Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Computing Nash Equilibrium in Interdependent Defense Games

Hau Chan and Luis E. Ortiz
Department of Computer Science, Stony Brook University
{hauchan,leortiz}@cs.stonybrook.edu

Abstract

Roughly speaking, Interdependent Defense (IDD) games, previously proposed, model the situation where an attacker wants to cause as much damage as possible to a network by attacking one of the sites in the network. Each site must make an investment decision regarding security to protect itself against a direct or indirect attack, the latter due to potential transfer-risk from an unprotected neighboring site. The work introducing IDD games discusses potential applications to model the essence of real-world scenarios such as the 2006 transatlantic aircraft plot. In this paper, our focus is the study of the problem of computing a Nash Equilibrium (NE) in IDD games. We show that an efficient algorithm to determine whether some attacker's strategy can be a part of a NE in an instance of IDD games is unlikely to exist. Yet, we provide a dynamic programming algorithm to compute an approximate NE when the graph/network structure of the game is a directed tree with a single source, and show that it is an FPTAS. We also introduce an improved heuristic to compute an approximate NE on arbitrary graph structures. Our experiments show that our heuristic is more efficient, and provides better approximations, than best-response-gradient dynamics for the case of Internet games, a class of games introduced and studied in the original work on IDD games.

Introduction

In this paper, we provide further computational results to Interdependent Defense (IDD) games, a model introduced by Chan, Ceyko, and Ortiz (2012). Roughly speaking, IDD games model the interaction among strategic agents, represented by sites embedded in a network, and an attacker. The sites have the objective to protect themselves from the attacker by making individual, voluntary investment decisions regarding their own security. The attacker aims to cause as much damage as possible. A key aspect is that the cost-effectiveness of each site's decision, as well as the attacker's own strategy, depend on both the risk of a successful *direct* attack initiated at a site and the potential one-hop *transfer* of the risk from one site to another through the network.

We now present a rough summary of IDD games. We delay a formal definition until the respective section. We refer the reader to (Chan, Ceyko, and Ortiz 2012) for a more thorough motivation, account of their design decisions and derivation, related work and description of a few example applications of IDD games, including models for airline-baggage security. (Other applications include computer security, vaccination, and house/apartment fire protection. There is a recent survey by Laszka, Felegyhazi, and Buttyan, 2014, to highlight other aspects of IDS games.) IDD games build on the work of economists and risk-analysis experts (Heal and Kunreuther 2005) on *interdependent security (IDS) games*. Hence, they can, in principle, apply to any setting in which IDS games do. A major distinction here is that the "bad event" is modeled as a result of some deliberate strategic attack by an attacker. The attacker determines which site to attack, given the potential one-hop attack risk transfer, to maximize the network's overall damage from attacks to sites on the system.

The adaptation of IDS games via the introduction of a *strategic* attacker, which *explicitly* induces risk via a deliberate attack, partly motivate the creation of IDD games (Chan, Ceyko, and Ortiz 2012). By explicitly modeling the attacker as a strategic agent, IDD games move away from the previously standard in security-type models in which the attack was considered as a non-strategic random event (e.g., Johnson et al., 2010).

As Chan, Ceyko, and Ortiz (2012) state, there are three main components of IDD games:

1. *There is a notion of, and the means for, attack transfer and transfer risks.*

   In IDD games, there is a notion of indirect risk, because a site can, with some probability, "transfer" a deliberate attack only once to another neighboring site in the network graph. As a result, sites must make investment decisions based not only on the potential of a deliberate direct attack, but also on the potential indirect risk from neighboring sites. As a side note, if the neighboring sites invest in protection against a direct attack, then it is not possible for the attack to transfer. Hence, one can alternatively view the transfer probabilities as implicitly inducing a directed graph over the network of sites.

2. *There is an attacker who aims to cause the most damage.*

   Moreover, the attacker determines which site to attack based on the amount of damage it can cause to a site directly, or to the site's neighbors indirectly, less the at-

tacker's cost to attack the site and the probability of and expected gains from a successful attack to that site.

3. *There are sites that can be viewed as targets of an attack, each of which wants to protect itself from the attack.*

   A site's investment decision is based on whether the site's neighbors invest in security and whether the attacker will attack the site or its neighbors, less the site's investment cost and its potential loss from a successful attack on the site.

Our Contribution

The objective of this paper is to study the problem of computing Nash equilibrium (NE) in IDD games. Here is a list summarizing our contributions.

- We show that verifying whether some attacker's strategy can be a part of a NE in an instance of IDD games is unlikely to have an efficient algorithm. We do this by showing a related problem is NP-complete. As a consequence, we may not be able to find a NE if this is the only strategy left (all other strategies are eliminated because they cannot be part of a NE).

  This is surprising because there is a polynomial-time algorithm to compute all NEs in IDD games when $\alpha = 1$ (Chan, Ceyko, and Ortiz 2012). (We define α in the next section.) However, that algorithm is unlikely to generalize to compute NEs for different α values.

- We study the question of computing an approximate NE. We show that there is an FPTAS to compute an approximate NE when the graph of the sites is a directed tree.

  This is surprising because it is not possible to use the dynamic programming algorithms of Kearns, Littman, and Singh (2001) and Elkind, Goldberg, and Goldberg (2006) to compute an ϵ-MSNE in IDD games efficiently, mostly because of the more compact representation of IDD games (i.e., linear in the number of edges in the network). Moreover, finding ϵ-MSNE in general degree-3 graphical games is PPAD-hard (Elkind, Goldberg, and Goldberg 2006). Our α-IDD games have more than 3 degrees. In fact, because the attacker is connected to all the nodes in the network, as a graphical game with normal-form representation of the local payoff matrices, the graph of the IDD games is completely connected (i.e., the attacker's mixed strategy imposes a global constraint). But the local payoff functions in our case are compactly representable in parametric form. Still, we provide an FPTAS to compute ϵ-MSNE in interesting subclasses of IDD games.

- We introduce a heuristic, based on smoothed best-response dynamics (Fudenberg and Levine 1999), to compute an approximate NE in IDD games with arbitrary network structures.

- We perform experiments using our heuristic on Internet games, as introduced by Chan, Ceyko, and Ortiz (2012), and compare our results against that from *best-response-gradient dynamics (BRGD)*, the heuristic used by Chan, Ceyko, and Ortiz (2012). The experiments show that our heuristic computes approximate NEs considerably faster and produces better/tighter approximations.

  The BRGD is already a reasonably efficient approach to compute (small) ϵ-MSNEs, up to about $\epsilon = 0.001$ (Chan, Ceyko, and Ortiz 2012). (Although, it was often incomplete for such small ϵ values.) Given the general simplicity and effectiveness of BRGD, there is no reason, a priori*, to expect to find a significantly better heuristic for most instances of the IDD games (i.e., Internet games) that Chan, Ceyko, and Ortiz (2012) evaluated. Yet, here, we do introduce a simple heuristic that is considerably faster and provides ϵ-MSNEs with a smaller ϵ than BRGD for the same class of IDD games.*

On the Complexity of Computing NE

To put our contribution in context, we now provide recent progress on computing a NE in general games. Nash (1950) showed that every finite, normal-form game has at least one NE (Fudenberg and Tirole 1991). Chen, Deng, and Teng (2009) first showed that computing exact one NE, or even an approximate one of quality better than an inverse polynomial of the approximation parameter, is PPAD-complete. The PPAD-completeness holds even for 2-player games and graphical games with graphs of maximum degree 3 and constant pathwidth (Daskalakis, Goldberg, and Papadimitriou 2006; Elkind, Goldberg, and Goldberg 2006). Kearns, Littman, and Singh (2001) and Elkind, Goldberg, and Goldberg (2006) provide quasi-polynomial-time algorithms to compute approximate NE on trees. However, we cannot directly or naively apply those algorithms to our setting. The graph structure of IDD games is not a tree because the attacker is connected to every site.

Interdependent Defense Games

We will closely follow the notations and definitions by Chan, Ceyko, and Ortiz (2012) to define a (directed) graphical-games version of IDD games. Let $[n] \equiv \{1, \ldots, n\}$ denote the set of sites. For each site $i \in [n]$, let $a_i = 1$ and $a_i = 0$ denote the invest and no-invest individual *action/pure-strategy* of player i, respectively. For each player i, the *cost of investing* in security is C_i, while the potential *loss* due to a successful attack on site i is L_i. For the attacker, let $\mathbf{b} \in \{0,1\}^n$ denote the joint-attack vector, such that $b_i = 1$ denotes the decision to attack site i. The *cost of an attack* on site i is C_i^0. We denote by $\mathbf{a} \equiv (a_1, \ldots, a_n) \in \{0,1\}^n$ the *joint-action/joint-pure-strategy* of the n sites, $\mathbf{a}_{-i}$ the joint-action of $n-1$ sites except i, and $\mathbf{a}_I$ the joint-action of sites in the set $I \subseteq [n]$.

From the sites' perspective, an attack can occur directly or indirectly through other neighboring sites. The parameter $\hat{p}_i$ captures the conditional probability that a direct attack on i is successful. The parameter $\hat{q}_{ij}$ captures the conditional probability that an attack, initiated at site i but transferred (undetected) to site j where the attack is finally successful. An implicit, simplifying assumption, carried over from the original IDS model (Kunreuther and Heal 2003; Heal and Kunreuther 2005), is that there is at most one transfer between sites. This condition is not as restrictive as it first appears. One way, albeit naive, to model multiple transfers would be to set the corresponding q_{ij} to a non-zero value.

Indeed, as an implication of the *transfer probabilities* $\hat{q}_{ij}$'s, there is a *directed (game) graph* $G = ([n], E)$ such that $E = \{(i,j)|\hat{q}_{ij} > 0\}$. We denote the *parents* and *children* of i in the graph as $\text{Pa}(i)$ and $\text{Ch}(i)$, respectively.

Given the graph structure and the risk-related parameters, we define the safety and (indirect) risk functions of a site. It is not hard to see that the safety of site i from site j depends whether j invests and whether the attacker attacks j. Moreover, the *safety* of i from j is $e_{ij}(a_j, b_j) \equiv a_j + (1-a_j)(1 - b_j\hat{q}_{ji}) = (1-\hat{q}_{ji})^{b_j(1-a_j)}$. The *overall safety function* is $s_i(\mathbf{a}_{\text{Pa}(i)}, \mathbf{b}_{\text{Pa}(i)}) \equiv \prod_{j\in\text{Pa}(i)} e_{ij}(a_j, b_j)$. The *overall risk function* is simply $r_i(\mathbf{a}_{\text{Pa}(i)}, \mathbf{b}_{\text{Pa}(i)}) \equiv 1 - s_i(\mathbf{a}_{\text{Pa}(i)}, \mathbf{b}_{\text{Pa}(i)})$.

Of course, the sites can fully protect themselves from a direct attack if they invest. However, they can only partially protect themselves from the (indirect) risk. The parameter $\alpha_i \in [0,1]$ denotes the probability that a transfer of a potential attack will go unblocked by i's security even if i invests.

Given the above, we define the *cost function of site* i

$$M_i(a_i, \mathbf{a}_{\text{Pa}(i)}, b_i, \mathbf{b}_{\text{Pa}(i)}) \equiv a_i[C_i + \alpha_i r_i(\mathbf{a}_{\text{Pa}(i)}, \mathbf{b}_{\text{Pa}(i)})L_i] + (1-a_i)[b_i\hat{p}_i + (1-b_i\hat{p}_i)r_i(\mathbf{a}_{\text{Pa}(i)}, \mathbf{b}_{\text{Pa}(i)})]L_i \ .$$

The above cost function states that if site/player i plays $a_i = 1$, site i will have to pay for the cost of investment but may still incur a potential loss from indirect risk. If site i plays $a_i = 0$, site i may incur the potential loss from a direct attack or the (full) potential loss from an indirect attack.

The attacker in IDD games maximizes the cost of the sites, without consideration for the sites' cost of investment, minus the cost to attack the sites. Formally, the *utility function of the attacker* is $U(\mathbf{a}, \mathbf{b}) \equiv \sum_{i=1}^{n} M_i(a_i, \mathbf{a}_{\text{Pa}(i)}, b_i, \mathbf{b}_{\text{Pa}(i)}) - a_i C_i - b_i C_i^0$.

As in Chan, Ceyko, and Ortiz (2012), here, we only consider IDD games under the following conditions: (1) there is at most one attack (i.e., $\sum_{i=1}^{n} b_i \leq 1$); (2) for a given site, the cost of investment is always less than the expected potentially loss (i.e., $0 < C_i < \hat{p}_i L_i$ for all $i \in [n]$); and (3) for a given site, the site's cost of an attack is always less than the maximum utility that can be obtained by attacking that site (i.e., $0 < C_i^0 < \hat{p}_i L_i + \sum_{j\in Ch(i)} \hat{q}_{ij}\alpha_j L_j$ for $\forall i \in [n]$).

Pure-strategy Nash equilibrium (PSNE) is one solution concept in non-cooperative game theory. Roughly speaking, a pure-strategy $(\mathbf{a}^*, \mathbf{b}^*)$ of an IDD game is a PSNE if (1) each site i's pure-strategy a_i^* minimizes the site's cost function M_i given the pure-strategies $\mathbf{a}^*_{-i}$ and $\mathbf{b}^*$ of the other sites and the attacker's, respectively; and (2) the attacker's pure-strategy $\mathbf{b}^*$ maximizes its utility function U_i given the pure strategies of all the sites $\mathbf{a}^*$. Unfortunately, Chan, Ceyko, and Ortiz (2012) showed that no PSNE exists under the conditions given above. As a result, we focus on *mixed strategies*.

Let x_i denote *player i's mixed strategy:* the probability that site i invests. Let $\mathbf{y}$ denote the *attacker's mixed strategy*, in which each component y_i denotes the probability that the attacker attacks site i. We denote by y_0 the probability of 'no attack', so that $\sum_{i=0}^{n} y_i = 1$. In this setting, players take expectations with respect to everybody's mixed strategy to evaluate their expected costs or utilities and take an optimal action. In particular, the sites evaluate the expect value of their individual cost functions and the attacker evaluates the expected value of the utility function. With a slight abuse of notation, the *(expected) overall safety function of site* i becomes $s_i(\mathbf{x}_{\text{Pa}(i)}, \mathbf{y}_{\text{Pa}(i)}) \equiv 1 - \sum_{j\in\text{Pa}(i)} y_j(1-x_j)\hat{q}_{ji}$, and as before $r_i(\mathbf{x}_{\text{Pa}(i)}, \mathbf{y}_{\text{Pa}(i)}) \equiv 1 - s_i(\mathbf{x}_{\text{Pa}(i)}, \mathbf{y}_{\text{Pa}(i)})$. The *expected cost of site* i becomes

$$M_i(x_i, \mathbf{x}_{\text{Pa}(i)}, y_i, \mathbf{y}_{\text{Pa}(i)}) \equiv x_i[C_i + \alpha_i r_i(\mathbf{x}_{\text{Pa}(i)}, \mathbf{y}_{\text{Pa}(i)})L_i] + (1-x_i)[\hat{p}_i y_i + r_i(\mathbf{x}_{\text{Pa}(i)}, \mathbf{y}_{\text{Pa}(i)})]L_i \ .$$

Similarly, the *expected utility of the attacker* becomes $U(\mathbf{x}, \mathbf{y}) \equiv \sum_{i=1}^{n} M_i(x_i, \mathbf{x}_{\text{Pa}(i)}, y_i, \mathbf{y}_{\text{Pa}(i)}) - x_i C_i - y_i C_i^0$. A simple rewriting of U provides a more natural interpretation. The expected utility of the attacker from attacking i depends on how much it can get from an attack to i directly or from the potential transfer to the children of i, minus the cost to attack i: $U(\mathbf{x}, \mathbf{y}) \equiv \sum_{i=1}^{n} y_i U_i(\mathbf{x})$, where $U_i(\mathbf{x}) \equiv$

$$(1-x_i)\left(\hat{p}_i L_i + \sum_{j\in\text{Ch}(i)} (x_j\alpha_j + (1-x_j))\hat{q}_{ij}L_j\right) - C_i^0 \ .$$

To conclude the description of the IDD games, we provide the general definition of a NE in terms of the best-response functions. Let $R_i \equiv \frac{C_i}{L_i}$ and $\hat{s}_i(\mathbf{x}_{\text{Pa}(i)}, y_i, \mathbf{y}_{\text{Pa}(i)}) \equiv y_i\hat{p}_i + (1-\alpha_i)r_i(\mathbf{x}_{\text{Pa}(i)}, \mathbf{y}_{\text{Pa}(i)})$ The *best-response correspondence of site* i is $\mathcal{BR}_i(\mathbf{x}_{\text{Pa}(i)}, y_i, \mathbf{y}_{\text{Pa}(i)}) \equiv$

$$\begin{cases} \{1\}, & \text{if } \hat{s}_i(\mathbf{x}_{\text{Pa}(i)}, y_i, \mathbf{y}_{\text{Pa}(i)}) > R_i, \\ \{0\}, & \text{if } \hat{s}_i(\mathbf{x}_{\text{Pa}(i)}, y_i, \mathbf{y}_{\text{Pa}(i)}) < R_i, \\ \{0,1\}, & \text{if } \hat{s}_i(\mathbf{x}_{\text{Pa}(i)}, y_i, \mathbf{y}_{\text{Pa}(i)}) = R_i. \end{cases}$$

The *best-response correspondence for the attacker* is $\mathcal{BR}_0(\mathbf{x}) \equiv \arg\max_{\mathbf{y}} U(\mathbf{x}, \mathbf{y})$.

Definition A mixed-strategy $(\mathbf{x}^*, y^*)$ is a *mixed-strategy Nash equilibrium (MSNE)* of an IDD game if (1) for all $i \in [n]$, $x_i^* \in \mathcal{BR}_i(\mathbf{x}^*_{\text{Pa}(i)}, y_i^*, y^*_{\text{Pa}(i)})$ and (2) $y^* \in \mathcal{BR}_0(\mathbf{x})$. If $(\mathbf{x}^*, y^*)$ corresponds to a (deterministic) joint-pure-strategy, then we refer to the MSNE simply as a PSNE.

On the Complexity of Computing an MSNE

Given that there is no PSNE in any IDD games (Chan, Ceyko, and Ortiz 2012), we shift our focus to computing an MSNE. Chan, Ceyko, and Ortiz (2012) provides an algorithm to compute all MSNEs in an instance of IDD games where $\alpha_i = 1$ for all sites i. The interpretation is that investment cannot protect the sites from indirect risk. However, there is no result for the harder case of general α_i.

Here, we consider the computational complexity of computing an MSNE in general α-IDD games. A closer look at the model reveals something interesting about IDD games: we can view computing an MSNE in IDD games as a two-part process. Given an attacker's strategy, we need to determine the MSNE of the underlying game of the sites, or *sites-game* for short. The sites-game could have many MSNEs and each MSNE could yield a different utility for the attacker (and the sites). Naively, the attacker can verify whether each of the MSNEs is in the attacker's best response. Clearly, doing so depends on whether we can efficiently compute all MSNEs in the sites-game, which of

course depends on the given attacker's strategy. For example, if $\sum_{i=1}^{n} y_i = 0$, then the sites-game would have 'none invest' as the only outcome, because of condition (2) above.

Our goal here is to show that there is an instance of IDD games, and an attacker's strategy in that instance, such that should we fix that attacker's strategy, we cannot compute all of the MSNEs efficiently in the underlying sites-game, unless $P = NP$. The implication is that the existence of an efficient algorithm to compute an MSNE of IDD games based on the iterative process just described, of checking whether each attacker's strategy can be part of an MSNE, would be unlikely.

To formally prove that we cannot always compute all of the MSNEs in an instance of the sites-games, as induced by an IDD game and an attacker's strategy, efficiently, we consider the *pure-Nash-extension problem* (Kearns and Ortiz 2004) for binary-action n-player games, which is NP-complete. The problem takes a description of the game and a *partial* assignment $\mathbf{a} \in \{0, 1, *\}^n$ as input. We want to determine whether there is a *complete* assignment $\bar{\mathbf{a}} \in \{0, 1, *\}^n$ consistent with $\mathbf{a}$. Note that proving that computing an MSNE in IDD games is PPAD-complete would be more appropriate, since there is always an MSNE, but we will leave this for future work.

Theorem 1 *Consider a variant of IDD games in which* $\sum_{i=1}^{n} R_i/\hat{p}_i \leq 1$. *There is an attacker's strategy* $\mathbf{y}$ *such that if we fix* $\mathbf{y}$*, then the pure-Nash extension problem for the induced* n*-player sites-game is NP-complete.*

Proof *(Sketch)* First, we construct a graph structure and set the parameters to define the IDD game based on an NP-complete problem. Next, we show that if $\mathbf{y}$ exists, then the induced sites-game solves the NP-complete problem. Finally, we show that such a $\mathbf{y}$ exists.

We take an instance of Monotone 1-in-3 SAT (Garey and Johnson 1979) with a set of clauses C and a set of variables V. We consider a bipartite graph structure between the clauses and the variables. We connect the variables to their corresponding clauses via direct edges (from the variables to the clauses). There is exactly one player for each variable, and we set the player's parameters such that the player is indifferent between invest and no-invest (i.e., $R_v = y_v \hat{p}_v$ for all variable players $v \in V$). For each clause, we introduce two clause players, denoted by a and b. For clause players a and b, we set $R_a > 0$, $R_b > 0$, $\hat{p}_a > 0$, $\hat{p}_b > 0$, $\alpha_a > 0$, $\alpha_b > 0$, and some transfer probability q such that $R_a > (1 - \alpha_a) R_v \hat{q}$ and $R_b > (1 - \alpha_b) 2 R_v \hat{q}$. Given a $\mathbf{y}$, we have that $\frac{2R_v\hat{q}}{\hat{p}_v} > \frac{1}{(1-\alpha_a)}(\frac{C_a}{L_a} - y_a\hat{p}_a) > \frac{R_v\hat{q}}{\hat{p}_v}$ and $\frac{3R_v\hat{q}}{\hat{p}_v} > \frac{1}{(1-\alpha_b)}(\frac{C_b}{L_b} - y_b\hat{p}_b) > \frac{2R_v\hat{q}}{\hat{p}_v}$. Clause player a invests if at least two of its variable players do not invest and clause player b invests if at least three, or all three, of its variable players do not invest. Finally, we give partial pure-strategy assignments to the clause players in which a invests and b does not to guarantee that exactly one invests. It is not hard to see that the solution to the Monotone 1-in-3 SAT is also an MSNE of the resulting sites-game and vice versa.

The existence of such a $\mathbf{y}$ follows immediately from the constraint $\sum_{i \in C \cup V} R_i/\hat{p}_i \leq 1$. For $v \in V$, $y_v = \frac{R_v}{\hat{p}_v}$. For each clause player a (and b), $y_a < \frac{R_a}{\hat{p}_a}$ $\left(y_b < \frac{R_b}{\hat{p}_b}\right)$. $\square$

Worst case, we need to consider the $\mathbf{y}$ just described, should other strategies fail to be a part of any MSNE. Another challenge is that even if we can compute all exact MSNEs, there could be exponentially many of them to check. In the next section, we look for efficient algorithms to compute an approximate MSNE in various graph structures.

FPTAS for ϵ-MSNEs in Tree-like IDD Games

In this section, we compute ϵ-MSNEs in a subclass of IDD games. In particular, we study different graph structures among the sites. We note that the attacker is connected to all of the sites even if we do not point it out explicitly.

Definition A mixed-strategy $(\mathbf{x}^*, \mathbf{y}^*)$ is an *ϵ-MSNE* of an IDD game if (1) for all $i \in [n]$, $M_i(x_i^*, \mathbf{x}^*_{\text{Pa}(i)}, y_i^*, \mathbf{y}^*_{\text{Pa}(i)}) \leq \min_{x_i} M_i(x_i, \mathbf{x}^*_{\text{Pa}(i)}, y_i^*, \mathbf{y}^*_{\text{Pa}(i)}) + \epsilon$, and (2) $U(\mathbf{x}^*, \mathbf{y}^*) \geq \max_{\mathbf{y}} U(\mathbf{x}^*, \mathbf{y}) - \epsilon$.

An exact MSNE $\equiv$ 0-MSNE. Moreover, we assume that all the cost and utility functions are individually normalized to $[0, 1]$ and $\epsilon \in [0, 1]$; otherwise ϵ is not well-defined.

We will start off simple by considering a *directed line/chain (DL)* graph structure. We show that there is a *fully polynomial-time approximation scheme (FPTAS)* to compute an ϵ-MSNE in DL-IDD games. Then we generalize the result to *directed trees (DT)*. Despite the simplicity of the graphs, one can envision very important real-world applications such as protection of supply chains and other hierarchical structures (e.g. see Agiwal and Mohtadi, 2008).

Directed Lines/Chains

Let *directed line* $G = ([n], E)$ where $E = \{(i, i+1) | \hat{q}_{i,i+1} > 0, i = 1, ..., n-1\}$. The line starts at site 1 and ends at site n. The risk functions for $i = 1, 2, ..., n$ become $r_i(x_{i-1}, y_{i-1}) \equiv y_{i-1}(1 - x_{i-1})\hat{q}_{i-1i}$ and $r_1 \equiv 0$.

Since the domain of the variables (i.e., mixed strategies) is $[0, 1]$, a direct optimization method to compute an MSNE would require solving a highly non-linear optimization problem: cubic objective function for the attacker with quartic constraints for the sites. An alternative is to discretize the continuous space of the x_i's and y_i's.

Let $\mathcal{X} \equiv \mathcal{X}(\Delta_x) \equiv \{0, \tau_x, 2\tau_x, \ldots, (\Delta_x - 1)\tau_x, 1\}$ and $\mathcal{Y} \equiv \mathcal{Y}(\Delta_y) \equiv \{0, \tau_y, 2\tau_y, \ldots, (\Delta_y - 1)\tau_y, 1\}$ be the respective *discretization* of the interval $[0, 1]$ where $\tau_x \equiv \lfloor \frac{1}{\Delta_x} \rfloor$ and $\tau_y \equiv \lfloor \frac{1}{\Delta_y} \rfloor$ are the respective *discretization lengths*, and Δ_x and Δ_y are the respective *discretization sizes*. The discretization defines the domains of x_i and y_i to be $\mathcal{X}$ and $\mathcal{Y}$, respectively. Moreover, $|\mathcal{X}| = \lceil \Delta_x \rceil$ and $|\mathcal{Y}| = \lceil \Delta_y \rceil$. Of course, there is an extra constraint for the y_i's in $\mathcal{Y}$: $\sum_{i=1}^{n} y_i \leq 1$ for $\mathbf{y} \in \mathcal{Y}^n$. We will determine the values of Δ_x and Δ_y to guarantee an ϵ-MSNE later in the section, but for now, assume they are given. A simple brute-force algorithm to compute an ϵ-MSNE is to check all possible discrete combinations and would take, $O\left((\frac{1}{\Delta_x}\frac{1}{\Delta_y})^n\right)$ time, to run in the worst case.

However, looking at the objective function of the attacker more carefully reveals something interesting:

the problem is equivalent to computing $(\mathbf{x}^*, \mathbf{y}^*) \in \arg\max_{(\mathbf{x},\mathbf{y})\in\mathcal{X}^n\times\mathcal{Y}^n} U(\mathbf{x},\mathbf{y})$ such that $\mathbf{x}^* \in \mathcal{X}^n$ satisfies the ϵ-best-response constraints and $\mathbf{y}^* \in \mathcal{Y}^n$ satisfies $\sum_{i=1}^n y_i^* \leq 1$. Note that at a solution $(\mathbf{x}^*, \mathbf{y}^*)$, the attacker has no incentive to deviate to another (discrete) strategy because it cannot obtain anything more than ϵ higher, given $\mathbf{x}^*$. In fact, we can show that $(\mathbf{x}^*, \mathbf{y}^*)$ is an ϵ-MSNE. All we need to do now is to find a way to compute a value in the set $\arg\max_{(\mathbf{x},\mathbf{y})\in\mathcal{X}^n\times\mathcal{Y}^n} U(\mathbf{x},\mathbf{y})$. Moreover, the utility function of the attacker for each site is "local" in the sense that the maximization depends on the immediate children. Hence, $\max_{(\mathbf{x},\mathbf{y})\in\mathcal{X}^n\times\mathcal{Y}^n} U(\mathbf{x},\mathbf{y})$

$$
\begin{aligned}
&= \max_{\mathbf{x},\mathbf{y}} \textstyle\sum_{i=1}^{n-1} y_i U_i(\mathbf{x}) + y_n U_n(x_n) \\
&= \max_{\mathbf{y}} \max_{x_1,x_2} y_1 U_1(x_1,x_2) + \max_{x_3} y_2 U_2(x_2,x_3) + \\
&\quad \ldots + \max_{x_n} y_{n-1} U_{n-1}(x_{n-1},x_n) + y_n U_n(x_n)\ .
\end{aligned}
$$

This is despite $\mathbf{y}$ *being globally constrained!* More importantly, because of the line structure, to compute an ϵ-best-response for a site i, we only need to know y_i, y_{i-1}, and x_{i-1} which we can compute systematically during the maximization using the principle of *dynamic programming*.

Indeed, we design a simple dynamic-programming algorithm to compute an ϵ-MSNE that is provably an FPTAS. Let $B[i][x][y] = yU_i(x, x_{i+1})$ be the utility of the attacker that can be obtained from site i when $y \in \mathcal{Y}$ and $x \in \mathcal{X}$. Note that $B[i][x][y]$ is a function of x_{i+1}. Throughout the algorithm, we will make sure that $\sum_i^n y_i \leq 1$. Because site 1 is the root and it does not have any parent, to compute its ϵ-best-response, we only need y_1. Therefore the base case will start off with site 1. The base case starts with computing the possible utility values for each combination of $x_1 \in \mathcal{X}$ and $y_1 \in \mathcal{Y}$. For those combinations that are not ϵ-best response, we assign the value $-\infty$ so that it will not be considered. This utility, of course, will be a function of x_2 (the children of 1). In the inductive ith step, for each combination of $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$, we take the maximum utility of the $(i-1)$th site condition on the value x_i and subject to the condition that $\sum_{j=1}^i y_j \leq 1$, while satisfying the ϵ-best response of site i. The pseudocode appears in the supplementary material.

The key to show that the dynamic-programming algorithm produces an ϵ-MSNE for the DL-IDD games is the discretization sizes. The question is, how small can we make Δ_x and Δ_y and still guarantee an ϵ-MSNE in the discretized space? A more general result about sparse discretization for graphical games (Ortiz 2014) provides the answer.

Corollary 1 *Given an n-player IDD game, let $k \equiv \max_{i\in[n]} |\mathrm{Pa}(i)| + 1$. Setting $\Delta_x = \lceil \frac{4k}{\epsilon} \rceil = O(\frac{k}{\epsilon})$ and $\Delta_y = \lceil \frac{2nk}{\epsilon} \rceil = O(\frac{nk}{\epsilon})$ is sufficient to obtain an ϵ-MSNE.*

Proposition 1 *There is a dynamic-programming algorithm that computes an ϵ-MSNE in DL-IDD games in time $O\left(n(\Delta_x\Delta_y)^2\right) = O\left(\frac{n^3}{\epsilon^4}\right)$, where the discretization lengths are set to $\Delta_x = \lceil \frac{8}{\epsilon} \rceil = O(\frac{1}{\epsilon})$ and $\Delta_y = \lceil \frac{4n}{\epsilon} \rceil = O(\frac{n}{\epsilon})$.*

Corollary 2 *There is a fully polynomial time approximation scheme (FPTAS) to compute an ϵ-MSNE in DL-IDD games.*

Directed Stars

Let the source node corresponds to player n, and the remaining $n-1$ sink nodes correspond to players' $1, \ldots, n-1$. The directed star (DS) is equivalent to a directed tree with a single root at n with $n-1$ leaves and no internal nodes.

Corollary 3 *There is an FPTAS to compute an ϵ-MSNE in DS-IDD games.*

The key idea is to realize that given a strategy (x_n, y_n) of the root n, the leaves' decisions are independent of each other. However, there is a sum less than or equal to one constraint for the attacker (i.e., $\sum_{i=1}^n y_i \leq 1$). Notice that, in this situation, this is very similar to the DL case earlier; the difference is that the nodes are not implicitly ordered given (x_n, y_n). If we impose an ordering on the nodes, we can run a very similar version of the dynamic programming algorithm used for the DL. Indeed, for each possible combination of (x_n, y_n), we can run such dynamic programming algorithm on the (same) ordering of the nodes and obtain a (best) value for each (x_n, y_n). Clearly, the best (x_n^*, y_n^*) that obtains the maximum value among all other (x_n, y_n)'s is the best possible strategy for the attacker. This guarantees that the attacker would not deviate to a different strategy. Moreover, the dynamic programming algorithm would produce solutions that ensure the leave players are best-responding. More formally, we define the following mathematical expressions for the dynamic programming algorithm. This will give us an FPTAS for DS-IDD games.

Upstream pass: Collection of conditional ϵ-MSNE computation. First, we impose an ordering on the leaves, that is, we order the leaves in increasing order. Let $\overline{M}_i(x_i, y_i, x_n, y_n) \equiv M_i(x_i, y_i, x_n, y_n) - x_i C_i - y_i C_i^0$ be the attacker's utility for attacking i. For each leaf $i = 1, \ldots, n-1$, we compute the set of individual conditional tables (in this order),

$$
\begin{aligned}
&\overline{T}_{i,n}(x_n, y_n, v_i, x_i, y_i, v_{i-1}) \equiv \\
&\overline{M}_i(x_i, y_i, x_n, y_n) + \\
&\log\left(\mathbb{1}[v_i = y_i + v_{i-1}]\right) + \\
&\log\left(\mathbb{1}\left[x_i \in \mathcal{BR}^{\epsilon}_{x_i}(y_i, x_n, y_n)\right]\right) + \\
&T_{i-1,n}(x_n, y_n, v_{i-1})
\end{aligned}
$$

$$
T_{i,n}(x_n, y_n, v_i) \equiv \max_{(x_i, y_i, v_{i-1})} \overline{T}_{i,n}(x_n, y_n, v_i, x_i, y_i, v_{i-1})
$$

$$
W_{i,n}(x_n, y_n, v_i) \equiv \arg\max_{(x_i, y_i, v_{i-1})} \overline{T}_{i,n}(x_n, y_n, v_i, x_i, y_i, v_{i-1})
$$

where $T_{0,n}(x_n, y_n, s_0) = 0$ for all (x_n, y_n, s_{i_0}). Each $T_{i,n}$ specifies the maximum possible utility an attacker can get by attacking all the leaves up to i given that the attacker will attack the root n with probability y_n, the root n to invest with probability x_n, and the allowable remaining probability of an attack v_i. The first and the second log-terms are to ensure that the overall probability of attack does not exceed the allowable limit and player i is playing best-respond strategies, respectively. This is similar to the DL case. Computing each

"table of sets" T's and W's, given above, all take $O(\Delta_x^2 \Delta_y^4)$ each. For n, the *root* of the tree, we compute

$$\begin{aligned}\overline{R}_0(s_0, x_n, y_n, s_n) \equiv & \overline{M}_n(x_n, y_n) + \\ & \log\left(\mathbb{1}[s_0 = s_n + y_n]\right) + \\ & \log\left(\mathbb{1}[x_n \in \mathcal{BR}_n^{\epsilon}(y_n)]\right) + \\ & R_n(x_n, y_n, s_n)\end{aligned}$$

$$R_0(s_0) \equiv \max_{(x_n, y_n, s_n)} \overline{R}_0(s_0, x_n, y_n, s_n)$$

$$W_0(s_0) \equiv \arg\max_{(x_n, y_n, s_n)} \overline{R}_0(s_0, x_n, y_n, s_n)$$

Clearly, computing R_0 and W_0 takes $O(\Delta_x \Delta_y^3)$. As mentioned earlier, for each combination of (x_n, y_n), we are going to compute the best value an attacker can obtain. The computation of R_0 does exactly this.

Downstream pass: assignment phase. The assignment phase is essentially the backtracking phrase in the dynamic programming algorithm where we follow the "back pointers" to find the mixed-strategies for the players and the attacker. For the "downstream" or assignment pass, we are going to start with the root and find $s_0^* \in \arg\max_{s_0} R_0(s_0)$. Because of the discretization result of Corollary 1, there always exists an ϵ-NE, and thus, there is a s_0^* such that $R_0(s_0^*) < -\infty$. We set the mixed-strategy of the root to be some $(x_n^*, y_n^*, s_n^*) \in W_0(s_0^*)$. Starting from the opposite order of upstream pass (i.e., $n-1, ..., 1$), we set the mixed-strategies of the leaves according to $v_{n-1}^* \leftarrow s_n^*$, and for $i = n-1, \ldots, 1$,

$$(x_i^*, y_i^*, s_i^*, v_{i-1}^*) \in W_i(x_n^*, y_n^*, v_i^*) \ .$$

By construction (i.e., the properties of dynamic programming and the discretization), the resulting $(\mathbf{x}^*, \mathbf{y}^*)$ is an ϵ-MSNE of the DS-IDD game.

We now generalize the last result even further to arbitrary DT-IDD games, yielding one of our main technical results.

Theorem 2 *There is an FPTAS to compute an ϵ-MSNE in DT-IDD games.*

Proof Let n denote a site/node in the directed tree with a single source (i.e., the root of the tree). Let $(i_1, \ldots, i_{k_n})$ be a sequence ordering the set of children of n, $\text{Ch}(n) \equiv \{i_1, \ldots, i_{k_n}\}$, where $k_n \equiv |\text{Ch}(n)|$. The following conditions express the dynamic programming corresponding to the "upstream pass" of the algorithm. For all n, *except the root* of the directed tree, we (recursively) define

$$R_n(x_n, y_n, s_n) \equiv T_{i_{k_n}, n}(x_n, y_n, s_n)$$

such that, for all $j = 1, \ldots, k_n$, we define

$$\begin{aligned}T_{i_j, n}(x_n, y_n, v_{i_j}) \equiv & \max_{(x_{i_j}, y_{i_j}, s_{i_j}, v_{i_{j-1}})} \overline{M}_{i_j}(x_{i_j}, y_{i_j}, x_n, y_n) \\ & + \log\left(\mathbb{1}\left[v_{i_j} = s_{i_j} + y_{i_j} + v_{i_{j-1}}\right]\right) \\ & + \log\left(\mathbb{1}\left[x_{i_j} \in \mathcal{BR}_{x_{i_j}}^{\epsilon}(y_{i_j}, x_n, y_n)\right]\right) \\ & + R_{i_j}(x_{i_j}, y_{i_j}, s_{i_j}) \\ & + T_{i_{j-1}, n}(x_n, y_n, v_{i_{j-1}}) \ ,\end{aligned}$$

$W_{i_j, n}(x_n, y_n, v_{i_j})$ is the arg max of the same optimization (i.e., the set of "witnesses" containing the values of $(x_{i_j}, y_{i_j}, s_{i_j}, v_{i_j - 1})$ that achieve the maximum values of the optimization given each (x_n, y_n, v_{i_j})), and, to simplify the presentation, we use the boundary conditions (1) $T_{i_0, n}(x_n, y_n, s_{i_0}) = 0$ for all (x_n, y_n, s_{i_0}); and (2) if i_j is a *leaf* of the tree, then $R_{i_j}(x_{i_j}, y_{i_j}, s_{i_j}) = 0$ for all $(x_{i_j}, y_{i_j}, s_{i_j})$. If n is the *root* of the tree, we compute

$$\begin{aligned}R_0(s_0) \equiv & \max_{(x_n, y_n, s_n)} \overline{M}_n(x_n, y_n) \\ & + \log\left(\mathbb{1}[s_0 = s_n + y_n]\right) \\ & + \log\left(\mathbb{1}[x_n \in \mathcal{BR}_n^{\epsilon}(y_n)]\right) \\ & + R_n(x_n, y_n, s_n) \ , \text{ and}\end{aligned}$$

$W_0(s_0)$ is the arg max of the same optimization (i.e., the set of "witnesses" containing the values of (x_n, y_n, s_n) that achieve the maximum values of the optimization given each s_0 in the discretized grid of probability values in $[0, 1]$).

For the "downstream" or assignment pass, first find $s_0^* \in \arg\max_{s_0} R_0(s_0)$. Note that such s_0^* with $R_0(s_0^*) < -\infty$ because of the properties of the discretization and the existence of MSNE. Set the values of the root node, denoted by n, to some $(x_n^*, y_n^*, s_n^*) \in W_0(s_0^*)$. Then (recursively) set the values of the children of n, in the reversed order in which the the dynamic program computes the maximizations: set $v_{i_{k_n}}^* \leftarrow s_n^*$, and for $j = k_n, \ldots, 1$.

$$(x_{i_j}^*, y_{i_j}^*, s_{i_j}^*, v_{i_{j-1}}^*) \in W_{i_j}(x_n^*, y_n^*, v_{i_j}^*) \ .$$

We repeat the same assignment process for all of the nodes in the tree. Recall that there will always be at least one witness value during the assignment phase because of the properties of the discretization and the existence of MSNE. By construction (i.e., the properties of dynamic programming and the discretization), the resulting $(\mathbf{x}^*, \mathbf{y}^*)$ is an ϵ-MSNE of the DT-IDD game. This completes the proof. □

The running time would be $O(|E| \Delta_x^2 \Delta_y^4)$.

Note that our results are nontrivial within the context of the state-of-the-art in computational game theory. We are working a graph structure where there is one node (the attacker) connecting to *all* the nodes of the tree (the sites). Naively applying the traditional well-known dynamic programming algorithms by Kearns, Littman, and Singh (2001) and Elkind, Goldberg, and Goldberg (2006) to our problem would not give us any FPTAS. In fact, their game representation size is exponential in the number of neighbors instead of our linear representation size.

A Heuristic to Compute ϵ-MSNEs

In this section, we introduce a heuristic to compute ϵ-MSNEs on *arbitrary* graphs. Chan, Ceyko, and Ortiz (2012) showed that *best-response-gradient dynamics (BRGD)* (Fudenberg and Levine 1998; Nisan et al. 2007; Shoham and Leyton-Brown 2009) can efficiently solve *Internet games (IGs)*, as introduced by Chan, Ceyko, and Ortiz (2012) [1],

[1] An IG is an instance of IDD games where the underlying graph structure corresponds to the topology of the Autonomous Systems in the Internet, as measured by from DIMES (Shavitt and Shir 2005). The graph has 26,424 nodes and 100,402 directed edges.

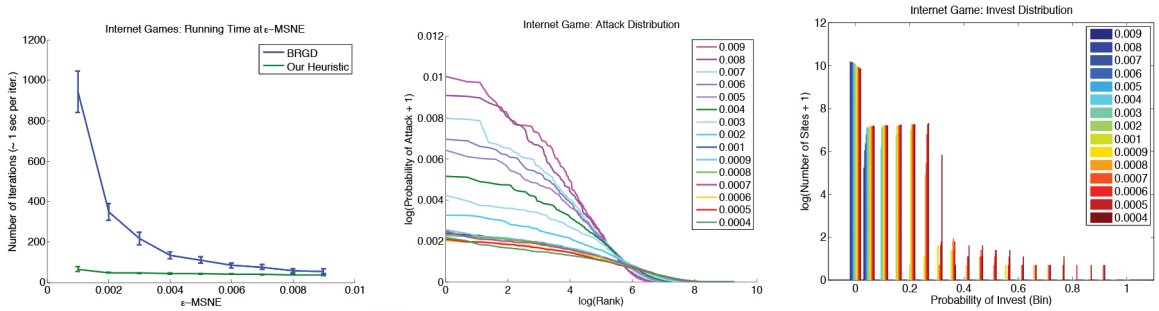

Figure 1: (a) BRGD vs. our heuristic running time; (b) Attacker's attack and sites' investment distribution on ϵ-MSNE

and can output ϵ-MSNEs up to $\epsilon = 0.001$. [2] Here, we evaluate our heuristic using IGs randomly generated according to Table 1 of Chan, Ceyko, and Ortiz (2012).

First we look at the attacker's behavior at an ϵ-MSNE. We generate a few IG instances and run BRGD until it converges to an ϵ-MSNE for $\epsilon \in \{0.001, 0.002, \ldots, 0.009\}$. We observe that in a 0.001-MSNE, (1) there is a positive, almost-deterministic correlation between the probability of an attack and the utility the attacker obtained from attacking the sites and (2) the attacker always target the sites with the highest potential utility (i.e., the maximum utility the attacker can get by attacking the sites with probability 1). This observation is consistent with other IGs and holds across the different ϵ-MSNEs for various ϵ values. Figure 2 shows evidence of this behavior. Indeed, the main take away is that the attacker tends to favor (or target) sites with highest expected utility. As observed, the attack seems to have some distributional form.

In what follows, we assume that the attacker performs *smoothed-best-response* based on the addition of an *entropic penalty/smoothing term* to the attacker's utility (Fudenberg and Levine 1999) leading to a Gibbs-Boltzmann distribution, with penalty parameter $c > 0$, as mixed strategy: $y_i \propto \exp(U_i(x_i, \mathbf{x}_{\mathrm{Ch}(i)}))/c)$. The interpretation of c is that it controls the precision of the attacker and make the utility more distinct. We observe that, as we decrease c, we can obtain a smaller ϵ-MSNE.

This form for the attacker's mixed strategy $\mathbf{y}$ has several attractive properties: (1) sites with high utility will have higher probability of an attack and (2) the respective expected utility and the probability of an attack are positively correlated (higher probability of attack implies higher expected utility gain). We observe these characteristics in our experiments (Figure 2).

Based on the previous discussion, we propose the following heuristic to compute ϵ-MSNEs. The heuristic starts by initializing all of the sites investment level x_i to 0. It then updates the probability of attack for each site and increments

[2] Roughly speaking, BRGD begins by initializing x_i and y_i in $[0, 1]$ for all sites i such that $\sum_{i=1}^{n} y_i \leq 1$. At each round, BRGD update $x_i \leftarrow x_i - 10 * (M_i(1, y_i, \mathbf{x}_{\mathrm{Pa}(i)}, \mathbf{y}_{\mathrm{Pa}(i)}) - M_i(0, y_i, \mathbf{x}_{\mathrm{Pa}(i)}, \mathbf{y}_{\mathrm{Pa}(i)})$ and $y_i \leftarrow y_i + 10 * (U_i(\mathbf{x}) - U(\mathbf{x}, \mathbf{y}))$, where the M_i's and U functions are normalized to $[0, 1]$.

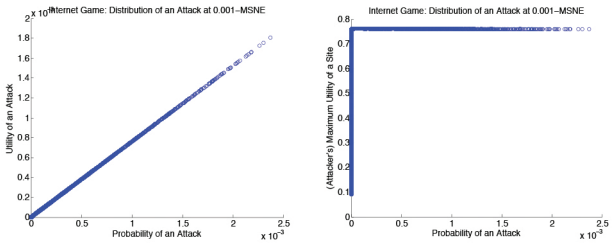

Figure 2: Internet Game: Attack Distribution

the investment level of the site by a small amount (currently 0.001) for sites that do not satisfy the following condition: $R_i \geq y_i \hat{p}_i + (1 - \alpha_i) \sum_{j \in \mathrm{Pa}(i)} y_j (1 - x_j) \hat{q}_{ji}$. The algorithm terminates either when all of the sites satisfy the condition or when it reaches the maximum number of iterations. (See the supplementary material for details.)

The condition just stated above for site i is the threshold for i to not invest. A nice property of this is that given the attacker's Gibbs-Boltzmann distribution, for any site i, given the strategies of others, the attack decreases monotonically with x_i. As a result, no site has an incentive to increase its investment to violate the constraint above. Consequently, to justify the use of the condition in our heuristic in IGs, we observe that in all of the IGs we generated, the percentage of the sites at the 0.001-MSNE we obtained that satisfies the above condition is $\geq 98\%$. The quality of an ϵ-MSNE obtained by our heuristic depends on the percentage of the sites that satisfy the condition at an ϵ-MSNE. Note that if a high percentage of the sites do not satisfy the condition at the ϵ-MSNE, we can reverse the heuristic by initializing all of the sites investment level x_i to 1 and lower the x_i's until all sites satisfy the opposite constraint.

Evaluation of Heuristic on Internet Games

To evaluate our heuristic, we randomly generated ten IGs and compare the results to those obtained using BRGD.

Evaluating Our Proposed Heuristic The first question we address is, what is the relation between the constant c and the actual approximation quality ϵ achieved in practice? Table 1 shows the impact c has on ϵ, for the smallest ϵ-MSNE

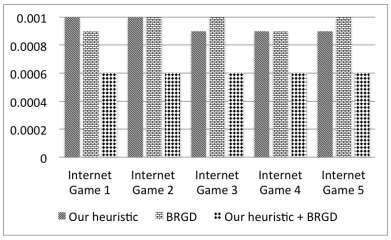

Figure 3: Internet Games: BRGD Improvement

we can obtain for an instance of the IGs (others are similar). Take-home message: ϵ deceases with c. For the remaining of this section, we will fix $c = 0.001$ when comparing to BRGD as BRGD cannot find ϵ-MSNE beyond 0.0009-MSNE within 10,000 iterations (1 sec. per iteration).

c	smallest ϵ
0.05	0.06
0.01	0.008
0.005	0.004
0.001	0.0009
0.0005	0.0006
0.0001	0.0004

Table 1: Selection of the constant c for our heuristic

Comparing Running Time of BRGD and Our Proposed Heuristic Next we study the time that the ten IG instances took to converge to an ϵ-MSNE using BRGD and our heuristic. We consider the running time in terms of the number of iterations the algorithm takes to achieve a particular ϵ-MSNE. Each iteration is roughly 1 sec. for both BRGD and our heuristic. Figure 1(a) shows that the running time of our heuristic is considerably faster than BRGD. The rate at which the number of iterations increases as ϵ decreases seems extreme for our heuristic—it is almost constant!—relative to that for BRGD. Not only is our heuristic faster than BRGD but it can also find better ϵ-MSNE with smaller ϵ.

As an application, we could run our heuristic until it reaches an ϵ-MSNE or converges. Then use the output of our ϵ-MSNE to initialize BRGD. Figure 3 shows the relative improvement over our heuristic on some IGs. It improves our $0.001/0.0009$-MSNE to 0.0006-MSNE.

Attacker and Sites' Equilibrium Behavior We study whether the same equilibrium behavior by the attacker and sites that Chan, Ceyko, and Ortiz (2012) present in their Figure 3 continues as we lower ϵ. The following results are a direct output of our heuristic. Figure 1(b) shows the attack distribution (left) and the investment distribution (right) at ϵ-MSNEs, for different ϵ values, on an IG instance. Our results are consistent with those of Chan, Ceyko, and Ortiz (2012), and persist for lower ϵ values. We see that as ϵ decreases, the attacker targets more sites while lowering the probability of the direct attack, and more sites move from not invest to partially invest.

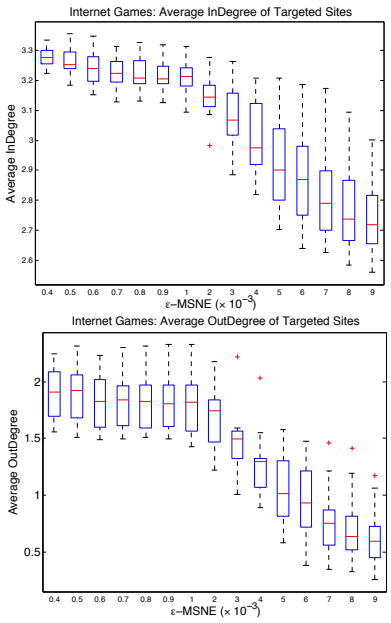

Figure 4: Internet Games: average indegree (top) and average outdegree (bottom) of the targeted sites over ϵ-MSNE

Network Structure of an Attack Next, we present experimental results on the average indegree and outdegree of the targeted sites at ϵ-MSNEs to understand the "network structure of the attack" as in Chan, Ceyko, and Ortiz (2012). Figure 4 shows exactly this. To summarize our experimental results, we can clearly observe that as ϵ decreases both the average indegree and outdegree increase. The results for lower ϵ values indicate the average indegree and outdegree are stabilizing and converging as ϵ decreases. This is also consistent with the observations made by Chan, Ceyko, and Ortiz (2012). This consistency also adds evidence to the effectiveness of our proposed heuristic for very low ϵ values.

Conclusion and Open Problem

We study the problem of computing an ϵ-MSNE in IDD games. We show that determining whether an attacker's strategy can be a part of a MSNE is unlikely to have an efficient algorithm. However, there is an FPTAS to compute an ϵ-MSNE when the underlying game-graph is a directed tree. For general IDD games, we construct a heuristic that computes ϵ-MSNEs in IGs effectively and efficiently. An open problem is to show that computing an MSNE in IDD games is PPAD-hard. Generalizing the FPTAS for DT to directed acyclic graphs of bounded-width of some kind is also open.

Acknowledgements

This material is based upon work supported by an NSF Graduate Research Fellowship (first author) and an NSF CAREER Award IIS-1054541 (second author).

References

Agiwal, S., and Mohtadi, H. 2008. Risk mitigating strategies in the food supply chain. *American Agricultural Economics Assocation (Annual Meeting)*.

Chan, H.; Ceyko, M.; and Ortiz, L. E. 2012. Interdependent defense games: Modeling interdependent security under deliberate attacks. In de Freitas, N., and Murphy, K. P., eds., *UAI*, 152–162. AUAI Press.

Chen, X.; Deng, X.; and Teng, S.-H. 2009. Settling the complexity of computing two-player Nash equilibria. *J. ACM* 56(3):14:1–14:57.

Daskalakis, C.; Goldberg, P. W.; and Papadimitriou, C. H. 2006. The complexity of computing a Nash equilibrium. In *Proceedings of the Thirty-eighth Annual ACM Symposium on Theory of Computing*, STOC '06, 71–78. New York, NY, USA: ACM.

Elkind, E.; Goldberg, L. A.; and Goldberg, P. 2006. Nash equilibria in graphical games on trees revisited. In *Proceedings of the 7th ACM Conference on Electronic Commerce*, EC '06, 100–109. New York, NY, USA: ACM.

Fudenberg, D., and Levine, D. K. 1998. *The Theory of Learning in Games*, volume 1 of *MIT Press Books*. The MIT Press.

Fudenberg, D., and Levine, D. 1999. *The Theory of Learning in Games*. MIT Press.

Fudenberg, D., and Tirole, J. 1991. *Game Theory*. The MIT Press.

Garey, M., and Johnson, D. 1979. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Freeman.

Heal, G., and Kunreuther, H. 2005. IDS models of airline security. *Journal of Conflict Resolution* 49(2):201–217.

Johnson, B.; Grossklags, J.; Christin, N.; and Chuang, J. 2010. Uncertainty in interdependent security games. In *Proceedings of the First International Conference on Decision and Game Theory for Security*, GameSec'10, 234–244. Berlin, Heidelberg: Springer-Verlag.

Kearns, M., and Ortiz, L. E. 2004. Algorithms for interdependent security games. In *In Advances in Neural Information Processing Systems*. MIT Press.

Kearns, M. J.; Littman, M. L.; and Singh, S. P. 2001. Graphical models for game theory. In *UAI '01: Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence*, 253–260. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Kunreuther, H., and Heal, G. 2003. Interdependent security. *Journal of Risk and Uncertainty* 26(2-3):231–249.

Laszka, A.; Felegyhazi, M.; and Buttyan, L. 2014. A survey of interdependent information security games. *ACM Comput. Surv.* 47(2):23:1–23:38.

Nash, J. F. 1950. Equilibrium points in n-person games. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 36, 48–49.

Nisan, N.; Roughgarden, T.; Éva Tardos; and Vazirani, V. V., eds. 2007. *Algorithmic Game Theory*. Cambridge University Press.

Ortiz, L. E. 2014. On sparse discretization for graphical games. arXiv:1411.3320 [cs.AI].

Shavitt, Y., and Shir, E. 2005. DIMES: Let the Internet measure itself. *ACM SIGCOMM Computer Communication Review* 35(5):71–74.

Shoham, Y., and Leyton-Brown, K. 2009. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge, UK: Cambridge University Press.