This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/LWC.2018.2814576, IEEE Wireless Communications Letters

Novel Sensor Scheduling Scheme for Intruder Tracking in Energy Efficient Sensor Networks

Raghuram Bharadwaj Diddigi[1], Prabuchandran K.J[12], and Shalabh Bhatnagar[13]

Abstract—**We consider the problem of tracking an intruder using a network of wireless sensors. For tracking the intruder at each instant, the optimal number and the right configuration of sensors has to be powered. As powering the sensors consumes energy, there is a trade off between accurately tracking the position of the intruder at each instant and the energy consumption of sensors. This problem has been formulated in the framework of Partially Observable Markov Decision Process (POMDP). Even for the state-of-the-art algorithm in the literature, the curse of dimensionality renders the problem intractable. In this paper, we formulate the Intrusion Detection (ID) problem with a suitable state-action space in the framework of POMDP and develop a Reinforcement Learning (RL) algorithm utilizing the Upper Confidence Tree Search (UCT) method to solve the ID problem. Through simulations, we show that our algorithm performs and scales well with the increasing state and action spaces.**

Index Terms—**Wireless Sensor Networks, Intruder Tracking, Monte Carlo Tree Search.**

I. Introduction

The problem of detecting an intruder using a network of sensors arises in applications like wild-life or military surveillance. In this problem, the objective of the Intrusion Detection (ID) system is to track one or more intruders moving in the field of a wireless sensor network (WSN). Typically, WSNs operate on limited power supply. This imposes a limitation on the number of sensors (energy budget) that can be switched ON over a time period when tracking the intruders and thus, constrains the maximum achievable tracking accuracy. In this paper, we propose a novel RL ID algorithm that respects such resource constraints.

In [1], a comprehensive survey of adaptive sensing models for ID and POMDP solution methodologies has been provided. In [2], a POMDP model for target tracking using multiple sensors has been formulated. The sensor scheduling problem considered in [2] involves determining the subset of sensors that needs to be activated in order to manage the tradeoff between tracking performance and the sensor usage cost. A Monte Carlo solution that combines particle filters for belief state estimation and a Q-value approximation for solving POMDP has been proposed. In [3], the ID problem has been formulated as a POMDP and approximate solution methods like Q-MDP, point-based value iteration under different sensing, motion and cost models with increasing levels of difficulty has been studied. In [4], under a realistic detection model, mobile sensors along with static sensors has been utilized for detecting the position of the intruder.

In many practical scenarios, the curse of dimensionality effect (exponential growth of the state and action space of POMDP) renders the problem challenging. Point-based techniques attempt to alleviate the curse of dimensionality effect in POMDPs by carrying out value function estimation at only chosen belief points rather than at all the belief points [5]. In [5], an effective way of choosing the belief points has been discussed. This technique is then combined with value iteration to arrive at a Point-Based Value Iteration (PBVI) algorithm. However, this method is not scalable as it uses full-width computation accounting all possible actions, observations and the next state in the tree search [6]. In [7], an algorithm VDCBPI that combines the Value Directed Compression (VDC) and Bounded Policy Iteration (BPI) techniques to tackle the state explosion problem has been proposed. Using VDC, the belief space is compressed into smaller subspace containing only the necessary information that is required to evaluate a policy. BPI is then employed to perform the policy improvement only on the reachable beliefs to determine an optimal policy.

In [8], a hybrid value iteration algorithm for POMDPs that incorporates the advantages of both the point-based techniques and the tree search methods has been proposed. In the first step of this hybrid algorithm, an offline computation is performed to obtain the upper bounds on the optimal value function. In the subsequent step, this computation is used in the online tree search for obtaining the optimal action. In [9], a two time-scale Q-Learning algorithm with function approximation has been developed to mitigate the curse of dimensionality problem. In their algorithm, policy gradient update is carried out on the faster timescale and Q-values are updated on slower timescale. The challenge however with function approximation primarily lies in choosing the right features for approximating the Q-values. While all the above mentioned algorithms mitigate the problem of state explosion, the problem of action explosion needs to be handled to obtain scalable solutions.

In this paper we reformulate the POMDP model considered in [3] suitably and develop an RL algorithm (ID_γ_MCTS) that scales well to large sensor networks. Unlike in [3], our algorithms do not make any assumptions on the structure of the network or the movement of the intruder.

The following are our contributions:

- We solve the problem of state space explosion in ID by using the Monte Carlo Tree Search (MCTS) algorithm [6]. Unlike the search carried out in the prior methods considered for ID to mitigate the state space explosion like Monte-Carlo sampling methods and the point-

[1] Department of Computer Science and Automation, Indian Institute of Science (IISc), Bangalore, India, 560012
[2] Supported by Amazon-IISC Postdoctoral fellowship
[3] Robert Bosch Centre for Cyber-Physical Systems, IISc

based value iteration methods [10], the MCTS algorithm searches the belief space in a sequentially best first order.

- We solve the problem of action space explosion in ID by suitably reformulating the action space of POMDP in [3]. This reduces the exponential number of actions to be considered in ID to a few actions. Furthermore, this makes it amenable to the application of the MCTS algorithm.
- Our reformulation of the action space is inspired from our simple ID_TG algorithm (see Section III) that greedily selects the top sensors (implicitly ignoring the current belief value). The key idea in our ID_γ_MCTS algorithm is to choose the top sensors dynamically based on the current belief value.
- Our ID_γ_MCTS algorithm is a scalable solution to ID and requires no extra assumptions. This makes our solution very practical.

II. Problem and Model

Let us consider a sensor grid where sensors are placed one on each block of the grid. The intruder moves from one block to another in the grid in each time period. The sensors that are kept ON in the grid send their observation on whether the intruder was seen on that block to a central controller. Based on this information, the central controller decides the action to be taken i.e., how many and which sensors to be switched ON during the next time period. This sequential decision making problem can be posed in the framework of POMDP. Note that we consider a simple sensing model in which there is no overlapping of sensor regions.

A Markov Decision Process (MDP) is defined via the tuple $< S, A, P, R >$. Here, S is the set of states that represent the different block positions of the intruder, A is the set of actions corresponding to the number and the configuration of sensors that needs to be switched ON at a given time period, P is the transition probability matrix governing the state evolution where $P(s_k = i, s_{k+1} = j, u_k = a)$ represents the probability dynamics of the intruder, i.e., probability of moving from the current block position, the state $s_k = i$ at time k to the next state $s_{k+1} = j$ at $k+1$ under action a, and $R(s_k = i, s_{k+1} = j, u_k = a)$ corresponds to the single stage cost incurred by taking action a when intruder position at time k is i and the position at $k+1$ is j.

In our setting, whenever the intruder moves to a block in which the sensor is not turned ON the intruder cannot be tracked and the position of the intruder (state) at the next time instant will remain unknown. Thus, it is not possible to solve this problem directly in the framework of MDP and our problem lies in the realm of POMDP as we only have access to partial information about the state as opposed to having complete state information in MDP.

We now formally present the POMDP formulation for the ID problem [3].

- Let n denote the total number of sensors in the network and the actual position of the intruder at time k be denoted by s_k. The stochastic matrix P of dimension $(n+1) \times (n+1)$ models the actual transition probability of the intruder movement between various blocks.
- Action Space : At time k, let u_k denote the vector that indicates the decision on which sensors to be switched on for the next time period. Thus,
$$u_k = (u_{k,l})_{l=1,\dots,n} \in \{0,1\}^n, \tag{1}$$
where 0 denotes the action to keep the sensor OFF and 1 denotes the action to keep the sensor ON. Here, $u_{k,l}$ denotes the action chosen by the lth sensor in the kth time period.
- Observations (o_k): There can be three possibilities. If we track the position of the intruder, then the observation is the state of the system s_k itself. If we do not track the position of the intruder, we denote this by ϵ and if the intruder moved out of the network, we denote this by τ. We assume that if the intruder moves out of the network (i.e., observation = τ) this information gets immediately known to the controller.
- State Space: The belief vector p is an $(n+1) \times 1$ vector, in which $p_k(l)$ indicates the probability of the intruder being at position l at time k. This evolves as follows:
$$\begin{aligned} p_{k+1} =& e_{\tau} \mathbb{1}_{\{s_{k+1}=\tau\}} + e_{s_{k+1}} \mathbb{1}_{\{u_{k+1,s_{k+1}}=1\}} \\ &+ [p_k P]_{\{j: u_{k+1,j}=1\}} \mathbb{1}_{\{u_{k+1},s_{k+1}=0\}}, \end{aligned} \tag{2}$$
where e_i is the vector that represents 1 at position i and 0 at other positions. The notation $[V]_S$ represents the probability vector obtained by setting all components V_i such that $i \in S$ to 0 and then normalizing this vector so that the sum of the components in V is 1.
- We define the single-stage cost model for the ID problem using two cost components. The system incurs a unit cost, if we do not track the position of intruder in a given time period and 0, if we track the position. Let us define
$$T(s_k, u_k, s_{k+1}) = \mathbb{1}_{\{u_{k,s_{k+1}}=0\}}, \tag{3}$$
where $\mathbb{1}(.)$ is the indicator function. This cost $T(s_k, u_k, s_{k+1})$ captures whether intruder is tracked or not in the kth period. The other cost component $C(s_k, u_k, s_{k+1})$ counts the number of sensors that are kept awake in the kth period, i.e.,
$$C(s_k, u_k, s_{k+1}) = \sum_{l=1}^{n} \mathbb{1}_{\{u_{k,l}=1\}}. \tag{4}$$
- The long-run objective is to minimize the average expected tracking error while satisfying the limit on the number of sensors to be ON (energy budget constraints):
$$\min_{\{u_{k,k\geq 0}\}} \lim_{m\to\infty} \frac{\mathbb{E}[\sum_{k=0}^{m-1} T(s_k, u_k, s_{k+1})]}{m} \tag{5}$$
subject to
$$\lim_{m\to\infty} \frac{\mathbb{E}[\sum_{k=0}^{m-1} C(s_k, u_k, s_{k+1})]}{m} \leq b, \tag{6}$$
where $\mathbb{E}[\cdot]$ denotes the Expectation over state transitions and b specifies the energy budget for a time period.
- Relaxed Cost Function: We modify the single-stage cost

function to include the constraint as follows:

$$g(s_k, u_k, s_{k+1}) = T(s_k, u_k, s_{k+1}) \\ + \lambda \times C(s_k, u_k, s_{k+1}), \qquad (7)$$

where $\lambda \in [0, 1]$ is a suitable threshold for tracking error and budget constraints.

III. Our ID Algorithms

In this section, we develop algorithms that obtain action sequences for minimizing average tracking error (see (5)) respecting the budget constraints (see (6)). We first describe a simple greedy algorithm ID_TG and the MCTS algorithm ID_MCTS. We then describe our ID_γ_MCTS RL algorithm that combines the advantages of both the algorithms. For lack of space, we briefly discuss our algorithms and for more details, we refer to our technical report [11].

A. ID_TG Algorithm

This is a greedy algorithm in which the top probable sensor positions the intruder might move to will be turned ON for tracking during each time period. The number of sensors to be kept ON is decided based on a parameter $\gamma \in (0, 1)$. The first step in this algorithm involves obtaining the approximate belief vector for the next time period. Note the belief vector gives the probability of the intruder reaching various possible positions in the next time period. We then select the minimum number of positions in the belief vector whose sum is at least γ in the order of decreasing probability values. This algorithm is easy to implement and at the same time solves the problem of state and action space explosions. However, as γ is constant and does not account for the structure of belief vector, its performance is poor (see Section IV).

B. ID_MCTS Algorithm

Monte Carlo Tree Search (MCTS) has gained a lot of attention in recent times and employed in many applications like the game of Go, Chess. The idea behind this algorithm is that the Monte-Carlo simulations are run from the current state by constructing a tree in a sequential best first search order. The value function is then approximated by averaging the rewards obtained during the search. In the tree, the nodes represent the states and edges represent the selected actions. While constructing the tree, the selection rule for choosing actions called Upper Confidence Bound for Trees (UCT) is employed to perform the best-first search.

In [6], MCTS for the general POMDP setting has been proposed. For the ID POMDP setting, we apply the MCTS algorithm and denote this as ID_MCTS. At each iteration based on the belief vector, the position of the intruder needs to be estimated for simulator to generate the next state. We can estimate this from the belief vector in two ways. The first method involves sampling a position from the belief vector according to its distribution at each iteration. The second method selects the position with maximum probability at each iteration. The second approach leads to faster search and thus, we employ this in our algorithm.

This algorithm solves the problem of state explosion. However, the algorithm requires all the (exponential) number of actions to be tried out a sufficient number of times before timeout. There are a total of 2^n actions where n is the total number of sensor positions. As the size of the network increases, the action space exponentially grows and it takes large amounts of time to explore all the actions due to which the value function estimates for such actions are poor [11].

C. ID_γ_MCTS Algorithm

In the ID_γ_MCTS algorithm, we reformulate the action space to a predefined set of γ values in the range 0 to 1 (for instance, with a uniform gap of 0.05). This set of values plays the same role as the parameter γ in the ID_TG algorithm. Note that the number of possible actions for this algorithm is constant unlike the exponential actions in the ID_MCTS algorithm. The details of this algorithm are given in Algorithm 1. In step 2 of Algorithm 1, we construct a Monte-Carlo tree with belief vector as nodes and the gamma values as edges. The value functions of belief-action pairs are updated based on the received rewards while constructing the tree (see [6]). We choose the optimal γ as the one that minimizes the value function estimate of the current belief p. In this way, the γ value is selected dynamically as opposed to a fixed γ in the ID_TG algorithm. Note that all actions are chosen a sufficient number of times . Now it is enough to know how many sensors to be kept awake in terms of γ instead of the explicit exact configuration of the sensors. In step 3, using the optimal γ value, we switch ON the top probable sensor positions in the approximate belief vector for the next time period as in the ID_TG algorithm. This algorithm imbibes ideas from both ID_TG and ID_MCTS and it does not suffer from the problems of state and action explosions. In [6], it has been shown that given the true belief state, the MCTS algorithm converges to the optimal solution. In our ID_γ_MCTS algorithm, as the belief computations are exact this algorithm converges to the optimal solution.

Algorithm 1 ID_γ_MCTS

```
1: while Object is within the Network do
2:     $\gamma$ = get_gamma(p) - Constructs a tree with belief $p$ as the initial node, gamma values as edges and immediate cost given by (7). At the end of this step, we obtain the best gamma value for $p$.
3:     $\mathbf{a_k}$ = get_action($\mathbf{p_k}, \gamma$) - selects the top sensor positions using the gamma value as described in $ID\_TG$. These sensors are switched on in the next time period.
4:     $\mathbf{s_{k+1}}$ = next_position($\mathbf{s_k}$) - Intruder moves to a new position.
5:     $\mathbf{p} \leftarrow$ Update belief based on the received observation as in (2)
6: end while
```

IV. Experiments and Results

We considered two different configurations of the sensor network. In our first setting, we considered a 1-dimensional sensor network with 41 sensors with probability transition matrix being the same as the one considered in [3]. At the start of the experiment, the intruder is placed at the center of the network with the movement constraint that it can either move 3 positions left or 3 positions right from it's current position. In the second setting, we ran our algorithms on a two-dimensional sensor grid of dimension 16×16. The feasible movements of the intruder are left, right, down, up and along the diagonals.

In the plots, the X-axis corresponds to the average number of sensors awake which is computed as the ratio of the total number of sensors switched ON during the run of the algorithm and the total number of time periods. The Y-axis corresponds to the average tracking error which is obtained as the ratio of the number of time periods in which the intruder is not tracked and the total number of time periods. For the ID_γ_MCTS algorithm, we chose the set of γ values by discretizing the interval $[0,1]$ in steps of 0.05. For the ID_TG algorithm, we obtain these values by fixing different discretized γ values between 0 and 1.

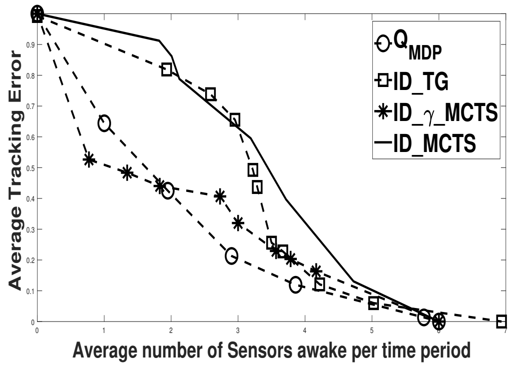

Fig. 1. Performance comparison of algorithms on 1D sensor network with 41 sensors

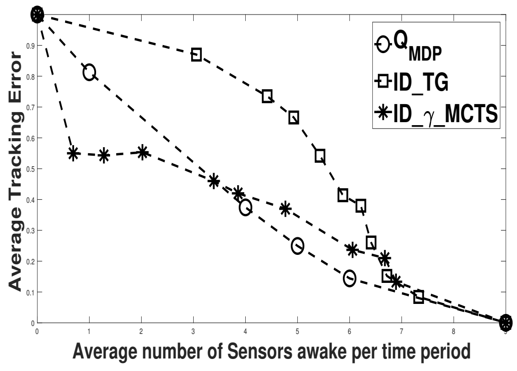

Fig. 2. Performance comparison of algorithms on 2D sensor network of size 16×16

We obtain the plots in the following manner: We run each algorithm with different discretized values of $\lambda \in [0,1]$ in the cost formulation. For each λ, we obtain the pair of average tracking error and the average number of sensors awake per time period. Note that each algorithm could attain the same average number of sensors at different λ values. In our setting, λ acts as a handle for obtaining different pairs of these values. Therefore, even though Q_MDP with the unrealistic assumption of "observation after control" achieves minimum long-run cost for a fixed λ value among the other algorithms, it is not guaranteed to yield minimum tracking error for a given energy budget as seen in Figs.1 and 2. For lack of space, the table of $\lambda-$values is shown in [11] and not here.

As ID_MCTS has to explore exponential number of actions its performance is poor (see Fig. 1). On the other hand, as the action space of ID_γ_MCTS is independent of the size of state space but rather depends mainly on the hardness of the POMDP, it performs well. This can be seen as (in the larger 16×16 grid setting) the performance of ID_γ_MCTS does not get affected (as is not the case with other algorithms). As discussed earlier, since ID_TG, uses a constant γ its performance is poor (see Figs. 1 and 2). In summary, we can conclude that ID_γ_MCTS without making any strong assumptions performs and scales well.

V. Conclusion

In this work, we proposed an RL algorithm for the problem of intruder detection under energy budget constraints. Our algorithm (ID_γ_MCTS) totally solves the problem of both the state and action space explosion. Furthermore, we observed that the ID_γ_MCTS algorithm performs well when the network size becomes large without making any assumption on the intruder movement or the intruder position information.

Acknowledgments

The authors thank the anonymous reviewers for their helpful comments. This work was partly supported through a project under the Joint DRDO-IISc Frontiers Programme.

References

[1] E. K. Chong, C. M. Kreucher, and A. O. Hero, "Partially observable markov decision process approximations for adaptive sensing," *Discrete Event Dynamic Systems*, vol. 19, no. 3, pp. 377–422, 2009.

[2] Y. He and E. K. Chong, "Sensor scheduling for target tracking: A monte carlo sampling approach," *Digital Signal Processing*, vol. 16, no. 5, pp. 533–545, 2006.

[3] G. K. Atia, V. V. Veeravalli, J. Fuemmeler *et al.*, "Sensor scheduling for energy-efficient target tracking in sensor networks," *Signal Processing, IEEE Transactions on*, vol. 59, no. 10, pp. 4923–4937, 2011.

[4] T. Wang, Z. Peng, J. Liang, S. Wen, M. Z. A. Bhuiyan, Y. Cai, and J. Cao, "Following targets for mobile tracking in wireless sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 12, no. 4, p. 31, 2016.

[5] J. Pineau, G. Gordon, and S. Thrun, "Anytime point-based approximations for large pomdps," *Journal of Artificial Intelligence Research*, vol. 27, pp. 335–380, 2006.

[6] D. Silver and J. Veness, "Monte-carlo planning in large POMDPs," in *Advances in neural information processing systems*, 2010, pp. 2164–2172.

[7] P. Poupart and C. Boutilier, "Vdcbpi: an approximate scalable algorithm for large pomdps," in *Advances in Neural Information Processing Systems*, 2005, pp. 1081–1088.

[8] D. Maniloff and P. J. Gmytrasiewicz, "Hybrid value iteration for pomdps," in *FLAIRS Conference*, 2011.

[9] L. Prashanth, A. Chatterjee, and S. Bhatnagar, "Two timescale convergent q-learning for sleep-scheduling in wireless sensor networks," *Wireless networks*, vol. 20, no. 8, pp. 2589–2604, 2014.

[10] M. T. Spaan and N. Vlassis, "Perseus: Randomized point-based value iteration for POMDPs," *Journal of artificial intelligence research*, vol. 24, pp. 195–220, 2005.

[11] R. B. Diddigi, K. J. Prabuchandran, and S. Bhatnagar, "Novel Sensor Scheduling Scheme for Intruder Tracking in Energy Efficient Sensor Networks," *ArXiv e-prints, 1708.08113*, Aug. 2017.