Patrolling in A Stochastic Environment

Student Paper Submission
(Suggested Track: Modeling and Simulation)

Sui Ruan \(^1\) (Student)
E-mail: sruan@engr.uconn.edu

Candra Meirina \(^1\) (Student)
E-mail: meirina@engr.uconn.edu

Feili Yu \(^1\) (Student)
E-mail: yu02001@engr.uconn.edu

Krishna R. Pattipati \(^{13}\)
University of Connecticut, Dept. of Electrical and Computer Engineering
371 Fairfield Road, Unit 1157
Storrs, CT 06269-1157
Fax: 860-486-5585
Phone: 860-486-2890
E-mail: krishna@engr.uconn.edu

Robert L. Popp \(^2\)
Information Exploitation Office, DARPA
3701 N. Fairfax Drive, Arlington, VA 22203-1714
Phone: (703)248-1520
E-mail: rpopp@darpa.mil

---

* This work is supported by the Office of Naval Research under contract #N00014-00-1-0101
\(^1\) Electrical and Computer Engineering Department, University of Connecticut, Storrs, CT 06269-1157, USA.
\(^2\) Information Exploitation Office, DARPA, 3701 N. Fairfax Drive, Arlington, VA 22203, USA
\(^3\) Correspondence: krishna@engr.uconn.edu
1. REPORT DATE  
JUN 2005

2. REPORT TYPE

3. DATES COVERED
00-00-2005 to 00-00-2005

4. TITLE AND SUBTITLE
Patrolling in A Stochastic Environment

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

6. AUTHOR(S)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Connecticut, Department of Electrical and Computer Engineering, 371 Fairfield Road Unit 1157, Storrs, Ct, 06269-1157

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR’S ACRONYM(S)

11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:
   a. REPORT unclassified
   b. ABSTRACT unclassified
   c. THIS PAGE unclassified

17. LIMITATION OF ABSTRACT

18. NUMBER OF PAGES 27

19a. NAME OF RESPONSIBLE PERSON

Standard Form 298 (Rev. 8-98)
Proscribed by ANSI Std Z39-18
Patrolling in a Stochastic Environment

Sui Ruan\textsuperscript{a}, Candra Meirina\textsuperscript{a}, Feili Yu\textsuperscript{a}, Krishna Pattipati\textsuperscript{a} and Robert L. Popp \textsuperscript{b}

Abstract

The patrolling problem considered in this paper has the following characteristics: Patrol units conduct preventive patrolling and respond to call-for-service. The patrol locations (nodes) have different priorities, and varying incident rates. We design a patrolling scheme such that the locations are visited based on their importance and incident rates. The solution is accomplished in two steps. First, we partition the set of nodes of interest into subsets of nodes, called sectors. Each sector is assigned to one patrol unit. Second, for each sector, we exploit a response strategy of preemptive call-for-service response, and design multiple sub-optimal off-line patrol routes. The net effect of randomized patrol routes with immediate call-for-service response would allow the limited patrol resources to provide prompt response to random requests, while effectively covering the nodes of different priorities having varying incidence rates. To obtain multiple routes, we design a novel learning algorithm (Similar State Estimate Update) under a Markov Decision Process (\textit{MDP}) framework, and apply softmax action selection method. The resulting patrol routes and patrol unit visibility would appear unpredictable to the insurgents and criminals, thus creating the impression of virtual police presence and potentially mitigating large scale incidents.

I. INTRODUCTION

In a highly dynamic and volatile environment, such as a post-conflict stability operation or a troubled neighborhood, military and/or police units conduct surveillance via preventive patrolling, together with other peace keeping or crime prevention activities. Preventive patrol constitutes touring an area, with the patrol units scanning for threats, attempting to prevent incidents, and intercepting any threats in progress. Effective patrolling can prevent small scale events from cascading into large scale incidents, and can enhance civilian security. Consequently, it is a major component of stability operations and crime prevention. In crime control, for example, for the greatest number of civilians, deterrence through ever-present police patrol, coupled with the prospect of speedy police action once a report is received, appears crucial in that the presence or potential presence of police officers on patrol severely inhibits criminal activity\cite{1}. Due to limited patrolling resources (e.g., manpower, vehicles, sensing and shaping resources), optimal resource allocation and planning of patrol effort are critical to effective stability operations and crime prevention\cite{2}.

The paper is organized as follows: In section II, the stochastic patrolling problem is modeled. In section III, we propose a solution approach based on a \textit{MDP} framework. Simulation results are presented in section IV. In section V, the paper concludes with a summary and future research directions.

II. STOCHASTIC PATROLLING MODEL

The patrolling problem is modeled as follows:

- A finite set of nodes of interest: $\mathcal{N} = \{i; \ i = 1, \ldots, I\}$. Each node $i \in \mathcal{N}$ has the following attributes:

\textsuperscript{a}Electrical and Computer Engineering Department, University of Connecticut, Storrs, CT 06269-1157, USA. E-mail: [sruan, meirina,\textsuperscript{,} yu02001, krishna]@engr.uconn.edu. \textsuperscript{b}Information Exploitation Office, DARPA, 3701 N. Fairfax Drive, Arlington, VA22203, USA. Email: rpopp@darpa.mil. This work is supported by the Office of Naval Research under contract No. 00014-00-1-0101.
- fixed location \((x_i, y_i)\);
- incident rate \(\lambda_i(1/\text{hour})\): we assume that the number of incident occurring at node \(i\) in a time interval \((t_1, t_2)\), denoted by \(n_i(t_2, t_1)\), is a Poisson random variable with parameter \(\lambda_i(t_2 - t_1)\): 
  \[
P(n_i(t_2, t_1) = k) = \frac{e^{-\lambda_i(t_2-t_1)}(\lambda_i(t_2 - t_1))^k}{k!}
  \]
- importance index \(\delta_i\): a value indicating the relative importance of node \(i\) in the patrolling area.

- The connectivity of the nodes: for any node \(j\) directly connected to node \(i\), we denote it as \(j \in \text{adj}(i)\), and the length of the edge connecting them as \(e(i, j)\);
- A finite set of identical patrol units, each with average speed \(v\), i.e., the estimated time for a unit, \(\bar{T}\), to cover a distance \(d\), is \(\bar{T} = \frac{d}{v}\). Each unit would respond to a call-for-service immediately when a request is received; otherwise, the patrol unit traverses along prescribed routes.

In this paper, we focus our attention on the problem of routing for effective patrolling, and assume that whenever a patrol unit visits a node, the unit can clear all incidents on that node immediately. Some real world constraints, such as the resources required and incident clearing times are not considered; future work would address these extensions.

### III. Proposed Solution

Our solution to the patrolling problem consists of two steps. First, we partition the set of nodes of interest (corresponding to a city for example) into subsets of nodes called sectors. Each sector is assigned to one patrol unit. Second, for each sector, we exploit a response strategy of preemptive call-for-service response, and design multiple off-line patrol routes. The patrol unit randomly selects predefined routes to conduct preventive patrolling; whenever a call-for-service request is received, the patrol unit would stop the current patrol and respond to the request immediately; after completing the call-for-service, the patrol unit would resume the suspended patrol route. The net effect of randomized patrol routes with immediate call-for-service response would allow limited patrol resources to provide prompt response to random requests, while effectively covering the nodes of different priorities having varying incidence rates.

The sector partitioning sub-problem is formulated as a combinatorial optimization problem, and solved via political districting algorithms presented in [5]. The off-line route planning subproblem for each sector is formulated as an infinite-horizon Markov Decision Process (MDP)[4], based on which a novel learning method, viz., Similar State Estimate Update, is applied. Furthermore, we apply Softmax action selection method[8] to prescribe multiple patrol routes to create the impression of virtual patrol presence and unpredictability.

#### A. Area partitioning for patrol unit assignment

The problem of partitioning a patrol area can be formulated as follows:

- A region is composed of a finite set of nodes of interest: \(N = \{i; \ i = 1, ..., I\}\). Each node \(i \in N\) is centered at position \((x_i, y_i)\), and value \(\varphi_i = \lambda_i \delta_i\);
There are $r$ areas to cover the region, such that all nodes are covered, with minimum overlap, and the sum of values for each area is similar, and areas are compact.

This is a typical political districting problem. Dividing a region, such as a state, into small areas, termed districts, to elect political representatives is called political districting[6]. A region consists of $I$ population units such as counties (or census tracks), and the population units must be grouped together to form $r$ districts. Due to court rulings and regulations, the deviation of the population per district cannot exceed a certain proportion of the average population. In addition, each district must be contiguous and compact. A district is contiguous, if it is possible to reach any two places of the district without crossing another district. Compactness essentially means that the district is somewhat circular or a square in shape rather than a long and thin strip. Such shapes reduce the distance of the population units to the center of the district or between two population centers of a district. This problem was extensively studied in [5], [6].

B. Optimal Routing in a Sector

1) MDP modeling: In a sector, there are $n$ nodes of interest, $N = \{1, \ldots, n\} \subseteq \mathbb{R}$. A Markov Decision Process (MDP) representation of the patrolling problem is as follows:

- Decision epochs are discretized such that each decision epoch begins at the time instant when the patrol unit finishes checking on a node, and needs to move to a next node; the epoch ends at the time instant when the patrol unit reaches the next node, and clears all incidents at that node.
- States $\{s\}$: a state, defined at the beginning of decision epoch $t$, is denoted as $s = (i, w)$, where $i \in N$ is the node the patrol unit is currently located at, and $w = \{w_j\}_{j=1}^{n}$ denotes the times elapsed since the nodes are last visited;
- Actions $\{a\}$: an action, also defined at the beginning of decision epoch $t$, is denoted as $a = (i, j)$, where $i$ is the patrol unit’s current location, and $j \in \text{adj}(i)$, an adjacent node of $i$, denotes the next node to be visited;
- State transition probabilities $P(s'|s, a)$: given state $s$, and action $a$, the probability of $s'$ being the next state;
- Reward $g(s, a, s')$: the reward for taking action $a = (i, j)$ at state $s = (i, w)$ to reach next state $s' = (j, w')$. At time $t'$, the patrol unit reaches node $j$ and clears $n_j(t')$ incidents, and earns the reward at time $t'$ of $g(s, a, s') = \delta_j n_j(t')$.
- Discount mechanism: the reward $g$ potentially earned at future time $t'$ is valued as $ge^{-\beta(t' - t)}$ at current time $t$, where $\beta$ is the discount rate;
- Objective is to determine an optimal policy, i.e., a mapping from states to actions, such that the overall expected reward is maximized.

The value function (expected reward) of a state, $s$ at time $t_0$, for policy $\Pi$ (a mapping from state to action) is defined as:

$$V^\Pi(s) = \mathbb{E}[\sum_{k=0}^{\infty} g_{k+1} e^{-\beta(t_{k+1} - t_0)}],$$

(1)
where $g_{k+1}$ is the reward earned at time $t_{k+1}$. Note that $V^\Pi(s)$ is independent of time, $t$, i.e., a constant state-dependent stationary value corresponding to a stationary policy.

Dynamic Programming [4][7] and Reinforcement Learning [8] can be employed to solve the MDP problem. In this work, we first prove that under any deterministic policy $\Pi$, the structure of value function ($V^\Pi$) of a state, $s = (i, w)$, is a linear function: $V^\Pi(s = (i, w)) = (c^\Pi(i))^T w + d^\Pi(i)$. Therefore, the optimal policy satisfies $V^\star(s = (i, w)) = (c^\star(i))^T w + d^\star(i)$. Here, we denote $c^\Pi(s)$, $d^\Pi(s)$ as the parameters for policy $\Pi$, while $c^\star(s)$, $d^\star(s)$ are the concomitant parameters for the optimal policy $\Pi^\star$. Based on this structure, we construct the linear function as an approximation of optimal value function, denoted as: $\tilde{V}^\star(s = (i, w)) = (\varphi^\star(i))^T w + \varphi^\star(i)$. This special structure of the value function enables us to design a novel learning algorithm, the so-called Similar State Estimate Update (SSEU) to obtain a deterministic near-optimal policy, from which a near-optimal patrolling route can be obtained. The SSEU algorithm employs the ideas from Monte-Carlo and Temporal Difference (specifically, TD(0)) methods[8]), while overcoming the inefficiencies of these methods on the patrolling problem.

At state $s = \{i, \{w\}\}$, when action $a = (i, j)$ is undertaken, the state transverses to $s' = \{j, \{w\}\}$. Note that under our modeling assumption, the state transition by action $a$ is deterministic, while the reward accrued by action $a$ at state $s$ is stochastic in the sense that the number of incidents at node $j$ is random. Therefore, the Bellman’s equation for the patrolling problem can be simplified as:

$$V^\star(s) = \max_a \alpha(s, s') \{ E[g(s, a, s')] + \gamma V^\star(s') \}$$

(2)

Here $g(s, a, s')$ is the reward for taking action $a = (i, j)$ at state $s = (i, w)$ to reach state $s' = (j, w')$. The expected reward is $E[g(s, a, s')] = \delta_j \lambda_j[w_j + \varphi^\star(i)]$, and $\alpha(s, s') = e^{-\beta D(i,j)}$ accounts for discount factor for state transition from $s$ to $s'$.

The greatest challenge in using MDPs as the basis for decision making lies in discovering computationally feasible methods for the construction of optimal, approximately optimal or satisfactory policies[7]. Arbitrary MDP problems are intractable; producing even satisfactory or approximately optimal policies is generally infeasible. However, many realistic application domains exhibit considerable structure and this structure can be exploited to obtain efficient solutions. Our patrolling problem falls into this category.

**Theorem 1**: For any deterministic policy in the patrolling problem, i.e., $\Pi : s \rightarrow a, \forall s \in S, \forall a \in A(s)$, the state value function has the following property:

$$V^\Pi(s = (i, w)) = (c^\Pi(i))^T w + d^\Pi(i) \quad \forall i \in N$$

(3)

**Proof**: Under any deterministic policy, $\Pi$, for an arbitrary state $s = (i, w)$ at $t$, the follow-on state trajectory is deterministic as the state transition is deterministic in the patrolling problem. We denote the state trajectory in a format “node(time, reward)” as:
\[ i_0(= i)(t, 0) \rightarrow i_1(t + T_1, r_1) \rightarrow \ldots \rightarrow i_N(t + T_N, r_N) \rightarrow \ldots \] (4)

Thus, the value function of state \( s \) under policy \( \Pi \) is

\[ V^\Pi(s = (i, \mathbf{w})) = E \left[ \sum_{k=0}^{\infty} r_k e^{-\beta T_k} \right] = \sum_j f_{ij} \] (5)

where \( r_k \) is the reward earned at decision epoch \( t_k \) and \( f_{ij} \) signifies its expected sum of rewards earned at node \( j \). Since the sequence of visits to node \( j \) is:

\[ j(t + T_{j,1}, r_{j,1}) \rightarrow \ldots \rightarrow j(t + T_{j,2}, r_{j,2}) \rightarrow \ldots \rightarrow j(t + T_{j,N}, r_{j,N},) \ldots \] (6)

and expected reward of first visit to node \( j \) following state \( s \) is: \( E(r_{j,1}) = \delta_j \lambda_j (w_j + T_{j,1}) e^{-\beta T_{j,1}}, \) and \( k^{th} \) \((k > 1)\) visit to node \( j \) is \( E(r_{j,k}) = \delta_j \lambda_j (T_{j,k} - T_{j,k-1}) e^{-\beta T_{j,k-1}} \). Therefore, we have

\[ f_{ij} = \delta_j \lambda_j [w_j + T_{j,1}] e^{-\beta T_{j,1}} + \delta_j \lambda_j [T_{j,2} - T_{j,1}] e^{-\beta T_{j,2}} + \ldots \delta_j \lambda_j [T_{j,N} - T_{j,N-1}] e^{-\beta T_{j,N}} \ldots \] (7)

Here, \( c_{ij} = \delta_j \lambda_j e^{-\beta T_{j,1}} \), and \( d_{ij} = \sum_{k=1}^{\infty} \delta_j \lambda_j [T_{j,k} - T_{j,k-1}] e^{-\beta T_{j,k-1}} \). Since \( T_{j,k} - T_{j,k-1}, \) \((k = 1, \ldots, \infty)\) are dependent on policy \( \Pi \) and state \( s \), we have \( V^\Pi(s = (i, \mathbf{w})) = (\mathbf{c}_i^\Pi(s))^T \mathbf{w} + d_i(s) \).

Based on this observation, we employ linear function approximation for \( V^* (s) \) as follows:

\[ V^*(s = (i, \mathbf{w})) = \widetilde{V}^*(s = (i, \mathbf{w})) \approx (\mathbf{c}_i^*)^T \mathbf{w} + d_i^*; \forall i \in N \] (8)

where \( \mathbf{c}_i^* = \{c_{ij}\}_{j=1}^{n}, c_{ij}^* \) is the expected value of \( \delta_j \lambda_j e^{-\beta T_{j,1}} \), \( j = 1, \ldots, n \) under optimal policy \( \Pi^* \); \( d_i^* \) is the expected value of \( \sum_{j=1}^{n} \sum_{k=1}^{\infty} \delta_j \lambda_j [T_{j,k} - T_{j,k-1}] e^{-\beta T_{j,k-1}} \) under optimal policy \( \Pi^* \).

Starting from an arbitrary policy, we could employ the following value and policy iteration method\[8\] to evaluate and improve the policies iteratively to gradually approach an optimal policy,

\[ V^{t+1} = \max_\forall a=(i,j), \ j \in \text{adj}(i) \alpha(s, s') \{E[g(s, a = (i, j), s')] + V^t(s')\}. \] (9)

\[ a^{t+1} = \arg \max_\forall a=(i,j), \ j \in \text{adj}(i) \alpha(s, s') \{E[g(s, a = (i, j), s')] + V^t(s')\}. \] (10)

2) Similar State Estimate Update Method (Learning Algorithm): We seek to obtain estimates \( r^* \) of optimal policy, where \( r^* = (\mathbf{c}, \mathbf{d})^* \), by minimizing the Mean-Squared-Error as:

\[ \min_r MSE(r) = \min_r \sum_{s \in S} (V^*(s) - \widetilde{V}(s, r))^2, \] (11)

where \( V^*(s) \) is the true value at state \( s \) under optimal policy, \( \widetilde{V}(s, r) \) is the linear approximation as defined in Eq(3).

At iteration step \( t \), we observe a new example \( s_t \rightarrow V^t(s_t) \). Stochastic gradient-descent methods adjust the parameter vector by a small amount in the direction that would most reduce the error on that example:

\[ r^{t+1} = r^t + \gamma_t [V^t(s_t) - \widetilde{V}(s_t, r^t)] \nabla \widetilde{V}(s_t, r^t) \] (12)
Here \( \nabla \) is the gradient operator with respect to \( r^t \), and \( \gamma_k \) is a positive step-size parameter. Stochastic approximation theory \([3]\) requires that \( \sum_{k=1}^{\infty} \gamma_k = \infty \) and \( \sum_{k=1}^{\infty} \gamma_k^2 < \infty \).

There are two classes of simulation-based learning methods to obtain \( r^* \), viz., Monte-Carlo and Temporal-Difference learning methods\([8]\). These methods require only experience - samples of sequences of states, actions, and rewards from on-line or simulated interaction with environment. Learning from simulated experience is powerful in that it requires no a priori knowledge of the environment’s dynamics, and yet can still attain optimal behavior. Monte-Carlo methods are ways of solving the reinforcement learning problem based on averaging the sample returns. In Monte Carlo methods, experiences are divided into episodes, and it is only upon the completion of an episode that value estimates and policies are changed. Monte-Carlo methods are thus incremental in an episode-by-episode sense. In contrast, Temporal Difference methods update estimates based in part on other learned estimates, without waiting for a final outcome\([8]\).

Monte-Carlo method, as applied to the patrolling problem, works as follows: based on current estimated \( r^t \), run one pseudo-episode (sufficiently long state trajectory); gather the observations of rewards of all states along the trajectory; apply the stochastic gradient descent method as in Eq(12) to obtain \( r^{t+1} \). Then, repeat the process until converged estimates \( (r^*) \) are obtained. A disadvantage of Monte-Carlo method here is that, for infinite \( MDP \), to make the return, \( V^t(s_t) \), accurate for each state, the episode has to be sufficiently long; this would result in large memory requirement and a long learning cycle.

Temporal Difference, \( TD(0) \) method, as applied to the patrolling problem works as follows: simulate one state transition with \( r^t \); then immediately update estimates to be \( r^{t+1} \). Define \( d_t \) as the return difference due to transition from state \( s \) to \( s' \):

\[
d_t = \alpha(s, s') [g(s, a, s') + \tilde{V}(s', r^t)] - \tilde{V}(s, r^t)
\]  

where \( \alpha(s, s') \) is the discount factor for state transition from \( s \) to \( s' \). The \( TD(0) \) learning method updates estimates \( r^{t+1} \) according to the formula

\[
r^{t+1} = r^t + \gamma d_t \nabla \tilde{V}(s, r^t)
\]  

A disadvantage of \( TD(0) \) as applied to the patrolling problem is the following. Since adjacent states are always from different nodes, \( r^t_j (r_j = (e_j, d_j)) \) is used to update \( r^{t+1}_i \) \((i \neq j)\); this could result in slow convergence or even divergence.

To overcome the disadvantages of Monte-Carlo and \( TD(0) \) methods, while exploiting their strengths in value learning, we design a new learning method, termed the Similar State Estimate Update (SSEU). We define states where the patrol unit is located at the same node as being similar, e.g., \( s_1 = (i, w_1) \) and \( s_2 = (i, w_2) \) are similar states. Suppose that the generated trajectory under current estimation \((c^t \text{ and } d^t)\) for two adjacent similar states of node \( i \), i.e., state \( s = (i, w^t) \) and \( s' = (i, w'^{t_N}) \) is: \( i_0(=i)(t, 0), i_1(t_1, g_1), i_2(t_2, g_2), ..., i_N(=i)(t_N, g_N) \). Based on this sub-trajectory, we obtain the new observations of \( C_{ij}^{new} \), for nodes \( j = i_1, i_2, ..., i_N \) as follows:

\[
c_{ij}^{new} = \delta_j A_j e^{\beta (t_i^j - t)}
\]
and the new observations of \( d_i^{\text{new}} \):

\[
d_i^{\text{new}} = \sum_{k=1}^{N} g_k e^{-\beta (t_k - t)} + V'\left(s'\right) e^{-\beta (t_{N} - t)} - \sum_{j=i}^{i_N} c_{ij}^{\text{new}} w_j
\]  

Consequently, the parameters \( c_{ij} \) and \( d_i \) are updated by:

\[
c_{ij}^{t+1} = c_{ij}^t + \frac{c_{ij}^{\text{new}} - c_{ij}^t}{N_{ij}^c} \\
d_i^{t+1} = d_i^t + \frac{d_i^{\text{new}} - d_i^t}{N_i^d}
\]  

where \( N_{ij}^c \) is the number of update of \( c_{ij} \), and \( N_i^d \) is the number of update of \( d_i \).

To make our learning algorithm effective, there are two other issues to consider. First, to avoid the possibility that some nodes are much less frequently visited than others, we apply exploring-starts rule, where we intentionally begin episodes from those nodes that are less frequently visited based on the simulation histories. Second, to escape from local minima, we employ the \( \epsilon \)-greedy method. The simplest action selection rule is to select the action with highest estimated action value as in Eq(10). This method always exploits current knowledge to maximize immediate reward, and it spends no time at all sampling apparently inferior actions to verify whether they might be profitable in the long term. In contrast, \( \epsilon \)-greedy behaves greedily most of the time, but every once in a while, with a small probability \( \epsilon \), selects an action at random, uniformly, and independently of the action-value estimates. In \( \epsilon \)-greedy, as in Eq(18), all non-greedy actions are given the minimal probability of selection, \( 1 - \frac{\epsilon}{|A(s)|} \), and the remaining bulk of the probability, \( 1 - \epsilon + \frac{\epsilon}{|A(s)|} \), is given to the greedy action [8], where \( |A(s)| \) is the cardinality of action set, \( A(s) \) in state \( s \). This enables the learning method to get out of local minima, and thus provides the balance between exploitation and exploration.

The details of Similar State Update learning algorithm can be found in Fig.1. The \( c^* \) and \( d^* \) obtained by this method can provide a near-optimal patrol route by concatenating greedy actions for each state, as described in Eq(10).

C. Strategy for Generating Multiple Patrolling Routes

In this section, we design a method for generating multiple satisfactory routes by Softmax action selection strategy. In order to impart virtual presence and unpredictability to patrolling, the unit needs multiple and randomized patrol routes. We employ Softmax action selection method[8], where the greedy action is still given the highest selection probability, but all the others are ranked and weighed according to their value estimates. The most common softmax method uses a Gibbs distribution. It chooses action \( a \) at state \( s \) with probability:

\[
e^{[Q^*(s,a) - Q^*]/\tau} \sum_{a' \in A(s)} e^{[Q^*(s,a') - Q^*]/\tau}
\]

where \( Q^* = \max_a Q^*(s,a) \);  

\[
Q^*(s,a) = \alpha(s,s')\{E[g(s,a,s')] + V^*(s')\}.
\]
Learning Algorithm: Similar State Estimate Update  
(With Exploring Starts and \(\varepsilon\)-greedy rules)

Initialize: 
\(c = 0, d = 0, Frequencies = 0\)

Repeat
- **Step 0 (Episode Initialization):** beginning with an empty episode \(\rho\), pick up a node 
  \(i_0 = \arg \min Frequencies\) and initialize \(w = 0\), append the state \(s = (i_0, w_0)\) to \(\rho\).
  
  Set \(t = 0\),
  \(\text{Frequencies}(i_0) + +\);
- **Step 1 (Parameters Update):** Get the last node of episode, i.e., \(s' = (i, w')\), find the latest similar state of \(s'\) in \(\rho\), i.e., \(s = (i, w)\), if no such node, go to step 2; 
  else obtain the sub-trajectory beginning at \(s\) and ending at \(s'\), update \(c_t + 1\) and \(d_t + 1\) as in Eq.(17), then go to step 2;
- **Step 2 (Policy Improvement):** Decide the action for state \(s\):
  \[
  j = \left\{ \begin{array}{ll}
  \max_{k \in \text{adj}(i)} \alpha(s,s')\{E[g(s,a = (i,k),s')] + V_t(s')\} & \text{w.p. } 1 - \varepsilon, \\
  \text{rand}(\text{adj}(i)) & \text{w.p. } \varepsilon,
  \end{array} \right.
  \]
  set \(\Delta_t = \varepsilon(i,j)\);
  calculate \(w'_t = w + \Delta_t\), \(w'_t = 0\);
  update \(t = t + \Delta_t\);
  append state \(s' = (j, w'_t)\) to episode \(\rho\);
  \(\text{Frequencies}(j) + +\);
  if \(\rho\) is sufficiently long, go to step 0; else go to step 1.

until \(c\) and \(d\) converge.

Fig. 1. Similar State Estimate Update (Learning Algorithm)

Here, \(\tau\) is a positive parameter called temperature. High temperatures cause the actions to be nearly equiprobable. Low temperatures cause a greater difference in selection probability for actions that differ in their value estimates. In the limit as \(\tau \to 0\), softmax action selection reverts to a greedy action selection.

IV. Simulation and Results

We illustrate our approach to patrol routing using a simple example that represents a small county, as in Fig. 2. The nodes, incident rates \((\lambda_i)\) and importance indices \((\delta_i)\) are given Table I.

The results for patrolling strategies from the similar state estimate update (SSEU) method and the one-step greedy strategy are compared in Table II. In the one-step greedy strategy, at each state, the neighboring node which results in the best instant reward is chosen as the next node, i.e., \(j = \arg \max_{k \in \text{adj}(i)} \alpha(s,s')\{E[g(s,a = (i,k),s')]\}\). If this patrol area is covered by one patrol unit, the expected overall reward of the unit following the route obtained by the SSEU method is 2,330 and the reward per unit distance is 17.4; while following the route from one-step greedy strategy, the expected overall reward is 1,474, and the expected reward per unit distance is 6.00. If this patrol area is divided into two sectors, i.e., sector a and sector b, as in Fig. 2, the SSEU method results in the following rewards: for sector a, the overall expected reward is 1,710 and the expected reward per unit distance is
Velocity of patrol ($v$): 1 unit distance/ unit time

discount rate ($\beta$): 0.1/unit time

<table>
<thead>
<tr>
<th>node</th>
<th>$\lambda_i$</th>
<th>$\delta_i$</th>
<th>node</th>
<th>$\lambda_i$</th>
<th>$\delta_i$</th>
<th>node</th>
<th>$\lambda_i$</th>
<th>$\delta_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>N1</td>
<td>2</td>
<td>2</td>
<td>N12</td>
<td>2</td>
<td>2</td>
<td>N23</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>N2</td>
<td>2</td>
<td>2</td>
<td>N13</td>
<td>2</td>
<td>2</td>
<td>N24</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>N3</td>
<td>2</td>
<td>2</td>
<td>N14</td>
<td>2</td>
<td>2</td>
<td>N25</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>N4</td>
<td>2</td>
<td>2</td>
<td>N15</td>
<td>2</td>
<td>2</td>
<td>N26</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>N5</td>
<td>2</td>
<td>2</td>
<td>N16</td>
<td>2</td>
<td>2</td>
<td>N27</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>N6</td>
<td>2</td>
<td>2</td>
<td>N17</td>
<td>3</td>
<td>4</td>
<td>N28</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>N7</td>
<td>2</td>
<td>2</td>
<td>N18</td>
<td>2</td>
<td>2</td>
<td>N29</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>N8</td>
<td>4</td>
<td>2</td>
<td>N19</td>
<td>1</td>
<td>2</td>
<td>N30</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>N9</td>
<td>2</td>
<td>2</td>
<td>N20</td>
<td>4</td>
<td>10</td>
<td>N31</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>N10</td>
<td>1</td>
<td>2</td>
<td>N21</td>
<td>2</td>
<td>1</td>
<td>N32</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>N11</td>
<td>1</td>
<td>2</td>
<td>N22</td>
<td>1</td>
<td>2</td>
<td>N33</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

19.43; for sector $b$, the overall expected reward is 1,471 and the expected reward per unit distance is 13.8. The one-step greedy strategy results in the following rewards: for sector $a$, the expected overall reward is 1,107, and the expected reward per unit distance is 10.9; for sector $b$, the expected overall reward is 1,238, and the expected reward per unit distance is 8.94. Thus, patrol routes obtained by the $SSEU$ method are highly efficient compared to the short-sighted one-step greedy strategy in this example. In this scenario, the nodes with high incident rates and importance indices are spread out and sparse. Typically, the $SSEU$ method is effective for general configurations of patrol area. Another observation from the simulation is that the net reward from sector $a$ and sector $b$, i.e., 3,181, with two patrolling units, is 36% better than the net reward (2,330) when there is only one patrol unit.
Furthermore when a unit patrols on a smaller area, higher overall reward per area and higher reward per unit distance are expected. After applying softmax action selection method on the near-optimal strategy from SSEU method on sector \( a \), we obtained multiple sub-optimal routes for this sector; four of them are listed in Table III.

### Table II

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Patrol Route</th>
<th>Expected Reward</th>
<th>Reward /distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSEU (whole county)</td>
<td>1, 10, 20, 21, 31, 32, 33, 34, 41, 40, 39, 36, 37, 26, 25, 28, 29, 30, 34, 33, 32, 31, 21, 20, 19, 18, 17, 24, 29, 35, 40, 39, 38, 37, 36, 35, 34, 33, 32, 31, 21, 20, 10, 1, 2, 3, 8, 12, 13, 17, 24, 23, 30, 34, 41, 40, 39, 36, 37, 27, 26, 15, 6, 5, 4, 7, 8, 9, 11, 10, 20, 21, 31, 32, 43, 42, 41, 34, 35, 40, 39, 38, 37, 26, 25, 24, 17, 18, 19, 20, 21, 22, 23, 30, 34, 33, 32, 43, 42, 41, 40, 39, 36, 37, 27, 26, 16, 15, 14, 13, 17, 18, 19, 20, 10, (1)</td>
<td>2,330</td>
<td>17.4</td>
</tr>
<tr>
<td>One-step greedy (whole county)</td>
<td>1, 9, 8, 7, 8, 3, 2, 1, 9, 8, 12, 18, 17, 13, 14, 15, 16, 5, 4, 7, 8, 3, 2, 1, 9, 11, 12, 18, 24, 25, 26, 16, 15, 14, 7, 8, 3, 2, 1, 9, 8, 12, 17, 18, 13, 14, 15, 16, 5, 4, 7, 8, 3, 2, 1, 10, 20, 19, 22, 23, 30, 34, 41, 40, 39, 38, 37, 36, 35, 34, 33, 32, 31, 20, 19, 22, 23, 30, 34, 41, 40, 39, 40, 35, 34, 33, 32, 43, 42, 41, 40, 39, 38, 37, 27, 26, 25, 28, 29, 24, 17, 13, 7, 4, 5, 6, 15, 16, 26, 25, 28, 36, 37, 38, 39, 40, 35, 34, 30, 23, 18, 12, 8, 3, 2, (1)</td>
<td>1,474</td>
<td>6.00</td>
</tr>
<tr>
<td>SSEU (sector a)</td>
<td>1, 10, 20, 21, 31, 32, 33, 34, 30, 23, 18, 19, 20, 21, 31, 32, 33, 34, 41, 42, 43, 32, 31, 32, 43, 42, 33, 34, 30, 29, 28, 25, 26, 16, 15, 14, 13, 17, 24, 23, 18, 12, 8, 3, 2, 11, 9, 10, 11, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 17, 24, 29, 28, 25, 26, 16, 15, 14, 7, 8, 3, 2, 1, 9, 8, 12, 17, 18, 13, 14, 15, 16, 5, 4, 7, 8, 3, 2, 1, 10, 20, 19, 22, 23, 30, 34, 41, 40, 39, 38, 37, 36, 35, 34, 33, 32, 31, 20, 19, 22, 23, 30, 34, 41, 40, 39, 40, 35, 34, 33, 32, 43, 42, 41, 40, 39, 38, 37, 27, 26, 25, 28, 29, 24, 17, 13, 7, 4, 5, 6, 15, 16, 26, 25, 28, 36, 37, 38, 39, 40, 35, 34, 30, 23, 18, 12, 8, 3, 2, (1)</td>
<td>1,710</td>
<td>19.43</td>
</tr>
<tr>
<td>One-step greedy (sector a)</td>
<td>1, 9, 8, 3, 2, 9, 8, 12, 18, 17, 24, 25, 26, 16, 15, 14, 13, 17, 24, 23, 30, 34, 41, 40, 39, 38, 37, 36, 35, 34, 33, 32, 31, 20, 19, 22, 23, 18, 12, 8, 3, 2, 9, 11, 10, 20, 21, 31, 32, 43, 42, 41, 34, 30, 23, 18, 12, 8, 3, 2, 9, 11, 19, 20, 10, (1)</td>
<td>1,107</td>
<td>11.0</td>
</tr>
</tbody>
</table>

### V. Summary and future work

In this paper, we considered the problem of effective patrolling in a dynamic and stochastic environment. The patrol locations are modeled with different priorities and varying incident rates. We identified a solution approach, which has two steps. First, we partition the set of nodes of interest into sectors. Each sector is assigned to one patrol unit. Second, for each sector, we exploited a response strategy of preemptive call-for-service response, and designed multiple off-line patrol routes. We applied the MDP methodology and designed a novel learning algorithm to obtain a deterministic optimal patrol route. Furthermore, we applied Softmax action selection method to device multiple patrol routes for the patrol unit to randomly choose from. Future work includes the following: a) considering
## TABLE III

### MULTIPLE PATROLLING ROUTES

<table>
<thead>
<tr>
<th>Route</th>
<th>Patrol route</th>
<th>Expected Reward</th>
<th>Reward /distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Route I:</td>
<td>1, 10, 20, 21, 22, 23, 30, 34, 33, 32, 43, 42, 33, 34, 41, 42, 33, 34, 30, 23, 22, 19, 20, 10, 11, 12, 8, 9, 2, 3, 8, 12, 11, 19, 20, 21, 31, 32, 43, 42, 41, 34, 33, 32, 31, 21, 22, 23, 18, 19, 20, 10, 11, 12, 8, 9, 2, 3, 8, 12, 18, 23, 30, 34, 41, 42, 33, 34, 30, 23, 18, 19, 22, 21, 20, 19, 18, 23, 30, 34, 33, 32, 31, 21, 20, 10, 11, 9.</td>
<td>1,525</td>
<td>17.05</td>
</tr>
<tr>
<td>Route II:</td>
<td>1, 10, 20, 21, 22, 19, 20, 21, 31, 32, 43, 42, 41, 34, 33, 32, 31, 21, 20, 10, 11, 9, 8, 3, 2, 9, 8, 12, 18, 23, 22, 19, 20, 10, 11, 9, 8, 12, 18, 23, 30, 34, 41, 42, 33, 32, 31, 21, 20, 19, 22, 23, 30, 34, 41, 42, 33, 32, 43, 42, 33, 34, 30, 23, 18, 19, 22, 23, 18, 19, 20, 10, 11, 12, 8, 9, 11, 12, 18, 23, 30, 34, 33, 32, 31, 21, 20, 19, 11, 12, 8, 3, 2, 9, 8, 3, 2, 1.</td>
<td>1,831</td>
<td>18.68</td>
</tr>
<tr>
<td>Route III:</td>
<td>1, 2, 9, 11, 10, 20, 19, 11, 9, 8, 3, 2, 9, 8, 3, 2, 9, 11, 10, 20, 21, 31, 32, 43, 42, 41, 34, 33, 32, 34, 30, 23, 18, 12, 8, 9, 11, 19, 20, 21, 31, 32, 33, 34, 30, 23, 22, 19, 20, 10, 11, 12, 8, 9, 2, 3, 8, 9, 2, 3, 8, 12, 18, 23, 30, 34, 33, 32, 31, 21, 20, 19, 11, 12, 8, 3, 2, 9, 8, 3, 2, 1.</td>
<td>1,300</td>
<td>15.22</td>
</tr>
<tr>
<td>Route VI:</td>
<td>1, 2, 9, 8, 12, 18, 23, 22, 21, 20, 19, 18, 12, 8, 3, 2, 9, 8, 12, 18, 23, 30, 34, 41, 42, 33, 32, 31, 21, 20, 10, 11, 9, 8, 12, 18, 23, 30, 34, 33, 32, 43, 42, 33, 34, 30, 23, 18, 12, 11, 19, 20, 10, 11, 9.</td>
<td>1,389</td>
<td>15.20</td>
</tr>
</tbody>
</table>

the incident processing time and resource requirement at each node; b) including patrol unit’s resource capabilities in the patrolling formulation; c) and applying adaptive parameter updates for incident rates and importance rates at each node.

**REFERENCES**

Patrolling in Stochastic Environments

Sui Ruan*
Candra Meirina*
Feili Yu*
Prof. Krishna R. Pattipati*
Dr. Robert L. Popp

*Dept. of Electrical and Computer Engineering
University of Connecticut
Contact: krishna@engr.uconn.edu (860) 486-2890

10th International Command and Control Research and Technology Symposium
June 13 - 16, 2005
Outline

- Introduction
- Stochastic Patrolling Model
- Our Proposed Solution
- Simulation Results
- Summary and Future Work
Introduction

- **Motivation**
  - Preventive patrolling is a major component of stability operations and crime prevention in highly volatile environments.
  - *Optimal resource allocation and planning* of patrol effort are critical to effective stability and crime prevention due to limited patrolling resources.

- **Model and Design Objective**
  - Introduce a model of patrolling problems that considers patrol nodes of interest to have *different priorities* and *varying incident rates*.
  - Design a patrolling strategy such that the net effect of *randomized patrol routes* with immediate call-for-service response allows limited patrol resources to *provide prompt response to random requests*, while *effectively covering the entire nodes*.
Consider a finite set of nodes of interest: \( N = \{i; i=1, \ldots, n\} \)

Each node \( i \) has the following attributes:

- Fixed location: \((x_i, y_i)\)
- Incident rate: \(\lambda_i\) (incidents/hour)  
  \(\Rightarrow\) assume a Poisson process
- Important index: \(\delta_i\)  
  \(\Rightarrow\) indicate relative importance of node \( i \) in the patrolling area

Assume \( r \) patrol units  
\(\Rightarrow\) each with average speed \( v \)
Step 1: Partition the set of nodes of interest into sectors – subsets of nodes. Each sector is assigned to one patrol unit.
⇒ Sector partitioning sub-problem

Step 2: Utilize a response strategy of preemptive call-for-service response and design multiple off-line patrol routes for each sector

Step 2.1: Response strategy
- Put higher priority to call-for-service requests ⇒ stop current patrols and respond to the requests
- Resume suspended patrols after call-for-service completion

Step 2.2: Off-line route planning sub-problem
- Optimal routing in a sector ⇐ Similar State Estimate Update (SSEU) in Markov Decision Process framework
- Strategy for generating multiple patrol routes ⇐ randomized (“softmax”) action selection method
The problem is formulated as a political districting problem:

- Let the finite set of nodes of interest form a region.
- Each node in the region is centered at \((x_i, y_i)\), and has an importance value of \(\varphi_i = \lambda_i \delta_i\).
- Define \(r\) areas (commensurate to the number of patrol units) over the region such that:
  - All nodes are covered with minimum overlaps
  - Similar sums of importance values between areas
  - Geography of the areas must be compact and contiguous

This problem has been extensively studied in combinatorial optimization [Garfinkel1970].
2.2: Off-line Route Planning Sub-problem
Markov Decision Process (MDP) Representation

States \( \{s\} \):
- A state is denoted by \( s = \{i,w\} \)
- \( i \) represents the node that has been most recently cleared by a patrol unit (and \( i \) is also the current location of the patrol unit)
- \( w = \{w_k\}_{k=1}^n \) denotes elapsed time of all nodes since last visits from the patrol unit

Action \( \{a\} \):
- An action is denoted by \( a = (i,j) \)
- \( j (\neq i) \) is an adjacent node of \( i \), the next node to be visited

Reward \( g(s,a,s') \):
Define the reward for taking action \( a = (i,j) \) at state \( s = \{i,w\} \) to reach next state \( s' = \{j,w'\} \)

Discount mechanism:
- The reward \( g \) potentially earned at time \( t' \) is valued as \( ge^{-\beta(t'-t)} \) at time \( t \), where \( \beta \) is the discount rate
- Encourage prompt actions

Objective:
Determine an optimal policy, i.e., a mapping from states to actions, that maximizes the overall expected reward
2.2: Off-line Route Planning Sub-problem
Linear State Value Structure

- Arbitrary MDP problems are intractable
- Fortunately, our patrolling problem exhibits a special structure: linearity

For any deterministic policy in the patrolling problem, the state value function has the property:

\[ V^\Pi(s = (i, \underline{w})) = (c_i^\Pi(s))^T \underline{w} + d_i^\Pi(s) \quad \forall i \in \mathbb{N} \]

linear w.r.t. \( \underline{w} \) (elapsed time of nodes since last visits from a patrol unit)

- Thus, a linear approximation of state value function for optimal policy is:

\[ \overline{V}^*(s = (i, \underline{w})) = (c_i^*)^T \underline{w} + d_i^* \]

- The problem becomes one of finding \( c_i^*, d_i^*, \forall i \in \mathbb{N} \) \( \Rightarrow \) determine the optimal policy
Introduce a variant of Reinforcement Learning (RL) method, **Similar State Estimate Update (SSEU)** method, to learn the optimal parameters $c^*_i$ and $d^*_i$, $\forall i \in \mathbb{N}$

- Reinforcement learning is a simulation-based learning method, which requires only experience, i.e., sample of sequences of states, actions and rewards from on-line or simulated interaction with the system environment.

- Given an arbitrary policy, $\Pi$, policy iteration method of RL iteratively improves the policy to gradually approach $\Pi^*$ as follows:

$$
\begin{align*}
    k^* &= \arg \max_{\forall a \in (i,k), \ k \in \text{adj}(i)} \alpha(s, s') \{ E[g(s, a = (i,k), s')] \} + V^\Pi(s') \\
    \alpha(s, s') &= e^{-\beta \text{dist}(i,j)} / v \\
    V^\Pi(s') &= (c_k)^T w + d_k
\end{align*}
$$

$\beta$: discount rate
$v$: average speed
$a$: action

Reward for taking action $a$ at state $s$, and reaching state $s'$$

State value of $s'$ under $\Pi$
2.2.a: Optimal Routing in a Sector
Similar State Estimate Update Method

Generate a trajectory via policy iteration utilizing current parameter estimates, $c_{ij}^t$ and $d_i^t$, for two adjacent similar states of node $i$, state $s = \{i, w\}$, $\tilde{s} = \{i, w'\}$:

Evaluate new values of $c_{ij}^{\text{new}}$ and $d_i^{\text{new}}$:

$$c_{ij}^{\text{new}} = \delta_j \lambda_j e^{-\beta(t_j^1 - t_0)}$$
$$d_i^{\text{new}} = \sum_{j=1}^{m} g_j e^{-\beta(t_j - t_0)} + \alpha(s, s')V^t(s') - (c_{ij}^{\text{new}})w^T$$

Thus

$$c_{ij}^{t+1} = c_{ij}^t + \frac{c_{ij}^{\text{new}} - c_{ij}^t}{m_{ij}^c}$$
$$d_i^{t+1} = d_i^t + \frac{d_i^{\text{new}} - d_i^t}{m_i^d}$$

$m_{ij}^c$ : number of $c_{ij}$ previous updates
$m_i^d$ : number of $d_i$ previous updates

Similar States: same node location, different visitation time

$w$ (elapsed time of nodes since last visits from a patrol unit)

$t_j^1$ denotes the first time node $j$ is visited in the trajectory; and

$$c_{ij}^* = \delta_j \lambda_j e^{-\beta(t_j^1 - t_0)}$$

$\delta_j$ : Important index of node $j$
$\lambda_j$ : Incident rate
$\beta$ : discount rate

$j$ represents a node along the trajectory

$w$ (elapsed time of nodes since last visits from a patrol unit)
2.2.b: Strategy for Generating Multiple Patrolling Routes

- **Why multiple patrolling routes?**
  - To impart virtual presence and unpredictability to patrolling
    ⇒ the patrol unit randomly selects one of many patrol routes

- **Softmax: random action selection method**
  - At each state,
    - The best action is given the highest selection probability
    - The second best action is given lesser probability
    - The third best action is given even less and …
  - Temperature – tunable parameter – decides probability differences among the actions
    - High temperatures ⇒ virtually equal probability
    - Low temperatures ⇒ greater difference in selection probabilities for actions having different value estimates
Results from the Illustrative Patrol Problem

<table>
<thead>
<tr>
<th>Range</th>
<th>Method</th>
<th>Expected Reward</th>
<th>Reward per Unit Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Whole Region</td>
<td>SSEU</td>
<td>2,330</td>
<td>17.4</td>
</tr>
<tr>
<td></td>
<td>Greedy</td>
<td>1,474</td>
<td>6.0</td>
</tr>
<tr>
<td>Sector a</td>
<td>SSEU</td>
<td>1,710</td>
<td>19.43</td>
</tr>
<tr>
<td></td>
<td>Greedy</td>
<td>1,455</td>
<td>13.8</td>
</tr>
<tr>
<td>Sector b</td>
<td>SSEU</td>
<td>1,471</td>
<td>13.8</td>
</tr>
<tr>
<td></td>
<td>Greedy</td>
<td>1,107</td>
<td>10.9</td>
</tr>
</tbody>
</table>

↑ Reward:
- ↑ Number of cleared incidents
- ↑ Incident importance
- ⇓ Latency

Greedy refers to one-step greedy strategy, i.e., for each state, select the neighboring node with best instant reward

- Patrol routes obtained by the SSEU method are highly efficient compared to the one-step greedy strategy
- Net reward from two patrolling units (for sectors a and b) is 36% higher with the SSEU method when compared to that of one patrol unit in the whole region
Present an analytical model of patrolling problem with varying incident rates and priorities

Propose a solution approach in two steps:

- **Step 1**: Solve the sector partitioning sub-problem via Political Districting Method ⇒ assign each sector to one patrol unit
- **Step 2**: Utilize a response strategy of preemptive call-for-service and define an optimal and near-optimal patrol routes for each sector via SSEU and “softmax”-based method, respectively

**Future work**:

- Incorporate incident processing time and resource requirements for each node
- Include patrol unit’s resource capabilities and workload constraints
- Introduce dynamic rerouting in the presence of changes in the incident rates and node priorities


Thank You!