Patrol Scheduling Against Adversaries with Varying Attack Durations

Hao-Tsung Yang, Shih-Yu Tsai, Kin Sum Liu, Shan Lin, Jie Gao
Stony Brook University, NY, US
{haotyang,shitsai,kiliu }@cs.stonybrook.edu, shan.x.lin@stonybrook.edu, jgao@cs.stonybrook.edu

ABSTRACT

We consider a generalization of zero-sum patrolling security game that allows the attacker choosing when, where, and how long to launch an attack, under three different attacker models. The attacker's payoff is the acquired utilities of the attack minus a penalty if the attacker is caught by the defender in patrol. The goal is to reduce the payoff of the attacker. To find the optimal defender/ attacker strategy, the game is converted to a combinatorial minimax problem with a closed-form objective function. Due to the complexity of the utility functions, we show that the minimax problem is not convex for all attacker models, even when the defender strategy is assumed as the time-homogeneous first-order Markov chain (i.e., the patroller's next visit only depends on his current location). However, for the zero penalty case, we prove that the optimal solution is either minimizing the expected hitting time or return time, based on different attacker models. We also observe that increasing the randomness of the patrol schedule helps to reduce the attacker's expected payoff for high penalty cases. Thus, to find solutions for general cases, we formulated a bi-criteria optimization problem and proposed three algorithms that support finding a trade-off between the expected maximum reward and the randomness. Another characteristic is that the third algorithm is able to find the optimal deterministic patrol schedule, although the running time is exponential on the number of patrol spots. Experiments demonstrate the effectiveness and scalability of our solutions. It also shows that our solutions outperform the baselines from state of the art in both artificial and real-world crime datasets.

KEYWORDS

Patrol Security Game; Vehicle Routing Problems; Markov Process

ACM Reference Format:
Hao-Tsung Yang, Shih-Yu Tsai, Kin Sum Liu, Shan Lin, Jie Gao. 2019. Patrol Scheduling Against Adversaries with Varying Attack Durations. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 10 pages.

1 INTRODUCTION

Public safety is crucial to everyday life. When responding to events of a criminal nature, it is necessary to consider game theoretic models and strategic behaviors, which is the focus of security games (see [44]). The problem is modeled as a Stackelberg game, consisting of a defender with a limited set of resources to protect a set of targets and an attacker who casts attacks after learning the defender's strategy and conducting careful planning. In this setting, the goal is to compute a *Stackelberg equilibrium*, a mixed strategy for the defender that maximizes the defender's utilities, under the condition that the attacker knows the defender's strategy and chooses the best response strategy. A subfamily of this domain is the *patrolling security games* or *adversarial patrolling game* [2, 11, 12, 14, 16, 25, 56]. These games are modeled as a two-player multi-stage game with infinite time horizon, where the defender moves the patroller on the vertices of a given graph to protect the targets while the attacker decides when and where to launch an attack on a vertex.

A standard way of analyzing/ solving patrolling security game is to formulate it as a mixed-integer linear programming problem and compute approximately optimal policy for the defender. However, with an infinite time horizon in the patrolling game, there are infinitely many pure strategies. Thus additional constraints are introduced to reduce the strategy space, for example, ignoring the time needed for a patroller to move between different locations [14, 24, 48, 52], if the time of moving is indeed negligible compared to the time spent for guarding. Other works assume special attacker models – the attacker taking a fixed period of time to complete an attack [11] or introducing an exponential discount factor on the attacker's utility [56]. In general, even when the number of pure strategies is bounded by these constraints, it is still challenging to handle the scalability issue due to the exponential size [44].

Our Contribution We consider a generalization of zero-sum patrolling security game in which the attacker is given not only the freedom to decide when and where to launch the attack but also the duration of the attack in order to maximize the expected payoff. The attacker's payoff is the acquired utilities of the attack minus a penalty if the attacker is caught by the defender in patrol. To the best of our knowledge, this is the first work considering varying attack duration in the patrolling game. We consider three different attacker models which affects how much information that the attacker can possibly gain by observing the patrol routes. The game is converted to a combinatorial minimax problem and one main challenge is the exponentially increased size in the solution space due to varying attack durations. Furthermore, for general utility functions, the problem of finding optimal defender strategy is not convex in general.

Despite the complexity of the problem, we show that when restricting the defender strategy as a time-homogeneous first-order Markov chain, finding the optimal defender strategy can be formulated as a closed-form minimax problem. In special cases with the zero penalties, the optimal solutions can be linked to minimizing the expected pairwise/ average hitting time or return time, depending on the visibility model of the attacker. In a scenario of high penalties, increasing the entropy of visiting time for each spot helps to reduce the attacker's expected payoff, since the attacker would pay

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

a high price if he is getting caught, even with a small chance. Thus, a randomness patrol schedule with high entropy of visiting time is beneficial to decrease the attacker's payoff. By the aforementioned observations, we formulate a bi-criteria problem of balancing the attacker's expected maximum reward and randomness of the patrol schedule and use the solution as the defender strategies for the original game. To the best of our knowledge, this is the first work to consider the randomized strategies in vehicle routing problems. We propose three algorithms: TSP-based solution (TSP-b), Biased random walk (Bwalk), and Walk on state graph (SG) where the first two are related to TSP and random walk solutions and the third is from the state machine mechanism. All proposed algorithms can balance the two criteria by a parameter α. In addition, SG can be used to find an optimal deterministic patrol schedule for the original game with any utility functions. Experiments show that three algorithms are adaptive to various utility functions/ penalties and both TSP-b and Bwalk are scalable with the increase to the number of spots. Our solutions also outperform (achieving lower expected payoff for the attacker) other baselines such as Markov chains of minimum hitting time [46], and Maxentropic Markov chains [26].

2 RELATED WORK

2.1 Surveillance and Security Game

Patrolling and surveillance problems have been widely studied in the robotics and operations research communities. In the non-strategic setting, algorithms are designed for traversing a given region with centralized optimization with specific objectives [23, 30, 40, 49, 53]. S. Alamdari et al. [3] focus on the problems of minimizing the maximum time duration between any two consecutive visits in a spot and provide a $\log n-$approximation algorithm for general graphs. A rigorous result was developed deeper on special graphs such as a chain, tree, and cyclic graphs [45]. In the strategic setting, the patrol strategies are designed to defend against intelligent intruders who would avoid getting detected. Thus, many works model the movement of the patrollers as Markov chains or random walks which embed the unpredictability of the patrol routes [8, 18, 22, 28, 46]. Patel et. al [46] study the minimization of the first passage time to quickly detect the intruder. Duan et. al [22] study Markov chains with maximum return time entropy, which provides an adapt solution to detect the intruder if he only lasts for a short period.

In a more complicated strategic setting, the behavior of the attackers is also included in the model, i.e., security game, which is first introduced by Kiekintveld et al [36] and has been widely applied in security domains to generate defender strategies [1, 13, 55]. Representative examples include deploying randomized checkpoints and canine patrol routes in LAX airport [48], game-theoretic scheduler for US Federal Air Marshals [54], scheduling patrolling routes of ports for US coast guards [24, 51], and aiding rangers for protecting wildlife in Uganda [57]. In addition, based on the various designs of the utility function and attacker-defender interactions, many different forms of security games have been developed to fit the application scenarios. Yevgeniy et al. [56] added discounted time factor in the attacker payoff function since delayed attack for a long time highly increases the likelihood of being caught. Bošanskỳ et al. [16] considered the case that targets are moving through areas according to deterministic movement schedules.

For handling scalability, current solutions are mainly application specific. For example, in ASPEN [31], there are more than one patroller in the setting. Thus, each patroller is solved independently to avoid a combinatorial explosion of schedule allocations. This idea is extended to both attacker's and defender's strategies for road network security in RUGGED [32]. Shieh et al. [52] combined previous ideas and used the TSP as a heuristic tool to order the search space that can efficiently provide a heuristic solution for each patroller. Basilico et al. [14] assumed that the adversary takes a period of times to attack a target and used some reduction techniques to handle the scalability issue. However, the formulation of their problem is only suitable for unweighted graphs and the attacker cannot control the attack duration.

2.2 TSP

The problem of planning patrol routes is related to the general family of vehicle routing problems (VRPs) and traveling salesman problems (TSPs) with constraints [9, 39, 47, 58]. This is a huge literature thus we only introduce the most relevant papers.

TSP is a well-known NP-complete problem in combinatorial optimization and has been discussed in operation research [5, 19, 27, 29, 37]. Christofides algorithm [20] provides a tour whose length is less or equal to 1.5 times of the minimum possible. Additionally, there are two independent papers that provide polynomial-time approximation scheme (PTAS) for Euclidean TSP by Mitchell and Arora [6, 42]. There are many variations of TSP that consider multiple objectives [10, 15]. However, in this work, one objective is to increase the randomness between generated tours, which, to the best of our knowledge, had not been studied in the literature. The other objective is related to minimize the maximal weighted latency among spots of the tour, which has been discussed in some works [3, 41]. One difference is that this work generalizes the "weight latency" as functions rather than constant weights.

3 MODEL OF PATROLLER AND ATTACKER

The patrol game is structured as a Stackelberg zero-sum game. That is, the defender executes a strategy first and the attacker chooses the best strategy based on the defender's executed strategy. The attacker's objective is to choose a strategy that maximizes his (expected) payoff and the defender's objective is to choose a strategy that minimizes the attacker's maximum expected payoff.

Mathematically, given a tuple (G, H, M), where $G = (V, E, W)$ is a weighted graph with vertices $V = \{1, 2, \cdots n\}$, edge set E, and edge-weight matrix W representing the traveling costs. M is the penalty cost ($M \geq 0$) and each vertex j has a utility function $h_j \in H$. Time is discretized into time slots. The attacker can launch one attack and can decide where (j), when (τ) and how long (T) the attack lasts. During the attack, at the $(\tau + t)$-th time slot the attacker collects a utility $h_j(t)$, where $1 \leq t \leq T$. Note that the utility function can be node dependent. We assume that $h_j(t) \geq 0$ always.

If the attacker is caught by the defender at the $(\tau + t')$-th time slot, the attacker would pay a penalty M and be forced to stop the attack. Thus, the total collected utilities of the attacker is $\sum_{t=1}^{t'} h_j(t) - M$. Otherwise, the total collected utilities is $\sum_{t=1}^{T} h_j(t)$ if the attacker is not caught.

Notice that in the adversarial patrolling games, it is possible that the attacker waits for a long time and acquires additional information such as when the patroller passes by. In the literature, there are different models which specify how much information the attacker can collect.

- *Full visibility*: The attacker has a probe in each spot such that it would notify the position of the patroller when he arrives any spot during the game. This model is used in *Patrolling Security Games* [14, 56].
- *Local visibility*: The attacker would have to choose a spot j first and would launch an attack right after the patroller leaves spot j [8].
- *No visibility*: The attacker cannot know the patroller's positions during the whole game. This is a common assumption in [4, 48].

In general assumption, the attacker knows the strategy used by the defender before the game starts in any attacker models.

4 STRATEGY WITH FIRST-ORDER MARKOV CHAIN

To tackle the problem, the defender's strategy is restricted as a time-homogeneous first-order Markov chain (only in this section). That is, the patroller movement is modeled as a Markov process over graph G with a transition matrix P, which is known by the attacker. Notice that any high-order Markov chain can be "flatten" into the first order one by some standard methods (which takes time exponential on the order of the Markov chain) [14].

To calculate the attacker's payoff we use the notation of *first visit matrix* F [8], where each element represents the visit probability distribution from a spot i to another spot j. In detail, given graph G and transition matrix P, the probability of taking k slots for the patroller, starting at i to reach j for the first time is given by

$$F_k(i,j) = \begin{cases} p_{ij}\mathbb{1}_{w_{ij}=k}, & k = 1 \\ \sum_{h\neq j} p_{ih} F_{k-w_{ih}}(h,j) + p_{ij}\mathbb{1}_{w_{ij}=k}, & k \geq 2, \end{cases}$$

where w_ij is the travel cost from spot i to j and $\mathbb{1}_{w_{ij}=k}$ is the indicator function which returns 1 if $w_{ij} = k$, and 0 otherwise. $F_k(i,j) = 0$ when k is non-positive. Extensively, we define *expected hitting time matrix* A, where each entry $a_{i,j} = \sum_{k=1}^{\infty} k \cdot F_k(i,j)$.

4.1 Attacker has full visibility

In the model of full visibility, the attacker knows the exact position of the patroller among all spots. Denote $Z_{i,j,T}$ as the expected payoff if the attacker launches an attack at j with the attack period T when the patroller is at i. In any time slot t during the attack, where $1 \leq t \leq T$, there are only 3 possible events: the patroller comes to spot j (after visiting i) in the period of time 1 to $t-1$, the patroller comes exactly at time t, or the patroller comes after time t. In the first case, the attacker cannot collect utility at time t since the attack is enforced to stop at t', where $t' < t$ (the penalty is also paid at time t' too). In the second case, the attack is caught at time t thus there is a penalty M substrated from the attacker's payoff. In the third case, the attacker collects utility $h_j(t)$. Thus, the expected payoff at time $t, 1 \leq t \leq T$, can be expressed as a closed form associated with F.

$$z_{i,j}(t) = (h_j(t) - M) \cdot F_t(i,j) + h_j(t)\Big(\sum_{k=t+1}^{\infty} F_k(i,j)\Big). \quad (1)$$

The total (expected) payoff during the whole attack period is $Z_{i,j,T} = \sum_{t=1}^{T} z_{i,j}(t)$, which is called as the payoff matrix. The attacker chooses an element of Z with the highest payoff, which describes his strategy of when, where, and how long the attack lasts.

For the defender, the problem of choosing a best strategy can be formulated as a minimax problem:

$$\min_P f(P), \text{ where } f(P) = \max_{i,j,T} Z_{i,j,T}.$$

For general utility function h_j and penalty M, the Hessian matrix of f is not guaranteed to be semi-definite thus $f(P)$ is not convex in general. However, in special cases $f(P)$ has strong connection with the expected hitting time matrix A.

Observation 1. *If $M = 0$ and the utility functions are all constant functions, then $f(P)$ is either ∞ or the maximum weighted expected hitting time of all pairs (i,j), with the weight for (i,j) as the constant of the utility function h_j.*

Proof. If the transition matrix P is reducible, i.e, there exists a pair of vertices i, j such that the patroller starting at i would never visit spot j, then the attacker can choose to attack j for infinitely long. In this case $Z_{i,j,\infty} = \infty$.

Now, assume that the transition matrix is irreducible. Denote by h_j the constant of the utility function at spot j. Given an attack period T, $M = 0$, from Equation 1, $Z_{i,j,T}$ can be simplified as

$$Z_{i,j,T} = h_j \cdot \sum_{k=1}^{T} k \cdot F_k(i,j) + h_j \cdot T \cdot \sum_{k=T+1}^{\infty} F_k(i,j). \quad (2)$$

Since $z_{i,j}(t) \geq 0$ for any t. Thus, taking $T = \infty$ period maximizes his payoff. That is,

$$f(P) = \max_{i,j} Z_{i,j,\infty} = \max_{i,j} h_j \cdot \sum_{k=1}^{\infty} F_k(i,j) \cdot k = \max_{i,j} h_j \cdot a_{i,j},$$

where $a_{i,j}$ is the expected first hitting time from i to j. □

At the defender's side, minimizing the maximum of all pairwise expected hitting times is still an open question to the best of our knowledge. One can find a relevant work which provides a lower bound and discusses the complication for this question [17].

4.2 Attacker has local visibility

In this model, assume the attacker's strategy is to attack spot j with the attack period T. Denote $z'_j(t)$ as the utility he collects for every time t where $1 \leq t \leq T$,

$$z'_j(t) = z_{j,j}(t) \quad (3)$$

By a similar discussion in Observation 1, one can infer that the best strategy for the attacker is to attack the spot with the longest expected (weighted) return time if the utility functions are all constants and the penalty is zero. If all edges have weight one, the optimal defender strategy can be derived by constructing an ergodic Markov chain with stationary distribution π^*, where $\pi^*_j = \frac{h_j}{\sum_{i=1}^{n} h_i}$, since the expected return time of a spot j is $1/\pi^*_j$ [50].

4.3 Attacker has no visibility

In this case, the attacker has no information of the patroller's trace thus it is meaningless for the attacker to choose when to launch an attack; instead, the payoff of attacking spot j is the expected payoff when the patroller is either at a random spot i or travels on a random edge (i,j). For the following analysis, we only consider

the attacks that starting at the time when the patroller is at exactly one of the spots. For general cases, it would underestimate the attacker's expected payoff at most $\max_{i,j} \sum_{t=1}^{w_{ij}} h_j(t)$ utilities.

Denote $Z''_{j,T}$ as the cumulative expected payoff for attacking j with period T and $z''_j(t)$ is the expected payoff at time t. Assume the attack is launched at a random time slot, $z''_j(t)$ is

$$z''_j(t) = \sum_{i=1}^{n} \pi_i \cdot z_{i,j}(t),$$

where π is the stationary distribution with transition matrix P. Thus, the cumulative expected payoff is

$$Z''_{j,T} = \sum_{t=1}^{T} z''_j(t) = \sum_{t=1}^{T} \sum_{i=1}^{n} \pi_i \cdot z_{i,j}(t) = \sum_{i=1}^{n} \pi_i Z_{i,j,T}. \quad (4)$$

Denote κ_i as the Kemeny constant [34], the expected hitting time when the walk starts at i, $\kappa_i = \sum_{j=1}^{n} a_{i,j}\pi_j$. It is known that the Kemeny constant is independent of the start node [35]. Thus, the Kemeny constant can be written as another formation

$$\kappa = \sum_{i=1}^{n} \pi_i \sum_{j=1}^{n} a_{i,j}\pi_j. \quad (5)$$

Equation 5 can be written as an expression with matrix A.

$$\kappa = \pi^T A \pi. \quad (6)$$

Now, suppose $f''(P) = \max_{j,T} Z''_{j,T}$ is the function maximizing the expected payoff, the following observation is shown.

Observation 2. *If $M = 0$ and the utility functions are all constant functions, $f''(P)$ is either ∞ or the Kemeny constant multiplying with the maximum constant among all utility functions.*

Proof. From the same argument in Observation 1, $f''(P)$ goes to ∞ when the Markov chain is reducible. Now, consider an irreducible Markov chain, from Equation 4, we have

$$f''(P) = \max_j \sum_{i=1}^{n} \pi_i Z_{i,j,\infty} = \max_j h_j \sum_{i=1}^{n} a_{i,j}\pi_i.$$

On the other hand, take transpose on both side in Equation 6, we have

$$\kappa = (\pi^T A \pi)^T = \pi^T A^T \pi.$$

Thus, A and A^T has the same Kemeny constant. The Kemeny constant of A^T is actually $\kappa_j = \sum_{i=1}^{n} a_{i,j}\pi_i$ for spot j, which means

$$f''(P) = \kappa \max_j h_j.$$

□

Observation 2 shows that when the penalty is zero with constant utility functions, the attacker's best strategy is to attack the spot with highest utility. From the defender side, it has to determine P such that the Kemeny constant is minimized. When all edges have weight 1, a simple solution is to construct P same as the adjacent matrix of a directed n-cycle in G [38]. In other cases, it has to minimize the Kemeny constant subject to a given stationary distribution [46].

4.4 High penalty scenarios

When $M \gg h_j(t)$ for all spots j and all time t, Equation 1 can be simplified as

$$z_{i,j}(t) = h_j(t)(\sum_{k=t+1}^{\infty} F_k(i,j)) - M \cdot F_t(i,j).$$

Assume that the attacker has full visibility and all utility functions are constants.

$$f(P) = \max_{i,j,T}(h_j \cdot T - (M+1) \cdot \sum_{t=1}^{T} F_t(i,j)). \quad (7)$$

At the defender side, it is beneficial to increase $\sum_{t=1}^{T} F_t(i,j)$ for all (i,j) pairs. Thus, having a schedule which is more random could help in this case. This observation also works in other two attacker models.

5 STRATEGY FOR GENERAL CASES

In the previous section, we show that in special cases (e.g. When the attacker has no visibility, the penalty is zero, and utility functions are all constants) the minimax problem of the zero-sum game is possibly solvable. In general, the optimization problem is not convex. Our solution for general cases is motivated by two observations. First, when the penalty is zero, the optimal schedule is to minimize the expected (pairwise/ average) hitting time or return time. Secondly, if the penalty is significant, it would be better to increase the randomness of the patrol schedule to "scare" the attacker away. In fact, there are prior works emphasizing each one as the objective for the patrol mission [22, 26, 46]. However, to the best of our knowledge, this is the first work to consider both objectives at the same time.

Specifically, we consider two optimization criteria: *expected maximum reward* (EMR) and *entropy rate* ($\mathcal{H}_\nabla$). Given a patrol schedule $X = (X_1, X_2, \cdots)$ as a random variable sequence and $(\omega_1, \omega_2, \cdots)$ is one of its possible realizations. Denote $U_j = (u_1, u_2, \cdots)$ is the sequence of times that the patroller visits j, i.e., $\forall u_r \in U_j, \omega_{u_r} = j$. Then, the maximum return time is

$$\phi_j = \max_{u_r \in U_j} \{ \sum_{k=u_r}^{u_{r+1}} w_{\omega_k \omega_{k+1}} \}$$

and the maximum cumulative rewards of j is $\sum_{t=1}^{\phi_j} h_j(t)$.

Since $\{\omega\}$ comes from a randomized process, we can define EMR as the expectation of the maximum (cumulative) rewards among all spots.

$$\text{EMR} = \max_{j \in \{1,2,\cdots n\}} \mathbb{E}[\sum_{t=1}^{\phi_j} h_j(t)].$$

In the following paragraphs, EMR(X) is used for emphasizing the value of EMR of schedule X. As a reminder, minimizing the maximum reward can be NP-hard since this problem has TSP as a special case.

On the other hand, the entropy rate is to quantify the randomness of a schedule X. It is defined as the following.

$$\mathcal{H}_\nabla(X) = \lim_{m \to \infty} \frac{\sum_{k=1}^{m} \mathcal{H}(X_k)}{m},$$

where $\mathcal{H}$ is the entropy function in information theory [33].

In the rest of this section, three algorithms are proposed: TSP-based (TSP-b), Biased Random Walk (Bwalk), and Walk on State Graph (SG) that balance the two criteria with an input parameter

α. One characteristic of SG is that it can generate the optimal deterministic solution and heuristic non-deterministic one.

5.1 TSP-based solution

The Algorithm *TSP-based solution* (TSP-b) is perturbing the optimal (or approximately optimal) deterministic EMR solutions by a parameter α. Adjusting this skipping parameter α will balance the two criteria. Roughly speaking, the main idea is to traverse on a deterministic tour but each vertex is only visited with probability α (i.e., with probability $1-\alpha$ it is skipped). Obviously, Algorithm TSP-b generates a randomized schedule. Also, since the algorithm works with a metric (with triangular inequality), the total travel distance after one round along the tour is bounded by the original tour length. Hence, the expected reward can be bounded.

The following is the analysis of EMR and entropy rate for TSP-b when the utility functions are polynomial functions with the maximum degree d.

5.1.1 Analysis of TSP-b with same utility functions. When the utility functions among all spots are the same, Algorithm TSP-b firstly generates a approximated-TSP tour $Q = \{q_1, q_2, \cdots q_n\}, q_i \in \{1, 2, \cdots n\}$ by, for example, a PTAS algorithm [7, 42]. Denote Y as the randomized schedule perturbed by α. Now, assume the spot of an arbitrary index k in the schedule is i, i.e., $Y_k = i$, without loss of generality, the tour Q is shifted such that $q_1 = i$. Thus, the probability of the next spot to visit being q_j is

$$\text{Prob}(Y_{t+1} = q_j | Y_t = q_1) = \begin{cases} \sum_{x=1}^{\infty}(1-\alpha)^{xn-1}\alpha & \text{if } j = 1 \\ \sum_{x=0}^{\infty}(1-\alpha)^{xn+j-2}\alpha & \text{if } j = \{2, 3, \cdots n\}. \end{cases} \quad (8)$$

Denote $\text{Prob}(Y_{k+1} = q_j | Y_k = q_1)$ as γ_j, then the entropy rate of Y would be

$$\mathcal{H}_\nabla(Y) = \sum_{j=1}^{n} \gamma_j \log \frac{1}{\gamma_j} \quad (9)$$

On the other hand, to bound $\mathbb{E}[\phi_i]$ we mainly need to determine how many rounds does the patroller tour around Q before spot i is visited again (a round is defined as the number of time slots for touring Q). Suppose the time taken for Q is $T(Q)$. Each such tour by triangle inequality has length at most $T(Q)$. Define β_i as the number of the rounds traveled until i is visited again. The probability of β_i is calculated as follows,

$$\text{Prob}(\beta_i = k) = (1-\alpha)^{k-1}\alpha.$$

Denote $\beta = \max_i \beta_i$. the probability distribution for β is bounded,

$$\text{Prob}([\beta \le k]) = \prod_i \text{Prob}(\beta_i \le k) = (1-(1-\alpha)^{k-1})^n.$$

The expected value of β is,

$$E[\beta] = \sum_{k=1}^{\infty} \text{Prob}(\beta \ge k) = \sum_{k=1}^{\infty} 1 - \text{Prob}(\beta \le k-1)$$

$$= \sum_{k=1}^{\infty}(1-(1-(1-\alpha)^{k-1})^n).$$

By tuning the probability α, TSP-b has different bounds on EMR and entropy rate $\mathcal{H}$. For a small α, lots of sites are skipped creating a schedule with high randomness, but EMR is also higher. On the other hand, for a large α, the sites are visited more frequently with lower reduced entropy rate. With some calculations, the analysis of α is summarized in Table 1. Remark that when α is sufficiently small ($\alpha < \frac{1}{n}$), TSP-b achieves maximum entropy and when α is sufficiently large ($\alpha > \frac{n-1}{n}$), it provides $(1+\frac{n}{n-1})^{d+1}$-approximation for EMR compared to the TSP tour Q, with the maximum degree d among all the utility functions. Despite that, when α is a constant between 0 to 1, A constant entropy and about $\log^{d+1} n$ extra factor of EMR are derived.

α	$\alpha < \frac{1}{n}$	$\alpha = \Theta(1)$	$\alpha > \frac{n-1}{n}$
EMR	$O(n^{d+1}\log^{d+1} n)$	$O(\log^{d+1} n)$	$O((1+\frac{n}{n-1})^{d+1})$
$\mathcal{H}_\nabla$	$\Theta(\log n)$	$\Theta(1)$	$\frac{\log n}{n}$

Table 1: The summary of the analysis for TSP-b when all the utility functions are the same with the maximum degree d ($0 < \alpha \le 1$).

5.1.2 Analysis of TSP-b with different utility functions. When the utility functions are different, TSP-b firstly generates the deterministic schedule by *Bamboo garden trimming* (BGT) algorithm [41] and then perturb it into a randomized schedule with α.

One can describe BGT as a vertex-weighted version of TSP. The objective is to output schedule such that the maximal weighted visited time among all spots is minimized. For the input, the graph is set up as G and each vertex j has a weight l_j, which is the coefficient of degree d in h_j, where d is the maximum degree among all spots. BGT divides spots into groups such that the weight of each group is less than 2. Then, the patroller visits one group with constant distance and switches to another until all spots are visited. In this way, it can not be hard to identify that the schedule generated by BGT gives $O(\log^{d+1} n)$ approximation of EMR.

For analyzing EMR in TSP-b, notice that when a certain spot i is skipped, the attacker can collect $O(\log^{d+1} n)$ additional utility if he attacks i. Thus, the expected reward of the attacker would be $E[\beta_i] \cdot O(\log^{d+1} n)$, where $\beta_i - 1$ is the number of times skipping i between two consecutive visits of i in this randomized schedule. Follow the similar analysis of β_i in the case of same utility functions, the bounds of EMR are the values of the second row in Table 1 multiplying with $O(\log^{d+1} n)$.

5.2 Biased Random Walk

Algorithm *Biased Random Walk (Bwalk)* uses a biased random walk to decide the patrol schedule. Define matrix $W' = (w'(i,j)) \in \mathbb{Z}_{\ge 0}^{n \times n}$. For each pair (i, j),

$$w'(i,j) = 1/\alpha^{w(i,j)}, \alpha > 1,$$

where α is an input parameter. Define stochastic matrix P' as

$$P'(i,j) = \frac{w'(i,j)}{\sum_{(i,j')\in E} w'(i,j')} \quad \text{if } (i,j) \text{ is an edge}$$

$$= 0 \quad \text{otherwise.}$$

5.2.1 Analysis of Bwalk with same utility functions. In this case, Bwalk repeatedly generates a set of randomized tours $\{S_1, S_2, \cdots\}$. Each tour S_l is an Euler-tour traversing on a randomized spanning tree Γ_l, where Γ_l is generated by the biased random walk with transition probability P'.

Let $(B_k; k \ge 0)$ be the biased walk on G with B_0 arbitrary. For each spot i, let v_i be the first hitting time:

$$v_i = \min\{k \ge 0 : B_k = i\}.$$

From $(B_k; k \geq 0)$, a randomized spanning tree Γ can be constructed, which consists of these $n-1$ edges,

$$(B_{v_i-1}, B_{v_i}); i \neq B_0.$$

Notice that the probability of generating a specific tree Γ is proportional to the product of $w'(i,j)$, for all edge $(i,j) \in \Gamma$ [43]. Thus, by controlling the input parameter α, the two criteria can be balanced.

Denote the schedule generated by Bwalk as Y_B. If $\alpha = 1$ and assume that graph G is a complete graph, S_l is actually a random permutation of n spots, which has the entropy $\log n + \log(n-1) + \cdots 1 = \log(n!) = O(n \log n)$. Thus, the entropy rate of Y_B is

$$\mathcal{H}_\nabla(Y_B) = \lim_{m\to\infty} \sum_{l=1}^{m} \frac{\mathcal{H}(S_l)}{m} = \lim_{m\to\infty} \sum_{l=1}^{m} \frac{\frac{n\log n}{n}}{m} = O(\log n).$$

On the other hand, the expected reward is bounded by the expected time of traversal on the uniform random spanning tree. Since each edge is traversed at most twice, the length of the tour is less than $2n\eta$, where η is the maximum distance among all edges. Thus, the maximum payoff of the attacker is actually $\max_j \sum_{t=1}^{2n\eta} h_j(t) = O(n^{d+1})$, if the utility function is polynomial with maximum degree d.

In other cases that $\alpha > 1$, the generated spanning tree is more likely a low-weight tree. Thus, the traversing distance is lower which makes EMR lower. However, the entropy would also become lower due to the probability distribution among all generated spanning tree is more "biased".

5.2.2 Analysis of Bwalk with different utility functions. When the utility functions are not the same, Bwalk would use BGT (which is introduced in *Analysis on TSP-b with different utility functions*) as a backbone. That is, when the patroller visits spots in each group with a constant distance, the tour which he has followed is not a deterministic tour but an Euler tour traversing on a randomized spanning tree of the vertices in the group. Similar to the case of the same utility functions, the randomized tour in each group is regenerated every time when the patroller visits all spots in the group.

5.3 Walk on State Graph

Algorithm *Walk on the State Graph* (SG) with a parameter α generates the schedule by a state machine with the transition process as another random walk.

5.3.1 Deterministic SG. One characteristic of deterministic SG is that it generates the optimal deterministic schedule for any utility functions and has the running time exponential in the number of sites.

Define D is a state machine and each state x is a $(n+1)$-dimension vector $x = (x_1, x_2, \cdots, x_n, k_x)$, where $x_j \in \mathbb{R}, x_j \geq 0$ and $k_x \in \{1, \cdots, n\}$. x_i represents the maximum utility the attacker could have collected since the last time the defender leaves spot i. The last variable represents the defender's current position.

State x, y is said to have an arc from x to y if $y = (y_1, y_2, \cdots, y_n, k_y)$, where

$$y_i = \begin{cases} h_i(x_i + d(k_x, k_y)), & \text{if } i \neq k_y \\ 0, & \text{otherwise.} \end{cases}$$

$d(k_x, k_y)$ represents the time needed to travel from k_x to k_y. An arc represents the change of state from x to y when the defender moves from k_x to k_y.

Clearly, any periodic R schedule of the defender can be represented as a cycle on the state machine defined above. Further, the state diagram captures all the information needed to decide on the next stop. Although there could be infinitely many states as defined above, only a finite number of them is needed. Basically, let's take a periodic schedule S with the kernel as some traveling salesman tour C. Suppose the maximum utility of this schedule is Z. Z is finite and is an upper bound of the optimal value. Thus, all states x that have any current utility of x_j greater than Z can be removed. This will reduce the size of the state machine to be at most $O(Z^n)$.

Now we attach with each edge (x, y) a weight as the maximum payoff among all variables within state x, y. That is,

$$w(x, y) = \max\{x_1, \cdots x_n, y_1, \cdots y_n\}.$$

For any cycle/path in this state machine, define *bottleneck* weight as the highest weight on edges of the cycle/path. The optimal deterministic schedule is actually the cycle of this state machine with the minimum bottleneck weight. To find this cycle, the first step is to find the *minimum bottleneck path* from any state u to any state v by Floyd-Marshall algorithm. The total running time takes time $O(|V|^3)$, where $|V|$ is the number of vertices (states) in the state machine. The optimal tour is obtained by taking the cycle $u \rightsquigarrow v \rightsquigarrow u$ with the minimum bottleneck value for all possible u, v. The total running time is still bounded by $O(|V|^3)$.

5.3.2 Non-deterministic SG. Since the state graph records the utility that would be collected at each site from the historical trace at each state, we run a random walk on the state graph with a probability dependent on the utility of the state.

Each state is defined as the aforementioned state machine D. From each state, the random walker can possibly move to $\deg(k_x)$ different states where $\deg(k_x)$ is the degree of spot k_x in G. The probability of moving from state x to y is

$$c_{x,y} = \min_{i\in\{1,2,\cdots,n\}} \frac{1}{y_i^\alpha},$$

where α is the given input parameter. Let the transition probability from state x to all possible y to be proportional to their edge weights. That is,

$$\text{Prob}(x, y) = \frac{c_{x,y}}{\sum_{(x,w)\in E(D)} c_{x,w}},$$

where $E(D)$ is the edge set of D.

Although there are (in the worst case) exponential states respect to the number of sites in the state graph, the probability of walking on each possible state is determined by local information $\{y_1, y_2, \cdots y_n\}$. Thus, the running time of the random walk depends only on the desired length of the output schedule.

5.4 Solving Patrolling Game with proposed algorithms

For anyone of our algorithms, once we have a family of schedules, parameterized by a parameter α, we can solve for the best choice of α when the penalty and the attacker model are introduced, to achieve the best balance of randomness and EMR. Given the tuple (G, H, M), we look for the optimal value for the parameter α to minimize the maximum payoff of the attacker.

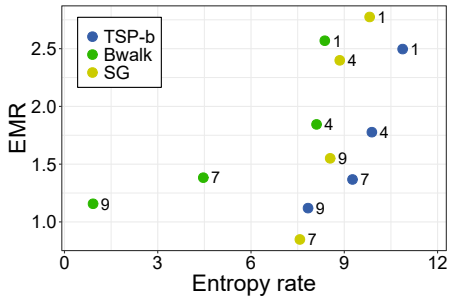

Figure 1: The values of expected maximum reward (EMR) and Entropy rate when the input parameter $\alpha = (1, 4, 7, 9)$. TSP-b has the most efficient tradeoff since it achieves the lowest EMR with the highest entropy rate.

6 EXPERIMENTS

We evaluate the proposed algorithms, *TSP-based* (TSP-b), *Biased random walk* (Bwalk), and *State graph walk* (SG), with two baselines *Markov chains with minimal Kemeny constant* (minKC) [46] and *Markov chains with maximum entropy* (maxEn) [26]. The experiments are based on artificial datasets and Denver crime dataset [21] with three different attacker models, Full visibility (Full vis.), Local visibility (Local vis.), and No visibility (No vis.). There are three major observations.

(1) Our algorithms realize the tradeoff between expected maximum reward (EMR) and entropy rate. For comparison, TSP-b and Bwalk have more freedom to control EMR and entropy rate with parameter α (Figure 1).
(2) For all algorithms, when the penalty increased, the attacker's (expected) payoff decreased. For the same evaluation setup, the attacker's payoff is the minimum when the attacker adopts the model of no visibility, and the highest, when the attacker adopts full visibility. Roughly speaking, MaxEn performs worse than our algorithms among all setups. TSP-b, Bwalk, and SG performs well when the utility function is not constant (Figure 3, 4). MinKC has comparable performance when the utility function is constant. In addition, our algorithms have low standard errors in all settings (Table 2).
(3) TSP-b and Bwalk are more scalable with the increase in the number of spots. One reason is that these algorithms are perturbed the tours from TSP/BGT, which are more delicate designed routes (Figure 8).

We define a unit length as the distance that the patroller takes a one-time slot to travel. For simulations, all spots are randomly generated from a 20000×20000 grid. Without specification, the number of spots in a setup is 30. For Denver dataset, the geographic range is in Denver City only, which has 78 neighborhoods. We set up the utility functions as polynomial ones with maximum degrees 0 (constant) or 1 (linear) for demonstration. Each one represents a different type of crimes. The coefficients of utility functions are generated uniform randomly between .001 to 1. In the real-world setting, we use the Denver Crime Dataset to learn the coefficient based on the number of crimes among different types in each neighborhood. For the baselines, minKC, and maxEn subject to stationary distribution constraints. To fit the setup of utility functions, we set up the stationary distribution for each spot j propositional to b_j, where b_j is the coefficient of the maximum degree in its utility function h_j.

In the game, each solution generates a patrol schedule as the defender's strategy. The attacker's payoff Z is realized by attacking spot i from time t_s to t_e. In this setting, we empirically calculated the expected payoff to all possible spot i according to all possible attack period t_s, t_e under specific attacker models. We then derived the attacker's maximum (expected) payoff. Due to the raw values are high, all payoff values are divided by ζ^{d+1}, where ζ is the diameter of all spots. In each experiment, each bi-criteria algorithm generates around 8 to 10 schedules based on different values of parameter α. The values of α are uniformly generated in the following domains. TSP-b: $[0.1, 1]$, Bwalk: $[0, 4.5]$, SG: $[0, 80]$. Generally speaking, increasing the number of α values would increase the performance of the algorithm but takes more computation time, which is a performance-complexity trade-off.

6.1 EMR v.s. entropy rate

Figure 1 reports the performance of algorithms under *expected maximum reward* and *entropy rate*, which are mentioned in Section 5. In y-axis, we scale the EMR as 1 if the maximum reward is generated by BGT. Each point represents the schedule which is generated by different algorithms and the digit aside each point denotes the value of the input parameter α. α is unified from 1 to 10 among three proposed algorithms, where the real-value may from different domains. For example, in TSP-b, the skip probability is 0 as $\alpha = 1$ and the skip probability is 90% as $\alpha = 10$. For TSP-b and Bwalk, the higher the value of α indicates the higher randomness of the schedule. For SG, the lower the value of α indicates the higher randomness of the schedule. In fact, one can see that the entropy rate and EMR increased with higher α value in TSP-b and Bwalk. However, this tradeoff is not that clear in SG for a high α value.

6.2 Attacker's payoff in artificial and real-world scenario

The experiments are examined with the following variables; penalty values, the maximum degree of utility functions (1, 2, 3), and the attacker models (Full vis., Local vis., No vis.). The last figure reports the simulation result of Denver crime dataset.

Each figure shows the attacker's (expected) payoff under different penalties. Each realization has been run 10 times and the y-axis is the average attacker's payoff with standard errors. We interpret an algorithm has better performance if and only if the attacker has the lower payoff in the schedule generated by this algorithm.

In the experiments of constant utility functions (Figure 2, 5, 6), one can see that the payoff drop down when the penalty increased. In addition, attackers with lower visibility (E.g., Full visibility v.s. No visibility) also has a lower payoff. Although TSP-b performs the best in Full vis., minKC is comparable in Local and No vis.. This reflects our observation in Section 4.2 and 4.3, that the optimal solution has a strong correlation with the minimum hitting time.

In the experiments of non-constant utility functions (Figure 3, 4, 7), our algorithms clearly outperform the baselines in most cases. For example, the attacker's payoff is 344 for minKC but only 0.46 for SG in the case of linear utility functions, 600 penalty value. One possible reason is that minKC and maxEn are designed only for

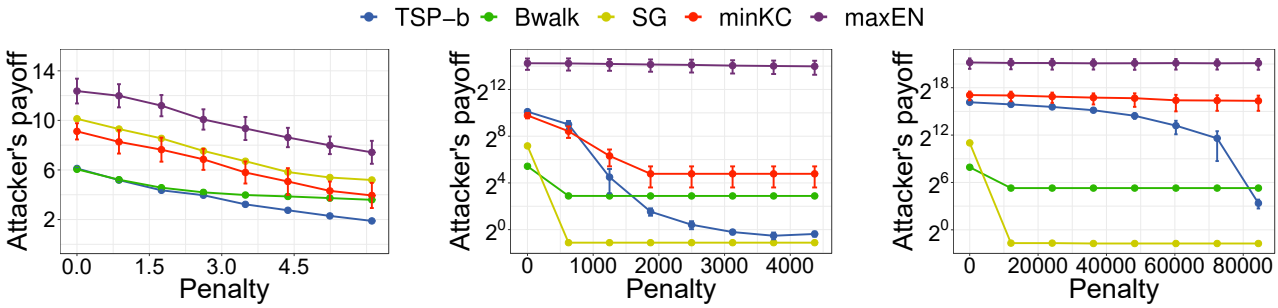

Figure 2: Constant utility, Full vis.

Figure 3: Linear utility, Full vis.

Figure 4: Quadratic utility, Full vis.

Figure 5: Constant utility, Local vis.

Figure 6: Constant utility, No vis.

Figure 7: Denver, Full vis.

Table 2: Comparing attacker's expected payoff (the lower the value, the better the performance of the patrol route) of our algorithms (TSP-b, Bwalk, SG) and baselines (minKC, maxEN) with different settings and attacker models. Figure 2, 5, 6 are constant utility functions under Full visibility, Local visibility, and No visibility. Figure 3, 4 are linear and quadratic utility functions under Full visibility. Figure 7 is the simulation on Denver Crime Dataset with full visibility.

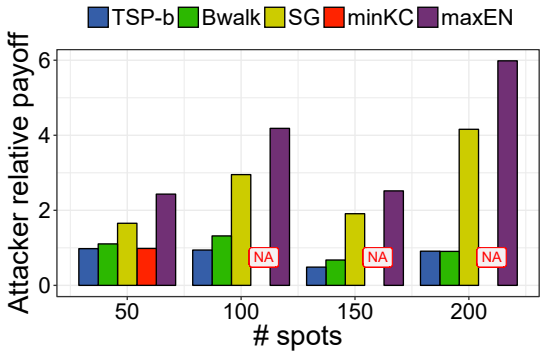

Figure 8: The attacker relative payoff when the number of spots increased. TSP-b and Bwalk show better and stable performance in high scale scenario.

linear vertex weight and they are not suitable for non-constant utility functions. On the other hand, our algorithms focus on the two objectives: EMR and entropy rate, which are not limited to the degree of utility functions.

6.3 Scalability

Figure 8 reports the scalability of the solutions. The settings are full visibility attacker model, constant utility functions, and 0 penalty value for demonstration. To compare the performance under different setups, all attacker's expected payoff is divided by the payoff of BGT patrol route. When the number of spots is more than 100, the solution of minKC is incalculable in our machine due to memory constraint (the solution minKC is calculated by CXYOPT in a laptop of Intel i7-4700 MQ 240GHz CPU with 32 GB ram).

For other solutions, the 3 proposed algorithms have better performance than maxEn. Comparing within the proposed algorithms, SG has the worst performance. One reason is that schedules generated by SG have higher randomness. When the number of spots increasing which make the topology becomes complicated, it requires a schedule with more delicate designed routes. Thus, TSP-b and Bwalk perform better since their schedules are perturbed from TSP/BGT tours.

7 CONCLUSION

We look into a general patrolling game that the attacker can also choose the attack period. Instead of formulating it as a mixed-integer linear programming problem and searching for combinatorial defend strategies which are exponential growth, we focus on two objectives, minimizing the maximum reward and the entropy rate. Based on that, we formulate the Randomized TSP problem and propose three algorithms to achieve the tradeoff between the two criteria. We also design a framework that uses the proposed algorithms to solve patrol security games efficiently. Experiments show that our work is scalable and adaptable to various utility functions and penalties.

Acknowledgements: J. Gao, K. Liu, and H. Yang would like to acknowledge support by NSF DMS-1737812, CNS-1618391, and CCF-1535900. S. Lin would like to acknowledge support by NSF CNS-1553273. The authors would like to acknowledge sociologist Prof. Yue Zhuo for helpful discussions on criminology literatures.

REFERENCES

[1] Noa Agmon, Sarit Kraus, and Gal A. Kaminka. 2011. Multi-Robot Adversarial Patrolling: Facing a Full-Knowledge Opponent. 42 (December 2011), 887–916.

[2] Noa Agmon, Vladimir Sadov, Gal A Kaminka, and Sarit Kraus. 2008. The impact of adversarial knowledge on adversarial planning in perimeter patrol. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 55–62.

[3] Soroush Alamdari, Elaheh Fata, and Stephen L Smith. 2014. Persistent monitoring in discrete environments: Minimizing the maximum weighted latency between observations. *The International Journal of Robotics Research* 33, 1 (2014), 138–154.

[4] Bo An, Eric Shieh, Milind Tambe, Rong Yang, Craig Baldwin, Joseph DiRenzo, Ben Maule, and Garrett Meyer. 2012. PROTECT–A Deployed Game Theoretic System for Strategic Security Allocation for the United States Coast Guard. *Ai Magazine* 33, 4 (2012), 96.

[5] RD Angel, WL Caudle, R Noonan, and ANDA Whinston. 1972. Computer-assisted school bus scheduling. *Management Science* 18, 6 (1972), B–279.

[6] Sanjeev Arora. 1996. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*. IEEE, 2–11.

[7] Sanjeev Arora. 1998. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *Journal of the ACM (JACM)* 45, 5 (1998), 753–782.

[8] Ahmad Bilal Asghar and Stephen L Smith. 2016. Stochastic patrolling in adversarial settings. In *American Control Conference (ACC), 2016*. IEEE, 6435–6440.

[9] Giorgio Ausiello, Stefano Leonardi, and Alberto Marchetti-Spaccamela. [n. d.]. On Salesmen, Repairmen, Spiders, and Other Traveling Agents. In *Algorithms and Complexity*, Giancarlo Bongiovanni, Rossella Petreschi, and Giorgio Gambosi (Eds.). Springer Berlin Heidelberg, 1–16. https://doi.org/10.1007/3-540-46521-9_1

[10] Baruch Awerbuch, Yossi Azar, Avrim Blum, and Santosh Vempala. 1995. Improved Approximation Guarantees for Minimum-Weight k-Trees and Prize-Collecting Salesmen. In *SIAM JOURNAL ON COMPUTING*. 277–283.

[11] Nicola Basilico, Giuseppe De Nittis, and Nicola Gatti. 2017. Adversarial patrolling with spatially uncertain alarm signals. *Artificial Intelligence* 246 (2017), 220–257.

[12] Nicola Basilico, Nicola Gatti, and Francesco Amigoni. 2009. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 57–64.

[13] Nicola Basilico, Nicola Gatti, and Francesco Amigoni. 2009. Leader-follower Strategies for Robotic Patrolling in Environments with Arbitrary Topologies. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 1 (AAMAS '09)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 57–64.

[14] Nicola Basilico, Nicola Gatti, and Francesco Amigoni. 2012. Patrolling security games: Definition and algorithms for solving large instances with single patroller and single intruder. *Artificial Intelligence* 184 (2012), 78–123.

[15] Daniel Bienstock, Michel X. Goemans, David Simchi-Levi, and David Williamson. 1993. A Note on the Prize Collecting Traveling Salesman Problem. *Math. Program.* 59, 3 (May 1993), 413–420. https://doi.org/10.1007/BF01581256

[16] Branislav Bošanskỳ, Viliam Lisỳ, Michal Jakob, and Michal Pěchouček. 2011. Computing time-dependent policies for patrolling games with mobile targets. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*. International Foundation for Autonomous Agents and Multiagent Systems, 989–996.

[17] Jane Breen and Steve Kirkland. 2017. Minimising the largest mean first passage time of a Markov chain: The influence of directed graphs. *Linear Algebra Appl.* 520 (2017), 306–334.

[18] Giorgio Cannata and Antonio Sgorbissa. 2011. A minimalist algorithm for multirobot continuous coverage. *IEEE Transactions on Robotics* 27, 2 (2011), 297–312.

[19] Arthur E Carter and Cliff T Ragsdale. 2002. Scheduling pre-printed newspaper advertising inserts using genetic algorithms. *Omega* 30, 6 (2002), 415–421.

[20] Nicos Christofides. 1976. *Worst-case analysis of a new heuristic for the travelling salesman problem.* Technical Report. Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group.

[21] City and County of Denver. 2016. Denver Open Data Catalog. *City and County of Denver* (2016).

[22] Xiaoming Duan, Mishel George, and Francesco Bullo. 2018. Markov Chains with Maximum Return Time Entropy for Robotic Surveillance. *arXiv preprint arXiv:1803.07705* (2018).

[23] Yehuda Elmaliach, Asaf Shiloni, and Gal A. Kaminka. 2008. A Realistic Model of Frequency-based Multi-robot Polyline Patrolling. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1 (AAMAS '08)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 63–70.

[24] Fei Fang, Albert Xin Jiang, and Milind Tambe. 2013. Optimal patrol strategy for protecting moving targets with multiple mobile resources. In *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 957–964.

[25] Nicola Gatti. 2008. Game Theoretical Insights in Strategic Patrolling: Model and Algorithm in Normal-Form.. In *ECAI*. 403–407.

[26] Mishel George, Saber Jafarpour, and Francesco Bullo. 2018. Markov chains with maximum entropy for robotic surveillance. *IEEE Trans. Automat. Control* (2018).

[27] Samuel Gorenstein. 1970. Printing press scheduling for multi-edition periodicals. *Management Science* 16, 6 (1970), B–373.

[28] Jeremy Grace and John Baillieul. 2005. Stochastic strategies for autonomous robotic surveillance. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*. IEEE, 2200–2205.

[29] Martin Grötschel, Michael Jünger, and Gerhard Reinelt. 1991. Optimal control of plotting and drilling machines: a case study. *Mathematical Methods of Operations Research* 35, 1 (1991), 61–84.

[30] L. Iocchi, L. Marchetti, and D. Nardi. 2011. Multi-robot patrolling with coordinated behaviours in realistic environments. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2796–2801. https://doi.org/10.1109/IROS.2011.6094844

[31] Manish Jain, Erim Kardes, Christopher Kiekintveld, Fernando Ordónez, and Milind Tambe. 2010. Security Games with Arbitrary Schedules: A Branch and Price Approach.. In *AAAI*.

[32] Manish Jain, Dmytro Korzhyk, Ondřej Vaněk, Vincent Conitzer, Michal Pěchouček, and Milind Tambe. 2011. A double oracle algorithm for zero-sum security games on graphs. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 327–334.

[33] Edwin T Jaynes. 1957. Information theory and statistical mechanics. *Physical review* 106, 4 (1957), 620.

[34] John G Kemeny and J Laurie Snell. 1960. Finite Markov Chains. D Van Nostad Co. *Inc., Princeton, NJ* (1960).

[35] John G Kemeny and J Laurie Snell. 1983. *Finite Markov chains: with a new appendix" Generalization of a fundamental matrix"*. Springer.

[36] Christopher Kiekintveld, Manish Jain, Jason Tsai, James Pita, Fernando Ordóñez, and Milind Tambe. 2009. Computing optimal randomized resource allocations for massive security games. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 689–696.

[37] Kap Hwan Kim and Young-Man Park. 2004. A crane scheduling method for port container terminals. *European Journal of operational research* 156, 3 (2004), 752–768.

[38] Steve Kirkland. 2010. Fastest expected time to mixing for a Markov chain on a directed graph. *Linear Algebra Appl.* 433, 11-12 (2010), 1988–1996.

[39] Gilbert Laporte. [n. d.]. The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research* 3 ([n. d.]), 345 – 358. https://doi.org/10.1016/0377-2217(92)90192-C

[40] Kin Sum Liu, Tyler Mayer, Hao Tsung Yang, Esther Arkin, Jie Gao, Mayank Goswami, Matthew P JohnsonS, Nirman KumarP, and Shan Lin. 2017. Joint Sensing Duty Cycle Scheduling for Heterogeneous Coverage Guarantee. In *INFOCOM 2017-IEEE Conference on Computer Communications, IEEE*. IEEE, 1–9.

[41] Jie Min and Tomasz Radzik. 2017. Bamboo Garden Trimming Problem. In *SOFSEM 2017: Theory and Practice of Computer Science: 43rd International Conference on Current Trends in Theory and Practice of Computer Science, Limerick, Ireland, January 16-20, 2017, Proceedings*, Vol. 10139. Springer, 229.

[42] Joseph SB Mitchell. 1999. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. *SIAM Journal on computing* 28, 4 (1999), 1298–1309.

[43] Mohamed Mosbah and Nasser Saheb. 1999. Non-uniform random spanning trees on weighted graphs. *Theoretical computer science* 218, 2 (1999), 263–271.

[44] Thanh H. Nguyen, Debarun Kar, Matthew Brown, Arunesh Sinha, Albert Xin Jiang, and Milind Tambe. 2016. Towards a Science of Security Games. In *New Frontiers of Multidisciplinary Research in STEAM-H*, B. Toni (Ed.).

[45] F. Pasqualetti, A. Franchi, and F. Bullo. 2012. On Cooperative Patrolling: Optimal Trajectories, Complexity Analysis, and Approximation Algorithms. *IEEE Transactions on Robotics* 28, 3 (June 2012), 592–606. https://doi.org/10.1109/TRO.2011.2179580

[46] Rushabh Patel, Pushkarini Agharkar, and Francesco Bullo. 2015. Robotic surveillance and Markov chains with minimal weighted Kemeny constant. *IEEE Trans. Automat. Control* 60, 12 (2015), 3156–3167.

[47] Victor Pillac, Michel Gendreau, Christelle Guéret, and Andrés L. Medaglia. [n. d.]. A review of dynamic vehicle routing problems. *European Journal of Operational Research* 1 ([n. d.]), 1 – 11. https://doi.org/10.1016/j.ejor.2012.08.015

[48] James Pita, Manish Jain, Janusz Marecki, Fernando Ordóñez, Christopher Portway, Milind Tambe, Craig Western, Praveen Paruchuri, and Sarit Kraus. 2008. Deployed ARMOR protection: the application of a game theoretic model for security at the Los Angeles International Airport. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems: industrial track*. International Foundation for Autonomous Agents and Multiagent Systems, 125–132.

[49] D. Portugal and R. P. Rocha. 2011. On the performance and scalability of multi-robot patrolling algorithms. In *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*. 50–55. https://doi.org/10.1109/SSRR.2011.6106761

[50] Richard Serfozo. 2009. *Basics of applied stochastic processes*. Springer Science & Business Media.

[51] Eric Shieh, Bo An, Rong Yang, Milind Tambe, Craig Baldwin, Joseph DiRenzo, Ben Maule, and Garrett Meyer. 2012. Protect: A deployed game theoretic system to protect the ports of the united states. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 13–20.

[52] Eric Shieh, Manish Jain, Albert Xin Jiang, and Milind Tambe. 2013. Efficiently solving joint activity based security games. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*. AAAI Press, 346–352.

[53] E. Stump and N. Michael. 2011. Multi-robot persistent surveillance planning as a Vehicle Routing Problem. In *Automation Science and Engineering (CASE), 2011 IEEE Conference on*. 569–575. https://doi.org/10.1109/CASE.2011.6042503

[54] Jason Tsai, Christopher Kiekintveld, Fernando Ordonez, Milind Tambe, and Shyamsunder Rathi. 2009. IRIS-a tool for strategic security allocation in transportation networks. (2009).

[55] Yevgeniy Vorobeychik, Bo An, and Milind Tambe. 2012. Adversarial Patrolling Games. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 3 (AAMAS '12)*. International Foundation for Autonomous Agents and Multiagent Systems, Richland, SC, 1307–1308.

[56] Yevgeniy Vorobeychik, Bo An, Milind Tambe, and Satinder P Singh. 2014. Computing Solutions in Infinite-Horizon Discounted Adversarial Patrolling Games. In *ICAPS*.

[57] Rong Yang, Benjamin Ford, Milind Tambe, and Andrew Lemieux. 2014. Adaptive resource allocation for wildlife protection against illegal poachers. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 453–460.

[58] Wei Yu and Zhaohui Liu. 2014. Vehicle routing problems with regular objective functions on a path. *Naval Research Logistics (NRL)* 61, 1 (2014), 34–43. https://doi.org/10.1002/nav.21564