Designing the Game to Play: Optimizing Payoff Structure in Security Games

Zheyuan Ryan Shi[1⋆], Ziye Tang[2⋆], Long Tran-Thanh[3], Rohit Singh[4], and Fei Fang[2]

[1] Swarthmore College, Swarthmore PA 19081, USA
zshi1@swarthmore.edu
[2] Carnegie Mellon University, Pittsburgh PA 15213, USA
ziyet@andrew.cmu.edu, feifang@cmu.edu
[3] University of Southampton, Southampton SO17 1BJ, United Kingdom
ltt08r@ecs.soton.ac.uk
[4] World Wide Fund for Nature, Cambodia
rsingh@wwfnet.org

Abstract. We study Stackelberg Security Games where the defender, in addition to allocating defensive resources to protect targets from the attacker, can strategically manipulate the attacker's payoff under budget constraints in weighted L^p-norm form regarding the amount of change. For the case of weighted L^1-norm constraint, we present (i) a mixed integer linear program-based algorithm with approximation guarantee; (ii) a branch-and-bound based algorithm with improved efficiency achieved by effective pruning; (iii) a polynomial time approximation scheme for a special but practical class of problems. In addition, we show that problems under budget constraints in L^0 and weighted L^∞-norm form can be solved in polynomial time.

Keywords: Security games · Optimization · Game theory.

1 Introduction

Research efforts in security games have led to success in various domains, ranging from protecting critical infrastructure [18, 26] and catching fare invaders in metro systems [29], to combating poaching [9] and preventing cyber intrusions [7, 1]. In these games, a defender protects a set of targets from an attacker through allocating defensive resources. One key element that characterizes the strategies of the players is the payoff structure. Existing work in this area typically treats the payoff structure of the players as given parameters, sometimes with uncertainties known a priori given the nature of the domain. However, under various circumstances, the defender is able to change the attacker's payoff, thus rendering the existing models inadequate in expressiveness. For example, in wildlife poaching, the law enforcement agency may charge a variable fine if the poacher

⋆ Z. R. Shi and Z. Tang contributed equally to this work.

is caught at different locations, e.g., in the core area v.s. in the non-core area. [Ziye: Similarly in the case of catching fare invaders in metro systems.] In cybersecurity, the network administrator may change the actual or appeared value of any network node for a potential attacker. In these cases, the defender's decision making is two-staged: she needs to choose the payoff structure, as well as the strategy of allocating defensive resources. With a properly chosen payoff structure, the defender can potentially achieve much better utility with the same or even less amount of resources, saving the effort of the defender.

As existing work in security games do not provide adequate tools to deal with this problem (see Section 2 for more details), we aim to fill this gap as follows. We study how to design the attacker's payoff structure in security games given budget constraints in weighted L^p-norm (it is easy to show that the problem becomes trivial without any budget constraints). The intuition behind this setting is that the defender can change the payoffs to make a target that is preferable to the defender more attractive to the attacker and disincentivize the attacker from attacking targets that can lead to a significant loss to the defender, but more change incurs a higher cost to the defender. Our findings can be summarized as follows:

1. L^1-norm case: When the budget constraint is in weighted L^1-norm form, i.e. additive cost, our contribution is threefold. (i) We exploit several key properties of the optimal manipulation and propose a *mixed integer linear program* (MILP)-based algorithm with *approximation guarantee*. (ii) We propose a *novel branch-and-bound approach* with improved efficiency achieved by effective pruning for the general case. (iii) Finally, we show that a *polynomial time approximation scheme* (PTAS) exists for a special but practical case where the budget is very limited and the manipulation cost is uniform across targets. The PTAS is built upon the key observation that there is an optimal solution where no more than two targets' payoffs are changed in this restricted case.

2. L^0 and L^∞-norm cases: We propose *a $O(n^3)$ and a $O(n^2 \log n)$ algorithm* for problems under budget constraints in L^0-norm form and weighted L^∞-norm form, respectively, where n is the total number of targets. For L^0-norm form budget, i.e. limited number of targets to manipulate, our algorithm converts the problem into n^2 subproblems and reduce each subproblem into a problem of finding a subset of items with maximum average weight, which can be solved in $O(n)$ time. For L^∞-norm form budget, i.e. limited range of manipulation on each target, the proposed algorithm reduces the problem to traditional Stackelberg Security Games with fixed payoff structure, which again admits an efficient algorithm.

3. Numerical evaluation: We provide extensive experimental evaluation for the proposed algorithms. For problems with L^1-norm form budget constraint, we show that the branch-and-bound approach with an additive approximation guarantee can solve up to hundreds of targets in a few minutes. This is faster than other baseline algorithms we compare to. Somewhat surprisingly, naively solving n non-convex subproblems using interior point method achieves good performance in practice. Yet there is no theoretical guarantee in solution quality.

We also evaluate the proposed $O(n^3)$ algorithm for the L^0-norm form case and show its superior performance over two greedy algorithms and a MILP based algorithm.

2 Preliminaries and Related Work

The security game that we consider in this paper features a set of n targets, $T = \{1, 2, \ldots, n\}$. The defender has r units of defensive resources, each can protect one target. The attacker can choose to attack one target after observing the defender's strategy. If the defender covers target i when it is attacked, the defender gets a reward $R_i^d \geq 0$ and the attacker gets a penalty $P_i^a \leq 0$. Otherwise, the defender gets a penalty $P_i^d \leq 0$ and the attacker gets a reward $R_i^a \geq 0$. When the defender commits to a mixed strategy c, that is, covering target i with probability c_i, the defender's and attacker's expected utilities when target i is attacked are $U_i^d = c_i R_i^d + (1-c_i)P_i^d$ and $U_i^a = c_i P_i^a + (1-c_i)R_i^a$, respectively.

We adopt the commonly used solution concept of Strong Stackelberg Equilibrium (SSE). At an SSE, the defender chooses an optimal strategy that leads to the highest expected utility for her when the attacker chooses a best response (assumed to be a pure strategy w.l.o.g), breaking ties in favor of the defender. Given a coverage c, the attack set $\Gamma \subseteq T$ contains all targets which have a weakly higher attacker's expected utility than any other target, i.e.,

$$\Gamma = \{j \in T : U_j^a \geq U_k^a, \forall k \in T\} \tag{1}$$

[13] show that there exists an SSE where the defender only covers the targets in the attack set.

Given the game parameters, the optimal defender strategy in such a game can be computed using multiple Linear Programs (LPs) [5] or an efficient $O(n^2)$ algorithm called ORIGAMI [13] based on enumerating the possible attack sets. We leverage insights from both works to devise our algorithms.

Although many algorithms have been developed to solve security games under various settings, in most existing work, the payoff structure is treated as fixed and cannot be changed by the defender, either in the full information case [16, 21, 17], or in the presence of payoff uncertainties [12, 15, 30, 19, 4]. As mentioned in the introduction, in many real-world scenarios the defender has control over the attacker's payoffs. The above-mentioned approaches ignore this aspect and thus leave room for further optimization.

Indeed, despite its significance, jointly optimizing the payoff structure and the resource allocation is yet under-explored. A notable exception, and a most directly related work to ours, is the audit game model [2, 3]. The defender can choose target-specific 'punishment rates', in order to maximize her expected utility offset by the cost of setting the punishment rate. Compared with their model, ours is more general in that we allow not only manipulation of attacker's penalty, but also attacker's reward. This realistic extension makes their core techniques inapplicable. Also, we treat the manipulation cost as a constraint instead of a regularization term in the objective function, for in some real-world settings,

payoffs can be manipulated only once, yet the defender may face multiple attacks afterwards, This makes it hard to determine the regularization coefficient. Another closely related work [23] extends previous work on the use of honeypot [14, 6, 7, 22]. It studies the problem of deceiving a cyber attacker through manipulating the attacker's (believed) payoff. However, this work assumes the defender can only change the payoff structure, ignoring the allocation of defensive resources after the manipulation.

If we conceptually decouple the payoff manipulation from resource allocation, the defender faces a two-stage decision. She first chooses the structure of the game, and then plays the game. Thus, our problem may be viewed as a mechanism design problem, albeit not in a conventional setting. Most work in mechanism design considers private information games [10, 20], while in our work, and most security game literature, the payoff information is public. Some design the incentive mechanism using a Stackelberg game [11], with applications to network routing [24], mobile phone sensing [28], and ecology surveillance [27]. However, these work solves the Stackelberg game to design the mechanism, rather than designing the structure of the Stackelberg game.

3 Optimizing Payoff with Budget Constraint in Weighted L^1-norm Form

In this section, we focus on computing the optimal way of manipulating attacker's payoffs and allocating defensive resources. Payoff manipulation is subject to a budget constraint in weighted L^1-norm form, i.e., the defender can change the attacker's reward and penalty, at a cost that grows linearly in the amount of change. The cost rate, referred to as weights, may be different across targets. This is an abstraction of several domains. For example, a network administrator may change the actual or appeared value of any network node although such change often incurs time and hardware costs.

Let R^a, P^a, $\bar{R}^a$, $\bar{P}^a$ denote the attacker's reward and penalty vectors before and after the manipulation. Similar to the initial payoff structure, we require that $\bar{R}^a \geq 0 \geq \bar{P}^a$ and denote $D_j = R^a_j - P^a_j$. Let $\epsilon = \bar{R}^a - R^a$ and $\delta = \bar{P}^a - P^a$ be the amount of change in attacker's reward and penalty and μ, θ the weights on ϵ, δ resp.. The weighted L^1 budget constraint is then $\sum_j (\mu_j |\epsilon_j| + \theta_j |\delta_j|) \leq B$ where B is the budget. The defender's strategy is characterized by (c, ϵ, δ). Given this strategy, in the manipulated game the attacker attacks some target t, which belongs to the attack set Γ. We first show some properties of the optimal solution.

Property 1. There is an optimal solution (c, ϵ, δ) with attack target t and attack set Γ which satisfies the following conditions:

1. $c_j = 0, \epsilon_j = 0, \delta_j = 0, \forall j \notin \Gamma$.
2. $\epsilon_t \geq 0, \delta_t \geq 0$; $\epsilon_j \leq 0, \delta_j \leq 0$, $\forall j \neq t$.
3. $\delta_t \epsilon_t (\delta_t + P^a_t) = 0$ and $\delta_j \epsilon_j (R^a_j + \epsilon_j) = 0$, $\forall j \neq t$.

Proof (Proof sketch). Condition 1: If any $\epsilon_j, \delta_j \neq 0$ with $j \notin \Gamma$, we may either set $\epsilon_j, \delta_j = 0$ or push j into the attack set. There is no need to protect a target

that is not in the attack set. Condition 2: We may flip the sign of ϵ and δ without affecting the solution structure. Condition 3: For each target i, changing R_i^a and P_i^a are equivalent while one is more budget efficient than the other depending on coverage. We can show that these manipulations can be done simultaneously.[5]

Similar to the multiple LPs formulation in [5], we consider n subproblems $\mathcal{P}_i$, each assuming some target $i \in T$ is the attack target, and the best solution among all n subproblems is the optimal defender strategy. Condition 2 in Property 1 shows it is possible to infer the sign of ϵ and δ given the attack target. So in the sequel, we abuse the notation by treating ϵ, δ as the absolute value of the amount of change, and assume w.l.o.g. that in $\mathcal{P}_i$, $\bar{R}_i^a = R_i^a + \epsilon_i, \bar{P}_i^a = P_i^a + \delta_i$ and $\forall j \neq i, \bar{R}_j^a = R_j^a - \epsilon_j, \bar{P}_j^a = P_j^a - \delta_j$. Thus, a straightforward formulation for $\mathcal{P}_i$ is

$$\max_{c,\epsilon,\delta} \quad U_i^d = R_i^d c_i + P_i^d(1 - c_i) \tag{2}$$

$$\text{s.t.} \quad U_i^a = c_i(P_i^a + \delta_i) + (1 - c_i)(R_i^a + \epsilon_i) \tag{3}$$

$$\geq U_j^a = c_j(P_j^a - \delta_j) + (1 - c_j)(R_j^a - \epsilon_j), \forall j \neq i$$

$$\sum\nolimits_j (\mu_j \epsilon_j + \theta_j \delta_j) \leq B \tag{4}$$

$$\sum\nolimits_j c_j \leq r \tag{5}$$

$$R_j^a - \epsilon_j \geq 0, \quad \forall j \neq i \tag{6}$$

$$P_i^a + \delta_i \leq 0 \tag{7}$$

$$c_j, \epsilon_j, \delta_j \geq 0, \quad c_j \leq 1, \quad \forall j \in T \tag{8}$$

The above formulation is non-convex due to the quadratic terms in Constraint 3, and thus no existing solvers can guarantee global optimality in polynomial time for the above formulation.

3.1 A MILP-based Solution with Approximation Guarantee

To find a defender strategy with approximation guarantee, we solve the atomic version of the subproblems with MILPs. We show an approximation guarantee which improves with the fineness of discretization. We further propose a branch-and-bound-like framework for pruning subproblems to improve runtime efficiency.

In the atomic version of the payoff manipulation problem, we assume the defender can only make atomic changes, with the minimum amount of change given as ρ_0. We refer to the atomic version of $\mathcal{P}_i$ as $\mathcal{AP}_i$. $\mathcal{AP}_i$ can be formulated as the MILP in Equations 9-19. We simplify the objective function as c_i since $D_i \geq 0$. All constraints involving sub/super-script j, k without a summation apply to all proper range of summation indices. We use binary representation

[5] Due to limited space, the omitted full proofs are included in the online appendix: https://www.dropbox.com/s/d2bv1v8kzvf85i0/APPENDIX.pdf?dl=0

for $\bar{R}_i^a/\rho_0$ and $\bar{P}_i^a/\rho_0$ in constraints 10-14. The binary representation results in bilinear terms like $y_j^k c_j$. We introduce variables α_j^k, β_j^k and constraints 18-19 to linearize them.

$$\max_{y_j^k, z_j^k, c_j} \quad c_i \tag{9}$$

$$s.t. \quad \text{Constraint 4-8}$$

$$\epsilon_i = \rho_0 \sum\nolimits_k 2^k y_i^k - R_i^a \tag{10}$$

$$\epsilon_j = R_j^a - \rho_0 \sum\nolimits_k 2^k y_j^k, \quad \forall j \neq i \tag{11}$$

$$\delta_i = -\rho_0 \sum\nolimits_k 2^k z_i^k - P_i^a \tag{12}$$

$$\delta_j = P_j^a + \rho_0 \sum\nolimits_k 2^k z_j^k, \quad \forall j \neq i \tag{13}$$

$$y_j^k, z_j^k \in \{0, 1\} \tag{14}$$

$$v_i \geq v_j \tag{15}$$

$$v_i = R_i^a + \epsilon_i - \rho_0 \sum\nolimits_k 2^k (\alpha_i^k + \beta_i^k) \tag{16}$$

$$v_j = R_j^a - \epsilon_j - \rho_0 \sum\nolimits_k 2^k (\alpha_j^k + \beta_j^k), \forall j \neq i \tag{17}$$

$$0 \leq \alpha_j^k \leq y_j^k, \quad c_j - (1 - y_j^k) \leq \alpha_j^k \leq c_j \tag{18}$$

$$0 \leq \beta_j^k \leq z_j^k, \quad c_j - (1 - z_j^k) \leq \beta_j^k \leq c_j \tag{19}$$

The optimal defender strategy for the atomic payoff manipulation problem can be found by checking the solution to all the subproblems and compare the corresponding U_i^d. We can also combine all the subproblems by constructing a single MILP, with additional variables indicating which subproblem is optimal. The full MILP is included in Appendix.

A natural idea to approximate the global optima of the original L^1-constrained payoff manipulation problem is, for each attack target i, approximate $\mathcal{P}_i$ with $\mathcal{AP}_i$ using small enough ρ_0. Theorem 1 below shows such an approximation bound.

Theorem 1. *The solution of the atomic problem is an additive* $\max_i 2\rho_0(R_i^d - P_i^d)/R_i^a$*-approximation to the original problem.*

Proof (Proof sketch). The floor and ceiling notations are about the "integral grid" defined by ρ_0. Suppose $(c^*, \epsilon^*, \delta^*)$ is an optimal solution to $\mathcal{P}_i$. Let $\epsilon' = \lfloor \epsilon^* \rfloor$, $\delta' = \lfloor \delta^* \rfloor$, and $c' = c^*$ except $c_i' = c_i^* - 2\rho_0/(D_i + \epsilon_i' - \delta_i')$. We can show such feasible solutions yield the desired approximation bound.

We note that the idea of discretizing the manipulation space is similar to [2, 3]. Yet allowing changes in both reward and penalty and the difference in objective function make our formulation different and the reduction to SOCP technique inapplicable. We can further improve the practical runtime of the MILPs by pruning and prioritizing subproblems as shown in Alg 1. We first compute a

global lower bound by checking a sequence of greedy manipulations. Inspired by Condition 2 and 3 in Property 1, we greedily spend all the budget on one target to increase its reward or penalty, leaving all other targets' payoff parameters unchanged (Line 2 - 8).

Upper bounds in $\mathcal{P}_i$ can be computed with budget reuse: we increase R_i^a and P_i^a as much as possible and decrease R_j^a and P_j^a, $j \neq i$, as much as possible. For the ease of notation, in Algorithm 1 we assume manipulations have uniform cost. The weighted case can be easily extended.

The subproblem $\mathcal{P}_i$ is pruned if its upper bound is lower than the global lower bound. To make the pruning more efficient, we solve subproblems in descending order of their corresponding lower bounds, hoping for an increase in the global lower bound. For subproblems that cannot be pruned, we set ρ_0 to a desired accuracy and solve the MILP to approximate the subproblem optima. We also add the linear constraint on c_i derived from the global lower bound to the MILP.

To get the bounds, we call an improved version of the ORIGAMI algorithm in [13] by doing a binary search on the size of the attack set Γ, and solve the linear system. It is denoted as ORIGAMI-BS in Alg 1. Recall r is the defender's total resource. Let M be the attacker's expected utility for attack set and $\bar{E}_k = \frac{1}{\bar{R}_i^a - \bar{P}_i^a}$. From $U_i^a = U_j^a, \forall j \in \Gamma$ and $\sum_{j\in\Gamma} c_j = r$, we obtain

$$M = \frac{\sum_{k\in\Gamma} \bar{R}_k^a \bar{E}_k - r}{\sum_{k\in\Gamma} \bar{E}_k} \tag{20}$$

$$c_j = \bar{E}_j \left(\frac{\sum_{k\in\Gamma} (\bar{R}_j^a - \bar{R}_k^a)\bar{E}_k + r}{\sum_{k\in\Gamma} \bar{E}_k} \right), \qquad \forall j \in \Gamma \tag{21}$$

We iteratively cut the search space by half based on c_j and M. The complexity improves from $O(n^2)$ to $O(n \log n)$. A complete description can be found in Appendix.

We end this subsection by remarking that atomic payoff manipulation arises in many real-world applications. For example, it is infeasible for the wildlife ranger to charge the poacher a fine of \$100/3. In those cases, our proposed MILP formulation could be directly applied.

3.2 PTAS for Limited Budget and Uniform Costs

We show that for a special but practical class of problems, there exist a PTAS. In many applications, the defender has only a limited budget $B \leq \min_{j\in T} \left|P_j^a\right|$. Additionally, the weights on ϵ and δ are the same. W.l.o.g., we assume $\mu_j = \theta_j = 1$. We first show a structural theorem below and then discuss its algorithmic implication.

Theorem 2. *When budget $B \leq \min_{j\in T} \left|P_j^a\right|$ and $\mu_j = \theta_j = 1, \forall j \in T$, there exists an optimal solution which manipulates the attack target and at most one other target.*

Algorithm 1 Branch-and-bound

Input: Payoffs $\sigma = \{R^d, P^d, R^a, P^a\}$, budget B
Initialize $LB \leftarrow \emptyset, globalLB \leftarrow -\infty, N \leftarrow \emptyset$ containing set of indices of pruned subproblems. Set ρ_0 to be a desired accuracy.
for Subproblem $\mathcal{P}_i$ **do**
  Greedy Modifications (GM):
  $\text{GM}_1 \leftarrow \bar{R}_i^a = R_i^a + B$
  $\text{GM}_2 \leftarrow \bar{P}_i^a = \min\{P_i^a + B, 0\}, \bar{R}_i^a = \max\{R_i^a, R_i^a + B + P_i^a\}$
  $LB_i \leftarrow \max_{j\in\{1,2\}} \text{ORIGAMI-BS}(GM_j)$
end for
$globalLB \leftarrow \max_{i\in[n]} LB_i$
Sort $\mathcal{P}_i$ in decreasing LB_i.
for sorted $\mathcal{P}_i$ **do**
  Overuse Modifications (OM): $\forall j \neq i, \bar{R}_j^a = \max\{0, R_j^a - B\}, \bar{P}_j^a = \min\{P_j^a, P_j^a - B + R_j^a\}$
  $\text{OM}_1 \leftarrow \bar{R}_i^a = R_i^a + B.$
  $\text{OM}_2 \leftarrow \bar{P}_i^a = \min\{P_i^a + B, 0\}, \bar{R}_i^a = \max\{R_i^a, R_i^a + B + P_i^a\}$
  $UB_i \leftarrow \min_{j\in\{1,2\}} \text{ORIGAMI-BS}(\text{OM}_j)$
  if $UB_i \leq globalLB$ **then**
    Prune $\mathcal{P}_i$
  else
    run MILP of $\mathcal{P}_i$ with additional constraint $c_i \leq \frac{globalLB - P_i^d}{R_i^d - P_i^d}$
  end if
end for
Output: Best solution among $globalLB$ and all $\mathcal{P}_i$'s.

Proof. Let t and Γ be the attack target and the attack set in the optimal solution. Let $V = \{k : k \in \Gamma \backslash \{t\}, \epsilon_k \neq 0 \text{ or } \delta_k \neq 0\}$. Since B is limited, either R_t^a or P_t^a is unchanged according to Condition 3 of Property 1. Further, with uniform cost, it is easy to show $\forall j \in T, \epsilon_j > 0$ only if $c_j \leq 1/2$. Below we assume all manipulations happen on attacker's reward; other cases also hold due to symmetry. We may assume targets in V are sorted in increasing order by the value of $(1-c_k)\epsilon_k$. First, we increase ϵ_t by $\Delta\epsilon_t = \frac{\epsilon_1(1-c_1)}{1-c_t}$. Then, for each target $j \in V$, we decrease ϵ_j by some $\Delta\epsilon_j$ such that j's utility increases to be the same as target t, i.e. $\Delta\epsilon_j(1-c_j) = \Delta\epsilon_t(1-c_t)$. This is possible if $\sum_{j\in V} \Delta\epsilon_j \geq \Delta\epsilon_i$. To show this, we lower bound $\sum_{j\in V} \Delta\epsilon_j$ as follows:

$$\sum\nolimits_{j\in V} \Delta\epsilon_j = \Delta\epsilon_t(1-c_t) \sum\nolimits_{j\in V} \frac{1}{1-c_j} \tag{22}$$

$$\geq \Delta\epsilon_t(1-c_t)\,|V| \tag{23}$$

where inequality (23) follows since $1 - c_j \leq 1$. Recall we assume $\epsilon_t > 0$, which implies $c_t \leq \frac{1}{2}$, then if $|V| \geq 2$, we have $\sum_{j\in V} \Delta\epsilon_j \geq \Delta\epsilon_t$. This process stops when $|V| \leq 1$, i.e. when the payoff of at most one non-attack target in the attack set is manipulated.

Note the assumption $B \leq \min_{j\in T} \left|P_j^a\right|$ is needed when the optimal solution increases the attack target's penalty.

We remark the above theorem is tight, i.e. there exists an instance where two targets are manipulated. See Appendix for details. When $B \leq \min_{j\in T} \left|P_j^a\right|$ and $\mu_j = \theta_j = 1, \forall j \in T$, Thm. 2 naturally suggests a PTAS – we can use linear search for manipulations on all pairs of targets as shown in Alg. 2, where e_i is a unit vector with a single one at position i . Thm. 3 shows the approximation guarantee, with a proof similar to Thm. 1 (full proof included in Appendix).

Algorithm 2 PTAS for limited budget and uniform costs in L^1

Input: Payoffs $\{R^d, P^d, R^a, P^a\}$, budget B, tolerance η.
Initialize $M \leftarrow -\infty$
for all ordered pairs of targets (i, j) **do**
  for $s = 0, 1, \ldots, \lfloor B/\eta \rfloor$ **do**
    $M \leftarrow \max\{M, \text{ORIGAMI-BS}(R^d, P^d, R^a + se_i - (B-s)e_j, R^d)\}$
    $M \leftarrow \max\{M, \text{ORIGAMI-BS}(R^d, P^d, R^a + se_i, R^d - (B-s)e_j)\}$
    $M \leftarrow \max\{M, \text{ORIGAMI-BS}(R^d, P^d, R^a, R^d + se_i - (B-s)e_j)\}$
    $M \leftarrow \max\{M, \text{ORIGAMI-BS}(R^d, P^d, R^a - (B-s)e_j, R^d + se_i)\}$
  end for
end for
Output: M

Theorem 3. *Alg. 2 returns an additive* $\max_{i\in[n]} \frac{2\eta(R_i^d - P_i^d)}{R_i^a}$ *approximate solution.*

4 Optimizing Payoff with Budget Constraint in Other Forms

In this section, we explore budget constraints in other forms and show polynomial time algorithms correspondingly.

4.1 Weighted L^∞-norm Form

When the budget constraint is in weighted L^∞-norm form, the defender can make changes to R^a and P^a for every target to the extent specified by B_i^r and B_i^p respectively. Equivalently, the defender can choose R^a and P^a from a given range. A real-world setting for this problem is when a higher level of authority specifies a range of penalty for activities incurring pollution and allow the local agencies to determine the concrete level of penalty for different activities.

We observe that Condition 2 of Property 1 still holds in this setting. Therefore, such problem can be solved by simply solving n subproblems. In the i^{th} subproblem which assumes i is the attack target, set reward and penalty of i to be the upper bound in the given range and choose the lower bound for other targets. With our improved ORIGAMI-BS algorithm, this problem can be solved in $O(n^2 \log n)$ time.

Theorem 4. *With budget constraint in weighted* L^∞*-norm, solving for defender's optimal strategy reduces to solving for defender's optimal coverage in fixed-payoff security games.*

4.2 L^0-norm Form

When the budget constraint is in L^0-norm form, the defender needs to choose which targets to make changes, while the amount of change can be unconstrained. Notice that allowing the defender to arbitrarily modify the attacker's reward R^a is not practical and will lead to a trivial solution: the defender will place all coverage on one attack target $t = \arg\max_i R_i^d$ and set $R_t^a = \infty$. So we only

allow the defender to change P^a to $\bar{P}^a$ and write $\bar{R}^a = R^a$. This corresponds to the domains where the defender can make some of the targets special. For example, in wildlife protection, legislators can designate some areas as "core zones", where no human activity is allowed and much more severe punishment can be carried out. But the defender cannot set all the areas to be core zones.

We assume the defender has a budget which allows her to change the penalty of B targets. Similar to L^∞ case, we first observe that the defender will choose an extreme penalty value once he decides to change the penalty of a target.

Property 2. There exists an optimal solution where either $\bar{P}^a_j = -\infty$ for B targets or $\bar{P}^a_j = -\infty$ for $(B-1)$ targets and $\bar{P}^a_t = 0$ for 1 target. If $\bar{P}^a_j = -\infty$, then $c_j = 0$.

Proof. When t is the attack target, the defender would like to maximize $\bar{P}^a_t$ and minimize $\bar{P}^a_j$ for all $j \neq t$. If $\bar{P}^a_j = -\infty$ and $c_j > 0$, target j will not be attacked as $U^a_j = -\infty$. In such case, target j is effectively removed from the game.

The defender's problem becomes non-trivial when the budget $B < T$, and we now provide a $O(n^3)$ algorithm for solving this problem. We note that several intuitive greedy algorithms do not work, even in more restrictive game settings. A detailed comparison of our algorithm, several greedy algorithms, and a baseline MILP is provided in Section 5.

First, we sort the targets in decreasing attacker's reward R^a_k. Let $E_k = \frac{1}{R^a_k - P^a_k}$ and $\bar{E}_k = \frac{1}{\bar{R}^a_k - \bar{P}^a_k}$ for all $k \in T$. When i is the attack target, by Property 2, we have $\bar{E}_i \in \{1/R^a_i, E_i\}$ and $\bar{E}_j \in \{0, E_j\}$. Let $\Gamma_l = \{1, 2, \ldots, l\}$ for $l = 1, 2, \ldots, n$. The ORIGAMI algorithm implies that in a fixed-payoff game, the defender's optimal strategy can be found by assuming each of Γ_l *is* the attack set [13]. We leverage this idea by noting that one of the Γ_l's *contains as a subset* the attack set in the optimal solution to our problem, i.e. after some targets are removed or set $\bar{P}^a = 0$. This allows us to formally define a subproblem $Q_{l,i}$: assume (i) the optimal attack set is contained in Γ_l, (ii) the attack target is $i \in \Gamma_l$, and (iii) no target is covered with certainty, what is the defender's optimal strategy $(c, \bar{E})$? A subproblem may be infeasible. First, we show that $Q_{l,i}$ can be solved in $O(n)$ time. From Equation 21, for subproblem $Q_{l,i}$, we have

$$\frac{c_i}{\bar{E}_i} = \frac{\sum_{k \in \Gamma_l \setminus \{i\}} (R^a_i - R^a_k)\bar{E}_k + r}{\bar{E}_i + \sum_{k \in \Gamma_l \setminus \{i\}} \bar{E}_k} \tag{24}$$

Let $s = \min\{B, l-2\}$ if $\bar{E}_i = E_i$ and $s = \min\{B-1, l-2\}$ if $\bar{E}_i = \frac{1}{R_i}$. Then $Q_{l,i}$ reduces to finding s out of the $(l-1)$ $\bar{E}_k$'s to set to 0, and set the rest $\bar{E}_k = E_k$, so as to maximize the above quotient. Below we show that finding the optimal $\bar{E}$ is equivalent to a problem of choosing subsets with maximum weighted average.

Proposition 1. *[8] Given a set S where $|S| = n$, real numbers $\{v_k : k \in S\}$, positive weights $\{w_k : k \in S\}$, and an integer r. Among all subsets of S of order $n - r$, a subset $T \subset S$ maximizing $A(T) = \frac{\sum_k v_k}{\sum_k w_k}$ can be found in $O(n)$ time.*

Algorithm 3 Algorithm for budget in L^0-norm form

Input: Payoffs $\{R^d, P^d, R^a, P^a\}$, budget B
1: Initialize $U^d(1..n, 1..n) \leftarrow -\infty$.
2: **for** attack set Γ_l **do**
3: **for** attack target $i \in \Gamma_l$ **do**
4: $\{Value, \Gamma_l^{drop}, \Gamma_l^{keep}\} \leftarrow \text{Random}(\langle v, w\rangle, s)$
5: Set $\bar{E}_j \leftarrow 0$ for $j \in \Gamma_l^{drop}$, $\bar{E}_j \leftarrow E_j$ for $j \in \Gamma_l^{keep}$.
6: Update $U^d(l, i)$ if solution is valid
7: Repeat inner iteration with $s \leftarrow \min\{B-1, l-2\}$ and $\bar{E}_i \leftarrow 1/R_i^a$.
8: **end for**
9: **end for**
10: **for** target k with largest s P_k^a **do**
11: **for** attack target i **do**
12: Update $U^d(l, i)$ if r big enough for $c_k = 1$
13: **end for**
14: **end for**
15: **Output:** $\max(U^d)$

Lemma 1. *The subproblem $Q_{l,i}$ can be solved in $O(n)$ time.*

Proof. Consider Equation 24. We equate $v_k = (R_i - R_k)E_k + \frac{m}{l-s-1}$ and $w_k = E_k + \frac{1}{l-s-1}\bar{E}_i$. Let $v = \{v_k : k \in \Gamma_l \backslash \{i\}\}$, $w = \{w_k : k \in \Gamma_l \backslash \{i\}\}$. By Property 2, we may assume s targets will be removed. Finding a subset $T \subset \Gamma_l\{i\}$, $|T| = l-s-1$, to maximize $A(T)$ is equivalent to our problem to maximize the quotient in Equation 24.

After we find the optimal choices for the s targets, we need to verify on Line 6 of Algorithm 3 that the attack set is valid. Since $c_k = 0$ for $k \notin \Gamma_l$, we need $M \geq R_{l+1}^a$, with valid coverage probabilities c_j's. These could have been violated by setting several $\bar{P}_i$'s to $-\infty$.

We are now ready to show the main result of this section.

Theorem 5. *There is a $O(n^3)$ algorithm for finding the optimal defender strategy with budget constraint in L^0-norm.*

Proof (Proof sketch). Consider Alg. 3. Since R^a is fixed, Γ_l's cover all the attack sets that need to be checked. There are n^2 subproblems $Q_{l,i}$. For each $Q_{l,i}$, we run a randomized algorithm for the maximum weighted average problem with expected running time $O(n)$ (Line 4). A deterministic $O(n)$ algorithm exists in [8]. The subproblems $Q_{l,i}$ miss the solutions where some target j is covered with certainty. In this case, Γ_n is the only possible attack set, and the solution is found on Lines 10-14. A solution is feasible if by removing targets we can keep the sum of coverage probabilities below the defender's resources.

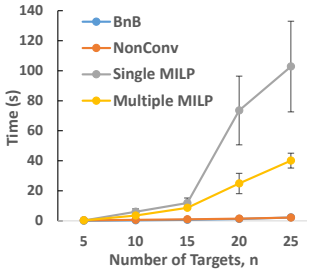
5 Experimental Results

5.1 Simulation results for L^1 budget problem

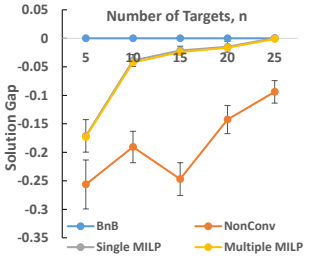
We compare our branch-and-bound (BnB) algorithm (Alg. 1) with three baseline algorithms – NonConv, multiple MILP and single MILP shown in Appendix. NonConv refers to solving n non-convex optimization problems as shown in 2-8

using IPOPT [25] solver with default parameter setting, which converges to local optima with no global optimality guarantee. Multiple MILP and single MILP formulations are equivalent and have an approximation guarantee specified in Thm. 1. The original payoff structures are randomly generated integers between 1 and $2n$ with penalties obtained by negation (recall n is the number of targets). Budget and weights of manipulations are randomly generated integers between 1 and $4n$.

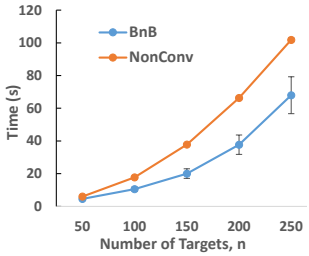
We set $\rho_0 = \min_{i \in T} \frac{R_i^a}{4(R_i^d - P_i^d)}$ which gives an additive $\frac{1}{2}$-approximate solution. Gurobi is used for solving MILPs, which is terminated when either time limit (15 min) or optimality gap (1%) is achieved. For each problem size, we run 60 experiments on a PC with Intel Core i7 processor. The solution quality of a particular algorithm is measured by the multiplicative gap between that algorithm and BnB, i.e. $\frac{Z_A - Z_{BnB}}{Z_{BnB}}$ where Z_A is best solution value by algorithm A. Thus a positive (negative) gap indicates better (worse) solution value than BnB. We report mean and standard deviation of the mean of runtime and solution quality in Fig. 1. Small instances refer to problem size from 5 to 25. Large instances refer to problem size from 50 to 250.



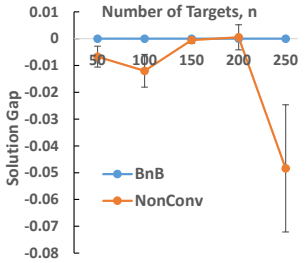
(a) Runtime, small instances



(b) Gap, small instances



(c) Runtime, large instances



(d) Gap, large instances

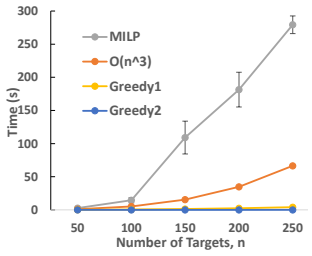
Fig. 1: Runtime and solution quality for L^1 case with standard deviation of the mean shown as vertical line

For problems of small size (Fig. 1a and 1b), BnB finds better solutions in nearly the same time as NonConv, faster than the other two. Since budget size can easily be indivisible by ρ_0 which is the atomic change we can make, greedy manipulation cannot be achieved by MILPs when such indivisibility happens. On the other hand, BnB first computes a global lower bound using such greedy

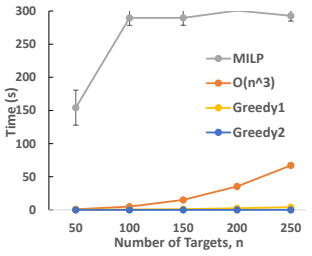
manipulations, thus creating a gap between BnB and other two MILP-based algorithms. Indeed the multiplicative gap between greedy solution and optimal solution is reported as 0.39% with a variance of 0.14%. For problems of large size (Fig. 1c and 1d), we only compare BnB and NonConv as other two algorithms timed out in solving MILP. BnB runs faster than NonConv. It returns better solutions for three problem sizes and nearly the same solution for other two cases. MILP-based solution including BnB also has a larger standard deviation in runtime than NonConv.

5.2 Simulation results for L^0 budget problem

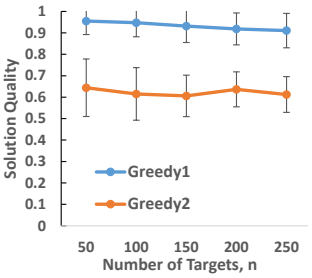
We compare the performance of our $O(n^3)$ algorithm with a baseline MILP and two greedy algorithms. Greedy1 uses ORIGAMI to remove one target at a time. Greedy2 starts from the target with highest $|P^d|$ and determines whether to remove it based on ORIGAMI. Details of these algorithms are in Appendix.



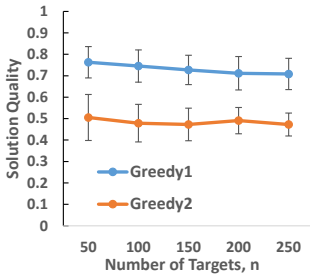
(a) Resource $r = 1$



(b) Resource $r = n/10$



(c) Resource $r = 1$



(d) Resource $r = n/10$

Fig. 2: Runtime and solution quality for L^0 case averaged over 22 trials. MILP has a time limit of 300 seconds. The error bars are standard deviations of the mean.

Initial payoffs are generated in the same way as in previous subsection. In Fig. 2a, we assume the defender has $r = 1$ resource and budget $B = n/2$, the worst case for the $O(n^3)$ algorithm. The runtime of MILP starts to explode with more than 100 targets, while the $O(n^3)$ algorithm solves the problem rather efficiently. We also note that MILP exhibits high variance in runtime. The variances of other algorithms, including the $O(n^3)$ algorithm, are relatively trivial and thus not plotted. We then test the algorithms with multiple defender resources, as

shown in Fig. 2b. With n targets, we assume the defender has $r = n/10$ units of resources and a budget $B = n/2$. Most MILP instances reach the time limit of 5 minutes when $n \geq 100$. Yet the $O(n^3)$ algorithm's runtime is almost the same as the single resource case.

Our $O(n^3)$ algorithm and MILP are guaranteed to provide the optimal solution. In contrast, the greedy algorithms exhibit fast runtime but provide no solution guarantee. We measure the solution quality in Fig. 2c and 2d using $\frac{U^d_{\text{greedy}} - p}{U^d_{\text{opt}} - p}$ where $p = \min_j P^d_j$. Greedy1, which runs slightly slower than Greedy2, achieves higher solution quality but both greedy algorithms can lead to a significant loss. In fact, extreme examples exist and are shown in Appendix.

References

1. Basilico, N., Lanzi, A., Monga, M.: A security game model for remote software protection. In: ARES '16. pp. 437–443 (Aug 2016). https://doi.org/10.1109/ARES.2016.96
2. Blocki, J., Christin, N., Datta, A., Procaccia, A.D., Sinha, A.: Audit games. In: IJCAI '13. pp. 41–47 (2013), http://dl.acm.org/citation.cfm?id=2540128.2540137
3. Blocki, J., Christin, N., Datta, A., Procaccia, A.D., Sinha, A.: Audit games with multiple defender resources. In: AAAI'15. pp. 791–797 (2015), http://dl.acm.org/citation.cfm?id=2887007.2887117
4. Blum, A., Haghtalab, N., Procaccia, A.D.: Learning optimal commitment to overcome insecurity. In: NIPS. pp. 1826–1834 (2014)
5. Conitzer, V., Sandholm, T.: Computing the optimal strategy to commit to. In: EC '06. pp. 82–90 (2006). https://doi.org/10.1145/1134707.1134717, http://doi.acm.org/10.1145/1134707.1134717
6. Durkota, K., Lisỳ, V., Bošanskỳ, B., Kiekintveld, C.: Approximate solutions for attack graph games with imperfect information. In: GameSec. pp. 228–249. Springer (2015)
7. Durkota, K., Lisỳ, V., Bosanskỳ, B., Kiekintveld, C.: Optimal network security hardening using attack graph games. In: IJCAI. pp. 526–532 (2015)
8. Eppstein, D., Hirschberg, D.S.: Choosing subsets with maximum weighted average. J. Algorithms **24**(1), 177–193 (1997). https://doi.org/10.1006/jagm.1996.0849, http://dx.doi.org/10.1006/jagm.1996.0849
9. Fang, F., Nguyen, T.H., Pickles, R., Lam, W.Y., Clements, G.R., An, B., Singh, A., Tambe, M., Lemieux, A.: Deploying paws: Field optimization of the protection assistant for wildlife security. In: AAAI'16. pp. 3966–3973 (2016), http://dl.acm.org/citation.cfm?id=3016387.3016464
10. Fujishima, Y., Leyton-Brown, K., Shoham, Y.: Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. In: IJCAI'99. pp. 548–553 (1999), http://dl.acm.org/citation.cfm?id=1624218.1624297
11. Kang, X., Wu, Y.: Incentive mechanism design for heterogeneous peer-to-peer networks: A stackelberg game approach. IEEE Transactions on Mobile Computing **14**(5), 1018–1030 (2015)
12. Kiekintveld, C., Islam, T., Kreinovich, V.: Security games with interval uncertainty. In: AAMAS '13 (2013), http://dl.acm.org/citation.cfm?id=2484920.2484959
13. Kiekintveld, C., Jain, M., Tsai, J., Pita, J., Ordóñez, F., Tambe, M.: Computing optimal randomized resource allocations for massive security games. In: AAMAS '09. pp. 689–696 (2009), http://dl.acm.org/citation.cfm?id=1558013.1558108

14. Kiekintveld, C., Lisỳ, V., Píbil, R.: Game-theoretic foundations for the strategic use of honeypots in network security. In: Cyber Warfare, pp. 81–101. Springer (2015)
15. Kiekintveld, C., Marecki, J., Tambe, M.: Approximation methods for infinite bayesian stackelberg games: Modeling distributional payoff uncertainty. In: AAMAS'11 (2011)
16. Korzhyk, D., Conitzer, V., Parr, R.: Complexity of computing optimal stackelberg strategies in security resource allocation games. In: AAAI'10. pp. 805–810 (2010), http://dl.acm.org/citation.cfm?id=2898607.2898737
17. Laszka, A., Vorobeychik, Y., Fabbri, D., Yan, C., Malin, B.: A game-theoretic approach for alert prioritization. In: AAAI-17 Workshop on Artificial Intelligence for Cyber Security (AICS) (2017), http://cps-forces.org/pubs/248.html
18. Letchford, J., Conitzer, V.: Solving security games on graphs via marginal probabilities. In: Twenty-Seventh AAAI Conference on Artificial Intelligence (2013)
19. Letchford, J., Conitzer, V., Munagala, K.: Learning and approximating the optimal strategy to commit to. In: Mavronicolas, M., Papadopoulou, V.G. (eds.) Algorithmic Game Theory. pp. 250–262. Springer Berlin Heidelberg (2009)
20. Myerson, R.B.: Mechanism design. The New Palgrave: Allocation, Information, and Markets pp. 191–206 (1989)
21. Paruchuri, P., Pearce, J.P., Marecki, J., Tambe, M., Ordonez, F., Kraus, S.: Playing games for security: An efficient exact algorithm for solving bayesian stackelberg games. In: AAMAS '08. pp. 895–902 (2008), http://dl.acm.org/citation.cfm?id=1402298.1402348
22. Píbil, R., Lisỳ, V., Kiekintveld, C., Bošanskỳ, B., Pěchouček, M.: Game theoretic model of strategic honeypot selection in computer networks. In: International Conference on Decision and Game Theory for Security. pp. 201–220. Springer (2012)
23. Schlenker, A.: Deceiving cyber adversaries: A game theoretic approach. In: AAMAS '18 (2018)
24. Sharma, Y., Williamson, D.P.: Stackelberg thresholds in network routing games or the value of altruism. In: EC '07. pp. 93–102 (2007). https://doi.org/10.1145/1250910.1250925, http://doi.acm.org/10.1145/1250910.1250925
25. Wächter, A., Biegler, L.T.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical programming **106**(1), 25–57 (2006)
26. Wang, Z., Yin, Y., An, B.: Computing optimal monitoring strategy for detecting terrorist plots. In: AAAI'16. pp. 637–643 (2016), http://dl.acm.org/citation.cfm?id=3015812.3015907
27. Xue, Y., Davies, I., Fink, D., Wood, C., Gomes, C.P.: Behavior identification in two-stage games for incentivizing citizen science exploration. In: Rueher, M. (ed.) Principles and Practice of Constraint Programming. pp. 701–717. Springer International Publishing, Cham (2016)
28. Yang, D., Xue, G., Fang, X., Tang, J.: Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing. In: Mobicom '12. pp. 173–184 (2012). https://doi.org/10.1145/2348543.2348567, http://doi.acm.org/10.1145/2348543.2348567
29. Yin, Z., Jiang, A.X., Johnson, M.P., Kiekintveld, C., Leyton-Brown, K., Sandholm, T., Tambe, M., Sullivan, J.P.: Trusts: Scheduling randomized patrols for fare inspection in transit systems. In: IAAI (2012)
30. Yin, Z., Tambe, M.: A unified method for handling discrete and continuous uncertainty in bayesian stackelberg games. In: AAMAS '12. pp. 855–862 (2012)