Computing Solutions in Infinite-Horizon Discounted Adversarial Patrolling Games

Yevgeniy Vorobeychik[1], **Bo An**[2], **Milind Tambe**[3], **and Satinder Singh**[4]
[1]Vanderbilt University, Nashville, TN, yevgeniy.vorobeychik@vanderbilt.edu
[2]Nanyang Technological University, Singapore, boan@ntu.edu.sg
[3]University of Southern California, Los Angeles, CA, tambe@usc.edu
[4]University of Michigan, Ann Arbor, MI, baveja@umich.edu

Abstract

Stackelberg games form the core of a number of tools deployed for computing optimal patrolling strategies in adversarial domains, such as the US Federal Air Marshall Service and the US Coast Guard. In traditional Stackelberg security game models the attacker knows only the probability that each target is covered by the defender, but is oblivious to the detailed timing of the coverage schedule. In many real-world situations, however, the attacker can observe the current location of the defender and can exploit this knowledge to reason about the defender's future moves. We show that this general modeling framework can be captured using adversarial patrolling games (APGs) in which the defender sequentially moves between targets, with moves constrained by a graph, while the attacker can observe the defender's current location and his (stochastic) policy concerning future moves. We offer a very general model of infinite-horizon discounted adversarial patrolling games. Our first contribution is to show that defender policies that condition only on the previous defense move (i.e., Markov stationary policies) can be arbitrarily suboptimal for general APGs. We then offer a mixed-integer non-linear programming (MINLP) formulation for computing optimal randomized policies for the defender that can condition on history of bounded, but arbitrary, length, as well as a mixed-integer linear programming (MILP) formulation to approximate these, with provable quality guarantees. Additionally, we present a non-linear programming (NLP) formulation for solving zero-sum APGs. We show experimentally that MILP significantly outperforms the MINLP formulation, and is, in turn, significantly outperformed by the NLP specialized to zero-sum games.

Introduction

Game theoretic approaches to security based on Stackelberg game models have received much attention in recent years, with several finding deployment in real-world settings including Los Angeles International Airport, United States Federal Air Marshals Service, United States Transportation Security Agency, and United States Coast Guard (Jain et al. 2010; An et al. 2011). At the backbone of these applications are defender-attacker Stackelberg games in which the defender (leader) first commits to a randomized security policy, and the attacker (follower) uses surveillance to learn about the policy before attacking. A Stackelberg equilibrium of this game yields an optimal security policy for the defender, accounting for an optimal attacker response, and considerable literature now exists on computing a *Strong Stackelberg Equilibrium (SSE)*, which is a Stackelberg equilibrium with the follower breaking ties in the leader's favor (Conitzer and Sandholm 2006; Paruchuri et al. 2008; Kiekintveld et al. 2009).

To date, most Stackelberg game models assume that the attacker knows the probability that each target is covered by the defender, but is oblivious to the actual sequence of defender moves. For example, the defender may in fact visit targets according to some fixed (but randomly generated) patrolling schedule, but the attacker is presumed to be unable to observe the defender's location at any point during the patrol. In many realistic settings, such as the US Coast Guard it is likely that the attacker can in fact observe the patrol while it is in progress (e.g., the coast guard ships can be quite overt). Thus, a more plausible model in such a setting would allow the attacker to observe both the randomized policy of the defender (i.e., probability distribution over moves) as well as current defender location. Such a model, often termed *adversarial patrolling games*, has indeed been proposed in recent literature (Basilico, Gatti, and Amigoni 2009; Basilico et al. 2009; 2010; Basilico, Gatti, and Villa 2011; Basilico and Gatti 2011; Bosansky et al. 2011; Basilico, Gatti, and Amigoni 2012). All of this literature, however, shares an assumption which makes it difficult to deploy the proposed models and associated computational techniques in practice: it is assumed that both players are completely indifferent about the timing of an attack. This assumption is rather at odds with intuition: even very patient attackers far prefer attacking sooner rather than later, since delaying increases likelihood of being caught, whereas defenders clearly would prefer to delay an attack as long as possible (for example, to have more time to prepare for the attack and its consequences or to catch the attacker). A natural way to incorporate temporal preferences of this nature is to introduce exponential discounting. It turns out, however, that solution methods previously applied in the undiscounted setting are no longer sensible as soon as discounting is introduced, as they leverage the very indifference in timing that discounting is meant to eliminate. Handling discounting, therefore, requires entirely dif-

Copyright © 2013, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

ferent techniques. Surprisingly, while abundant work exists in the undiscounted setting, ours is the first effort to handle infinite-horizon discounted adversarial patrolling games. While our work is related to several recent treatments of stochastic Stackelberg games (Letchford et al. 2012; Vorobeychik and Singh 2012), none of these consider the special structure of our problem, and scale poorly in general.

We generalize previous approaches to adversarial patrolling games (APGs) in several important directions. First, we introduce exponential discounting, as we had already motivated above. Second, while we maintain the assumption that patrol moves are constrained by a network (for example, representing physical barriers), we additionally introduce costs that the defender incurs for traversing edges of the network. This cost models variations in terrain and/or equipment/vehicle that one needs to use to traverse a given edge (for example, a boat or a helicopter). Third, with a few exceptions (e.g., (Basilico and Gatti 2011)), most of the previous work assumed a single defense resource; our model allows for multiple homogeneous resources. In addition, we make several contributions towards computing Stackelberg equilibria in this very general class of infinite-horizon discounted APGs. First, we demonstrate that even in APGs in which attacks take a single time unit, a stationary Markov defense policy (i.e., a policy which only conditions on the last defense move) may be arbitrarily suboptimal, thereby resolving an important open question. Second, we provide a mathematical programming formulation for computing optimal defender policies that condition on arbitrary, but bounded, horizon of previous defense moves. Third, we present a mixed-integer linear programming formulation for approximating solutions to APGs, with provable approximation guarantees. Fourth, we present a non-linear programming formulation for the special case of zero-sum APGs. In our experiments, we demonstrate this formulation to be highly scalable compared to alternative approaches.

Adversarial Patrolling Games

An adversarial patrolling game (APG) is described by the tuple $\{T, G, C, U_D^c(i), U_D^u(i), U_A^c(i), U_A^u(i), \gamma_D, \gamma_A\}$, where T is the set of n targets patrolled by the defender, $G = (T, E)$ is a graph with targets as vertices and E the set of directed edges constraining that the defender can only move from i to j if $(i, j) \in E$, and $C = \{c_{ij}\}$ is a matrix of costs of traversing an edge $(i, j) \in E$. $U_D^c(i)$ and $U_D^u(i)$ are the utilities to the defender if an attacker chooses a target $i \in T$ when it is visited by the defender, and not, respectively, while $U_A^c(i)$ and $U_A^u(i)$ are the corresponding attacker utilities. Some of the nodes in T may in fact not be potential targets of attack, either because they are too difficult to penetrate for the attacker, or because they are not worth anything to the attacker. We can easily capture both of these aspects by assigning a large negative attacker utility to impenetrable targets, and assigning zero utility to nodes that the attacker has no interest in attacking. Finally, $\gamma_D, \gamma_A \in (0, 1)$ are the discount factors of the defender and attacker (in some cases, we also allow $\gamma_A = \gamma_D = 1$). It is useful to consider the representation of this graph as an adjacency matrix A, where $A_{ij} = 1$ if and only if there is an edge from target i to target j. We assume that the defender uses R homogeneous resources to patrol the targets, and let $z = \{z_1, \ldots, z_n\}$ denote the coverage vector, i.e., a vector where $z_i = 1$ indicates that target i is covered by a defense resource and $\sum_i z_i = R$. We denote the space of all such coverage vectors by Z. In our discrete time model, the resolution of iterations is assumed to correspond to defense moves that can feasibly be taken in a chosen unit of time. We assume that this time resolution is sufficiently small that the attacker can only take as long, or longer, than a single time unit to execute an attack. We let $h \geq 1$ denote the number of time steps that an attack takes to execute.

The game is structured as follows. First, the defender *commits to* a patrolling policy, π, which is in general an arbitrary function from all observed history (i.e., the sequence of past coverage vectors) to a probability distribution over the coverage vectors patrolled in the next iteration. Second, the attacker finds out the patrolling policy π (e.g., through surveillance) and chooses a policy a as a best response, which, just like π, can condition on a previous sequence of the defender's moves, as well as defender's current coverage vector z. The attacker's response policy determines whether the attacker waits or attacks a particular target $i \in T$ at any point in time. Once both the defender and the attacker have chosen their respective policies, their payoffs are evaluated solely based on the timing of the attack decision, and whether or not the defense patrol passes through the attacked target within h steps of the attack. If an attack on target i commences at time t, for example, the utility to the attacker is $\gamma_A^{t+h} U_A^u(i)$ if the defender does not pass through i in the time interval $[t+1, t+h]$, while the attacker's utility is $\gamma_A^{t+v} U_A^c(i)$ if the patrol passes through i at time $v \leq h$. The corresponding payoffs to the defender are symmetric. To summarize, then, the utility to each player in this effectively two-stage game is the expected discounted utility (as a function of both π and a) computed at time step 1. (To be clear, even though we refer to this setup as an infinite-horizon discounted game, in our model both players only care about their expected utility as evaluated at the very first time step of the game).

Finally, we assume, entirely for convenience, that all defense resources begin at a fixed starting point ("base"), which we denote as a coverage vector z^0.

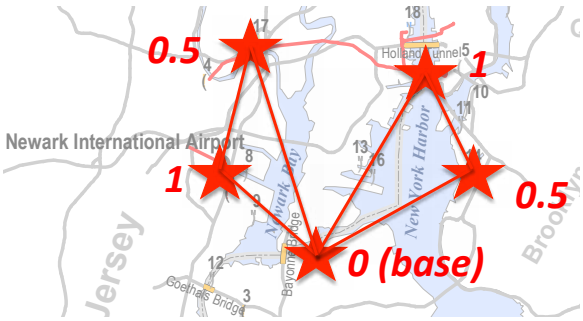

Figure 1: Example of a simple New York Bay patrolling scenario.

Example 1. ***USCG's Patrolling Problem as an APG****: USCG safeguards important infrastructure at US coasts, ports, and inland waterway. For this example, we chose a simple Newark Bay and New York Harbor patrolling scenario, shown in Figure 1. We chose five possible targets, with the graph roughly representing geographic patrolling constraints (assuming a boat patrol). The number near each target represents the amount that the attacker gains and the defender loses if the corresponding target is successfully attacked; both sides receive the utility of zero if an attack is unsuccessful. The highly connected target with zero value represents the base of operations.* □

While we are not the first to consider adversarial patrolling scenarios (see, e.g., (Basilico, Gatti, and Amigoni 2009; Basilico et al. 2009) for more details), we are the first to include discounting in our model. We can use Example 1 to demonstrate the importance of considering the discount factor in adversarial patrolling settings.

Proposition 1. *Assuming that the attacker is infinitely patient when in fact he is not can result in an arbitrarily suboptimal patrolling policy.*

Proof. Consider example 1, and suppose that $\gamma_D = \gamma_A = 1$, that is, the attacker is infinitely patient, $h = 1$, and the game is zero-sum. This means that the attacker is willing to wait indefinitely before attacking, and attack as soon as the defender commits to either of the most valuable targets. Consequently, the attacker will obtain utility of 1 no matter what the defender does, and any defense is optimal, including staying at base with probability 1. If we suppose that, instead, the attacker's discount factor is 0.5, a policy of always staying at base still gives the attacker utility of 1, while the defender can lower it to $1/2$ by defending the two most valuable targets with probability $1/2$ each. The ratio between the optimal policy, and a policy obtained if we assume that $\gamma_A = 1$ can be made arbitrarily large by amplifying target values. □

Stackelberg Equilibria in APGs

It is well known that in general-sum stochastic games there always exists a Nash equilibrium (NE) in Markov stationary policies (Filar and Vrieze 1997). The import of this result is that it allows one to focus NE computation on this very restricted space of strategies. In this section we show that this result *does not* in general extend to APGs, and, indeed, a Markov stationary policy could be arbitrarily suboptimal.

Proposition 2. *The leader's optimal policy in APGs may not be Markov stationary even if $c_{ij} = 0$ for all $(i, j) \in E$, $U_D^c(j) = -U_A^c(j)$, and $U_D^u(j) = -U_A^u(j)$ (i.e., the players merely have different discount factors, but agree on the relative importance of targets).*

Proof. Consider an APG with three targets, 1, 2, and 3. Targets 1 and 2 have a value of 1 and target 3 has a value of 0; that is, if the attacker attacks 1 or 2 and the corresponding target is not defended, the attacker gains, and the defender loses, 1, whereas if the attacked target is covered, no loss or gain is accrued to either player. Additionally, if target 3 is attacked, everyone receives 0 payoff no matter what the defender's strategy is. Targets 1 and 2 are connected to each other and each is connected to itself; target 3 can be reached from either 1 or 2, but does not connect back to these, although it is connected to itself; it is, effectively, an absorbing state. Visually, this setup is illustrated in Figure 2. Suppose

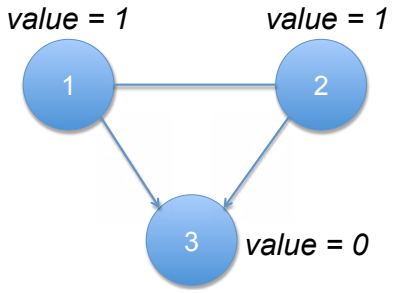

Figure 2: Counterexample: a Markov stationary policy for the defender need not be optimal in APGs.

that the defender's discount factor is 0.1 and the attacker's is 0.9, that is, the defender is largely concerned about what happens during the first time step, while the attacker is willing to wait. Finally, suppose that the defender starts at target 1.

Since targets 1 and 2 are identical in every way, there is an optimal Markov stationary policy that plays the same strategy in both of these. For the same reason, the probability of staying put at either target 1 or 2, or moving to the other valued target (that is, moving from 1 to 2 or from 2 to 1) is the same in some optimal Markov stationary policy. Let us call this probability p. Thus, the probabilities of moving from 1 to 2, from 2 to 1, or staying at 1 or at 2, all equal p. Then the probability of moving to target 3 from either 1 or 2 is $1-2p$, which implies that $p \leq 0.5$.

Since target 3 has a value of 0 to both players and is absorbing, clearly the attacker would attack either target 1 or target 2 (he is indifferent between these), accruing the utility of 1 (which is lost to the defender) if ever he observes the defender having landed on (and stuck at) target 3.

Suppose that the defender is at target 1 (target 2 can be considered symmetrically). Let V_A be the attacker's expected discounted value of observing defender at target 1. The attacker will wait another step if and only if his expected utility of waiting exceeds his expected utility of attacking immediately, that is, if $0.9((1-2p) + 2pV_A) \geq (1-p)$, where the quantity on the left-hand-side is the expected utility of waiting, and $(1-p)$ is the expected utility of attacking either target 1 or target 2 immediately. Since $V_A \leq 1$, the necessary condition for forcing the attacker to wait in our example is that $1 - p \leq 0.9$ or $p \geq 0.1$. Therefore, if the attacker were to wait, the defender's loss is bounded from below by $0.1((1-2p) + 2pV_D) \geq 0.1 - 0.2p \geq 0.08$. If the attacker were to attack immediately, on the other hand, the expected loss to the defender is $1 - p$, which is minimized when $p = 0.5$. Thus, the defender will force the attacker to wait and lose at least 0.08 in expectation.

Now consider the following non-stationary policy for the

defender. The defender plays $p = 0.5$ for the first two rounds, then moves to target 3 with probability 1. Clearly, the attacker will wait and attack in round 3, since his expected utility of waiting both rounds is $0.9^2 = 0.81 > 0.5$, which is what he would attain from attacking immediately. For the defender, however, the expected loss from this policy is $0.1^2 = 0.01$, much smaller than the expected loss from an optimal Markov stationary policy. □

Computing Solutions for APGs

A MINLP Formulation for General-Sum APGs

Our first step is to present a very general MINLP formulation for computing a Strong Stackelberg equilibrium in general-sum APGs. Recall that z^0 corresponds to the fixed initial vector of defense resource deployment, that is, the initial coverage vector, and let K be the fixed number of previous defense moves that the defender keeps track of in computing and executing a defense policy. We define $s = \{z^1, \ldots, z^K\}$ as the history of the K previous moves, the first $K-1$ of which may be $\emptyset$, if the defender has only begun less than K steps ago. Sometimes we casually refer to s as *state*. Let $B_{sz} = 1$ if and only if z is a valid transition for the defender following the last defender move in s, z_K. Below, we show how to construct this matrix given the adjacency matrix A for the graph constraining defender's moves. Let S be the set of all such *feasible* K-length histories (i.e., sequences of moves such that $B_{z^k, z^{k+1}} = 1$ for all $1 \leq k < K$), ignoring the leading $\emptyset$ entries. We overload notation to let z^0 denote the initial history s at the beginning of the game.

To obtain the matrix B, consider a pair of coverage vectors z and z', and let T_s be the set of covered targets under z and $T_{z'}$ the set of targets covered under z'. Let $G_{zz'} = \{T_z, T_{z'}, E\}$ be a bipartite graph with a directed edge $(i,j) \in E$ if and only if $i \in T_z$, $j \in T_{z'}$, and $A_{ij} = 1$. The following proposition is then relatively direct.

Proposition 3. *For each $s \in S$, $z' \in Z$, $B_{sz'} = 1$ if and only if $G_{zz'}$ has a perfect matching, where z is the last move by the defender in s.*

Proof. For one direction, suppose that $G_{zz'}$ has a perfect matching. This means that every covered target in z is matched to exactly one covered target in z', which implies that there is a feasible move for a resource situated at each target covered under z to z' and, since this is a matching, no two resources move to the same location. For the other direction, suppose that the maximum matching is not a perfect matching. Since this matching matches the largest number of covered targets, it must be that under every possible matching there exists an infeasible move for some resource. □

The convenience of this result is that the existence of a perfect matching can be checked in time polynomial in the number of resources r (Edmonds 1965).

Define Q as a binary tensor with $Q_{szs'} = 1$ if and only if s followed by z results in a new K-long sequence of moves s' (note that this tensor can be computed from the problem definition). The tensor Q will be useful below to allow us to cleanly express which state results by appending a move z to a previous state s. Let π denote the set of variables that compute defense policy, with π_{sz} the probability of choosing a coverage vector z after a history of previous moves s. Define b_s as an indicator variable with $b_s = 1$ if and only if the attacker chooses to wait in state s, and let a_{sj} be an indicator variable with $a_{sj} = 1$ if and only if the attacker attacks target j in state s, if he chooses to attack at all. Define $V_D(s)$ as the maximum expected utility of the defender after observing history s, and $V_A(s)$ is the corresponding expected utility of the attacker; both of these will be computed, along with the optimal policy, in the optimization program below. Let $P(s)$ be the total expected costs incurred by the defender's optimal patrolling strategy following the history of moves s.

An important step is computing the probability of making the move $z \in Z$ exactly t time steps after a sequence of defense moves s without passing through z in the meantime, which we denote by α^t_{sz}. Clearly,

$$\alpha^1_{sz} = \pi_{sz} \quad \forall \quad s \in S, z \in Z. \tag{1}$$

Moreover, we now show that it can be computed recursively for an arbitrary t. First, for each "neighbor" of s, r (excluding z itself), we compute the probability that z can be reached if s is followed by r in exactly $t-1$ steps without passing through z (this is computed by the quantity $\sum_{s' \in S} Q_{srs'} \alpha^{t-1}_{s'z}$, where Q serves to compute s' that follows after adding a move r to s). Next, since π_{sr} is the probability of reaching a neighbor r from s in one step, $\pi_{sr} \sum_{s' \in S} Q_{srs'} \alpha^{t-1}_{s'z}$ computes the probability of reaching z from s in exactly t steps by going through r. Finally, the quantity of interest is just the sum over all neighbors r, except z itself (which we do not pass in this calculation), i.e.,

$$\alpha^t_{sz} = \sum_{r \in Z | r \neq z} \pi_{sr} \sum_{s' \in S} Q_{srs'} \alpha^{t-1}_{s'z} \; \forall \, s, z, 1 < t \leq h. \tag{2}$$

The usefulness of defining and computing α^t_{sz} as we have done above is that the corresponding events are disjoint for different values of t. Therefore, we can compute the probability of reaching z starting at s in at most h steps directly as $\sum_{t=1}^{h} \alpha^t_{sz}$. If the attacker attacks target $j \in T$ upon observing a sequence of defense moves s, the utility of both players is the discounted expected utility over the h-step horizon, where utility at each step t is determined by the probability that there is some coverage vector that covers j in exactly t steps. Thus, the expected utility to the attacker is

$$\left(1 - \sum_{z \in Z | z_j = 1} \sum_{t=1}^{h} \alpha^t_{sz}\right) \gamma_A^{h-1} U_A^u(j) + \sum_{z \in Z | z_j = 1} \sum_{t=1}^{h} \alpha^t_{sz} \gamma_A^{t-1} U_A^c(j).$$

The defender's utility is symmetric.

Now, we need to deal with computing the expected costs $P(s)$, using the primitives c_{ij} that are given to us as a part of the problem definition. First, recall that there is a valid move

from a coverage vector z to another, z', iff there is a perfect matching in the bipartite graph $G_{zz'}$ from z to z' induced by G. Let $\mathcal{M}_{zz'}$ be the set of all perfect matchings on $G_{zz'}$. We define the cost of moving from z to z' as the cheapest way to achieve this move. Formally,

$$c_{zz'} = \min_{E \in \mathcal{M}_{zz'}} \sum_{(i,j) \in E} c_{ij}.$$

Next, notationally we use $c_{sz'}$ to mean $c_{zz'}$ where z is the last defender move in s. Finally, note that the expected total cost in state s, $P(s)$ depends on the attacker's decision whether to attack or to wait in that state, b_s. If the attacker chooses to attack, the game terminates in exactly h time steps, and we can compute the total cost over that horizon recursively. Specifically, let $P^t_{sj}(attack)$ denote the total cost with t steps before the game ends if the attacker attacks target j in state s, and let $P_s(attack)$ be the total expected cost if the attacker attacks.

$$P^1_{sj}(attack) = \sum_{z \in Z} \pi_{sz} c_{sz}, \tag{3}$$

while for all $s \in S, 1 < t \leq h$,

$$P^t_{sj}(attack) = \sum_{z \in Z | z_j = 1} \pi_{sz} c_{sz} + \sum_{z \in Z | z_j = 0} \pi_{sz} (c_{sz} + \gamma_D \sum_{s' \in S} Q_{szs'} P^{t-1}_{s'j}(attack)). \tag{4}$$

The first term is the cost accrued if the defender passes through target j immediately t steps prior to the end of the game, whereas the second term computes the cost in the case when the defender does not immediately pass through j with t steps remaining. Then $P_s(attack) = \sum_j a_{sj} P^h_{sj}(attack)$.

In the case that the attacker chooses to wait, $P_s(wait)$ is the sum of immediately incurred costs and discounted future costs:

$$P_s(wait) = \sum_{z \in Z} \pi_{sz} \left(c_{sz} + \gamma_D \sum_{s' \in S} Q_{szs'} P(s') \right).$$

Thus, the total expected costs are

$$P(s) = (1 - b_s) P_s(attack) + b_s P_s(wait). \tag{5}$$

The final set of pieces for the formulation is a set of constraints for computing the attacker's decisions about attacking or waiting (b_s), about which target to attack (a_{sj}), and corresponding expected utilities to both defender and attacker. Let $R_A(s)$ denote the expected utility the attacker receives in state s, and let $R_D(s)$ be the corresponding defender utility. The following set of constraints computes the decision to attack or wait:

$$0 \leq V_A(s) - R_A(s) \leq b_s M \quad \forall \quad s \in S \tag{6a}$$

$$0 \leq V_A(s) - \gamma_A \sum_{z \in Z} \pi_{sz} \sum_{s' \in S} Q_{szs'} V_A(s') \leq (1 - b_s) M \quad \forall \quad s \in S. \tag{6b}$$

Constraint 6a corresponds to choosing to attack immediately in state s, whereas constraint 6b is the expected utility if the attacker chooses to wait. A similar set of constraints computes the decisions a_{sj} as well as corresponding utilities $R_A(s)$ and $R_D(s)$.

Putting it all together, we obtain the following MINLP formulation (in which M is a large constant):

$$\max \quad V_D(z^0) - P(z^0) \tag{7a}$$

$$\text{s.t.:} \quad \sum_{z \in Z} \pi_{sz} = 1, \quad \sum_{j \in T} a_{sj} = 1 \quad \forall \quad s \in S \tag{7b}$$

$$\pi_{sz} \leq B_{sz} \quad \forall \quad s \in S, z \in Z \tag{7c}$$

$$0 \leq V_A(s) - R_A(s) \leq b_s M \quad \forall \quad s \in S \tag{7d}$$

$$0 \leq V_A(s) - \gamma_A \sum_{z \in Z} \pi_{sz} \sum_{s' \in S} Q_{szs'} V_A(s') \leq (1 - b_s) M \quad \forall \quad s \in S \tag{7e}$$

$$V_D(s) - R_D(s) \leq b_s M \quad \forall \quad s \in S \tag{7f}$$

$$V_D(s) - \gamma_D \sum_{z \in Z} \pi_{sz} \sum_{s' \in S} Q_{szs'} V_D(s') \leq (1 - b_s) M \quad \forall \quad s \in S \tag{7g}$$

$$0 \leq R_A(s) - \left(1 - \sum_{z \in Z | z_j = 1} \sum_{t=1}^{h} \alpha^t_{sz} \right) \gamma_A^{h-1} U^u_A(j) - \sum_{z \in Z | z_j = 1} \sum_{t=1}^{h} \alpha^t_{sz} \gamma_A^{t-1} U^c_A(j) \leq (1 - a_{sj}) M \quad \forall \quad s \in S, j \in T \tag{7h}$$

$$R_D(s) - \left(1 - \sum_{z \in Z | z_j = 1} \sum_{t=1}^{h} \alpha^t_{sz} \right) \gamma_D^{h-1} U^u_D(j) - \sum_{z \in Z | z_j = 1} \sum_{t=1}^{h} \alpha^t_{sz} \gamma_D^{t-1} U^c_D(j) \leq (1 - a_{sj}) M \quad \forall \quad s \in S, j \in T \tag{7i}$$

$$\pi_{sz} \geq 0, \ b_s, a_{sj} \in \{0, 1\} \quad \forall \quad s \in S, z \in Z, j \in T \tag{7j}$$

$$\text{constraints} \quad 1 - 5.$$

Mixed-Integer Linear Programming Approximation

What makes the MINLP formulation above difficult is the combination of integer variables, and the non-convex interaction between continuous variables involving π_{sz} with the variables computing expected utilities, patrolling costs, and α^t_{sz}. If at least one of these variables is binary, we can linearize these constraints using McCormick inequalities (McCormick 1976). To enable the application of this technique, we discretize the probabilities π_{sz} which the leader's policy can use, following the approach set forth in Vorobeychik and Singh (2012).

Let p_l denote a lth probability value and let $\mathcal{L} = \{1, \ldots, L\}$ be the index set of discrete probability values we use. Define binary variables d_{lsz} which equal 1 if and only if $\pi_{sz} = p_l$, and 0 otherwise. We can then write π_{sz} as

$$\pi_{sz} = \sum_{l \in \mathcal{L}} p_l d_{lsz} \quad \forall \quad s \in S, z \in Z. \tag{8}$$

The next step is to introduce new variables for the bilinear terms. Thus, we rewrite

$$\pi_{sz} \sum_{s' \in S} Q_{s,z,s'} V_A(s') = \sum_{l \in \mathcal{L}} p_l x_{lsz},$$

where $x_{lsz} = d_{lsz} \sum_{s' \in S} Q_{s,z,s'} V_A(s')$. Finally, we rewrite this equality as the following equivalent set of McCormick inequalities:

$$\sum_{s' \in S} Q_{s,z,s'} V_A(s') - M(1 - d_{lsz}) \leq x_{lsz} \leq \sum_{s' \in S} Q_{s,z,s'} V_A(s') + Z(1 - d_{lsz}) \quad (9)$$

$$-M d_{lsz} \leq x_{lsz} \leq M d_{lsz}. \quad (10)$$

Since all our bilinear terms involve π_{sz}, they can all be linearized in precisely the same manner, and so we omit the details. The bottom line is that it suffices to only discretize the defense probabilities π_{sz} to enable us to linearize all of the bilinear terms. The result is that we obtain a mixed-integer *linear* programming formulation, which now computes an *approximately* optimal policy for the defender. Moreover, we can bound the quality of this approximation using the results of Vorobeychik and Singh (2012) directly.

Special Case: Zero-Sum APGs

An important special cases of adversarial patrolling games is when they are zero-sum. Specifically, in this class we assume that $c_{ij} = 0$ for all $(i, j) \in E$ (that is, as long as an edge exists, it is free to traverse it). Additionally, $\gamma_A = \gamma_D = \gamma$ (i.e., both players have identical discount factors), $U_D^c(i) = -U_A^c(i)$, and $U_D^u(i) = -U_A^u(i)$. As a result, it suffices to consider only defender's decision variables π_{sz} and attacker's maximum expected utility starting in state s (and computed by the mathematical program below), $V_A(s)$. Optimal patrolling policy can then be computed using the following non-linear program (NLP):

$$\min \quad \sum_{s \in S} V_A(s) \quad (11a)$$

$$\text{s.t.:} \quad \sum_{z \in Z} \pi_{sz} = 1 \quad \forall \quad s \in S \quad (11b)$$

$$\pi_{sz} \leq B_{sz} \quad \forall \quad s \in S, z \in Z \quad (11c)$$

$$V_A(s) \geq \left(1 - \sum_{z \in Z | z_j = 1} \sum_{t=1}^{h} \alpha_{sz}^t\right) \gamma^{h-1} U_A^u(j) + \sum_{z \in Z | z_j = 1} \sum_{t=1}^{h} \alpha_{sz}^t \gamma^{t-1} U_A^c(j) \quad \forall \quad s \in S, j \in T \quad (11d)$$

$$V_A(s) \geq \gamma \sum_{z \in Z} \pi_{sz} \sum_{s' \in S} Q_{s,z,s'} V_A(s') \quad \forall \quad s \in S \quad (11e)$$

$$\pi_{sz} \geq 0 \quad \forall \quad s \in S, z \in Z \quad (11f)$$

$$\text{constraints} \quad 1 - 2.$$

The crucial difference between this formulation for zero-sum APGs from the general formulation is that this variant has no integer variables. Still, there are bilinear constraints that remain. Nevertheless, we demonstrate below that this NLP scales extremely well with the number of targets, indeed, far better than either the general MINLP or the discretized MILP approximation.

Experiments: Adversarial Patrolling on Exogenous Graphs

In our experimental studies below we use a somewhat simplified model in which $U_D^c(i) = U_A^c(i) = 0$ for all targets $i \in T$, and restrict attention to zero-sum adversarial patrolling settings, with $\delta = \gamma_D = \gamma_A$. We generate the values of successful attacks $U_A^u(i)$ i.i.d. from a uniform distribution on the unit interval; since the game is zero-sum, $U_D^u(i) = -U_A^u(i)$. Throughout, we use $\delta = 0.95$, except where specified otherwise.[1] We use a relatively standard Erdos-Renyi generative model to generate graphs over which the defender patrols (Newman 2010). In an Erdos-Renyi model every directed link is made with a specified and fixed probability p; we refer to this model by ER(p), or simply ER. Additionally, we consider graphs which are simple Cycles.

In our experiments, we assume for simplicity that patrols initially deploy from a base, labeled as target 0, which we connect to every other target, with network topology effective only on the rest of the targets.[2] Additionally, we assume that the base has no intrinsic value to the defender, and therefore fix $U_A^u(0) = 0$.

All computational experiments were performed on a 64 bit Linux 2.6.18-164.el5 computer with 96 GB of RAM and two quad-core hyperthreaded Intel Xeon 2.93 GHz processors. We did not make use of any parallel or multi-threading capabilities, restricting a solver to a single thread, when relevant. Mixed integer linear programs were solved using CPLEX version 12.2, mixed integer non-linear programs were solved using KNITRO version 7.0.0, and we used IPOPT version 3.9.3 to solve non-linear (non-integer) programs. The results we report are based on 100 samples from both the attacker utility distribution and (when applicable) from the network generation model. Throughout, we report 95% confidence intervals, where relevant.

Comparison to Basilico et al.

Basilico, Gatti, and Amigoni (2009) presented a multiple math programming approach to adversarial patrolling for a setting very similar to ours. While it is not difficult to see that assuming infinite patience for the attacker can lead to arbitrarily poor results, we now wish to make a more direct comparison. By setting $\delta = 1$, and reformulating the algorithm in Basilico et al. in a zero-sum setting and with a single-step attack, we can make a direct comparison between

[1] We considered other discount factors as well, but this one strikes the right balance: it creates interesting tradeoffs between attacking and waiting, and yet creates a setting that is significantly different from past work which only considers $\delta = 1$.

[2] We can motivate this by noting that location of a base is also a strategic choice, and a base located central to the high value targets makes patrols much more effective.

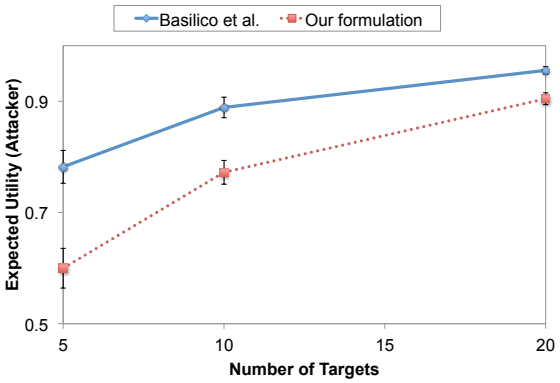

Figure 3: Comparison between our NLP formulation and that developed by Basilico et al. The graph is ER(0.1).

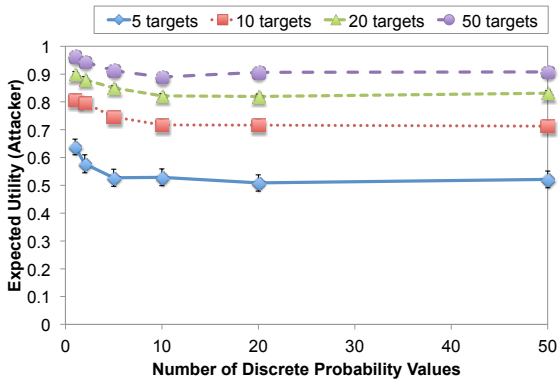

Figure 4: MILP objective value as a function of granularity of discretization. The graph is ER(0.1); the results are similar on other graph classes.

our approach (using the NLP formulation) and theirs. The results, shown in Figure 3, demonstrate that our approach yields significantly better solutions, though the difference between the two becomes less significant as the number of targets increases.

It is, at first, quite puzzling that our approach yields solutions better to those using the formulation of Basilico et al., even "playing on their turf", that is, having an attacker that is infinitely patient. In the online supplement (http://sites.google.com/site/onlineappendices/apgappendix.pdf) we show why the approach offered by Basilico et al. is suboptimal. (To our knowledge, we are the first to offer this analysis of what is currently the state-of-the-art; all of the related approaches build on the same core framework).

MILP Discretization

Our main approach for approximating solutions to APGs is by using a mixed-integer linear programming formulation with discretized probability values. The size and, consequently, complexity of the MILP depends greatly on how finely we discretize the probability interval. While we can achieve an arbitrarily optimal solution by discretizing the probability interval finely enough, an important question is: how finely do we need to discretize *in practice*? We address this question by considering a sequence of increasingly fine discretizations, starting at $L = 1$ ($p_0 = 0$ and $p_1 = 1$) and going up to $L = 50$ ($p_l \in \{0, 0.02, 0.04, \ldots, 1\}$). To ensure that whatever we find is not particular to a given setting, we also vary the number of targets between 5 and 50, as well as the network topology (Cycle, ER). The results, shown in Figure 4, are quite reassuring: $L = 10$ seems to suffice across all the settings we considered. From this point on, results based on a MILP approximation use $L = 10$, unless otherwise specified.

Comparison of the Alternative Formulations

We offered several alternative formulations of the defender's optimization problem: MINLP (the mixed integer non-linear programming approach for general-sum APGs), NLP (non-linear program specific to zero-sum APGs), and MILP.

We compare all these formulations in terms of objective value (i.e. average $V_A(0)$ over 100 random realizations of target values and network topologies) and average running time. The results in Figure 5 suggest that there is not a

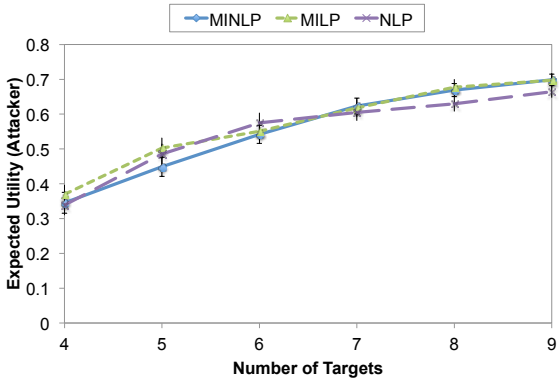

Figure 5: Comparison of average attacker utility achieved using MINLP, MILP, and NLP formulations, using the Cycle topology.

significant difference in efficacy of the programming approaches we propose. Running time, however, does in fact differentiate them. Experimentally we found that MINLP running time diverges rapidly from that of MILP: even with as few as 9 targets, KNITRO solver takes nearly 300 seconds, as compared to solving MILP using CPLEX, which takes under 2 seconds on comparable problem instances (see Figure 6). Figure 7 shows that the NLP formulation scales considerably better than MILP, solving instances with as many as 1000 targets in under 200 seconds (MILP already begins to reach its limit by $n = 50$). It is interesting to note that the graph topology plays a role in determining the difficulty of the problem: Cycles are solved much faster than ER graphs.

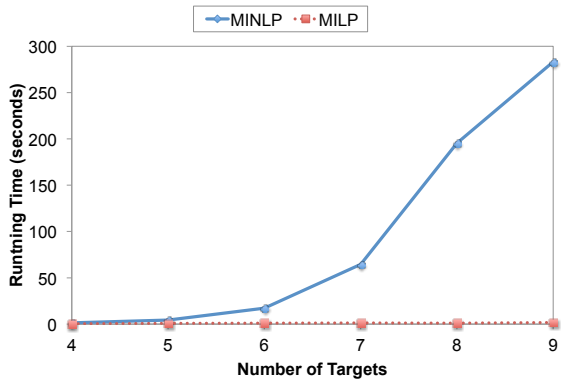

Figure 6: Running time comparison between MILP and MINLP formulations (on Cycle topology).

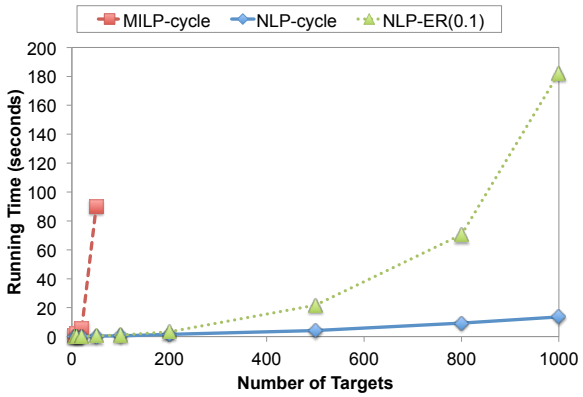

Figure 7: Running time comparison between MILP and NLP on Cycle and ER(0.1) graphs. We omit MINLP which does not scale, and the two MILP formulations yield similar results, so we only present MILP (baseline) here.

Attacks Taking Multiple Steps

As our formulation of the defender's optimization problem in the case when attacks can take more than a single step to unfold allows one to make a principled tradeoff between runtime and approximation quality, we now study this tradeoff. Specifically, we fix the number of steps an attack takes at $h = 3$, fix the number of targets at 10, and vary $1 \leq K \leq 3$,.

The results are shown in Table 1. It is quite clear that solving this problem optimally is an unlikely proposition: even with $h = 3$ and only 10 targets, solving to optimality requires, on average, over 20 minutes. Fortunately, it appears that both $K = 1$ and $K = 2$ approximations achieve near-optimal utility, and are much faster.

K	expected utility	runtime (s)
1	0.52±0.01	0.3±0.02
2	0.48±0.01	7.18±1.16
3	0.47±0.01	1325±243

Table 1: Comparison of attacker's expected value and defender's network design cost for the NLP (ND) formulation solved by IPOPT and KNITRO, and the MILP (ND) formulation. For all, the number of targets is 20 and per-edge cost is 0.02. For KNITRO, we used 4 restarts; we had not tried more, as even with 4 a significant fraction of instances (between 5 and 10%) simply stall.

Conclusion

We presented a very general model of discounted adversarial patrolling on exogenous networks. We showed that, in general, APGs do not have Markov stationary Strong Stackelberg equilibria. We then presented a collection of mathematical programming formulations for solving APGs. Our starting point is a mixed-integer non-linear programming formulation which computes an optimal randomized defense policy that conditions on bounded, but arbitrary, length history of previous defense moves, and accounts for attacks that can take an arbitrary number of steps. In itself, this formulation is quite clearly intractable. Our step towards a more practical approach is a mixed-integer linear programming approximation based on a discretized defense policies. Experimentally, we find that this MILP significantly outperforms the exact MINLP formulation, and, additionally, even a coarse discretization of the defender's strategy space already yields near-optimal solutions. Finally, we present a non-linear programming formulation for computing equilibria in zero-sum variants of adversarial patrolling games, which our experiments show to be far more scalable than the alternatives.

While we presented the first viable solution approach for discounted adversarial patrolling games, much remains to be done to manage scalability, which is still a concern, certainly in the general-sum version of the game, but even in the zero-sum variant. For general-sum games, there is some hope that minimax solutions provide good approximations, and a natural future direction is to investigate this further, perhaps generalizing similar results from Korzhyk et al. (2011). For zero-sum games, while scalability is limited due to exponential problem explosion in both the number of defense resources and the number of previous moves upon which defense decisions are conditioned, we offered some hope in our experiments that good solutions are already achievable when the defender keeps track of only a few moves. In future work, we would hope to explore this in greater depth, and tackle scalability in the number of resources by identifying structure that allows us to manage each resource independently, attempting to extend similar results from Kiekintveld et al. (2009). Yet another direction for future work would be to explore branch-and-bound search techniques that leverage the special structure of the problem.

Patrolling in adversarial settings is both a significant technical challenge, as well as of practical consequence, with US coast guard activities just one important example. Our contributions, both the general model that captures some of the most salient aspects of such settings, and the solution techniques, pave the way for deployment of game theoretic adversarial patrolling approaches in real security scenarios.

References

An, B.; Pita, J.; Shieh, E.; Tambe, M.; Kiekintveld, C.; and Marecki, J. 2011. Guards and protect: Next generation applications of security games. In *SIGECOM*, volume 10, 31–34.

Basilico, N., and Gatti, N. 2011. Automated abstraction for patrolling security games. In *Twenty-Fifth National Conference on Artificial Intelligence*, 1096–1099.

Basilico, N.; Gatti, N.; Rossi, T.; Ceppi, S.; and Amigoni, F. 2009. Extending algorithms for mobile robot patrolling in the presence of adversaries to more realistic settings. In *IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, 557–564.

Basilico, N.; Rossignoli, D.; Gatti, N.; and Amigoni, F. 2010. A game-theoretic model applied to an active patrolling camera. In *International Conference on Emerging Security Technologies*, 130–135.

Basilico, N.; Gatti, N.; and Amigoni, F. 2009. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *Eighth International Conference on Autonomous Agents and Multiagent Systems*, 57–64.

Basilico, N.; Gatti, N.; and Amigoni, F. 2012. Patrolling security games: Definition and algorithms for solving large instances with single patroller and single intruder. *Artificial Intelligence Journal* 184-185:78–123.

Basilico, N.; Gatti, N.; and Villa, F. 2011. Asynchronous multi-robot patrolling against intrusion in arbitrary topologies. In *Twenty-Forth National Conference on Artificial Intelligence*.

Bosansky, B.; Lisy, V.; Jakov, M.; and Pechoucek, M. 2011. Computing time-dependent policies for patrolling games with mobile targets. In *Tenth International Conference on Autonomous Agents and Multiagent Systems*, 989–996.

Conitzer, V., and Sandholm, T. 2006. Computing the optimal strategy to commit to. In *Seventh ACM conference on Electronic commerce*, 82–90.

Edmonds, J. 1965. Maximum matching and a polyhedron with 0-1 vertices. *Journal of Research of the National Bureau of Standards* 69:125–130.

Filar, J., and Vrieze, K. 1997. *Competitive Markov Decision Processes*. Springer-Verlag.

Jain, M.; Tsai, J.; Pita, J.; Kiekintveld, C.; Rathi, S.; Tambe, M.; and Ordóñez, F. 2010. Software assistants for randomized patrol planning for the lax airport police and the federal air marshal service. *Interfaces* 40:267–290.

Kiekintveld, C.; Jain, M.; Tsai, J.; Pita, J.; Ordóñez, F.; and Tambe, M. 2009. Computing optimal randomized resource allocations for massive security games. In *Seventh International Conference on Autonomous Agents and Multiagent Systems*.

Korzhyk, D.; Yin, Z.; Kiekintveld, C.; Conitzer, V.; and Tambe, M. 2011. Stackelberg vs. nash in security games: An extended investigation of interchangeability, equivalence, and uniqueness. *Journal of Artificial Intelligence Research* 41:297–327.

Letchford, J.; MacDermed, L.; Conitzer, V.; Parr, R.; and Isbell, C. 2012. Computing optimal strategies to commit to in stochastic games. In *Twenty-Sixth National Conference on Artificial Intelligence*.

McCormick, G. 1976. Computability of global solutions to factorable nonconvex programs: Part I - convex underestimating problems. *Mathematical Programming* 10:147–175.

Newman, M. 2010. *Networks: An Introduction*. Oxford University Press.

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 *Proc. of The 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 895–902.

Vorobeychik, Y., and Singh, S. 2012. Computing stackelberg equilibria in discounted stochastic games. In *Twenty-Sixth National Conference on Artificial Intelligence*.