Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Non-Additive Security Games

Sinong Wang, Fang Liu
Department of ECE,
The Ohio State University
Columbus, OH 43210, USA
{wang.7691, liu.3977}@osu.edu

Ness Shroff
Departments of ECE and CSE,
The Ohio State University
Columbus, OH 43210, USA
shroff.11@osu.edu

Abstract

Security agencies have found security games to be useful models to understand how to better protect their assets. The key practical elements in this work are: (i) the attacker can simultaneously attack multiple targets, and (ii) different targets exhibit different types of dependencies based on the assets being protected (e.g., protection of critical infrastructure, network security, etc.). However, little is known about the computational complexity of these problems, especially when there exist dependencies among the targets. Moreover, previous security game models do not in general scale well. In this paper, we investigate a general security game where the utility function is defined on a collection of subsets of all targets, and provide a novel theoretical framework to show how to compactly represent such a game, efficiently compute the optimal (minimax) strategies, and characterize the complexity of this problem. We apply our theoretical framework to the *network security game*. We characterize settings under which we find a polynomial time algorithm for computing optimal strategies. In other settings we prove the problem is NP-hard and provide an approximation algorithm.

Introduction

The nature of resource allocation in practical *security games* often results in exponentially many pure strategies for the defender, such that the defender's optimal mixed strategy is hard to solve. In the past few years, several works have tried to resolve this issue from both theoretical and practical perspectives (Kiekintveld et al. 2009; Korzhyk, Conitzer, and Parr 2010; Jain et al. 2011; Letchford and Conitzer 2013; Xu et al. 2014; Xu 2016). A common restriction in these works is to either assume that the attacker only attacks one target, or that different targets are independent. The latter implies that the payoff of a group of targets is the sum of the payoffs of each one (Korzhyk, Conitzer, and Parr 2011). In practice, there exists various dependencies among the targets such that attacking one target will influence the others. Traditional models that ignore the inherent synergistic effects among the targets could lead to catastrophic consequences (Buldyrev et al. 2010). Motivated by this phenomenon, some recent works have investigated the security game with dependent targets (Shakarian, Lei, and Lindelauf 2014; Vorobeychik and Letchford 2015).

However, these works are limited to specific dependencies and provide neither a systematic understanding of complexity properties, nor an efficient algorithm. For example, Shakarian et al. (2014) assumes that the attacker and defender can choose a subset of nodes in a power grid and their utilities are dependent on the set of disconnected loads. They show that the defender best response problem (DBR) can be solved in polynomial time if the attacker attacks at most one target, while NP-hard in other cases. However, their complexity results cannot be easily reduced to the complexity of determining the defender's mixed strategies.

In this paper, we introduce a new security game, which we call the Non-additivity Security Game (NASG). It is a non-zero-sum game including two players - the *defender* and *attacker*, and n targets, denoted by $[n] = \{1, 2, \ldots, n\}$. We model various dependencies among the targets by defining the strategy of each player as a subset of $[n]$ and adopt a general set function as the utilities. Specifically, the attacker obtains benefits for successfully attacked targets and pays a cost for its strategy. Also, the defender will lose benefits for those targets and also pays a cost. A critical feature of the NASG is that *the benefit and cost for several targets is not the summation of each target's utility, instead, it is dependent on the specific combination of targets.*.

At a high level, the main challenge of NASG is that both the size of the strategy space and the number of utility functions are $\Theta(2^n)$. We are interested whether the following well understood questions in the case of additive utility functions can be addressed under the non-additivity assumption.

- How to **compactly** represent the NASG and how to **efficiently compute** the mixed strategies of NASG?
- What is the **complexity** of computing the mixed strategies of NASG?

To answer these questions, we make the following contributions: (1) We provide the conditions for compactly representing the NASG and prove that there exists poly(n) number of variables in the compact model if the number of non-additive utility functions is poly(n). The main technique is isomorphism and projection of a polytope. (2) We design an algorithmic framework to efficiently compute the mixed strategies for NASGs by reducing the original problem to an oracle problem. The main technique is to design a polynomial-time vertex mapping algorithm from the low-

dimensional polytope to a simplex; (3) We prove that the above oracle problem and the computation of mixed strategies of NASG can be reduced to each other in polynomial-time under a reasonable restriction. Furthermore, we show that such an oracle problem is a problem of maximizing a *pseudo-boolean function*; (4) Finally, we apply our theoretical framework to the *network security game*. We provide polynomial-time algorithms for some kinds of networks and security measures, while for the general case, we show the NP-hardness and propose an approximation algorithm.

All the proofs in this paper are left to the supplemental material due to space constraints.

Problem Description and Preliminary

We begin by defining the NASG as a two-player normal-form non-zero-sum game.

Players and **targets:** The NASG contains two players (a *defender* and an *attacker*), and n targets, indexed by set $[n] \triangleq \{1, 2, \ldots, n\}$.

Strategies and utility functions: A *pure* strategy for each player is a subset of $[n]$. In the general case, we consider the *complete pure strategy space* of attacker and defender, defined as the power set $2^{[n]} \triangleq \{V | V \subseteq [n]\}$, denoted by $\mathcal{A}$ and $\mathcal{D}$, respectively. So there are $N \triangleq 2^n$ pure strategies for both players. Let **set function** $C_a(\cdot) : \mathcal{A} \to \mathbb{R}$ and $C_d(\cdot) : \mathcal{D} \to \mathbb{R}$ be the attacker's and defender's cost function, respectively, and the set function $B(\cdot) : \mathcal{A} \to \mathbb{R}$ be the benefit function.

Remark 1. *Traditional models do not consider a cost function, instead, they assume that there exists a resource constraint such that certain strategies, i.e., subsets of $[n]$, are restricted. In our paper, we explicitly consider the cost function but do not have such resource constraints*[1]*. In cyber-security applications, security resources are available for a cost and can be used to replace resource constraints, as illustrated in (Vorobeychik and Letchford 2015).*

Tie-breaking Rule: When the attacker and defender choose strategy $A \in \mathcal{A}$ and $D \in \mathcal{D}$, targets in the set $A \backslash D$ are successfully attacked by the attacker. Moreover, both players pay the cost for their strategy, and the attacker's and defender's payoff is given by $[B(A \backslash D) - C_a(A)]$ and $[-B(A \backslash D) - C_d(D)]$, respectively.

Normal-form representation: Suppose that the order of the attacker's pure strategy is given by index function $\sigma(\cdot) : \mathcal{A} \to \{1, 2, \cdots, N\}$, then define the index function $\mu(\cdot)$ for the defender's pure strategy: $\mu(U) = \sigma(U^c)$ for any $U \in \mathcal{D}$. This definition of the index function is to guarantee the symmetry of benefit matrix, which simplifies most theoretical results. Note that this assumption can be generalized, but makes the notation cumbersome. Then we can define the utility matrices including the cost matrices of attacker and defender: $\mathbf{C}^A, \mathbf{C}^D \in \mathbb{R}^{N \times N}$,

$$\mathbf{C}^A_{\sigma(A),\mu(D)} = C_a(A), \mathbf{C}^D_{\sigma(A),\mu(D)} = C_d(D), \forall A, D \in 2^{[n]},$$

and the benefit matrix $\mathbf{M} \in \mathbb{R}^{N \times N}$,

$$\mathbf{M}_{\sigma(A),\mu(D)} = B(A \backslash D), \forall A, D \in 2^{[n]}.$$

[1]Later, in this paper, we will consider the limited resource.

Let $\mathbf{M^a}$ and $\mathbf{M^d}$ be the attacker's and defender's payoff matrices. It's clear that $\mathbf{M^a} = \mathbf{M} - \mathbf{C}^A$ and $\mathbf{M^d} = -\mathbf{M} - \mathbf{C}^D$. The *mixed* strategy $\mathbf{p}, \mathbf{q} \in \Delta_N$ is a distribution over the set of pure strategies $\mathcal{A}, \mathcal{D}$, where $\mathbf{p}_{\sigma(A)}, \mathbf{q}_{\mu(D)}$ is the probability that the attacker chooses strategy A and the defender chooses strategy D. Δ_N represents a N-dimensional simplex. Then the expected payoffs for the attacker and defender is given by following bilinear form, when they play the mixed strategy $\mathbf{p} \in \Delta_N$ and $\mathbf{q} \in \Delta_N$, by

$$U_a(\mathbf{p}, \mathbf{q}) = \mathbf{p}^T \mathbf{M^a} \mathbf{q} \quad \text{and} \quad U_d(\mathbf{p}, \mathbf{q}) = \mathbf{p}^T \mathbf{M^d} \mathbf{q}.$$

Solution Concepts: In this paper, we assume that both players move simultaneously and the standard solution concept is the *Nash Equilibrium (NE)*. Our goal is to compute the defender's minimax mixed strategies and we call it the **min max problem**.

The following three definitions are used heavily in our theoretical development.

Definition 1. *The common utility function is defined as the following transform of the benefit function $B(\cdot)$, cost function $C_a(\cdot)$ and $C_d(\cdot)$ for all $U \in 2^{[n]}$,*

$$B^c(U) = \sum_{V \subseteq U} (-1)^{|U \backslash V|} B(V),$$
$$C^c_a(U) = \sum_{V \subseteq U} (-1)^{|U \backslash V|} C_a(V),$$
$$C^c_d(U) = \sum_{V : U \subseteq V} (-1)^{|V \backslash U|} C_d(V).$$

Intuitively, if we regard the benefit function as a measure defined on a given algebra of set $[n]$, then using the inclusion-exclusion principle to expand each term in the summand, the common utility is equivalent to measuring the utility of the intersection of the targets, which seems like measuring the synergy effect of targets.

Definition 2. *The support set of NASG is*

$$S = \{U \in \mathcal{A} | B^c(U) \text{ or } C^c_a(U) \text{ or } C^c_d(U^c) \neq 0\}, \quad (1)$$

and support index set $\sigma(S) = \{\sigma(U) | U \in S\}$.

Definition 3. *The projection operator $\pi_S : \mathbb{R}^N \to \mathbb{R}^{|S|}$ is*

$$\pi_S((\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)) = (\ldots, \mathbf{x}_i, \ldots)_{i \in \sigma(S)}, \quad (2)$$

and projection of polytope: $\Pi_S(\Delta_N) \triangleq \{\pi_S(\mathbf{x}) | \mathbf{x} \in \Delta_N\}$.

Strategically Zero-sum Form

Although the NASG contains non-zero-sum payoff, we prove the following proposition, which shows that it belongs to the strategically zero-sum game (Moulin and Vial 1978).

Proposition 1. *The set of Nash equilibriums of NASG is equivalent to the set of Nash equilibriums of zero-sum game with payoff matrix $\mathbf{M} - \mathbf{C^A} + \mathbf{C^D}$.*

Clearly, the Stackelberg equilibrium set is equivalent to the NE set of the NASG. This proposition allows us to solve the NASG via the equivalent zero-sum game, which can be tackled by a linear programming approach. In the sequel, we use $\mathbf{M}^\circ = \mathbf{M} - \mathbf{C^A} + \mathbf{C^D}$ to denote the payoff matrix.

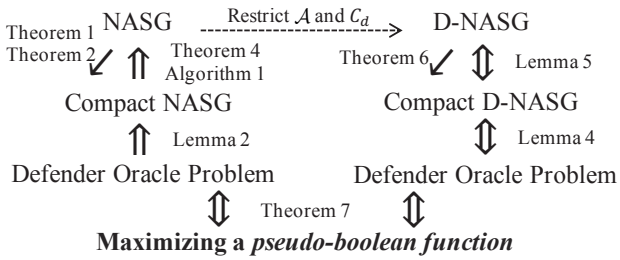

Figure 1: The summary of the main results. The double arrow denotes the polynomial time reduction. The single arrow denotes the compact representation

Remark 2. *The traditional zero-sum security game (Xu 2016) assumes that the defender gets a reward r_i if target i is covered, or incurs a cost c_i if uncovered. We can set specific values for our utility functions to recover their setting.*

The main results of this paper are summarized in Fig. 1.

The Compact Representation of NASG

Based on the equivalent zero-sum game $\mathbf{M}^\circ$ and von Neumann's minimax theorem, computing the NE of NASG can be formulated as the following min max problem,

$$\min_{\mathbf{q}\in\Delta^N}\max_{\mathbf{p}\in\Delta^N}\mathbf{p}^T\mathbf{M}^\circ\mathbf{q}. \tag{3}$$

This optimization model has 2^{n+1} variables, which implies that NASG is in general hard to solve. The goal of this section is to find a condition on the NASG that can be compactly represented with only poly(n) variables. To convey our idea more easily, we begin with following intuitive example.

Motivating Example

We first use gauss elimination of the matrix $\mathbf{M}^\circ$ to transform it into the row canonical form, which is to left and right multiply $\mathbf{M}^\circ$ by elementary matrices $\mathbf{E}$ and $\mathbf{F}$,

$$\begin{aligned}\min_{\mathbf{q}\in\Delta_N}\max_{\mathbf{p}\in\Delta_N}\mathbf{p}^T\mathbf{M}^\circ\mathbf{q} &= \min_{\mathbf{q}\in\Delta_N}\max_{\mathbf{p}\in\Delta_N}\mathbf{p}^T\mathbf{E}^{-1}\mathbf{E}\mathbf{M}^\circ\mathbf{F}\mathbf{F}^{-1}\mathbf{q}\\ &= \min_{\mathbf{q}\in\Delta_N}\max_{\mathbf{p}\in\Delta_N}\mathbf{p}^T\mathbf{E}^{-1}\begin{bmatrix}\mathbf{M}_r^\circ & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\mathbf{F}^{-1}\mathbf{q}.\end{aligned}$$

where r is the rank of payoff matrix $\mathbf{M}^\circ$, and $\mathbf{M}_r^\circ$ is the non-zero block of its row canonical form. If we define the affine projection $f(\mathbf{p}) = \left(\mathbf{p}^T\mathbf{E}^{-1}\right)^T, g(\mathbf{q}) = \mathbf{F}^{-1}\mathbf{q}$, and let $\Delta_N^a = \{f(\mathbf{p})|\mathbf{p}\in\Delta_N\}$, $\Delta_N^d = \{g(\mathbf{q})|\mathbf{q}\in\Delta_N\}$, we can obtain the following optimization problem,

$$\min_{\mathbf{q}'\in\Delta_N^d}\max_{\mathbf{p}'\in\Delta_N^a}\mathbf{p}'^T\begin{bmatrix}\mathbf{M}_r^\circ & \mathbf{0}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\mathbf{q}'. \tag{4}$$

Since the polyhedra Δ_N^a and Δ_N are isomorphic, their vertices exhibit one-one correspondence. Similar argument for the polyhedra Δ_N^d and Δ_N. Thus the optimization problem (3) and (4) are equivalent. Further, considering the fact that only the first r elements in vector $\mathbf{p}'$ and $\mathbf{q}'$ have the non-zero coefficients in (4), we can further simplify (4) as

$$\min_{\mathbf{q}'\in\Pi_r\left(\Delta_N^d\right)}\max_{\mathbf{p}'\in\Pi_r\left(\Delta_N^a\right)}\mathbf{p}'^T\mathbf{M}_r^\circ\mathbf{q}', \tag{5}$$

where the operator $\Pi_r(\cdot)$ is to project the $N-$dimensional polytope into its first $r-$coordinates.

Remark 3. *The observation is that the number of variables in the model (5) depends on the rank of the payoff matrix. For example, if the rank of $\mathbf{M}^\circ$ is* poly(n)*, we can compactly represent NASG with only* poly(n) *variables.*

The Formal Description of Compact NASG

Although the above conceptual derivation provides a possible path to compactly represent the NASG, there exists a significant technical challenge: the elementary matrices $\mathbf{E}$, $\mathbf{F}$ and their inverse matrices have exponential size, hence, the key question is whether we can **find both elementary matrices** efficiently? To tackle this problem, we first show that the payoff matrix $\mathbf{M}^\circ$ can be decomposed as the product of three matrices. The following technical lemma is critical in our decomposition.

Lemma 1. *For all $U\in 2^{[n]}$, the utility function satisfies*

$$\begin{aligned}B(U) &= \sum_{V\subseteq U}B^c(V),\\ C_a(U) &= \sum_{V\subseteq U}C_a^c(V),\\ C_d(U) &= \sum_{V:U\subseteq V}C_d^c(V).\end{aligned}$$

Note that similar results hold for the cost functions and their common utility. Then we can decompose the payoff matrix $\mathbf{M}^\circ$ in terms of common utilities. An illustrative example can be seen in the supplemental material.

Theorem 1. *The payoff matrix $\mathbf{M}^\circ = \mathbf{M} - \mathbf{C}^A + \mathbf{C}^D$ can be decomposed as*

$$\mathbf{M}^\circ = \mathbf{Q}(\mathbf{D} - \mathbf{L} + \mathbf{V})\mathbf{Q}^T, \tag{6}$$

where $\mathbf{D}$ is the diagonal matrix with $\mathbf{D}_{\sigma(A),\sigma(A)} = B^c(A)$. $\mathbf{V}$ and $\mathbf{L}$ are two sparse matrices with non-zero elements: $\mathbf{V}_{\mu([n]),\sigma(A)} = C_d^c(A), \mathbf{L}_{\mu(D),\sigma(\{\emptyset\})} = C_a^c(D^c)$. The $\mathbf{Q}$ is binary matrix with $\mathbf{Q}_{\sigma(A),\mu(D)} = \mathbb{1}\{D^c\subseteq A\}$.

Based on this result, we can let elementary matrices $\mathbf{E} = \mathbf{Q}^{-1}$, $\mathbf{F} = (\mathbf{Q}^T)^{-1}$, affine transformation $f(\mathbf{p}) = \mathbf{Q}^T\mathbf{p}$ and $g(\mathbf{q}) = \mathbf{Q}^T\mathbf{q}$ to yield two isomorphic polytopes: Δ_N^a and Δ_N^d with Δ_N. The whole procedure is listed in Fig. 2, and the following theorem answers part of our first question.

Theorem 2. *If $|S| =$ poly(n), the rank of the payoff matrix $\mathbf{M}^\circ$ is* poly(n)*, moreover, the NASG can be compactly represented by* poly(n) *number of variables and $(\mathbf{p}^*, \mathbf{q}^*)$ is a NE of NASG if and only if $(\pi_S(f(\mathbf{p}^*)), \pi_S(g(\mathbf{q}^*)))$ is the optimal solution of (8).*

Since we do not utilize the row canonical form of $\mathbf{M}^\circ$, instead, we extract the non-zero columns and rows of $\mathbf{D}-$

The Framework of Compact Representation

Isomorphic polytope: solving the NASG is equivalent to solving the following optimization problem,

$$\min_{\mathbf{q}' \in \Delta_N^d} \max_{\mathbf{p}' \in \Delta_N^a} \mathbf{p}'^T (\mathbf{D} - \mathbf{L} + \mathbf{V}) \mathbf{q}' \tag{7}$$

Projection of polytope: projects the polytope Δ_N^a and Δ_N^d into coordinates with indices in $\sigma(S)$, and further simplify (7) as the following compact represented model,

$$\textbf{Compact NASG} \quad \min_{\mathbf{q}' \in \Pi_S(\Delta_N^d)} \max_{\mathbf{p}' \in \Pi_S(\Delta_N^a)} \mathbf{p}'^T \mathbf{M}^S \mathbf{q}' \tag{8}$$

where matrix $\mathbf{M}^S$ is a sub-matrix of $\mathbf{D} - \mathbf{L} + \mathbf{V}$, which is obtained by extracting those rows and columns whose index belonging to $\sigma(S)$.

Figure 2: The isomorphism and projection of a polytope

$\mathbf{L} + \mathbf{V}$ to form the low-dimensional matrix $\mathbf{M}^S$, the Theorem 2 provides only a sufficient condition for our compact representation. Indeed, we can make it both sufficient and necessary by further conducting elementary elimination to transform the matrix $\mathbf{D} - \mathbf{L} + \mathbf{V}$ into an approximate diagonal matrix $\mathbf{D}$. However, this process will significantly complicate our affine transformation f and g, and make it impossible to map the optimal solution of compact model (8) to the original mixed strategy.

Implication of compact NASG: From the perspective of attacker's utility function, our compact representation (8) simplifies $U_a(\mathbf{p}, \mathbf{q})$ as (similar result holds for $U_d(\mathbf{p}, \mathbf{q})$),

$$\sum_{U \in S} \mathbf{p}'_{\sigma(U)} [\mathbf{q}'_{\sigma(U)} B^c(U) - C_a^c(U)]. \tag{9}$$

Based on the definition of affine transformations f, g and matrix $\mathbf{Q}$, each variable $\mathbf{p}'_{\sigma(U)} = \sum_{V:U \subseteq V} \mathbf{p}_{\sigma(V)}$ is the marginal probability that the attacker attacks all the targets in set U, while $\mathbf{q}'_{\sigma(U)} = \sum_{V \subseteq U^c} \mathbf{q}_{\mu(V)}$ is the marginal probability that the defender does not defend any target in set U. Therefore, we can regard $B^c(U)$, $C_a^c(U)$ (not $B(U)$ and $C_a(U)$!) as the benefit and cost function for a "virtual target U", and the formula (9) calculates the expected utility in such a new game.

Theorem 8 provides a sufficient condition of which kind of non-additive security game can be compactly represented. We then discuss several important applications of this result. One application is based on the following corollary.

Corollary 1. *If all the utility functions are additive, i.e., $B(U) = \sum_{i \in U} B(\{i\})$ (similarly for cost function), the common utility satisifies $B^c(U) = 0, |U| > 1$. Further, the cardinality of support set $|S| = n$.*

Remark 4. *The previous works (Kiekintveld et al. 2009; Korzhyk, Conitzer, and Parr 2011; Korzhyk et al. 2011) assuming that the utility functions are additive have provided a compact represented game, in which the defender's mixed strategy is represented by a $n-$dimensional marginal probability vector. Corollary 1 recovers and justifies this result.*

Another application is, in the network security domain, if we adopt some classic parameters as the security measure such as node's betweenness centrality (the number of shortest paths from all vertices to all others that pass through that node), the cardinality of support set $|S| =$poly(n).

In the sequel, *the terminology NASG refers to those NASGs that only have poly(n) variables in their compact model*. Based on our compact representation, a natural question that arises is, can we efficiently solve such a compact model and implement the optimal solution by the defender's mixed strategy? We will answer this question in the next section.

Oracle-based Algorithmic Framework

The main result of this section is given in the following theorem.

Theorem 3. *There is a* poly(n) *time algorithm to solve the min max problem if there is a* poly(n) *time algorithm to compute the defender oracle problem, which is defined as, for any vector $\mathbf{w} \in \mathbb{R}^{|S|}$, compute*

$$\mathbf{x}^* = \arg \max_{\mathbf{x} \in \Pi_S(\Delta_N^d)} \mathbf{w}^T \mathbf{x}. \tag{10}$$

It is not surprising that the compact NASG can be reduced to the defender oracle problem (DOP), and the reduction follows from an application of equivalence between separation and optimization (Grötschel, Lovász, and Schrijver 1981). What is interesting, however, is the reduction from the **min max problem** to the **compact NASG**. Namely, how to map the optimal solution of compact NASG to a defender's mixed strategy in poly(n) time. We show that this can be done by exploiting the geometric structure of polytope Δ_N^d.

Reducing Compact NASG to Oracle Problem

For simplicity, we use H_a and H_d to denote the polytope $\Pi_S(\Delta_N^a)$ and $\Pi_S(\Delta_N^d)$, and I_a and I_d to denote their vertices, respectively. The compactly represented NASG (8) can be formulated as the linear programing (LP) problem.

$$\textbf{Compact-LP} \quad \min \quad u \tag{11}$$

$$s.t. \quad \mathbf{v}^T \mathbf{M}^S \mathbf{q}' \le u \quad \forall \mathbf{v} \in I_a, \quad \mathbf{q}' \in H_d. \tag{12}$$

The compact LP has poly(n) number of variables and possibly exponentially many constraints. One can therefore apply the ellipsoid method to solve such an LP, given a poly(n) time separation oracle. Further, the separation oracle can be reduced to the following two parts: given any $(\mathbf{q}', u)$, (1) *membership problem*: decide whether $\mathbf{q}' \in H^d$. If not, generate a hyperplane separating $(\mathbf{q}', u)$ and H^d; (2) *inequality constraint problem*: decide whether all the inequality constraints hold. If not, find one violated constraint. We have the following result for these problems.

Lemma 2. *The* membership problem *and* inequality constraint problem *of compact LP (11) can be reduced to the defender oracle problem (10) in* poly(n) *time.*

Algorithm 1 Vertex Mapping from Vertex to Pure Strategy

Input: Vertex $\mathbf{v}^U \in I^d$
Output: Defender's pure strategy U of original NASG
$T = \emptyset$;
for each $i \in [n]$ **do**
  if $\mathbf{v}^U_{\sigma(\{i\})} \neq 0$ **then** $T = T \cup \{i\}$;
end for
$U = T^c$;

Reducing NASG to Compact NASG

A classical result in combinatorial optimization is that if the separation problem of polytope $P \in \mathbb{R}^n$ can be solved in poly(n) time, we can decompose any point $\mathbf{x} \in P$ into the convex combination of at most $(n+1)$ vertices of P (Grötschel, Lovász, and Schrijver 1981). Note that this is precisely the DOP required for the above reduction. Applying this result to the optimal solution $\mathbf{x}^*$ of compact LP (11), we can get a convex decomposition $\mathbf{x}^* = \sum_{i=1}^{|S|+1} \lambda_i \mathbf{v}^i$, where $\mathbf{v}^i \in I_d$. If we can map the vertices $\mathbf{v}^i$ back to the vertices (pure strategy) of original NASG, denoted by $h(\mathbf{v}^i)$, the mixed strategies of the defender can be expressed as

$$\mathbf{q}^* = \sum_{i=1}^{|S|+1} \lambda_i h(\mathbf{v}^i). \tag{13}$$

Thus, the key lies in how to compute $h(\mathbf{v}^i)$ in poly(n) time.

To tackle this problem, first, considering an arbitrary pure strategy $U \in 2^{[n]}$, the corresponding vertex is a unit vector $\mathbf{e}^U \in \mathbb{R}^N$ with only one non-zero element $\mathbf{e}^U_{\mu(U)} = 1$. Based on the affine transformation $g(\mathbf{q}) = \mathbf{Q}^T\mathbf{q}$, the corresponding vertex of isomorphic polytope Δ^d_N is

$$g(\mathbf{e}^U) = \mathbf{Q}^T\mathbf{e}^U = \mathbf{Q}^T_{\mu(U)}, \tag{14}$$

where $\mathbf{Q}_{\mu(U)}$ is the $\mu(U)$th row of matrix $\mathbf{Q}$. Then the corresponding point $\mathbf{v}^U$ of the projected polytope H^d is

$$\mathbf{v}^U = \pi_S(\mathbf{Q}^T_{\mu(U)}), \tag{15}$$

which is a sub-vector of $\mathbf{Q}^T_{\mu(U)}$. The problem is that the vertex in the high-dimensional polytope may not project to a vertex of its low-dimensional image. However, the following lemma will provide a positive result.

Lemma 3. *$\forall S \subseteq 2^{[n]}$ s.t. $[n] \subseteq S$, the vertices of the polytope H_d are the rows of a sub-matrix of $\mathbf{Q}$, which is formed by extracting the column whose index belongs to $\sigma(S)$.*

No matter which coordinate we project the polytope Δ^d_N into, the number of vertices is still N, and they form a sub-matrix of $\mathbf{Q}$. Therefore, we can exploit the property of matrix $\mathbf{Q}$ to construct a vertex mapping algorithm and the correctness of Algorithm 1 is justified by following theorem.

Theorem 4. *Vertex mapping algorithm runs in $O(n)$ time and maps each vertex of H_d to a unique pure strategy.*

Solving NASG is a Combinatorial Problem

In this section, we will answer our second question, i.e., what is the complexity of the NASG, in a restrictive class: the attacker attacks at most c targets, the defender can protect at most k targets, where c is a constant and k is arbitrary; the defender's cost functions $C_d(\cdot)$ are additive. Then, the attacker's and defender's pure strategy spaces are given by $\mathcal{A} = \{A \in 2^{[n]} || A| \leq c\}, \mathcal{D} = \{D \in 2^{[n]} || D| \leq k\}$.

Definition 4. *A D-NASG is given by the tuple $(\mathcal{A}, \mathcal{D}, \mathcal{B}, \mathcal{C}^a, \mathcal{C}^d)$, where the set of benefit function $\mathcal{B} = \{B(A) | A \in \mathcal{A}\}$, set of attacker's and defender's cost function $\mathcal{C}^a = \{C_a(A) | A \in \mathcal{A}\}$, $\mathcal{C}^d = \{C_d(i) | i \in [n]\}$.*

The first assumption is motivated by the fact that both players have limited resources (Kiekintveld et al. 2009) and they cannot cover any targets. In the realistic scenario, the attacker cannot simultaneously attack a large amount of targets such that c is a constant. The second assumption is reasonable because in the security game, the synergistic effect always occurs after the attack and defense, and will not influence the defense cost. For example, in the cybersecurity game, the defender is to deploy the anti-virus software in each communication node, in which case the cost (licensing cost) of deploying multiple nodes is equal to the summation of each one. Let $N_a = |\mathcal{A}|$ and $N_d = |\mathcal{D}|$, our main result is the following theorem.

Theorem 5. *There is a* poly(n) *time algorithm to compute the defender's mixed strategy in D-NASG,* ***if and only if*** *there is a* poly(n) *time algorithm to compute the defender oracle problem: for any $\mathbf{w} \in \mathbb{R}^{N_a}$,*

$$\mathbf{x}^* = \arg\max_{\mathbf{x} \in H'_d} \mathbf{w}^T\mathbf{x}, \tag{16}$$

where the definition of H'_d is given in (18).

Reduction between D-NASG and Oracle Problem

The reduction from D-NASG to DOP still follows our isomorphism and projection framework, and the main technical step is a partial decomposition of the payoff matrix. However, the reverse direction follows from a different path. First, let the payoff matrix of D-NASG be denoted by $\mathbf{M}^b \in \mathbb{R}^{|\mathcal{A}| \times |\mathcal{D}|}$, which is a sub-matrix of $\mathbf{M}^\circ$.

Theorem 6. *The payoff matrix $\mathbf{M}^b \in \mathbb{R}^{N_a \times N_d}$ can be decomposed as*

$$\mathbf{M}^b = \mathbf{I}\mathbf{M}^{\mathcal{A}}\mathbf{J}^T, \tag{17}$$

where the matrix $\mathbf{I} \in \mathbb{R}^{N_a \times N_a}$ and $\mathbf{J} \in \mathbb{R}^{N_d \times N_a}$ are binary matrices, and the matrix $\mathbf{M}^{\mathcal{A}} \in \mathbb{R}^{N_a \times N_a}$ contains one non-zero diagonal, non-zero row and non-zero column.

The detailed definition of each elements in matrix $\mathbf{I}, \mathbf{J}$ and $\mathbf{M}^{\mathcal{A}}$ can be seen in the supplemental material. Similarly, we have the affine transformation: $f'(\mathbf{p}) = \mathbf{I}^T\mathbf{p}$, $g'(\mathbf{q}) = \mathbf{J}^T\mathbf{q}$; transformed polytope: $\Delta^a_{N_a}$, $\Delta^d_{N_d}$; projected polytope:

$$H'_a = \Pi_S(\Delta^a_{N_a}), H'_d = \Pi_S(\Delta^d_{N_d}). \tag{18}$$

Note that in this case, the polytope $\Delta^d_{N_d}$ is not isomorphic with Δ_{N_d}, but the correctness of our compact representation

follows a similar proof of Theorem 2. Further, the compactly represented linear programming is expressed as

$$\textbf{Compact-D-LP} \quad \min \quad u \tag{19}$$

$$s.t. \quad \mathbf{v}^T \mathbf{M}^{\mathcal{A}} \mathbf{q}' \leq u \quad \forall \mathbf{v} \in I'_a, \quad \mathbf{q}' \in H'_d, \tag{20}$$

where I'_a is the set of vertices of polytope H'_a.

Lemma 4. *The separation problem for H'_d and the compact optimization problem (19) reduce to each other in* poly(n) *time.*

Considering the equivalence between separation (H'_d) and optimization (DOP), we arrive at the reduction between the DOP (16) and the compact model (19). The main technique in the reduction between D-NASG and the compact optimization (19) is: (i) for any arbitrary instance $\mathbf{M}^{\mathcal{A}}$ of compact optimization problem, we can construct the set of utility functions: $\mathcal{B}, \mathcal{C}^a$ in $O(2^c n) = O(n)$ time based on Lemma 1; (ii) vertex mapping algorithm from pure strategy to H'_d.

Lemma 5. *The min max problem of D-NASG and compact optimization (19) reduces to each other in* poly(n) *time.*

Lemma 4 and Lemma 5 together yield our desired result.

What is the Defender Oracle Problem

Through a series of reductions, we find that the NASG is essentially a defender oracle problem defined on a low-dimensional polytope H'_d, but the complicated form of polytope H'_d still prevents us from uncovering how the non-additive utility function change the internal combinatorial structure of the security game. Fortunately, based on the investigation of the geometric structure of the H'_d, we will prove that the DOP is indeed a problem of *maximizing a pseudo-boolean function*.

Theorem 7. *The defender oracle problem is equivalent to, for any vector $\mathbf{w} \in \mathbb{R}^{|S|}$, maximize a pseudo-boolean function under a cardinality constraint,*

$$\max_{\sum_{i=1}^{n} \mathbf{x}_i \geq n-k, \mathbf{x} \in \{0,1\}^n} \left[\sum_{V \in S} \mathbf{w}_{\sigma(V)} \left(\prod_{\{i\} \in V} \mathbf{x}_i \right) \right]. \tag{21}$$

The complexity of (21) is dependent on the support set S. For example, in the simplest case, $S = [n]$, we can efficiently solve such a problem by summing all the positive elements of vector $\mathbf{w}$, which corresponds to the traditional additive security game. Instead, if $S = \{U \in 2^{[n]} || U| \leq 2\}$, then the oracle problem is a binary quadratic programming problem, which is known to be NP-hard. If $k = n$, the above problem will degenerate to an unconstrained optimization. This result builds a connection between the NASG and optimizating a *pseudo-boolean function*, which enables us to design an efficient DOP solver or understand the complexity of NASG via analyzing the structure of the support set S and using the results of combinatorial algorithm design.

Application to the Network Security Domain

In this section, we will apply our theoretical framework to an important domain, in which the security game occurs in a network. The following definition is motivated by the works (Gueye, Marbukh, and Walrand 2012; Shakarian, Lei, and Lindelauf 2014).

Definition 5. *A network security game is given by the tuple $(G, T, \mathbf{F_a}, c)$, where $G = (V, E)$ with node set V, edge set E, T is the network value function, $\mathbf{F_a}$ is the failure operator, c is the maximum number of nodes attacker can choose and defender can protect any targets ($k = n$).*

The network value function $T : G \to \mathbb{R}$ is a security measure assessing the utility of a network, and failure operator $\mathbf{F_a} : 2^G \to 2^G$ is to generate a new network via a specific failure mode after removing some nodes. For example, Shakarian et al. (2014) adopt the number of connected load nodes as T, and edge cascading failure model as $\mathbf{F_a}$. The main result of our work is summarized as in TABLE 1.

Table 1: Solvability Status

CASES	SOLVABILITY	APPENDIX
Additive benefit function	poly(n)	Trivial
The separable support set S with $\max_i \|U_i\| = \Theta(\log(n))$	poly(n)	Theorem 6, Corollary 1
Constant c, negative common utilities except for singleton set	poly(n)	Theorem 7, Corollary 2
Constant $c \geq 2$	NP-hard and efficient approximation	Last Section

The first polynomial solvable class is trivial, because the size of support set is $O(n)$ when all the utility functions are additive. Regarding the second polynomial solvable class, we have the following result

Corollary 2. *(*Second solvable class*) If the support set S satisfies separability such that*

$$S = \bigcup_{i=1}^{m} S_i \text{ such that } A_i \cap A_j = \emptyset, \forall A_i \in S_i, A_j \in S_j,$$

and set $[n]$ is divided into m pairwise disjoint subsets by

$$U_i = \bigcup_{U \in S_i} U, \forall 1 \leq i \leq m,$$

and $\max_i |U_i| = \Theta(\log(n))$, then we can solve the defender oracle problem in poly(n) *time.*

The basic idea of the second solvable class is to show that when S is separable with size of largest component equal to $\Theta(\log(n))$, the DOP is separable and can be solved in poly(n) time via an enumerating algorithm.

Corollary 3. *(*Third solvable class*) If c is constant, all the common utilities $B^c(U)$ are negative except the singleton set, for which $C^c_d(U)$ are negative, and we can solve the defender oracle problem in* poly(n) *time.*

In the third solvable class, we will show that DOP under such a condition is a submodular minimization problem, which can be solved in poly(n) time. These special cases are interesting because they correspond to the following applications.

Algorithm 2 Separable Approximation

Calculate benefit $B(U) = T(G) - T(\mathbf{F_a}(G \backslash U)), \forall U \in \mathcal{A}$;
for each $U \in \mathcal{A}$ **do**
  Calculate $B^c(U) = \sum_{W \subseteq U} (-1)^{|U \backslash W|} B(W)$;
  if $|B^c(U)| \leq \epsilon_c$ **then** $\tilde{B}^c(U) = 0$;
end for
Create support set S and let $(S_1, \ldots, S_m) \leftarrow$ disjoint (S);

- The second class can be applied in a sparse network. For example, if the size of largest connected component of G is $\Theta(\log(n))$, S will satisfy the condition of second solvable class (Corollary 1 in supplemental material).
- The third class can be applied to a dense network where the most nodes are adjacent. For example, if $c = 2$, attacking any two nodes will lead to the superposition of the failure effect, resulting in negative common utilities.
- Another application of the third class: in cybersecurity, the sensor network often exhibits a tree topology. The game is such that the attacker attempts to invade some nodes to destroy the connectedness of the network and the IT manager is required to deploy anti-virus software in some nodes. We show that this game satisfies the condition of the third class. (Corollary 4 in supplemental material)

For the general network (not sparse, not dense or not a tree), the problem is clearly NP-hard, but we still need to answer two questions: (1) can we still compactly represent the game if c is large, i.e., $|S| = \text{poly}(n)$? (2) can we efficiently solve such a game? To tackle these problems, we propose a novel separable approximation framework (Algorithm 2), which can guarantee the approximation error of the original problem, instead of the DOP. One crucial observation is that *the common utility in the realistic network is well concentrated around zero.*

In Fig. 3, we examine the distributions of the benefit function and its common utility function in the following two kinds of network: Erdös-Renyi network $G(n, p)$ and scale-free network $G(n, \alpha)$, where n is the number of nodes, p is the probability that any two nodes are connected, α is the parameter of degree distribution of the scale-free network. Suppose that the network G consists of m connected components: $V_1, V_2, \ldots, V_m$ and we adopt the following two kinds of network value functions,

$$T_1(G) = \max_{1 \leq i \leq m} |V_i|, T_2(G) = \sum_{i=1}^{m} |V_i|^2.$$

The different form of network value functions have different assessment of the network. The detailed comparison can be found in (Gueye, Marbukh, and Walrand 2012). As can be seen in Fig. 3, in both Erdös-Renyi and scale-free networks, although the distribution of the benefit function is random, the distribution of the common utility function is well concentrated around zero and 90% of them are less than 0.05. In particular, when the number of nodes increases, this phenomenon is amplified such that almost 99% of the common utility functions are less than 0.05.

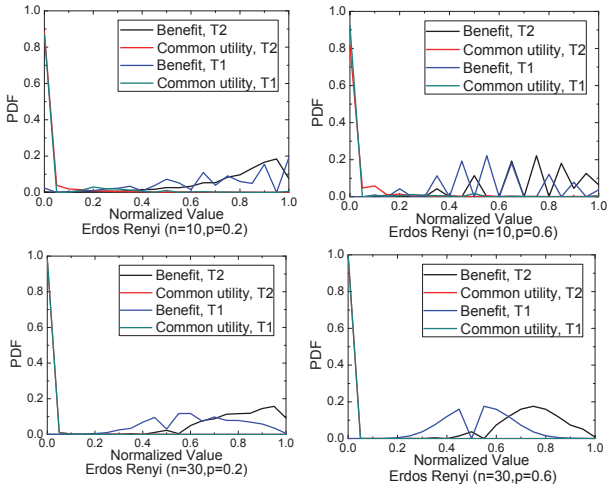

Figure 3: The distributions of common utility function and benefit function. All their value are absolute value and normalized in $[0, 1]$.

Based on the above observation, we can let most of the common utility functions equal to 0 according to a given threshold ϵ_c. Formally, let $\tilde{B}^c(\cdot)$ denote the new common utility function generated by Algorithm 2, then the corresponding approximate benefit function satisfies

$$|\tilde{B}(U) - B(U)| = \left| \sum_{W \subseteq U} \tilde{B}^c(W) - \sum_{W \subseteq U} B^c(W) \right|$$
$$\leq \sum_{W \subseteq U} \left| \tilde{B}^c(W) - B^c(W) \right| \leq 2^{|U|} \epsilon_c.$$

Since $|U| \leq c$, the maximum error between the original benefit functions and new generated benefit functions is less than $2^c \epsilon_c$. A classic result of game theory is that, if the maximum difference between the elements of two payoff matrices is bounded by ϵ, the difference of the optimal game values yielded by these two payoff matrices are bounded by 2ϵ (Lipton, Markakis, and Mehta 2003). Therefore, the approximation error of our game value is bounded by $2^{c+1} \epsilon_c$. Remark that the disjoint operator in Algorithm 2 separates the support set S into m parts that satisfies the conditions of separability of support set S.

As shown at the top of Fig. 4, for the Erdös Renyi, scale-free and Italian communication network, the size of support set will be reduced 90% by an extremely small approximation error 0.05. Moreover, this process also leads to a separable structure of S, and the resulting complexity of solving the NASG is $\text{poly}(n) O(2^{\max_i |U_i|})$. For example, in the bottom of Fig. 4, the complexity term $\max_i |U_i|$ can be greatly reduced to the order of $\Theta(\log(n))$ with an approximation error of 1%, regardless of the size and density of the network, and how many targets the attacker can choose. The more comprehensive numerical results can be found in

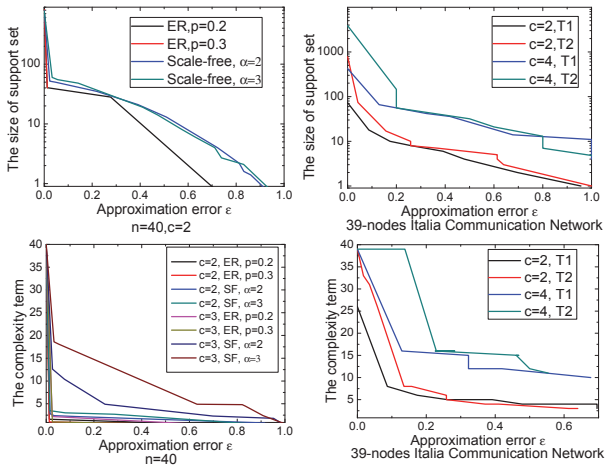

Figure 4: Top: the size of support set $|S|$ versus approximation error ϵ; Bottom: the complexity term $\max_i |U_i|$ versus approximation error ϵ. Remark that the ϵ represents the approximation error of the game value. Note that SF denotes the scale-free network.

the supplementary material. In summary, our approximation framework can reduce the complexity term $\max_i |U_i|$ to order $\Theta(\log(n))$ by only 10% approximation error in most networks including Erdös-Renyi, scale-free network and a $39-$nodes Italian communication network. Therefore, using our theoretical framework, we can **approximately and compactly represent a realistic network security game and solve it in poly(n) time with high accuracy**.

Conclusion

In this paper, we examined the security game under non-additive utility functions and a structured strategy space, i.e., uniform matroid. We showed that the size of the compact representation is dependent on the number of non-additive strategies, and NASG is essentially the problem of *maximizing a pseudo-boolean function*. Compared with previous results, this work greatly extends the polynomial solvable class, provides an understanding of the complexity properties, and partly answers the question proposed by Xu (2016) in zero-sum, uniform matroid scenario. For future directions, we plan to investigate (i) the relationship between the Oracle problem and the NASG when the defender has a non-structured strategy space; (ii) how to efficiently compute the defender's mixed strategy when attacker and defender have different benefit functions.

Acknowledgement

This work was supported in part by a grant from the Army Research Office AROW911NF-15-1-0277, and an Army Research Office MURI W911NF-12-1-0385.

References

Buldyrev, S. V.; Parshani, R.; Paul, G.; Stanley, H. E.; and Havlin, S. 2010. Catastrophic cascade of failures in interdependent networks. *Nature* 464(7291):1025–1028.

Grötschel, M.; Lovász, L.; and Schrijver, A. 1981. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica* 1(2):169–197.

Gueye, A.; Marbukh, V.; and Walrand, J. C. 2012. Towards a metric for communication network vulnerability to attacks: A game theoretic approach. In *Game Theory for Networks*. Springer. 259–274.

Jain, M.; Korzhyk, D.; Vaněk, O.; Conitzer, V.; Pěchouček, M.; and Tambe, M. 2011. A double oracle algorithm for zero-sum security games on graphs. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1 (AAMAS)*, 327–334. International Foundation for Autonomous Agents and Multiagent Systems.

Kiekintveld, C.; Jain, M.; Tsai, J.; Pita, J.; Ordóñez, F.; and Tambe, M. 2009. Computing optimal randomized resource allocations for massive security games. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 689–696. International Foundation for Autonomous Agents and Multiagent Systems.

Korzhyk, D.; Yin, Z.; Kiekintveld, C.; Conitzer, V.; and Tambe, M. 2011. Stackelberg vs. nash in security games: An extended investigation of interchangeability, equivalence, and uniqueness. *J. Artif. Intell. Res.(JAIR)* 41:297–327.

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

Korzhyk, D.; Conitzer, V.; and Parr, R. 2011. Security games with multiple attacker resources. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, 273. Citeseer.

Letchford, J., and Conitzer, V. 2013. Solving security games on graphs via marginal probabilities. In *AAAI*.

Lipton, R. J.; Markakis, E.; and Mehta, A. 2003. Playing large games using simple strategies. In *Proceedings of the 4th ACM conference on Electronic commerce (EC)*, 36–41. ACM.

Moulin, H., and Vial, J.-P. 1978. Strategically zero-sum games: the class of games whose completely mixed equilibria cannot be improved upon. *International Journal of Game Theory* 7(3-4):201–221.

Shakarian, P.; Lei, H.; and Lindelauf, R. 2014. Power grid defense against malicious cascading failure. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, 813–820. International Foundation for Autonomous Agents and Multiagent Systems.

Vorobeychik, Y., and Letchford, J. 2015. Securing interdependent assets. *Autonomous Agents and Multi-Agent Systems* 29(2):305–333.

Xu, H.; Fang, F.; Jiang, A. X.; Conitzer, V.; Dughmi, S.; and Tambe, M. 2014. Solving zero-sum security games in discretized spatio-temporal domains. In *AAAI*, 1500–1506. Citeseer.

Xu, H. 2016. The mysteries of security games: Equilibrium computation becomes combinatorial algorithm design. *arXiv preprint arXiv:1603.02377*.