Efficient Approximation for Security Games with Interval Uncertainty

Christopher Kiekintveld and Vladik Kreinovich

University of Texas at El Paso
Computer Science Department
El Paso, TX 79912

Abstract

There are an increasing number of applications of security games. One of the key challenges for this field going forward is to address the problem of model uncertainty and the robustness of the game-theoretic solutions. Most existing methods for dealing with payoff uncertainty are Bayesian methods which are NP-hard and have difficulty scaling to very large problems. In this work we consider an alternative approach based on interval uncertainty. For a variant of security games with interval uncertainty we introduce a polynomial-time approximation algorithm that can compute very accurate solutions within a given error bound.

Introduction

Security games have been used in recent work to make decisions in a variety of homeland security domains (Pita et al. 2008; Tsai et al. 2009; Pita et al. 2011) and they are increasingly used in cybersecurity applications as well (Alpcan and Basar 2003; Nguyen and Basar 2009). This research is driven by the development of fast algorithms that are able to solve large game models that capture the most important elements of the real world problems (Conitzer and Sandholm 2006; Paruchuri et al. 2008; Jain et al. 2010). One of the key practical concerns with real-world applications of security games is that the game models require very precise and accurate information about the capabilities and preferences of the players to construct. In practice, these models are constructed using elicitation from the security experts (i.e., the defenders making resource allocation decisions). It is particularly difficult for security experts to provide precise values for the payoffs of attackers, given the diversity, secrecy, and lack of information about many adversaries.

For this reason, there is growing emphasis on models and algorithms for security games that handle different types of uncertainty, often using Bayesian games (Paruchuri et al. 2008; Pita et al. 2009; Yin et al. 2011; Kiekintveld, Marecki, and Tambe 2011). All of these approaches share a common goal of increasing our capability to model uncertainty, and to improve the overall robustness of the game-theoretic results. However, most of the approaches listed above suffer from problems with computational scalability and/or solution quality. Exact approaches like Bayesian games are very

difficult to solve computationally, both in theory and in practice. Some scalable approximation techniques have been developed for games with distributional payoff uncertainty, but these lack any guarantees on solution quality.

In this paper we take an approach based on worst-case optimization. We extend the security game model to include interval payoffs for the attacker's values for successful attacks. The defender's objective is defined as maximizing the *worst case* outcome against any possible realization of the attackers' payoff values. The attacker knows her payoffs exactly and chooses a best response. This model is most closely related to the BRASS model introduced by Pita et al. for robustness against human decision-makers (Pita et al. 2009). BRASS does not directly model interval payoffs, but assumes that attackers will chose any target within some ϵ of the payoff for the best target (for the attacker). This type of behavior can be induced by a special case of our model when all of the payoffs have intervals of equal size.

The interval-based approach has some advantages over Bayesian approaches. It does not require a probability distribution over possible payoff values, and therefore it may be simpler for decision-makers to understand, and more straightforward to elicit the necessary values to instantiate the model. Bayesian games are also very difficult to solve computationally, even with recent advances in algorithms for Bayesian Stackelberg security games (Paruchuri et al. 2008; Jain, Tambe, and Kiekintveld 2011) We introduce here a polynomial-time approximation algorithm that is scalable and provides tight bounds on solution quality. This provides a viable technique for incorporating modeling uncertainty into very large security games where Bayesian approaches may be intractable.

Security Games with Interval Uncertainty

Our model is based on the security games first described by Kiekintveld et al. (Kiekintveld et al. 2009). A security game has two players, a *defender*, Θ, and an *attacker*, Ψ. There is a set of n targets $t_i \in T$ that the attacker wishes to attack and the defender wishes to protect. In our model, the attacker can choose to attack exactly one target from this set. The defender has a limited number of resources, $m < n$, that can be deployed to protect the targets. We assume throughout that resources are identical, can protect any target, and that at most one resource can protect each target.

The attacker's set of pure strategies consist of attacking each of the n targets. The defender's set of pure strategies comprise all possible ways to assign the m resources to the n targets. However, we can conveniently summarize the defender's strategy by defining the *coverage vector* which gives the probability that there is a defender resource assigned to each individual target. Let us denote these probabilities by c_i, so that $\sum_{i=1}^{n} c_i = m$. The vector of coverage probabilities we denote by C.

If the attacker chooses to attack target t_i, we call the attack *successful* if the target is left uncovered by the defender, and *unsuccessful* if the target is covered. In the original security game model, there are four payoff values defined for each target. The defender's payoff for an uncovered attack is denoted $U_{\Theta}^{u}(t)$, and for a covered attack $U_{\Theta}^{c}(t)$. Similarly, $U_{\Psi}^{u}(t)$ and $U_{\Psi}^{c}(t)$ are the attacker's payoffs in each case.

We modify this payoff structure in two ways. First, for simplicity we consider a restricted case where the payoffs for an unsuccessful attack are zero for both players ($U_{\Theta}^{c}(t) = U_{\Psi}^{c}(t) = 0$ for all t). Second, we introduce payoff intervals to represent the attacker's payoffs. Rather than a single value $U_{\Psi}^{u}(t)$ for each target, we have a pair of values, $U_{\Psi}^{u,max}(t)$ and $U_{\Psi}^{u,min}(t)$, that represent the maximum and minimum values the attacker could get for a successful attack on target t. The idea of this approach is that the defender knows only that the attacker's payoffs lie within some possible range of values, and not the precise value. For now we continue to represent the defender's payoffs using a single value for each target, consistent with the idea that the defender has greater knowledge of their own preferences than those of the attacker.

Security games are typically modeled as *Stackelberg games* in which the attacker can observe the defender's strategy $(c_1, \ldots, c_n)$ before planning an attack (modeling the capability of attackers to use surveillance to learn security policies). We keep this same basic assumption in our interval security games, but we cannot use the standard solution concept of Strong Stackelberg Equilibrium because the defender's expected payoff is not well-defined. We assume that the attacker knows the realization of the payoffs within the specified payoff intervals, but the defender knows only that the payoffs lie within the given intervals. Therefore, the defender does not have information about the distribution of the payoffs within these intervals, and cannot compute an expected payoff. Instead, we follow the literature on robust optimization and take a worst-case approach. The defender's goal in our framework is to select a coverage vector, C, that maximizes the defenders worst-case payoff over all of the possible ways that the attacker payoffs could be chosen from the defined intervals.

Analysis

In security games without intervals, we can define the *attack set* to be the set of all targets that give the attacker the maximum expected payoff, given some coverage strategy C. For some classes of security games, finding the optimal coverage strategy can be reduced to finding a coverage strategy

that induces the maximum attack set, while minimizing the attacker's expected payoff. This results is the basis of the ORIGAMI algorithm (Kiekintveld et al. 2009).

In our model we cannot directly apply the idea of the attack set, but we can generalize this idea as follows. We define the *potential attack set* for a coverage strategy C to be the set of all targets that could give the attacker the maximum expected value, for any realization of attacker payoffs consistent with the payoff intervals. For every target, the attacker has a range of expected payoffs:

$$v^{max}(t_i) = (1 - c_i) \cdot U_{\Psi}^{u,max}(t_i) \qquad (1)$$

$$v^{min}(t_i) = (1 - c_i) \cdot U_{\Psi}^{u,min}(t_i). \qquad (2)$$

Observe that the attacker is always guaranteed a payoff of at least the maximum of the minimum values over all targets; let us denote this value by $R = \max_{t_i} v^{min}(t_i)$. Given the value of R we can identify the targets that could be attacked. Any target t_i with a maximum expected value $v^{max}(t_i) \geq R$ could be the best target for the attacker to attack. To see this, suppose that the value for t_i is maximal, and the values for all other targets is minimal, so that the best possible value for attacking any target other than t_i is R. Therefore, the potential attack set, $\Lambda(C)$, is defined as:

$$\Lambda(C) = \{t_i : v^{max}(t_i) \geq R\} \qquad (3)$$

The defender's expected payoff for each target is:

$$d_i = (1 - c_i) \cdot U_{\Theta}^{u}(t_i). \qquad (4)$$

The defender'a objective is to select a strategy C to maximize the worst-case payoff over all of the targets in the potential attack set:

$$\max_{C} (\min_{t_i \in \Lambda(C)} d_i) \qquad (5)$$

The main idea of our algorithm is to use a binary search through the space of possible defender payoffs to turn this optimization problem into a series of feasibility problems. This is possible because the defender's expected payoff increases monotonically with the number of available resources (this follows directly from the fact the that the defender's strategy space becomes strictly larger as we add additional resources). Suppose that we wish to determine whether some defender payoff D^* is feasible given the number of resources available, m. For every target, one of two conditions must hold to guarantee the defender D^*:

1. The target is in the potential attack set, but the defender's expected payoff for attacking the target is greater than D^*

2. The target is not in the potential attack set.

We can easily calculate the coverage required on each target to satisfy condition 1 for each target (if it is in Λ) from the equation for the defender's payoff. The minimal coverage for each target is given by:

$$c_i^{1} = \max(0, 1 - \frac{D^*}{U_{\Theta}^{u}(t_i)}). \qquad (6)$$

So, the problem reduces to finding the set of targets for the potential attack set that will minimize the overall coverage probability required to meet conditions 1 and 2 for all

targets. A naïve approach would be to enumerate all of the possible attack sets and calculate the minimum coverage for each such set. For any given set, we can calculate the value of R, and the minimal coverage required for each target in Λ from Equation 6. Given the value of R we can also calculate the minimum coverage on every target that is *not* in Λ so that the maximum expected attacker for the target is not greater than R:

$$c_i^2 = \max(0, 1 - \frac{R}{U_\Psi^{u,max}(t_i)}). \qquad (7)$$

By summing the values of c_i^1 for targets in Λ and c_i^2 for the remaining targets, we get the minimum coverage required to guarantee D^* for this potential attack set. Unfortunately, the number of such sets is exponential in the number of targets, so this approach is highly inefficient. To avoid this problem we make another observation. For every possible set Λ there will be some target $\hat{t}$ that has the maximum minimum expected payoff R. There are only n targets, so if we test each of these targets as a possible $\hat{t}$ and construct the set Λ with minimal coverage for this choice we only need to explore a linear number of such cases. If any one of these cases is feasible (i.e., the coverage required is less than m), the value of D^* is feasible. In the following section we describe an algorithm that uses this solution strategy to efficiently approximate the optimal coverage vector C for the defender.

Algorithm for Interval Security Games

We call our algorithm ISEGS for *Interval SEcurity Game Solver*. The pseudocode is given in Algorithms 1 and 2. Algorithm 1 is a straightforward binary search in the space of possible defender payoffs. The feasibility check is presented in Algorithm 2, based on the analysis presented above. It iterates through every possible target as a candidate $\hat{t}$. For $\hat{t}$ we know that Equation 6 must hold, since the target is in Λ by definition. We take the value of $c_{\hat{t}}^1$ for and calculate the resulting value of R. Since the values of c_i^1 are independent of R and the values of c_i^2 are monotonically increasing as R decreases, it can only increase the total coverage required if we increase the coverage on $\hat{t}$ beyond the minimal $c_{\hat{t}}^1$.

Given the value of R, we calculate the value of c_i^2 for every other target. We can then calculate the minimum coverage required to satisfy one of the two conditions described in the previous section by taking $\min(c_i^1, c_i^2)$ for each target. There is one final condition that must be met for our initial assumption to hold: the calculated value of R must actually be the minimum expected attacker payoff. We guarantee this for each target by calculating one additional constraint:

$$c_i^3 = \max(0, 1 - \frac{R}{U_\Psi^{u,min}(t_i)}). \qquad (8)$$

This is the minimum coverage for each target other than $\hat{t}$ so that the initial assumption holds. The final calculation for the minimum coverage for each target becomes $\max(c_i^3, \min(c_i^1, c_i^2))$. We sum these coverages over all of the targets and compare this with the available resources m

to determine whether this selection of $\hat{t}$ yields a feasible solution. If not, we continue testing until all possible targets have been tried.

The worst-case complexity of the algorithm is $O(n^2 \cdot \log(1/\epsilon))$ where ϵ is the error tolerance parameter for the binary search. Each feasibility check requires one iteration to test each target as $\hat{t}$, and each iteration does several constant-time operations on each target to determine the minimal coverage. Therefore, the feasibility check is $O(n^2)$. Binary search requires $O(\log(1/\epsilon))$ iterations to converge within ϵ, giving the overall complexity of $O(n^2 \cdot \log(1/\epsilon))$.

Algorithm 1 ISEGS

> **for all** $t_i \in T$ **do**
> $\quad c_i \leftarrow 0$
> **end for**
> $maxPayoff \leftarrow 0$
> $minPayoff \leftarrow \min_{t_i \in T} U_\Theta^u(t_i)$
> **while** $maxPayoff - minPayoff > \epsilon$ **do**
> $\quad midPoint \leftarrow (maxPayoff - minPayoff)/2$
> $\quad$ **if** feasibilityCheck($midPoint, m, C$) **then**
> $\quad\quad minPayoff \leftarrow midPoint$
> $\quad$ **else**
> $\quad\quad maxPayoff \leftarrow MidPoint$
> $\quad$ **end if**
> **end while**
> **return** C

Algorithm 2 feasibilityCheck

> **for all** $t_i \in T$ **do**
> $\quad c_i^1 \leftarrow \max(0, 1 - \frac{midPoint}{U_\Theta^u(t_i)})$
> **end for**
> **for all** $t_i \in T$ **do**
> $\quad totalCov \leftarrow c_i^1$
> $\quad c_i \leftarrow c_i^1$
> $\quad R \leftarrow (1 - c_i^1) \cdot U_\Psi^{u,min}(t_i) - \epsilon'$
> $\quad$ **for all** $t_j \in \{T \backslash t_i\}$ **do**
> $\quad\quad c_j^2 \leftarrow \max(0, 1 - \frac{R}{U_\Psi^{u,max}(t_j)})$
> $\quad\quad c_j^3 \leftarrow \max(0, 1 - \frac{R}{U_\Psi^{u,min}(t_j)})$
> $\quad\quad minCov \leftarrow max(c_i^3, \min(c_i^1, c_i^2))$
> $\quad\quad totalCov \leftarrow totalCov + minCov$
> $\quad\quad c_j \leftarrow minCov$
> $\quad$ **end for**
> $\quad$ **if** $totalCov \leq m$ **then**
> $\quad\quad$ **return** TRUE, C
> $\quad$ **end if**
> **end for**
> **return** FALSE

In addition to ISEGS we have specified a mixed-integer program that computes an exact solution for our interval security games. This MIP model is used as a benchmark in the experimental evaluation. We do not described this MIP in detail here due to space constraints, but note that the formulation is a (relatively minor) generalization of the BRASS MIP formulation presented in Pita et al. (Pita et al. 2009).

Experimental Evaluation

We tested our algorithm against an exact MIP formulation on a class of randomly generated game instances. Defender payoffs for uncovered attacks were uniformly distributed between 0 and -100. The minimum attacker payoff was uniformly distributed between 0 and 100. The range for attacker payoffs was randomly generated between 0 and 20, with this value added to the minimum payoff to get the maximum payoff for the target (the size of the intervals is different for each target). The number of resources is fixed at 20% of the number of targets. We ran 30 sample games instances to test the average solution times.

In Figure 1 we present results for the MIP (solved using GLPK version 4.36) and ISEGS with three different tolerance settings. All three ISEGS algorithms are much faster, even with an error tolerance of just 0.0001. Figure 2 shows results for the three ISEGS settings on much larger games. Here we see a modest increase in solution time with increasing accuracy. Even for 10000 targets and the highest accuracy setting, ISEGS solves the game in half the time required by the MIP to solve games with only 300 targets.
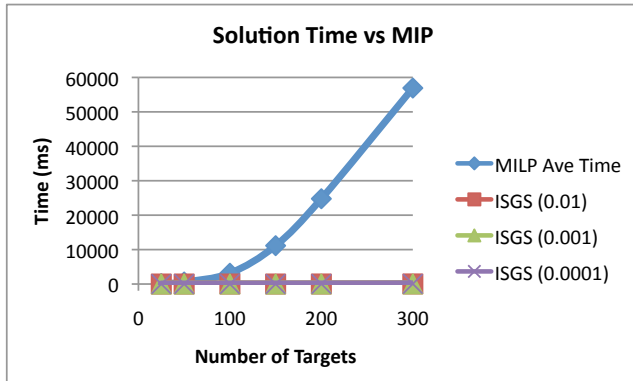

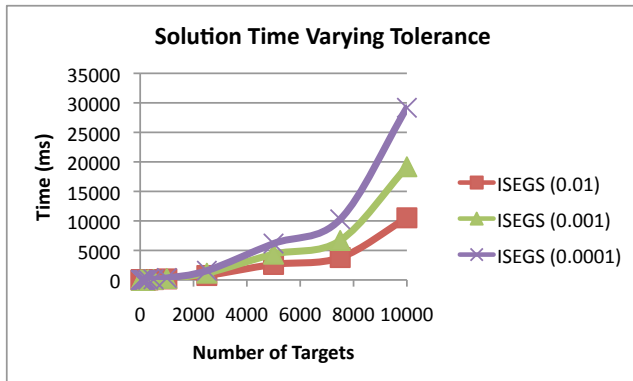Figure 1: Solution time comparison between ISEGS and MIP formulation.

Figure 2: Solution time for ISEGS on larger games with varying error tolerance.

Conclusion

We introduce a new security game model with interval uncertainty as an alternative to Bayesian approaches. For this model we provide an analysis that leads to a polynomial-time approximation algorithm for calculating defender strategies. This method has bounded error and can quickly calculate solutions within very small tolerances of the optimal solution. Empirical results show much faster performance than an exact MIP formulation.

References

Alpcan, T., and Basar, T. 2003. A game theoretic approach to decision and analysis in network intrusion detection. In *Proc. of the 42nd IEEE Conference on Decision and Control*, 2595–2600.

Conitzer, V., and Sandholm, T. 2006. Computing the optimal strategy to commit to. In *ACM EC-06*, 82–90.

Jain, M.; Kardes, E.; Kiekintveld, C.; Tambe, M.; and Ordonez, F. 2010. Security games with arbitrary schedules: A branch and price approach. In *AAAI-10*.

Jain, M.; Tambe, M.; and Kiekintveld, C. 2011. Quality-bounded solutions for finite bayesian stackelberg games: Scaling up. In *AAMAS-11*.

Kiekintveld, C.; Jain, M.; Tsai, J.; Pita, J.; Ordonez, F.; and Tambe, M. 2009. Computing optimal randomized resource allocations for massive security games. In *AAMAS-09*.

Kiekintveld, C.; Marecki, J.; and Tambe, M. 2011. Approximation methods for infinite bayesian stackelberg games: Modeling distributional payoff uncertainty. In *AAMAS-11*.

Nguyen, K. C., and Basar, T. A. T. 2009. Security games with incomplete information. In *Proc. of IEEE Intl. Conf. on Communications (ICC 2009)*.

Paruchuri, P.; Pearce, J. P.; Marecki, J.; Tambe, M.; Ordonez, F.; and Kraus, S. 2008. Playing games with security: An efficient exact algorithm for Bayesian Stackelberg games. In *AAMAS-08*, 895–902.

Pita, J.; Jain, M.; Western, C.; Portway, C.; Tambe, M.; Ordonez, F.; Kraus, S.; and Parachuri, P. 2008. Depoloyed ARMOR protection: The application of a game-theoretic model for security at the Los Angeles International Airport. In *AAMAS-08 (Industry Track)*.

Pita, J.; Jain, M.; Ordóñez, F.; Tambe, M.; Kraus, S.; and Magori-Cohen, R. 2009. Effective solutions for real-world stackelberg games: When agents must deal with human uncertainties. In *AAMAS-09*.

Pita, J.; Tambe, M.; Kiekintveld, C.; Cullen, S.; and Steigerwald, E. 2011. Guards - game theoretic security allocation on a national scale. In *AAMAS-11 (Industry Track)*.

Tsai, J.; Rathi, S.; Kiekintveld, C.; Ordóñez, F.; and Tambe, M. 2009. IRIS - A tools for strategic security allocation in transportation networks. In *AAMAS-09 (Industry Track)*.

Yin, Z.; Jain, M.; Tambe, M.; and Ordonez, F. 2011. Risk-averse strategies for security games with execution and observational uncertainty. In *AAAI-11*.