International Journal of Computer Applications (0975 - 8887)
Volume 156 - No.2, December 2016

Decentralized and Nondeterministic Multi-Robot Area Patrolling in Adversarial Environments

Tauhidul Alam
School of Computing and Information Sciences
Florida International University
Miami, FL 33199, USA

ABSTRACT

Multi-robot patrolling is the problem of repeatedly visiting a group of regions of interest in an environment with a group of robots to prevent intrusion. Early works have proposed deterministic patrolling algorithms which could be learned by an adversary observing them over time. More recent works provide non-deterministic patrolling schemes which only work for perimeter patrolling and require coordination and synchronization. In this paper, we investigate the problem of finding robust and scalable strategies for multi-robot patrolling under an adversarial environment. So, we present algorithms to find different decentralized strategies for a patroller in the form of Markov chains which use convex optimization to minimize the average commute time for an environment, a subset of the environment, or a specific region of an environment when we use uniform distribution over all regions for both patroller and adversary. Additionally, we use these strategies in a game theoretical setup to form a payoff matrix to obtain an optimal mixed strategy for patroller. We also propose an algorithm to find a decentralized strategy for patroller in the form of Markov chain which converges very fast as it is the optimal response from patroller against the adversary when we use non-uniform distribution over all the regions for both patroller and adversary. Our results show the scalability and applicability of our approach in different types of environments. Despite the lack of synchronization and coordination among patrollers, our approach performs competitively compared to existing methods.

General Terms

Multi-robot patrolling

Keywords

Patroller, Adversary, Markov chain, Game theory, Convex optimization

1. INTRODUCTION

Patrolling is the problem of repeatedly visiting a group of regions in an environment. This problem has applications in different areas such as environmental monitoring, infrastructure surveillance, and border security. In its multi-agent version, the action of patrolling is carried out by multiple robots as patrollers working together to ensure the safety of the area under surveillance. In its adversarial setup, one or more adversaries always try to penetrate the environment being patrolled by observing patroller?s strategies and patrollers do not know where adversaries are going to attack. These make the problem difficult and interesting to address. Patrolling schemes can be further categorized into perimeter patrolling and area patrolling, where perimeter patrolling is a subset of area patrolling. The focus of this work is area patrolling.

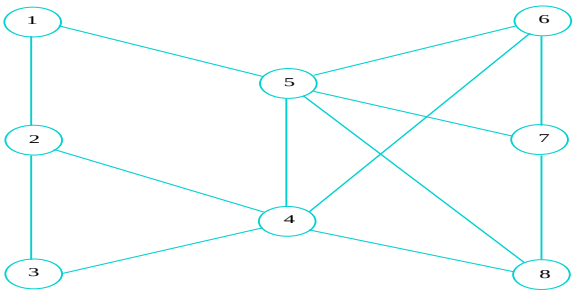


Fig. 1: An example patrolling scenario with eight different regions and connections among the regions

An example of a patrolling scenario has been demonstrated in Figure 1. The patrolling environment is given as a undirected graph where patrollers will have to patrol the different regions (vertices) across the connections (edges) among the regions. In this work, we consider both uniform and non-uniform distribution over all the regions (vertices) for both patroller and adversary. This problem is challenging to handle because if we use deterministic approaches based on the frequencies of visits to different regions of the environment, then it is easy for adversaries to penetrate the environment. Again, if we use multiple patrollers and we consider the synchronous and coordinated movement of those patrollers then these types of patrolling strategies are expensive in terms of computation. However, with a non-deterministic strategy, patrolling robots would move randomly around the graph, making it much more difficult for the adversary to effectively choose where to penetrate in the environment. Moreover, randomized strategies do not need the synchronization, communication or coordination among the patrollers following these strategies.

Taking the inspiration from the presence of adversaries and the need for unpredictability, Agmon et al. present perimeter patrolling strategies based on Markov chain models that maximize the probability of detecting an adversary [1, 2, 3]. In this stream of research,

only cyclic graphs were considered, and the robots have to be synchronized and coordinated, which may prevent practical deployments of these strategies. Furthermore, all robots have to be placed in known locations and with equal distance apart in the environment. Sak et al. [4] consider an empirical non-deterministic approach for general graphs rather than a perimeter. However, only experimental results were presented without formalization or algorithmic details.

The problem of patrolling in the presence of adversaries can also be formulated as a *Bayesian Stackelberg Game*. However, this problem of choosing an optimal strategy for the patroller to commit to in a *Bayesian Stackelberg Game* is NP-hard [5]. So, we have applied an efficient method for finding optimal mixed strategies for patroller in such games. In this game theoretical setting, the patroller first commits to an optimal mixed strategy generated through DOBSS (Decomposed Optimal Bayesian Stackelberg Solver) [6] where each strategy is a path in a fully connected graph. Once the patroller has decided its mixed strategy, the adversary uses the knowledge of the mixed strategy to choose a region to attack. One of the potential drawbacks of this approach is that the set of strategies (paths) should be chosen *a priori* from a large set of possible paths and the particular connectivity of the graph is not studied in detailed.

Our work is inspired by the work of minimizing the effective resistance of a graph through convex optimization. This work finds its original motivation in electrical networks where the solution is a measure of how well "connected" the network is [7]. The concept of effective resistance can be applied to Markov chains where minimizing the resistance between vertices on a graph corresponds to minimizing the commute times between vertices. With this application, robots patrolling on a graph with Markov chains would be able to quickly travel between every pair of vertices when we consider a uniform distribution of cost over all vertices. On the other hand, for non-uniform distribution of cost over all vertices, this work corresponds to the problem of finding fastest mixing Markov chain for any graph through convex optimization. This problem finds the probabilities to the edges of the graph in such a way that it minimizes the second largest eigenvalue modulus (SLEM) [8]. The problem of finding fastest mixing Markov chain is equivalent to the minimization of SLEM because the rate of convergence to the normal distribution of Markov chain is the mixing rate of the Markov chain. In the work of Boyd [9], he addresses the problem of choosing the edge weights of an undirected graph so as to maximize or minimize some function of eigenvalues of the associated Laplacian matrix subject to some constraints on the weights, such as nonnegativity or a given total value.

The purpose of our work is fourfold. First, we would like to extend current ideas into a more general class of graphs. The previous perimeter ideas can be extended to general graphs but this will first need finding a Hamiltonian cycle which is an NP-complete problem. Second, we would like to remove the need for communication, synchronization, and known initial placement of the randomized patrolling strategies. This will allow patrolling algorithms to be implemented with simple robots, in a decentralized fashion, and in communication denied environments. Third, we would like to propose patrolling algorithms for the case of both same or different preferences over the regions in the environment which lead us to consider either uniform distribution of over regions in the environment for both patroller and adversary. Fourth, we would like to adapt the game theoretical setup [6] to use Markov chains instead of deterministic strategies and to be applied to different sets of graphs.

Randomized strategies based on Markov chains are used in our work for several reasons: 1) These will make it harder for an adversary to successfully complete an attack due to the unpredictability of the strategies; 2) A randomized motion can be easily implemented in a mobile robot, since its communication, sensing, and computation requirements are minimal; 3) Efficient algorithms can calculate Markov chains with desired properties [7].

The contributions of the paper are as follows:

—We present algorithms that do not require communication, are based on convex optimization, can scale well, and can also be applied to any type of environment represented as a graph where distribution of cost over regions (vertices) is both uniform and non-uniform for patroller and adversary.

—We present a game-theoretical approach to patrolling in uniform cost distribution case where the set of strategies are Markov chains. We also calculate the payoffs of each strategy and present approaches to generate the optimal mixed strategy for patroller and the optimal strategy for the adversary.

The remainder of the paper is organized as follows: Section II presents the problem formulation; Section III introduces algorithms to find decentralized and distributed strategies for multi-robot patrolling in the cases of uniform and non-uniform distribution and a game theoretical approach for finding optimal mixed strategies in uniform distribution case; Section IV presents experimental results of our approach; Section V discusses our conclusions and future directions for research.

Parts of this work appeared in preliminary form in [10]. This version presents new theoretical and experimental results.

2. RELATED WORKS

Multi-robot patrolling has been investigated in many different studies. Initial research [11, 12, 13, 14] proposed deterministic approaches based on the optimization of the frequency of visits to the regions in the environment. Portugal et al. presented a multilevel partitioning algorithm (MSP) that assigns different regions to each patroller agent [15]. The performance of this approach is slightly better, but still deterministic. A survey of multi-agent patrolling strategies can be found in [16]. where strategies are evaluated based on robot perception, communication, coordination, and decision-making capabilities. A basic approach to creating a patrolling strategy is a deterministic approach which generates patrol paths in the environment for multiple robots as patrollers. However, an adversary can easily penetrate the perimeter or area if a deterministic patrolling is used. For example, if a patrolling strategy ensures that a region around a perimeter is visited every 20 seconds and it takes an adversary 15 seconds to break in, then the adversary is guaranteed success if it attacks just after the region is visited [1]. Nevertheless, with a non-deterministic strategy, patrolling robots would move more randomly around the graph, making it much more difficult for the adversary to effectively choose where to penetrate the perimeter environment.

In more recent works, Nicola et al. proposed the optimal strategy for patroller for patrolling the arbitrary environment like a set of connected cells, using single patroller or the smallest no. of patrollers required and based on several coordination dimensions among the robots [17, 18, 19]. They have considered different penetration time for different cells of the environment and different target preferences among those cells for an intruder. They have formulated bilinear mathematical programming problem to find the optimal patroller's strategy. They have used Markovian

property that maximizes the expected intruder-capture utility for patrollers. Nicola and Carpin again proposed probabilistic intrusions and a variable resolution sensing model that naturally applies to the domain of UAVs [20]. However, they have not used inherent Markov chain property to patrol the environment with optimal mixed strategy or fastest mixing strategy which is the best response from patroller against an intruder. Vorobaychik et al. presented a general model of infinite-horizon discounted adversarial patrolling games [21]. Here they assume payoffs for target regions are static over time.

3. PROBLEM FORMULATION

We define the patrolling environment as an *undirected graph*, $G = (V, E)$ with $|V| = n$, $|E| = m$. Each vertex ($v \in V$) corresponds to a region, and each edge ($e \in E$) corresponds to a connection between two regions in the environment. Let M be a discrete time Markov chain on the graph with M_{ij}, the probability of transitioning between vertex i and vertex j.
The worst case scenario for patrollers is when they do not know where an adversary is going to attack and they are not certain about their initial positions in the graph. Also, we consider a uniform distribution of cost over all vertices in the graph for both patroller and adversary. In this case, it is better to find a strategy for patrollers that minimizes the average commute or hitting time between every pair of vertices. The *hitting time*, H_{ij}, is the (random) time taken to reach vertex j starting from vertex i using the Markov chain. The *commute time*, C_{ij}, is the time it takes to travel from i to j and back using the Markov chain, $C_{ij} = H_{ij} + H_{ji}$. The average hitting time, $\bar{H}$, and average commute time, $\bar{C}$, for a Markov chain in a graph are the times averaged over all pairs of vertices. In this context, the patrolling problem will be defined as:

Problem 1: Finding decentralized patrolling strategies in case of uniform distribution over all vertices
Given the graph, $G = (V, E)$, find distributed strategies that minimize average commute time ($\bar{C}$) or average hitting time ($\bar{H}$): 1) for all vertices in V; 2) for a clique $v_1 ... v_d \in V$, $d <= n$; and 3) to a particular vertex $v \in V$
We also explicitly model an adversary who (1) knows all possible strategies that a patroller can choose, (2) has full knowledge of the environment, and (3) is able to optimally choose the vertex to attack in graph G. This scenario is known as a *Bayesian Stackelberg Game* where the two players of the game are the patroller and the adversary. In our proposed game theoretical setup, each patroller's strategy is a Markov Chain. The adversary also has a set of strategies to attack in any of the n vertices. The game is formulated as follows:

—A nonempty, finite set called the set of patrolling strategies $\mathcal{M} = \{M_1, ..., M_k\}$ where k is the number of Markov chains.
—A nonempty, finite set called the set of adversary strategies V. Each $v \in V$ is a vertex in the graph.
—A function $P : \mathcal{M} \times V \longrightarrow \mathbb{R}^+ \cup \{\infty\}$ called the payoff matrix for the patroller.
—A function $Q : \mathcal{M} \times V \longrightarrow \mathbb{R}^+ \cup \{\infty\}$ called the payoff matrix for the adversary.

In order to calculate both payoff matrices, we need the following values:

—d_i^{pat} : cost in region i to the patroller.
—d_i^{adv} : cost in region i to the adversary.
—c_i^{pat} : reward to the patroller of catching the adversary in the i-th region.
—c_i^{adv} : cost to the adversary of getting caught in the i-th region.
—p_i : the probability that the patroller will catch the adversary at the i-th region of the environment.

A patroller's reward must consider the factor c_i^{pat} for capturing the adversary (with probability p_i) and the reward value gets reduced when the adversary is not captured (probability $1-p_i$). Conversely, the adversary pays cost c_i^{adv} (with probability p_i) but gains c_i^{adv} (with probability $1 - p_i$). Additionally, p_i also depends on the i-th hitting time of the Markov chain for each patrolling strategy. Given these definitions, we are interested in the following problem:

Problem 2: Generating the optimal strategies in case of uniform distribution over all vertices
Given the payoff matrices, P and Q, the set of patroller strategies, $\mathcal{M}$, the set of strategies for adversary, V, find the optimal mixed strategy for the patroller and the optimal strategy for the adversary.
Again we consider non-uniform distribution of cost over all vertices in the graph. The distribution of cost at vertex i is $\pi_i = \frac{d_i^{pat}}{\sum_{i=1}^{n} d_i^{pat}}$.
We also define the non-uniform equilibrium distribution of Markov chain M, is π = (π_1, ...,π_n). Here we consider the best response for patrollers when an adversary uses the non-uniform equilibrium distribution of Markov chain and they are not certain about their initial positions in the graph. In this case, it is better to find a strategy for patrollers that converges Markov chain very fast using the non-uniform distribution of the cost of vertices in the graph. In this context, the patrolling problem will be defined as:

Problem 3: Finding decentralized patrolling strategies in case of uniform distribution over all vertices
Given the graph, $G = (V, E)$, find fastest mixing or convergence Markov chain (M) that converges very fast for the graph G with non-uniform distribution,π over all vertices in G.

4. METHOD

In this section, we present algorithms to generate patrolling strategy which is an optimal response for a patroller against an adversary when patroller and adversary use both uniform and non-uniform distribution over all regions in the environment.

4.1 Decentralized Patrolling Strategy for Uniform Distribution Case

As mentioned before, we consider patrolling strategies in a graph as Markov chains $\mathcal{M}$. Gosh et al. [7] define the effective resistance between two vertices i and j in a graph as R_{ij}. They also define the nonnegative conductance on edge l as g_l which is a weight that can be assigned to an edge of a graph. R_{ij} is small when there are many paths between vertices i and j with high edge weights and is large when there are fewer paths between vertices i and j with low edge weights. In [7], the *total effective resistance*, R_{tot} is the sum of the effective resistances between all pairs of vertices,

$$R_{tot} = \frac{1}{2} \sum_{i,j=1}^{n} R_{ij} = \sum_{i<j} R_{ij} \quad (1)$$

where R_{tot} is related to the average commute time ($\bar{C}$) of the Markov chain.
In this work, an environment with small total effective resistance corresponds to a Markov chain with small hitting or commute times between vertices, and a large total effective resistance corresponds

to a Markov chain with large hitting or commute times between at least some pairs of vertices. In [7], a convex optimization method is proposed for minimizing the total effective resistance of the graph by allocating a fixed total conductance among the edges:

$$\begin{aligned} &\text{minimize} && R_{tot} \\ &\text{subject to} && \mathbf{1}^T g = 1, \quad g \geq 0 \end{aligned} \tag{2}$$

The optimization variable is $g \in R^m$, the vector of edge conductances. In our patrolling strategies, the total *effective resistance minimization problem* (ERMP) is equivalent to the problem of selecting weights on edges to minimize the commute (or hitting) time between vertices. Once R_{ij} is considered as distances among the vertices, the ERMP is the problem of allocating the edge weights to a graph to make the graph small in terms of average distance between vertices.
The relationship between commute time, C_{ij}, and effective resistance, R_{ij}, between vertex i and j is the following [7]:

$$C_{ij} = (\mathbf{1}^T g) R_{ij}$$

Using this relationship, we present our first patrolling strategy of *minimization of average commute time* (MACT) on a graph following equation 2.
In the second patrolling strategy, we have also extended the MACT problem for a subset of vertices or clique. This extension ensures a higher probability of traveling along the edges within the subset of vertices while still allowing for travel to the remaining vertices of the graph. For example, the edges $e = \{shortestPath(i,j),\ shortestPath(j,k),\ shortestPath(k,i)\}$, where $e \in E$, are given priority. This means that the edges along this shortest cycle will be optimized such that the edge weights will consist of the majority of the sum of all edge weights. We have extended the ERMP as follows:

$$\begin{aligned} &\text{minimize} && R_{tot} \\ &\text{subject to} && \mathbf{1}^T g = 1, && g \geq 0 \\ & && e = \text{shortestCycle}(s), \\ & && \sum_e w(e) \geq t \end{aligned} \tag{3}$$

In this problem, the variable e represents an edge along the shortest cycle between the vertices in the subset, s, of V. The condition $\sum_e w(e) \geq t$ ensures that the sum of the edge weights of $w(e) \in g$ will be greater than or equal to a threshold that represents the percentage of edge weights allocated to the cycle.
Let A be an $n \times m$ incidence matrix of a graph $G = (V, E)$ where $n = |V|$ and $m = |E|$. Suppose edge l connects vertices i and j. We define $A_{il} = 1$, $A_{jl} = -1$, and all other elements 0. Let $b : E \rightarrow \mathbb{R}^+$ denotes the distance associated with each edge in graph G. We present an algorithm for MACT towards each vertex.
Algorithm 1 generates a set of n Markov chains, $\mathcal{M}$, that minimizes the commute time towards every vertex $v \in V$ for the graph G. For every Markov chain, the k shortest paths from v_j to the target vertex v_i are found (line 11). The k shortest paths are found using Yen's algorithm [22], which has a worst-case runtime of $O(n^2)$. After the k shortest paths are found, every edge transition found within the k paths is incremented in the Markov chain that is being generated (lines 12-14). When $v_j = v_i$, all possible edge transitions from the target vertex v_j to adjacent vertices are incremented (lines 5-8). Once all paths have been explored for a Markov chain, M is normalized and is added to the set of Markov chains $\mathcal{M}$.

Algorithm 1 MACTtowardsVertices(A, b)

Input: $A, b(.)$ {Incidence and edge costs of graph}
Output: $\mathcal{M} = \{M_1, .., M_n\}$ {A set of Markov chains}
1: $\mathcal{M} \leftarrow 0$
2: **for** $i = 1$ to n **do**
3: $M(n \times n) \leftarrow 0$
4: **for** $j = 1$ to n **do**
5: **if** $j == i$ **then**
6: **for** all edges $e_{jb} \in A$ **do**
7: $M_{jb} \leftarrow M_{jb} + 1$
8: **end for**
9: continue
10: **end if**
11: $k \leftarrow$ KSHORTESTPATHS$(b(.), j, i)$
12: **for** all edges $e_{ab} \in k$ **do**
13: $M_{ab} \leftarrow M_{ab} + 1$
14: **end for**
15: **end for**
16: normalize M
17: $\mathcal{M} \leftarrow \mathcal{M} \cup M$
18: **end for**
19: return $\mathcal{M}$

4.2 Game Theoretical Optimal Strategies for Uniform Distribution Case

The game theoretical approach has two stages; the first stage is payoff matrices calculation and the second stage is optimal strategies generation.

4.2.1 Payoff Matrices Calculation. The payoff matrices for patroller and adversary, P and Q, are calculated from the set of Markov chains, $\mathcal{M}$. We calculate n Markov chains from Algorithm 1 and we also have two other Markov chains from MACT for all vertices and MACT for a subset of vertices or clique. Now we have $n + 2$ Markov chains in the set of Markov chains, $\mathcal{M}$.
In Algorithm 2, we present a procedure to calculate the payoff matrices. As this is the uniform cost distribution case, the algorithm assigns uniform costs (d_i^{pat}, d_i^{adv}) in each vertex for both the patroller and the adversary. It calculates the hitting time matrix, H, (line 5) for each Markov chain. The mean hitting time vector (lines 7-9), mht, is calculated using the function HTOFPREFERRED-VERTICES. This function takes the vertices involved in the Markov chain optimization as follows: 1) In the case of MACT strategy, it calculates the mean of the hitting times for all vertices; 2) In the case of MACT for a subset of vertices or clique, it takes the mean hitting time for the subset of vertices; 3) In the case of MACT toward each vertex, it takes the hitting time towards that particular vertex. In lines 13-21, it assigns the values to the variables needed to calculate the payoff matrices. In lines 22-25, it calculates the payoff matrices using these variables.

4.2.2 Optimal Strategies Generation. The probability distribution of choosing the strategies for the patroller can be represented as an m-dimensional vector,

$$w = (w_1, w_2, ..., w_m) \tag{4}$$

Equation 4 should satisfy: 1) $w_i \geq 0$ for all $i \in 1...m$, and 2) $w_1 + w_2 + ... + w_m = 1$. The value w_i is the proportion of the patroller choosing strategy w_i.

Algorithm 2 PayoffMatrixCalculation($\mathcal{M}$)

Input: $\mathcal{M} = \{M_1, .., M_{n+2}\}$ {Markov chains for all patrolling strategies}
Output: P, Q {Payoff matrix for patroller and adversary}

for $i = 1$ to n **do**
  $d_i^{pat} \leftarrow 1/n \qquad d_i^{adv} \leftarrow 1/n$
end for
for $i = 1$ to $|\mathcal{M}|$ **do**
  $H \leftarrow$ HITTINGTIMES(M_i)
  $mht \leftarrow 0$
  for $j = 1$ to n **do**
    $mht_j \leftarrow$ HTOFPREFERREDVERTICES(H)
  end for
  $asc \leftarrow$ SORTASCENDING(mht)
  $desc \leftarrow$ SORTDESCENDING(mht)
  $c^{pat} \leftarrow 1 \qquad c^{adv} \leftarrow 1 \qquad hh \leftarrow n$
  for $k = 1$ to n **do**
    $c^{pat}_{asc_k} \leftarrow c^{pat}_{asc_k} hh$
    $c^{adv}_{desc_k} \leftarrow c^{adv}_{desc_k} hh$
    $hh \leftarrow hh - 1$
  end for
  $p \leftarrow 1$
  for $k = 2$ to n **do**
    $p_k \leftarrow 1/2^{k-2}$
  end for
  for $k = 1$ to n **do**
    $P_{ik} \leftarrow p_k c_k^{pat} + (1 - p_k) d_k^{pat}$
    $Q_{ik} \leftarrow p_k c_k^{adv} + (1 - p_k) d_k^{adv}$
  end for
end for
return P, Q

Similarly, z represents all the possible strategies for the adversary as an n-dimensional vector,

$$z = (z_1, z_2, ..., z_n) \tag{5}$$

Equation 5 should satisfy: 1) $z_i \in \{0, 1\}$ for all $i \in 1...n$, and 2) $z_1 + z_2 + ... + z_n = 1$. Since the adversary can attack any region $n \in |V|$ in the environment, the adversary's strategies, z_i, are strategies for attacking each region separately, and only one strategy can be chosen.
When we have few heterogeneous patrolling strategies preferencing some part of the graph or subset of vertices of the graph then we can also recover the homogeneous patrolling strategy from these different types of patrolling strategies.

PROPOSITION 1. *There exists an optimal homogeneous patrolling strategy for some heterogeneous patrolling strategies.*

PROOF. Assume there exist some heterogeneous patrolling strategies modeled as k different Markov chains $M_1, ..., M_k$, then we can construct from these strategies a single homogeneous Markov chain M_h which will be executed by the patrollers, is no worse. Suppose we have two heterogeneous patrolling strategies as Markov chain M_1 and M_2 with more vulnerabilities in left and right side of dumbbell graph respectively. These two strategies have a high probability of traveling between edges of vertices in left and right part of dumbbell graph. However, the homogeneous patrolling strategy is the first of our patrolling strategies which uniformly travels every vertex of the dumbbell graph optimally. □

4.3 Decentralized Patrolling Strategy for Non-Uniform Distribution Case

Again we define edge weights of the graph G as $(w_1, ...w_m) \in \mathbb{R}^+$ and edge $l \sim (i, j)$ is defined as l connects vertices i,j. We define incidence matrix A as:

$$A_{il} = \begin{cases} 1 & \text{edge } l \text{ connects vertex } i \\ -1 & \text{edge } l \text{ connects vertex } j \\ 0 & \text{otherwise} \end{cases}$$

From the incidence matrix A, we define weighted laplacian matrix, $L = Adiag(w)A^T$ which is in turn defined as

$$L_{ij} = \begin{cases} -w_l & l \sim (i, j) \\ w_l & i = j \\ 0 & \text{otherwise} \end{cases}$$

From the definition of Markov chain, Markov chain, M on vertex of G, with transition probabilities on edges $P_{ij} = \text{Prob}\ (X(t+1) = j \mid X(t) = i)$. Here we focus on symmetric transition probability matrices P (everything extends to a reversible case, with non-uniform equilibrium distribution) and Identifying P_{ij} with w_l for $l \sim (i, j)$, we have $P = I - L$ where I is identity matrix and L is weighted laplacian matrix. Again the non-uniform equilibrium distribution of Markov chain, M, is π = $(\pi_1, ...\pi_n)$ and let Π = $diag(\pi)$. So, the balanced condition for transition probability matrix P of Markov chain M for non-uniform distribution π is $\pi_i P_{ij} = \pi_j P_{ji}$ where $i, j = 1, ...n$. Since $\Pi P = P^T \Pi$ and **P1=1** which means $\Pi^{\frac{1}{2}} P \Pi^{-\frac{1}{2}}$ matrix is a symmetric and same eigenvalue as P. The eigenvector of $\Pi^{\frac{1}{2}} P \Pi^{-\frac{1}{2}}$ is $q = (\sqrt{\pi_1}, ...\sqrt{\pi_n})$. The second largest eigenvalue modulus (SLEM) [8] of M is $\mu(P) = \| \Pi^{\frac{1}{2}} P \Pi^{-\frac{1}{2}} - qq^T \|$. Also, the associated mixing time is $\tau = \frac{1}{\log(\frac{1}{\mu})}$.
From the above definition, fastest mixing reversible Markov chain (FMRMC) is formulated as:

$$\begin{aligned} \text{minimize} \quad & \mu = \| \Pi^{\frac{1}{2}} (I - L) \Pi^{-\frac{1}{2}} - qq^T \| \\ \text{subject to} \quad & w \geq 0, \mathbf{diag}(L) \leq 1 \end{aligned} \tag{6}$$

Here optimization variable is w and problem data is graph G. The two other existing common suboptimal sequences are:
Max-degree Chain: $w = (\frac{1}{\max_i d_i})\mathbf{1}$
Metropolis-Hastings Chain: $w_l = (\frac{1}{\max\{d_i, d_j\}})$ where $l \sim (i, j)$ which comes from Metropolis-Hastings method [23, 24] for generating reversible MC with non-uniform stationary distribution and d_i, d_j are the degree of vertices i, j respectively.

5. EXPERIMENTAL RESULTS

This section illustrates the simulation results of our method for both uniform and nonuniform distributions over all the regions in the environment.

5.1 Decentralized Patrolling Results for Uniform Distribution Case

We have compared our MACT method with the three patrolling methods proposed by Agmon et al. [1]. In our approach, we place patrollers on the graph either randomly or at an equal distance from each other (uniformly) around the graph; Agmon et al.'s three methods, BMP, DCP, DNCP, all require that patrollers are placed an

equal distance from each other on the graph. The patrollers are mobile robots such as UAVs or wheeled robots. In Figure 2, we show the mean first hitting time of MACT with both uniform and random placement of patrollers compared to their three methods on a graph of 64 vertices and 2016 edges, (K_{64} : 2016). Since their methods require a cycle graph to be tested on, we have tested their methods on a cycle graph of 64 vertices, (C_{64}), which is assumed to be equivalent to the Hamiltonian cycle of our original graph, K_{64}. In our test, we have simulated four patrollers so the distance between two consecutive robots, d, is 16. We also consider the number of state transitions it takes to penetrate the environment, t, as 12 units. Though our approach does not use synchronization and communication among robots, it still surpasses the mean hitting time of one of their methods and performance does not degrade much compared to other two.

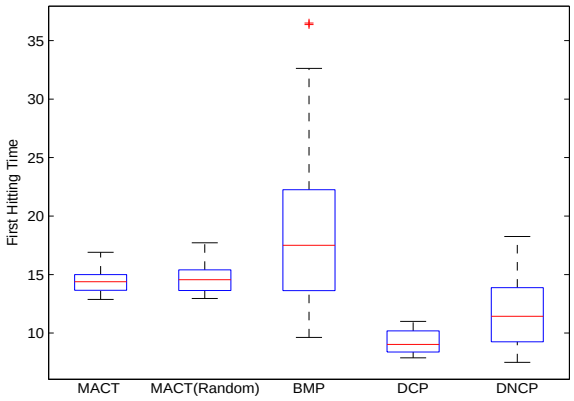


Fig. 2: Comparison result of our MACT method with uniform and random robot placement as well as three existing methods (e.g. BMP, DCP, DNCP) [1] for patrolling. Each line represents the maximum, minimum first hitting time and each box represents the median along with the mean hitting time in the middle.

We compare more closely our MACT method with DCP and DNCP methods using varying number of patrollers on the graphs mentioned above (K_{64} and C_{64}). The result of this comparison is shown in Table 1. It shows that the more the patrollers are present on the graph, the closer the first hitting times of DCP and DNCP's methods approach the data from our approach.

Table 1. Comparison of average first hitting time of our approach, MACT with uniform and random robot placement and two methods (DCP, DNCP) for 30 experiments

No. of Patrollers	Uniform MACT	Random MACT	DCP	DNCP
2 (d=32, t=24)	30.35	30.58	17.29	20.85
4 (d=16, t=12)	14.22	14.70	9.26	11.74
8 (d=8, t=6)	7.11	7.30	5.90	6.90
16 (d=4, t=3)	3.21	3.40	3.1	3.16

We have also tested our methods on different types of graphs, including line, tree, mesh, complete, and randomly generated graphs. Figure 3 shows the edge weight allocation on different graphs [25] for our MACT method. In each graph, the thickness of each edge corresponds to the optimal edge weight value, or the probability of that edge being chosen by a patroller. For example, a wider edge connection between two vertices represents a high probability of that edge being chosen for travel, and vice versa.

5.2 Game Theoretical Optimal Strategies Result for Uniform Distribution Case

We have tested DOBSS with a small graph [25] consisting of eight vertices and thirteen edges. The patroller has ten patrolling strategies available: eight which minimize the average commute time towards each of eight vertices, one that minimizes the average commute time towards a subset of vertices or clique, and one that minimizes the average commute time over all vertices. The resulting graphs for all ten patrolling strategies are shown in Figure 4.
As an illustration, payoff matrices of a small graph are shown in Table 2, where the values of the payoff matrices for patroller and adversary, P and Q, for ten patrolling strategies are calculated using Algorithm 2.

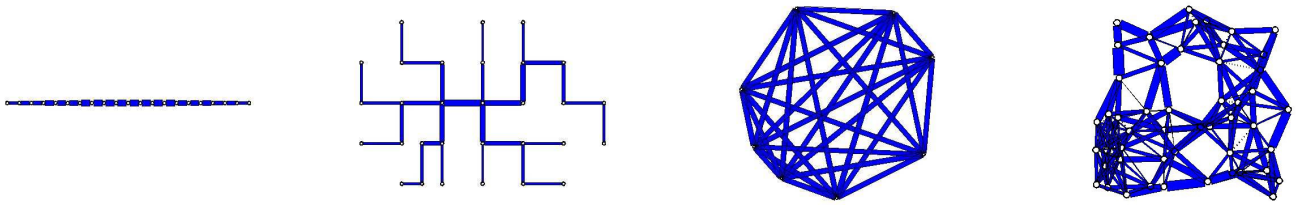
Table 2. Payoff matrices for small graph

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
P_1	8.000	1.594	0.154	0.132	3.563	0.367	0.734	0.215
Q_1	1.000	0.844	0.232	0.187	1.063	0.430	0.609	0.309
P_2	0.215	8.000	1.594	0.734	0.154	0.132	0.367	3.563
Q_2	0.309	1.000	0.844	0.609	0.232	0.187	0.430	1.063
P_3	0.215	1.594	8.000	3.563	0.154	0.132	0.734	0.367
Q_3	0.309	0.844	1.000	1.063	0.232	0.183	0.609	0.430
P_4	0.367	0.734	3.563	8.000	0.154	0.132	0.215	1.594
Q_4	0.430	0.609	1.063	1.000	0.232	0.187	0.309	0.844
P_5	0.734	0.367	0.154	0.132	8.000	3.563	1.594	0.215
Q_5	0.609	0.430	0.232	0.187	1.000	1.062	0.844	0.309
P_6	0.367	0.154	0.734	0.132	3.563	8.000	1.594	0.215
Q_6	0.430	0.232	0.609	0.187	1.063	1.000	0.844	0.309
P_7	0.132	1.594	0.734	0.154	0.215	0.367	8.000	3.563
Q_7	0.187	0.844	0.609	0.232	0.309	0.430	1.000	1.063
P_8	0.215	1.594	0.367	3.563	0.132	0.154	0.734	8.000
Q_8	0.309	0.844	0.430	1.063	0.187	0.232	0.609	1.000
P_{MACT}	0.215	3.563	0.734	0.154	1.594	0.132	8.000	0.367
Q_{MACT}	0.309	1.063	0.609	0.232	0.844	0.187	1.000	0.430
P_{Clique}	0.734	3.563	0.132	0.215	1.594	0.154	0.367	8.000
Q_{Clique}	0.609	1.063	0.187	0.309	0.844	0.232	0.430	1.000

DOBSS produces the optimal mixed strategy as shown in Table 3. The patroller will patrol the graph with an optimal mixed strategy consisting of strategies 7 and 9. Here the cost-minimizing strategy for the adversary generates an optimal response for attacking vertex 7 (i.e. $z_7 = 1, z_i = 0, i \neq 7$ for all $i \in 1..n$).

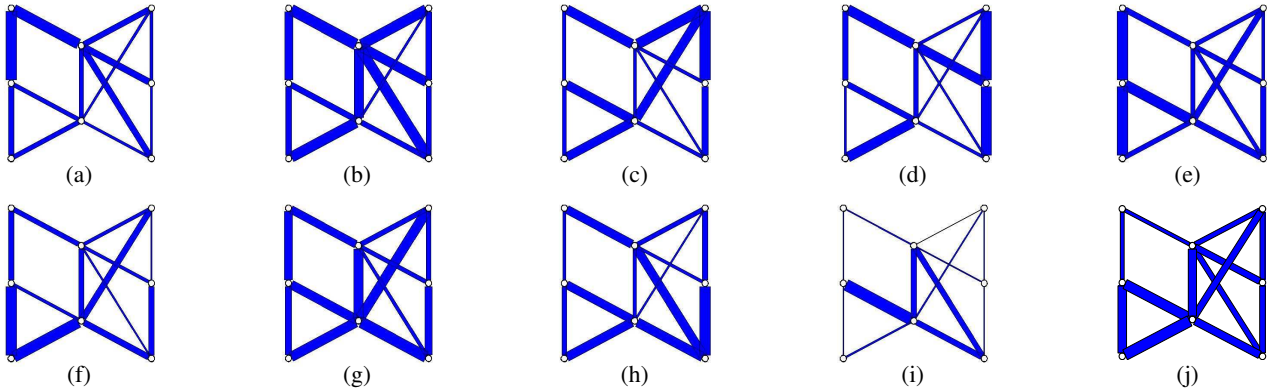
Table 3. Optimal mixed strategy result

Patrolling Strategy No., M_i	Proportion of using Strategy w_i
1	0
2	0
3	0
4	0
5	0
6	0
7	0.9012
8	0
9	0.0988
10	0



(a) (b) (c) (d)

Fig. 3: Different types of graph for MACT method: a) Edge weight allocation on a line graph of 20 vertices; b) Edge weight allocation on a tree with 30 vertices; c) Edge weight allocation on a complete graph, K_8; d) Edge weight allocation on a randomly generated graph with 50 vertices and 200 edges.



(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

Fig. 4: Edge weight allocation for ten patrolling strategies of a small graph: a)-h) Edge weight allocation for minimizing average commute time towards vertex 1 to vertex 8 respectively; i) Edge weight allocation for minimizing average commute time preferring a clique or subset of vertices, {2,5,8}; j) Edge weight allocation for minimizing average commute time over all vertices.

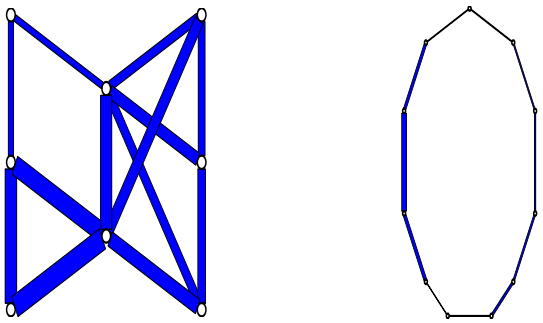
5.3 Decentralized Patrolling Results for Non-Uniform Distribution Case

We have tested our FMRMC method with a small graph [25] consisting of eight vertices and thirteen edges.
We compare more closely our FMRMC method with two heuristic methods such as Max-degree and Metropolis-Hastings (M.-H.) on the small graph mentioned above. The result of this comparison is shown in Table 4. It shows that the second largest eigenvalue modulus (SLEM) and mixing rate for FMRMC are less compared to the other two methods.

Table 4. Comparison of Max-degree and Metropolis-Hasting(M.-H.) and FMRMC method

	Max-degree	M.-H.	FMRMC
SLEM μ	0.779	0.774	0.643
mixing rate τ	4.01	3.91	2.27

We have also tested our methods on different types of graphs, including line, tree, mesh, complete, and randomly generated graphs. Figure 5 shows the edge weight allocation on different graphs [25] for our FMRMC method. In each graph, the thickness of each edge corresponds to the optimal edge weight value, or the probability of that edge being chosen by a patroller. For example, a wider edge connection between two vertices represents a high probability of that edge being chosen for travel, and vice versa. The importance of the vertices in the graph is proportional to width of edges between vertices in the graph.



(a) (b)

Fig. 5: Different types of graph for FMRMC method: a) Edge weight allocation on a small graph of 8 vertices; b) Edge weight allocation on a cycle graph with 11 vertices;

6. CONCLUSION AND FUTURE WORK

In this paper, we have introduced algorithms for finding decentralized patrolling strategies in case of uniform distribution over all vertices in the graph, for multiple robots, based on Markov chains which minimize the average commute time towards (1) a specific vertex, (2) a subset of the vertices and (3) the average commute time for all pairs of vertices in the graph. Methods (1) and (2) use convex optimization and method (3) seeks the shortest path in graphs. Also, we have also proposed an algorithm for finding decentralized patrolling strategy for multiple robots based on Markov chains which converges very fast for the graph with non-uniform distribution over vertices. Several interesting directions are left for future work as described in further detail below.
One of the goals of our work was to remove the communication and localization requirements in [1] and extend the class of graphs to which Markov chain based strategies can be applied. Even though

we have used simple Markov chain based strategies for patrollers, one surprising result of our work seems to indicate that the performance does not degrade despite the lack of communication and synchronization. Further experimental tests and analytical tools are going to be used to quantify the differences in performance between our approach and existing methods.

The most promising line of future work is to incorporate a game theoretic framework so that patroller can patrol the environment optimally even the adversary uses the non-uniform distribution of costs over different regions of the environment. We also have to model the adversary so that the adversary can see the patroller an attack instantaneously or the adversary can collect statistics and attack after some time as well as what will be the optimal time to attack after statistics collection.

Another immediate line of future work is the implementation of our ideas in physical deployments. Since our approaches do not require that we solve localization, communication, or synchronization problems, and the motions of the patrollers follow simple Markov chains, so they can be implemented in inexpensive robotic platforms. In order to apply our methods, we can take a 2D or 3D workspace with obstacles, create a grid where each element of the grid is a state in the Markov chain, and choose the appropriate neighborhoods for transitions.

We have the plan to incorporate a less rational adversary in our framework. In our current setup, the adversary has complete knowledge of the environment and is assumed to be perfectly rational. However, real world situations will involve adversaries that are less knowledgeable and rational and more unpredictable. Because of this, we wish to extend our ideas to use a more realistic adversarial model to help test the true effectiveness of the patrolling policies.

7. REFERENCES

[1] N. Agmon, S. Kraus, and G. A. Kaminka, "Multi-robot perimeter patrol in adversarial settings," in *Proc. of the Intl. Conf. on Robotics and Automation*, pp. 2339–2345, 2008.

[2] N. Agmon, "On events in multi-robot patrol in adversarial environments," in *Proc. of the Intl. Conf. on Autonomous Agents and Multiagent Systems*, pp. 591–598, 2010.

[3] N. Agmon, G. A. Kaminka, and S. Kraus, "Multi-robot adversarial patrolling: facing a full-knowledge opponent," *J. of Artificial Intelligence Research*, vol. 42, pp. 887–916, 2011.

[4] T. Sak, J. Wainer, and S. K. Goldenstein, "Probabilistic multiagent patrolling," in *Brazilian Symp. on Artificial Intelligence*, pp. 124–133, Springer, 2008.

[5] P. Paruchuri, J. P. Pearce, M. Tambe, F. Ordonez, and S. Kraus, "An efficient heuristic approach for security against multiple adversaries," in *Proc. of the Intl. Joint Conf. on Autonomous Agents and Multiagent Systems*, 2007.

[6] P. Paruchuri, J. P. Pearce, J. Marecki, M. Tambe, F. Ordonez, and S. Kraus, "Playing games for security: an efficient exact algorithm for solving bayesian stackelberg games," in *Proc. of the Intl. Joint Conf. on Autonomous Agents and Multiagent Systems*, pp. 895–902, 2008.

[7] A. Ghosh, S. Boyd, and A. Saberi, "Minimizing effective resistance of a graph," *SIAM Review*, vol. 50, no. 1, pp. 37–66, 2008.

[8] S. Boyd, P. Diaconis, and L. Xiao, "Fastest mixing markov chain on a graph," *SIAM Review*, vol. 46, no. 4, pp. 667–689, 2004.

[9] S. Boyd, "Convex optimization of graph laplacian eigenvalues," in *Proc. of the Intl. Congress of Mathematicians*, vol. 3, pp. 1311–1319, 2006.

[10] T. Alam, M. Edwards, L. Bobadilla, and D. Shell, "Distributed multi-robot area patrolling in adversarial environments," in *Intl. Work. on Robotic Sensor Networks, Seattle, WA, USA*, 2015.

[11] Y. Chevaleyre, F. Sempe, and G. Ramalho, "A theoretical analysis of multi-agent patrolling strategies," in *Proc. of the Intl. Joint Conf. on Autonomous Agents and Multiagent Systems*, pp. 1524–1525, 2004.

[12] A. Almeida, G. Ramalho, H. Santana, P. Tedesco, T. Menezes, V. Corruble, and Y. Chevaleyre, "Recent advances on multi-agent patrolling," in *Brazilian Symp. on Artificial Intelligence*, pp. 474–483, Springer, 2004.

[13] M. Ahmadi and P. Stone, "A multi-robot system for continuous area sweeping tasks," in *Proc. of the Intl. Conf. on Robotics and Automation*, pp. 1724–1729, 2006.

[14] Y. Elmaliach, N. Agmon, and G. A. Kaminka, "Multi-robot area patrol under frequency constraints," *Annals of Mathematics and Artificial Intelligence*, vol. 57, no. 3-4, pp. 293–320, 2009.

[15] D. Portugal and R. Rocha, "Msp algorithm: multi-robot patrolling based on territory allocation using balanced graph partitioning," in *Proc. of Symp. on Applied Computing*, pp. 1271–1276, 2010.

[16] D. Portugal and R. Rocha, "A survey on multi-robot patrolling algorithms," in *Doctoral Conf. on Computing, Electrical and Industrial Systems*, pp. 139–146, Springer, 2011.

[17] N. Basilico, N. Gatti, and F. Amigoni, "Leader-follower strategies for robotic patrolling in environments with arbitrary topologies," in *Proc. of the Intl. Conf. on Autonomous Agents and Multiagent Systems*, pp. 57–64, 2009.

[18] N. Basilico, N. Gatti, and F. Villa, "Asynchronous multi-robot patrolling against intrusions in arbitrary topologies.," in *AAAI*, 2010.

[19] N. Basilico, N. Gatti, and F. Amigoni, "Patrolling security games: Definition and algorithms for solving large instances with single patroller and single intruder," *Artificial Intelligence*, vol. 184, pp. 78–123, 2012.

[20] N. Basilico and S. Carpin, "Online patrolling using hierarchical spatial representations," in *Proc. of the Intl. Conf. on Robotics and Automation*, pp. 2163–2169, IEEE, 2012.

[21] Y. Vorobeychik, B. An, M. Tambe, and S. P. Singh, "Computing solutions in infinite-horizon discounted adversarial patrolling games.," in *Proc. of the Intl. Conf. on Automated Planning and Scheduling*, 2014.

[22] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Management Science*, vol. 17, no. 11, pp. 712–716, 1971.

[23] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The J. of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[24] W. K. Hastings, "Monte carlo sampling methods using markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, 1970.

[25] Convex Optimization in Matlab. Available at http://cvxr.com/cvx/examples/.