ordonez, Milind Tambe, and Richard John. Improving resource allocation strategy against human adversaries in security games. In *IJCAI*, 2011.

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

[Zhuang *et al.*, 2010] Jun Zhuang, Vicki M. Bier, and Oguzhan Alagoz. Modeling secrecy and deception in a multi-period attacker-defender signaling game. *European Journal of Operational Research*, 203:409–418, 2010.