Game-Theoretic Allocation of Security Forces in a City

Jason Tsai, Zhengyu Yin, Jun-young Kwak, David Kempe, Christopher Kiekintveld, Milind Tambe

University of Southern California, Los Angeles, CA 90089
{jasontts, zhengyuy, junyounk, dkempe, kiekintv, tambe} @usc.edu

ABSTRACT

Law enforcement agencies frequently must allocate limited resources to protect targets embedded in a network, such as important buildings in a city road network. Since intelligent attackers may observe and exploit patterns in the allocation, it is crucial that the allocations be randomized. We cast this problem as an attacker-defender Stackelberg game: the defender's goal is to obtain an optimal mixed strategy for allocating resources. The defender's strategy space is exponential in the number of resources, and the attacker's exponential in the network size. Existing algorithms are therefore useless for all but the smallest networks.

We present a solution approach based on two key ideas: (i) a polynomial-sized game model obtained via an approximation of the strategy space, solved efficiently using a linear program; (ii) two efficient techniques that map solutions from the approximate game to the original, with proofs of correctness under certain assumptions. We present in-depth experimental results, including an evaluation on part of the Mumbai road network.

Categories and Subject Descriptors

I.2.11 [**Computing Methodologies**]: Artificial Intelligence—*Distributed Artificial Intelligence - Intelligent Agents*

General Terms

Algorithms, Performance, Experimentation, Security, Theory

Keywords

Game Theory, Stackelberg Games, Algorithms, Uncertainty, Security, Randomization, Patrolling, Risk Analysis

1. INTRODUCTION

Protecting targets against potential attacks is an important problem for security forces worldwide. The general setting is as follows: An attacker assigns different values to reaching (and damaging or destroying) one of multiple targets. A defender wants to allocate resources (such as patrol cars or canine units) to capture the attacker before he reaches a target. In many of these situations, the domain has structure that is naturally modeled as a graph. For example, city maps can be modeled with intersections as nodes and roads as edges, where nodes are targets for attackers. In order to prevent attacks, security forces can schedule checkpoints on edges (e.g., roads) to detect intruders. For instance, in response to the devastating terrorist attacks in 2008 [3], Mumbai police deploy randomized checkpoints as one countermeasure to prevent future attacks [1]. The strategy for placing these checkpoints must necessarily be decided in advance of attack attempts, should account for targets of differing importance, and should anticipate an intelligent adversary who can observe the strategy prior to attacking.

In light of these requirements, game-theoretic approaches have been developed to assist in generating randomized security strategies in several real-world domains, including applications in use by the Los Angeles International Airport [14] and the Federal Air Marshals Service [15]. To account for the attacker's ability to observe deployment patterns, these methods model the problem as a Stackelberg game and solve for an optimal probability distribution over the possible deployments to ensure unpredictability. Novel solvers for classes of security games have recently been developed [2, 13, 4]. However, these solvers take time at least polynomial in the number of actions of both players. In our setting, every path from an entry point to a target is an attacker action, and every set of r or fewer edges is a defender action. (r is the maximum number of checkpoints.) Since the attacker's actions grow exponentially with the size of the network, and the defender's actions grow exponentially with r, existing methods quickly become too slow when applied to large real-world domains.

In this work, we develop an efficient procedure for generating checkpoint deployments based on two key ideas: (i) a polynomial-sized approximation of the strategy space solved using a linear program; (ii) two efficient sampling techniques to map solutions back to the original space. To avoid the exponential strategy space over all possible combinations of checkpoint placements (the joint distribution), our methods operate on the marginal probabilities of edges, i.e., the total probability of placing a checkpoint on an edge. Our linear program, RANGER, upper-bounds the capture probabilities along paths by the sum of marginal probabilities.

Our sampling algorithms efficiently generate joint distributions in the original problem space from RANGER's solution of marginal probabilities. We prove that under certain conditions, the actual capture probabilities of our algorithms match the upper bounds of RANGER, and thus necessarily give *optimal* payoff. *Radius Sampling* generates optimal joint distributions if certain conditions on the marginal distribution are met. *Comb Sampling* generates distributions which are optimal against an *approximating attacker* who calculates the expected value of an attack by summing the marginal probabilities on the path.

In addition to our theoretical results, we test our methods empirically. First, we evaluate the quality of RANGER against an optimal solution technique, DOBSS, to verify the accuracy of RANGER's approximation. Then, we evaluate the sampling procedures by testing against an *exact attacker* who plays a best response to the defender's true joint distribution. We also apply our methods to a game model of the city of Mumbai and the targets attacked in 2008.

2. RELATED WORK

Aside from the literature on Stackelberg games for security, our approach is also based on insights from network interdiction [17, 16, 9, 5]. These are the special case of our model when there is a single target, or — equivalently — all targets have identical values. For such games, Washburn and Wood (1995) give an algorithm finding optimal strategies for both players based on Min-Cut computations. However, different target values can cause their algorithm to perform arbitrarily poorly, as we see in our experiments.

Two additional lines of work are somewhat related. Mavronicolas et al. (2008) define and analyze a network security game where each attacker can attack any node of the network, and the defender chooses a path to patrol to capture as many attackers as possible. Because the attacker is not restricted to paths, the types of results for this game are different from ours, and the focus in [12, 11] is on understanding the impact of selfish behavior by defenders rather than optimal strategies. Hider-seeker games [2, 8] are also studied on graphs, but here, the attacker's goal is only to evade capture, not to reach any particular target.

3. PROBLEM DESCRIPTION

A graph-based security game models an attacker and a defender who take actions on a graph $G = (V, E)$, with $n = |V|$ nodes and $m = |E|$ edges. The attacker starts at one of the source nodes $s \in S \subseteq V$ *of his choosing* and travels along a path in an attempt to reach one of the targets $t \in T \subseteq V$. The attacker's pure strategies are thus all s-t paths P, denoted by $\mathcal{B}$, from some source s to some target t. The defender tries to capture the attacker before he reaches a target, by placing up to r resources on *edges* of the graph. The defender's pure strategies are subsets of r or fewer edges; we denote the set of all such sets by $\mathcal{L}$. Assuming that the defender plays $L \in \mathcal{L}$ and the attacker $P \in \mathcal{B}$, the attacker is captured whenever $P \cap L \neq \emptyset$, and succeeds in his attack when $P \cap L = \emptyset$.

Unsuccessful attacks always have a payoff of c for the defender, while successful ones have a penalty of $D(t)$. We make the natural restriction that $D(t) \leq c$. We also assume that the game is zero-sum, meaning that the attacker's payoff for a successful attack on target t is $-D(t)$, and $-c$ for an unsuccessful one. We stress here that targets may have vastly different payoffs associated with them, unlike in [16]. This distinction is crucial to model real security domains, and thus to bridge the gap between theory and practice.

Since the attacker can choose which source to enter from, for our analysis, we merge all sources into a *single source* without loss of generality. More formally, we reform the graph so that all original source-incident edges are incident to the new source. While this operation obviously changes the graph, it does so only in a way that does not impact the game: no rational attacker would ever include multiple sources in his path, and therefore, a defender will never select an edge between two sources. Those are the only edges that disappear from the problem. Thus, to simplify presentation and analysis, we will assume that the attacker always enters from a unique known source s.

In a world of increasingly sophisticated and determined attackers, a good defender strategy must take into account the fact that the attacker will observe and exploit patterns in the defender's behavior. Thus, the game is naturally modeled as a Stackelberg game, an approach also taken (for the same reasons) in past work in security settings [7, 10]. The defender is modeled as the *leader* and moves first, by selecting a mixed strategy $\lambda \in \Lambda$ that assigns a probability to each pure strategy $L \in \mathcal{L}$. The attacker is the *follower* and chooses a strategy after observing the defender's mixed strategy. There is always a pure-strategy best response for the attacker, so we restrict the attacker to pure strategies without loss of generality. Thus, the attacker's Stackelberg strategy is a function $f : \lambda \mapsto P$. For any pair of strategy profiles (λ, f), the expected rewards for the defender (R_D) and attacker (R_A) are given by:

$$R_D(\lambda, f) = p \cdot c + (1 - p) \cdot D(t) \quad (1)$$

$$R_A(\lambda, f) = p \cdot -c + (1 - p) \cdot -D(t), \quad (2)$$

where t is the target at the end of the path specified by $f(\lambda)$, and p the probability that the attacker is captured on the path to t given the defender's strategy λ. Although the optimal defender strategy is a Stackelberg Equilibrium, since our game is zero-sum, this is equivalent to a Maximin strategy [6]. Unfortunately, as $\mathcal{L}$ has size $\Theta(m^r)$, and $\mathcal{B}$ has size exponential in n, existing methods for computing such strategies do not scale to realistic problem sizes. We therefore develop a linear program and two accompanying sampling methods to efficiently solve graph-based security games.

4. RANGER

We first introduce RANGER (Resource Allocation for Network-based Games with Efficient Representations), a linear program for finding an optimal set of *marginal* checkpoint probabilities for the defender. We denote the marginal probability associated with edge e by x_e. Formally, $x_e = \sum_{L \in \mathcal{L}, e \in L} \lambda_L$, where λ_L is the probability of the set L under λ. We denote the marginal distribution by $\vec{x} = \langle x_e \rangle$.

By reasoning over $\vec{x}$, we avoid the exponential size of the defender's space. The key insight of our approach is the following simple consequence of the Union Bound: *For any path P, the capture probability under λ is at most $\sum_{e \in P} x_e$.* We use this upper bound (the sum of x_e) as an approximation of the true capture probability in deriving RANGER. The power of our approach is that we subsequently present ways to sample joint distributions where the total capture probability *matches* this upper bound under certain conditions; this immediately implies optimality of our procedures, and retroactively justifies the approximation.

In the RANGER linear program below, x_e is the marginal probability of placing a checkpoint on edge e. The d_v are, for each vertex v, the minimum sum of checkpoint probabilities along any path from the source s to v[1]. This is enforced by the constraints (4)–(6). Constraint (7) enforces that at most r checkpoints are placed, and Constraint (3) captures the payoff for the defender.

Maximize R_D, s.t.:

$$R_D \leq (1 - d_t) \cdot D(t) + d_t \cdot c \quad (3)$$

$$d_s = 0 \quad (4)$$

$$d_v \leq \min(1, d_u + x_e) \qquad \forall e = (u, v) \quad (5)$$

$$0 \leq x_e \leq 1 \qquad \forall e \in E \quad (6)$$

$$\sum_{e \in E} x_e \leq r \quad (7)$$

Notice that, as specified, the d_v values do not have a lower bound. However, we can show that in an optimal solution, we can, without loss of generality, raise all d_v values to their upper bound, the shortest sum of marginals of a path to v.

THEOREM 1. *Let $(R_D^*, \vec{x}^*, \vec{d}^*)$ be an optimal solution returned by RANGER. Define $\vec{d}' = \langle d'_v \rangle$ for $v \in V$, where $d'_v = \min(1, \min_{P \in \mathcal{P}_v} \sum_{e \in P} x_e)$. $\mathcal{P}_v$ is the set of paths from s to v. Then $(R_D^*, \vec{x}^*, \vec{d}')$ is also an optimal solution.*

[1]Recall that we can assume a single source w.l.o.g.

PROOF. We will show that $(R_D^*, \vec{x}^*, \vec{d'})$ satisfies all RANGER constraints and leads to a reward greater than or equal to R_D^* and, therefore, must also be optimal.

From Constraint 5 in RANGER, for any node v, $d_v^* \leq d'_v$. By definition of $\vec{d'}$, $d'_s = 0$. In addition, consider edge $e' = (u, v)$. Let $P_u = \arg\min_{P \in \mathcal{P}_u} \sum_{e \in P} x_e$. By definition $d'_v \leq 1$. And,

$$d'_v = \min\left(1, \min_{P \in \mathcal{P}_v} \sum_{e \in P} x_e\right) \leq \min\left(1, \sum_{e \in P_u} x_e + x_{e'}\right)$$
$$\leq \min\left(1, \sum_{e \in P_u} x_e\right) + x_{e'} = d'_u + x_{e'}.$$

Hence, $d'_v \leq \min(1, d'_u + x_{e'})$. Finally, given $c \geq D(t)$ and $d'_t \geq d_t^*$,

$$R_D^* \leq (1 - d_t^*)D(t) + (d_t^*)c$$
$$\leq (1 - d'_t)D(t) + (d'_t)c.$$

Thus, $(R_D^*, \vec{x}^*, \vec{d'})$ satisfies all RANGER constraints and leads to a reward greater than or equal to R_D^*. Since R_D^* is the optimal value of RANGER, $(R_D^*, \vec{x}^*, \vec{d'})$ must also be an optimal solution to RANGER. □

Thus, although RANGER may not produce these maximal d_v values, in an optimal solution each d_v can be set to exactly the shortest sum of marginals to v without loss of generality. We will define d_v as such going forward to simplify our analysis.

We verify the claim that RANGER's solution is an overestimate of an optimal solution.

THEOREM 2. *Let λ^* be the optimal strategy and R^* the corresponding defender reward. Then, $R_D^* \geq R^*$, where R_D^* is the defender reward returned by the LP.*

PROOF. Let $\vec{x^*}$ be the marginal probabilities of λ^*. Obviously, $0 \leq x_e^* \leq 1$, and $\sum_{e \in E} x_e^* = \sum_{L \in \mathcal{L}} |L| \cdot \lambda_L^* \leq r$. For each vertex v, let d_v^* be the probability (under λ^*) of capturing the intruder assuming he chooses the best path to reach v. Then $0 \leq d_v^* \leq 1$, and for each edge $e = (u, v)$, $d_v^* \leq d_u^* + x_e^*$, by the Union Bound. The attacker will choose the path to maximize his own payoff R_A^*. Because the game is zero-sum,

$$R^* = -R_A^* = -\max_t\{(1 - d_t^*) \cdot -D(t) + d_t^* \cdot -c\}$$
$$= \min_t\{(1 - d_t^*) \cdot D(t) + d_t^* \cdot c\}.$$

Thus, for any target t, $R^* \leq (1 - d_t^*) \cdot D(t) + d_t^* \cdot c$. Thus, the values R^*, $\vec{d^*}$ and $\vec{x}^*$ are feasible for the LP; because RANGER finds the optimum feasible solution, we obtain that $R_D^* \geq R^*$. □

RANGER is an exponentially more compact representation of both the attacker and defender strategy spaces. This can be seen by noticing that RANGER has a polynomial number of variables with respect to n and m. Any straightforward application of prior formulations would have $\Theta(m^r)$ variables for the defender and exponentially many ($|\mathcal{B}|$) constraints for the attacker.

5. CREATING JOINT DISTRIBUTIONS

To deploy security resources, we require joint schedules, drawn from a joint distribution over $\mathcal{L}$. We develop sampling procedures that use the $\vec{x}$ computed by RANGER to generate a distribution over joint schedules. The principle behind these methods is to bring the *actual* capture probability for a target t up to the value d_t. One way to ensure this would be if no deployment ever placed two checkpoints on any s-t path to any target. More generally (and informally), it is sufficient if this "checkpoint disjointness" is ensured for "critical" s-t paths: those whose sum of marginal probabilities are close to the minimal (d_t).

Notice that placing checkpoints by independently sampling from $\vec{x}$ violates this approach with increasing frequency as r increases, and yields very suboptimal solutions. Instead, we introduce two novel sampling procedures that achieve a certain "checkpoint disjointness", under some assumptions, and are therefore *optimal*.

5.1 Radius Sampling

Radius Sampling (RS) interprets the marginal probabilities as "distances", and places a security ring around the source. In this way, it avoids sampling multiple times on "critical paths", in a way we make precise now. For any $h \geq 0$, we define the *ring of radius* h around s as $R_h := \{e = (u, v) | d_u \leq h < d_v\}$, i.e., the set of edges from a node with probability of capture at most h from s to a node with probability of capture more than h from s.

We define $\alpha := \int_0^\infty |R_h| dh$ (a normalization constant), and the density function $\phi(h) := \frac{|R_h|}{\alpha}$. Notice that

$$\alpha \quad = \quad \sum_{e=(u,v)}(d_v - d_u) \quad \leq \quad \sum_e x_e \leq r. \tag{8}$$

Our algorithm works as follows: Choose a radius h from $[0, \infty]$ according to the density function ϕ. Now, choose r of the edges in R_h uniformly at random (or all edges in R_h if $|R_h| \leq r$). Place checkpoints on these edges. We call the resulting set L_R. Notice that both h and L_R are random variables, and L_R is a set of at most r edges.

THEOREM 3. *If for all h, $|R_h| \geq r$ or $|R_h| = 0$, then RS produces an optimal distribution for the defender.*

Theorem 3 follows from Lemma 4 and Theorem 2 as follows: By Lemma 4, the capture probability for any s-t path is at least d_t, i.e., RANGER's value. Therefore, the defender's payoff is at least RANGER's, which by Theorem 2 is at least the payoff with the optimum mixed strategy.

LEMMA 4. *Under the assumptions of Theorem 3, let P be any s-v path and w the node maximizing d_w among all nodes on P. The capture probability along P is at least d_w.*

PROOF. We prove the lemma by induction on $|P|$, the number of edges on path P. In the base case $|P| = 0$, the only node v with a path from s is s itself, and the statement holds.

For the inductive step, let P be a path of length $\ell + 1$ and $e = (v', v)$ the last edge of P. Let $P' = P \setminus \{e\}$ be the path of length ℓ from s to v', and w' the node on P' maximizing $d_{w'}$. By Induction Hypothesis, $\text{Prob}[L_R \cap P' \neq \emptyset] \geq d_{w'}$.

We distinguish two cases. If $d_{w'} \geq d_v$, then

$$\text{Prob}[L_R \cap P \neq \emptyset] \quad \geq \quad \text{Prob}[L_R \cap P' \neq \emptyset]$$
$$\geq \quad d_{w'} \geq d_v,$$

implying the claim.

If $d_v > d_{w'}$, then consider the event $\mathcal{E} = [h > d_{w'} \text{ and } e \in L_R]$. $\mathcal{E}$ is the event when we include e in L_R and h is sufficiently large that no edge from P' can also be sampled. The probability of $\mathcal{E}$ is

$$\int_{d_{w'}}^{d_v} \text{Prob}[e \in L_R \mid h = x]\phi(x)dx$$
$$= \int_{d_{w'}}^{d_v} \frac{r}{|R_x|} \cdot \frac{|R_x|}{\alpha} dx = \int_{d_{w'}}^{d_v} \frac{r}{\alpha} dx \geq d_v - d_{w'}.$$

Here, we substituted the definitions of the sampling process, then used that $\frac{r}{\alpha} \geq 1$ from Equation (8), and that $\text{Prob}[e \in L_R \mid h = x] = \frac{r}{|R_x|}$ using the assumption of Theorem 3.

Whenever L_R intersects P', by definition, we must have that $h \leq d_{w'}$ (because no edge $e' \in P'$ is in R_h for $h > d_{w'}$). Thus, the events $\mathcal{E}$ and $[R_h \cap P' \neq \emptyset]$ are disjoint, and

$$\begin{aligned} \text{Prob}[R_h \cap P \neq \emptyset] &\geq \text{Prob}[[R_h \cap P' \neq \emptyset] \cup \mathcal{E}] \\ &= \text{Prob}[R_h \cap P' \neq \emptyset] + \text{Prob}[\mathcal{E}] \\ &\geq d_{w'} + (d_v - d_{w'}) \\ &= d_v. \end{aligned}$$

The penultimate step used the induction hypothesis as well as the inequality $\text{Prob}[\mathcal{E}] \geq d_v - d_{w'}$ derived above. □

To implement RS, we need to find d_v values to each node, determine the edges and weights for each R_h, and sample according to the above procedure. Each step takes time polynomial in n. Thus, we have a polynomial-time procedure that *optimally* solves a graph-based security game under the conditions of Theorem 3. Previously known techniques either required time exponential in the graph size or can not provide quality guarantees. However, since we cannot guarantee performance when Radius Sampling's condition is not met, we also explore another sampling algorithm.

5.2 Comb Sampling

Now, we consider a somewhat simpler case for the defender: the attacker only observes marginal distributions and approximates the capture probability on any path by adding the probabilities. This may occur because observing the full joint probability distribution is much more time- and resource-intensive than observing the marginals. When only able to observe marginals, adding probabilities is a reasonable and conservative approximation for the attacker.

Comb Sampling (CS) is based on two ideas: (1) If the marginals of the joint distribution match RANGER's x_e values, then an attacker summing probabilities will choose the target t and a path P that is the best path for the attacker to use to reach t as calculated by RANGER. (2) If the edges on P are chosen mutually exclusively, then the capture probability on P matches that of RANGER.

Let $e_1, \ldots, e_{|P|}$ be the edges on the path P (in arbitrary order), and $e_{|P|+1}, \ldots, e_m$ the remaining edges, in arbitrary order. For each $1 \leq j \leq m$, let $X_j = \sum_{i<j} x_i$, and define the interval $I_j = [X_j, X_j + x_j)$. Because $\sum_i x_i = r$ (w.l.o.g.), the I_j form a disjoint cover of the interval $[0, r)$. We now generate a deployment, L_C, as follows: Pick a number $y \in [0, 1)$ uniformly at random, and include in L_C all edges e_j such that $y + k \in I_j$ for some integer k. In other words, include exactly the edges which "own" the intervals containing the points $y, y + 1, y + 2, \ldots, y + r - 1$. This samples exactly r edges.

LEMMA 5. *Given a marginal distribution $\vec{x}$, CS will exactly meet all marginal probabilities, x_e.*

PROOF. Consider any edge e_j, and two cases. If $I_j \subseteq [k, k+1)$ for some k (i.e., I_j contains no integer point), then e_j is included if and only if $k + y \in I_j$, which happens with probability $|I_j| = x_j$. On the other hand, if $I_j = [X_j, k) \cup [k, X_j + x_j)$, then e_j is included if and only if $y + k - 1 \in [X_j, k)$ or $y + k \in [k, X_j + x_j)$; because $x_k \leq 1$, this happens with probability $(k - X_j) + (X_j + x_j - k) = x_j$. □

Lemma 5 ensures that the attacker will follow the path predicted by RANGER. Now consider P. If $\sum_{e \in P} x_e \geq 1$, then for any y, some edge $e \in P$ will be included, so the attacker is always captured. This correctly matches the d_t value produced by RANGER, which would also be 1 by Constraint 5. Otherwise, an edge from P is included if and only if $y < \sum_{e \in P} x_e$, which happens with probability $\sum_{e \in P} x_e$, i.e., the sum of marginals on P. Combined with Lemma 5, this guarantees that RANGER's reward is achievable using CS if the defender faces an approximating attacker. The sampling time is clearly polynomial in the graph size. Thus, we have a polynomial-time procedure to optimally defend against an approximating attacker.

6. EXPERIMENTS

6.1 Quality Comparison

Our first evaluation studies the quality of solutions generated by Radius and Comb Sampling in the general case, against both exact and approximating attackers, as defined in the introduction. Neither method is guaranteed to achieve the optimal value against an exact attacker in all cases, so we are interested in whether these methods give good approximations. We compare them against DOBSS, which computes the optimal solution against an exact attacker. DOBSS may not be optimal against an approximating attacker, so we also report the quality of DOBSS against approximating attackers, labeled DOBSS Marginal. As a benchmark, we include a simple Independent Sampling (IS) strategy, wherein for each checkpoint, edge e is selected independently with probability $\frac{x_e}{r}$.

We generate random graphs that are representative of the domains where our methods are most relevant. First, we test on random geometric graphs to estimate performance for road network domains. Then we test on scale-free graphs as an analogy for subway networks. For each graph type, we generate 500 instances each of 4, 5, and 6 nodes, zero-sum games and report results in Table 1. Every graph has one source, between 1 and $|V| - 1$ targets with values from -1 to -10 for the defender when successfully attacked, and 1 to 3 checkpoints. The payoffs for capture are all 0. *Graphs were kept simple so DOBSS could solve them within a reasonable time limit*.

For each graph, we run each sampling method on the marginals produced by RANGER and calculate the *actual* capture probabilities for the joint distribution generated. Using the true capture probabilities, we select an optimal target for the attacker and compute expected payoffs based on this choice. We compare these rewards against DOBSS to evaluate the quality of our methodology.

For DOBSS Marginal, we calculate the marginal probabilities from the joint distribution given and calculate the path to each target with the least (approximate) probability of capture by summing the marginals along each path. Taking the (approximate) expected reward for attacking each target, we can determine an *approximating attacker*'s optimal action and the corresponding reward for the defender.

For Independent Sampling, the actual probability of capture is $1 - (1 - p/r)^r$, where p is the sum of RANGER's marginal probabilities on the edges on the path P. To see this, recall that since r checkpoints are placed independently, and the probability that the j^{th} checkpoint is not on P is $1 - \sum_{e \in P} x_e/r = 1 - p/r$, the probability that there is no checkpoint on P is $(1 - p/r)^r$. Thus, the probability for capture is $1 - (1 - p/r)^r$.

For Radius Sampling, we find all rings R_h and the probability for selecting each. Then, we calculate the probability of selecting edges within each ring. From these probabilities, we then obtain the marginal probabilities for each edge; when the conditions of Theorem 3 are violated, these marginal probabilities might be less than RANGER's values. Finally, we add the marginal probabilities on any path to find the actual probability of capture.

For Comb Sampling, recall that we have a joint probability dis-

tribution. For each path, we determine which joint actions place a checkpoint on the path and sum the probabilities associated with these actions. In general, this would be an exponential procedure, since there are exponentially many possible joint actions and exponentially many paths that must be calculated. However, Comb Sampling produces a joint distribution using only $O(m)$ joint actions, making experiments relatively efficient. In practice, we could randomize the order in which the x_e's are processed to create a more complex joint distribution, but experimental evaluation would become computationally infeasible.

For each of the sampling methods, we report the expected defender reward based on the *exact attacker*'s action as determined by the procedures outlined previously. The DOBSS value is simply the expected defender reward calculated by the corresponding MILP.

Table 1 shows the number of cases where a difference in quality of more than 0.001 exists between methods. Empirically, RANGER computes very good estimates of the optimal reward value, never differing from DOBSS for more than 5% of cases. Unsurprisingly, Independent Sampling frequently results in suboptimal distributions (44%–78%). Remarkably, CS attains the optimal expected reward in every single one of the 3,000 graphs tested. RS also performs very well, never differing from DOBSS for more than 11% of the games. DOBSS Marginal never differs from DOBSS or RANGER, indicating that DOBSS remains optimal against an approximating attacker (not shown). However, as runtime experiments will show, DOBSS is completely incapable of solving reasonable game sizes.

	Random Geo.			Scale-Free		
Nodes	4	5	6	4	5	6
RG $>$ DOBSS	12	8	5	0	4	22
IS $<$ DOBSS	220	283	280	389	347	247
CS $<$ DOBSS	0	0	0	0	0	0
RS $<$ DOBSS	0	0	3	0	29	53

Table 1: Results by number of cases (RG - RANGER).

6.2 Mumbai

As a real-world trial, we use our algorithms to create security policies for the southern tip of Mumbai, shown in Figure 1, which was an area of heavy terrorist activity in 2008. The region is modeled as a graph with 35 nodes and 58 edges. Attackers can potentially enter from *any* entry node, chosen based on historical and likely entry points. A total of four target nodes are chosen based on historical attacks, marked with black circles in Figure 1. These are held constant throughout testing. Payoffs are decided as in the Quality Comparison experiments.

Figure 2(a) shows the averaged defender rewards obtained across eight configurations, each with their own setup of target values and sources, with each configuration being run with checkpoints varying from 2 to 10. Figure 2(b) shows results averaged across a different set of eight configurations, each with their own setup of target values and 4 checkpoints, with each configuration being run with the number of sources increasing from 1 to 7.

DOBSS is unable to solve even the simplest case within the 20-minute limit; thus, we include only Comb Sampling and Radius Sampling's expected reward, Minimum Cut, as well as three natural defense strategies, *Uniform Random*, *Entry-Incident* and *Weighted-Target-Incident*. Minimum Cut, as introduced by [16], contracts all sources into a super-source and all targets into a super-target and finds the minimum cut on the resulting graph, uniformly randomizing resources across it. This effectively ignores target value variation, but is extremely efficient. Uniform Random places checkpoints uniformly randomly across all edges in the graph. Entry-Incident places checkpoints on edges incident to entry nodes with equal probability. Weighted-Target-Incident places checkpoints on edges incident to target nodes, weighted according to their payoff. The y-axis shows the expected reward in Figure 2(a) and 2(b). The x-axis of Figure 2(a) shows the number of checkpoints allowed and the number of sources in Figure 2(b). RANGER ran in ≈ 0.2 seconds in all trials.

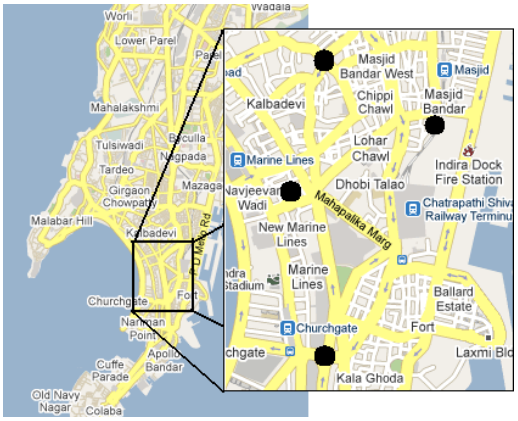


Figure 1: Target layout for southern Mumbai.

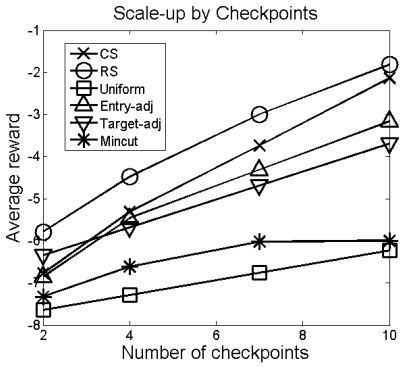
Throughout the tests, the two sampling methods developed here outperformed all others, with RS performing worse than CS when its optimality condition was violated more severely. Minimum Cut, which does not prioritize higher value targets, performs worse than the Uniform Random strategy in some situations, which sometimes happens to place more coverage on higher value targets simply because there are more roads to them.

RANGER actually exploits resources differently and to better effect, which we explore in Figure 3(a) and 3(b). Data was used from the games in Figure 2(b). Figure 3(a) shows the average number of edges with non-zero marginal probability in Entry-Incident and RANGER strategies as the number of sources increases (other methods would be a constant value). As can be seen, RANGER actually uses *fewer* edges than Entry-Incident as the number of entry points increases, but uses them to much better effect.

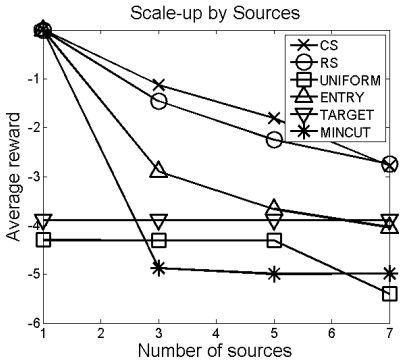
Figure 3(b) shows the standard deviation of marginal probabilities of edges. As expected, Entry-Incident remains constant at 0. RANGER's results vary from 0 (with only one source) to 0.2, which is actually a standard deviation of 20% in marginal probability on edges. Although we cannot provide guarantees on the performance of Radius or Comb Sampling against an exact attacker in general, the techniques yield non-trivial mixed strategies of high quality in practice that outperform the sensible alternatives explored here.

6.3 Runtime Comparison

We have discussed the exponentially smaller solution space RANGER, RS, and CS operate in; we now show runtime comparisons to verify our conclusions empirically. Specifically, we evaluate the runtime performance of RANGER against the fastest-known exact algorithm for solving general Bayesian Stackelberg games, DOBSS [13], as well as a faster solver for security games, ERASER [10]. DOBSS serves as a benchmark, since it provides the optimal solution against exact attackers. ERASER exploits structural proper-

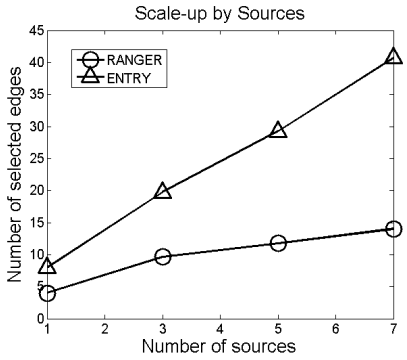


(a) Avg reward by checkpoints.

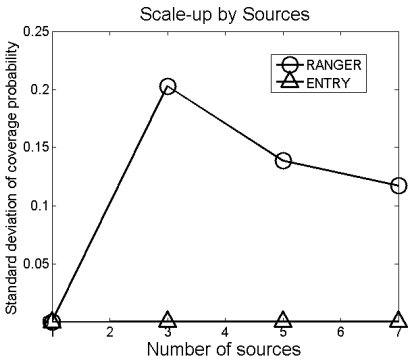


(b) Avg reward by sources.

Figure 2: Rewards in Mumbai domain.



(a) No. edges used.



(b) Standard dev. of Prob.

Figure 3: RANGER strategies in Mumbai domain.

ties that exist in many security domains to create a compact game representation. However, in order for the solution to be correct, it also requires that defender actions be independent from each other, which is not the case in our domain, since placing two checkpoints on one path will violate this assumption. Nevertheless, ERASER serves as another approximation algorithm that runs much more efficiently than DOBSS, so we compare RANGER's runtime against it.

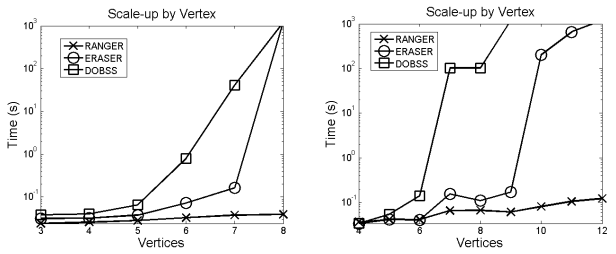
Experiments were run on quad-core Intel 3.0GHz processors with 3GB of RAM. Each approach was run 100 times on each problem, and we report the average time. All algorithms were given a maximum time of 20 minutes.

Figure 4 shows the scaling of each method's runtime with respect to n. In these experiments, we use complete graphs (3 to 8 vertices) and random geometric graphs (4 to 12 vertices), each with one source, one target (value -1 to -10 for defender), and two checkpoints. The x-axis shows the number of vertices in the graph, and the y-axis shows runtime in seconds. Each result is an average of 10 trials. Unsurprisingly, DOBSS and ERASER scale poorly and are only able to solve the problem with up to 7 vertices for complete graphs and 8 (DOBSS) and 11 (ERASER) vertices on random geometric graphs. RANGER is capable of solving games with 400 vertices within the time limit (not shown).

7. CONCLUSIONS

In this work, we provide three primary contributions. First, we develop a linear program, RANGER, to efficiently create optimal marginal distributions for placing checkpoints on the edges of a graph. We prove that the reward found by RANGER is an overestimate of the true optimal reward. Second, we introduce Radius Sampling, which we show produces optimal joint distributions under specific conditions. Third, we develop Comb Sampling, which we prove guarantees optimality against an approximating attacker. We complete the discussion by providing experimental verification of the high quality of our techniques on random graphs as well as a real-world domain.

The techniques introduced here can be put into direct use by city officials such as those in Mumbai, where terrorist attacks of the form we model here are very real and occur with tragic regularity. We show that RANGER strategies implemented using either Radius or Comb Sampling far outperform the simpler alternatives explored. Although we cannot know the current method used by Mumbai police, we offer an efficient, high-quality solution technique based on game-theoretic foundations and hope to transition



(a) Complete graphs. (b) Random geometric graphs.

Figure 4: Runtimes for RANGER, ERASER, and DOBSS.

this theory into practice in the near future.

8. FUTURE RESEARCH

While RANGER and Comb/Radius Sampling can be used immediately, we see a series of open research questions that could potentially improve upon these results for real-world deployment. One direction would be the generalization to general-sum domains, since it is a subject of debate whether real-world scenarios are always best modeled as zero-sum games.

Also, perhaps one can prove NP-hardness of the problem of finding optimal defender strategies in either our restricted games or in the general-sum cases. Assuming that no efficient algorithm can handle all inputs, it would then be desirable to identify conditions under which either optimality or approximation factors can be guaranteed. Even without guarantees, other practical and efficient heuristics may be of interest to warrant deployment in practice.

Another highly pertinent extension would be to introduce a "probability of capture" for each checkpoint, instead of assuming that a checkpoint will always capturing an attacker using the edge, as we do in our work. This would require major alterations in sampling, since our sampling techniques have focused on never placing two checkpoints along a single s-t path, which may no longer be desirable. All of these directions are critical in effectively implementing game-theoretic methods in real-world security domains such as the Mumbai example presented here and we hope to pursue them to further improve existing security practices.

9. ACKNOWLEDGEMENT

This research was supported by the United States Department of Homeland Security through the Center for Risk and Economic Analysis of Terrorism Events (CREATE) under grant number 2007-ST-061-000001. Any opinions, conclusions or recommendations herein are solely those of the authors and do not necessarily reflect views of the Department of Homeland Security.

10. REFERENCES

[1] S. A. Ali. Rs 18L seized in nakabandi at Vile Parle. *Times of India*, 4 August 2009.

[2] N. Basilico, N. Gatti, and F. Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *AAMAS-09*, 2009.

[3] R. Chandran and G. Beitchman. Battle for Mumbai ends, death toll rises to 195. *Times of India*, 29 November 2008.

[4] V. Conitzer and T. Sandholm. Computing the optimal strategy to commit to. In *EC '06: PRoceedings of the 7th ACM conference on Electronic commerce*, pages 82–90, New York, NY, USA, 2006. ACM.

[5] K. J. Cormican, D. P. Morton, and R. K. Wood. Stochastic network interdiction. *Oper. Res.*, 46(2):184–197, 1998.

[6] D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, 1991.

[7] N. Gatti. Game theoretical insights in strategic patrolling: Model and algorithm in normal-form. In *ECAI-08*, pages 403–407, 2008.

[8] E. Halvorson, V. Conitzer, and R. Parr. Multi-step Multi-sensor Hider-Seeker Games. In *IJCAI*, 2009.

[9] E. Israeli and R. K. Wood. Shortest-path network interdiction. *Networks*, 40(2):97–111, 2002.

[10] C. Kiekintveld, M. Jain, J. Tsai, J. Pita, M. Tambe, and F. Ordóñez. Computing Optimal Randomized Resource Allocations for Massive Security Games. In *AAMAS-09*, 2009.

[11] M. Mavronicolas, V. Papadopoulou, A. Philippou, and P. Spirakis. A network game with attackers and a defender: A survey. In *ECCS*, 2006.

[12] M. Mavronicolas, V. Papadopoulou, A. Philippou, and P. Spirakis. A network game with attackers and a defender. *Algorithmica*, 51(3):315–341, 2008.

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

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

[15] J. Tsai, S. Rathi, C. Kiekintveld, F. Ordóñez, and M. Tambe. IRIS - A Tool for Strategic Security Allocation in Transportation Networks. In *AAMAS-09 (Industry Track)*, 2009.

[16] A. Washburn and K. Wood. Two-person zero-sum games for network interdiction. *Operations Research*, 43(2):243–251, 1995.

[17] R. K. Wood. Deterministic network interdiction. *Mathematical and Computer Modeling*, 17(2):1–18, 1993.