Rollout Strategy for Hidden Markov Model (HMM)-based Dynamic Sensor Scheduling

Hyunsung Lee, Satnam Singh, Woosun An, Swapna S. Gokhale, Krishna Pattipati and David L. Kleinman

Abstract— In this paper, a hidden Markov model (HMM)-based dynamic sensor scheduling problem is formulated, and solved using rollout concepts to overcome the computational intractability of the dynamic programming (DP) recursion. The problem considered here involves dynamically sequencing a set of sensors to minimize the sum of sensor cost and the HMM state estimation error cost. The surveillance task is modeled as a single HMM with multiple emission matrices corresponding to each of the sensors. The rollout information gain (RIG) algorithm proposed herein employs the information gain (IG) heuristic as the base algorithm. The RIG algorithm is illustrated on an intelligence, surveillance, and reconnaissance (ISR) scenario of a village for the presence of weapons and terrorists/refugees. Extension of the RIG strategy to monitor multiple HMMs involves combining the information gain heuristic with the auction algorithm that computes the \( \kappa \)-best assignments at each decision epoch of rollout.

I. INTRODUCTION

A. Motivation

The dynamic sensor scheduling problem occurs in a number of applications, including target tracking, sensor networks, unmanned aerial vehicles for surveillance in remote or hostile environments, robotics, and supply chain management in a warehouse. In a target tracking application, sensors are dynamically activated to trade off tracking performance with the sensor usage cost [1]. For example, consider a target identification scenario where an incoming aircraft needs to be identified as an enemy or as a friendly target using various forms of sensors available at a base station [2]. This scenario requires sensor scheduling because the better sensors (e.g., radar) tend to reveal or to provide clues about the location of the aircraft relative to the base station, whereas the more stealthy sensors tend to be inaccurate [2].

In this paper, we motivate the sensor scheduling problem to support an intelligence, surveillance, and reconnaissance (ISR) officer in an expeditionary strike group (ESG) mission in coordinating the use of surveillance assets (sensors) to improve situational awareness [3]. An ESG provides a flexible force package, capable of tailoring itself to a wide variety of mission sets [3]. An important ESG mission involves dealing with asymmetric threats, such as terrorist groups who carry out attacks while trying to avoid direct confrontation. Terrorist groups are elusive, secretive, amorphously structured and decentralized entities that often appear unconnected. This stealthy behavior makes it very difficult to predict when and where they will strike. Moreover, the increased geographical range and unpredictable nature of this behavior require effective allocation and appropriate scheduling of sensors to achieve mission objectives. Effectively performing the ISR activities is a key step to gain situational awareness, which, in turn, enables the allocation of resources for the interdiction of terrorist activities.

We model the asymmetric threats using hidden Markov models, because these activities are concealed and their true state can be only inferred through the observations obtained using various ISR sensors. A pattern of these observations and its dynamic evolution over time is a potential realization of an asymmetric threat [4]. Performing the ISR activities requires multiple sensors to provide observations needed for accurately estimating the status of suspicious activities. The available sensors are limited in number, and possess different attributes (e.g. cost, deployment requirements, accuracy, capabilities, etc.) requiring judicious sensor allocation over time. For example, unmanned aerial vehicles (UAVs) are preferred assets for monitoring nearly all the ISR activities; however, they cannot be deployed in large numbers due to their limited availability. Therefore, an effective scheduling of ISR sensors over time is essential for accurate situation assessment and to the success of the overall mission.

B. Previous Work

Sensor scheduling problem has been widely studied in the area of target tracking (e.g. [1], [2]). In [5], Singh et al. provided a summary of previous research on sensor scheduling for targets, whose dynamics are modeled by linear Gauss-Markov processes. They formulated the sensor scheduling problem as one of minimizing the variance of the estimation error of the hidden states of a controlled HMM with respect to a given action sequence [5]. The authors proposed a stochastic gradient algorithm to determine the optimal schedule for a HMM where the state, observation and action spaces are continuous. Another effort, related to our
work, using the discrete HMM framework was presented by Krishnamurthy in [2]. In this case, the author proposed a stochastic dynamic programming (DP) framework to solve the sensor scheduling problem. Logothetis et al. [6] formulated the sensor scheduling problem as computing a sequence of active sensors to maximize the information on the state of the system.

Rollout algorithms were first proposed for the approximate solution of dynamic programming recursions by Bertsekas et al. in [7], [8]. They are a class of suboptimal solution methods inspired by the policy iteration of dynamic programming and the approximate policy iteration of neuro-dynamic programming. The rollout algorithm combined with the information gain heuristic (RIG) was first proposed in our previous research on sequential fault diagnosis [9]. In [9], we showed that rollout strategy, which can be combined with the one-step or multi-step lookahead heuristic algorithms as base algorithms, can solve test sequencing problems in real-world systems with a higher computational efficiency than the optimal strategies, while being superior to those using the base algorithms only. In this paper, we apply the RIG algorithm to solve the dynamic sensor scheduling problem.

II. PROBLEM FORMULATION

Consider a single ISR task modeled as a HMM. The problem is one of allocating a sensor at each epoch, from among the sensors available at that epoch, to minimize the sum of sensor cost and the HMM state estimation error cost. Figure 1 shows an HMM with a hidden state denoted by \( x(k) \) at time epoch \( k \), \( m \) sensors, \( K \) time epochs, and \( L \) observation symbols. The hidden state could be one of the \( n \) states which are denoted by \( s_i, i=1,2,...,n \). Note that the number of observation symbols \( L \) can be a function of the selected sensor \( u(k) \), i.e., \( L(u(k)) \). This models a realistic scenario that different sensors have different capabilities in generating the observation symbols. The HMM states can be inferred via the observations acquired using the sensors. At time epoch \( k \), a subset of \( m \) sensors \( \mu(k) \in \{1,2,...,m\} \) is available from which a single sensor should be selected for observing the hidden state. An observation at time epoch \( k \) is denoted by \( y(k) \), and the observation belongs to one of \( L(u(k)) \) symbols \( y(k) \in \{O_1(k),...,O_{L(u(k))}(k)\} \) where \( u(k) \in \mu(k) \) is the sensor selected at time epoch \( k \). Our HMM is parameterized by \( \Lambda = (A(k),B(k),\varphi) \) where

\[
A(k) = [a_{ij}(k)] = [P(x(k) = s_{j} \mid x(k-1) = s_{i})], \quad (i,j = 1,2,...,n) \tag{1}
\]

is the transition probability matrix of the underlying Markov chain,

\[
B(k) = [b_{iq}(k)] = [P(y(k) = O_i(k) \mid x(k) = s_i, u(k) = q)], \quad (i = 1,2,...,n; l = 1,2,...,L; q = 1,2,...,m) \tag{2}
\]

is the set of emission matrices (also known as confusion and observation matrices) for the \( m \) sensors, and

\[
\varphi = \{\varphi_i = p(x(1) = s_i), (i = 1,2,...,n) \}
\]

is the initial probability distribution of the underlying Markov states at time \( t = 1 \). We assume that the transition probability matrix \( A(k) \), the set of emission matrices \( B(k) \), and the initial probabilities \( \varphi \) are known. These probabilities could also be estimated based on historical data using the Baum-Welch algorithm [10].

The key point here is that the observation \( y(k) \) depends upon the current state and the selected sensor \( u(k) \) from among the available sensors at time \( k, \mu(k) \). At time epoch \( k \), we have \( \{Y^{k-1},U^{k-1}\} \), where \( Y^{k-1} = \{y(1),...,y(k-1)\} \) and \( U^{k-1} = \{u(0),u(1),...,u(k-1)\} \), the previously observed symbols and the sensor sequence used from time epoch \( t = 1 \) to time epoch \( t = k-1 \). Thus, the sensor scheduling problem is: How to optimally select a sensor \( u(k) \) from the available sensor set at time epoch \( k, \mu(k) \) based on \( \{Y^{k-1},U^{k-1}\} \), the information available at time epoch \( k \)? Our criterion for optimality involves minimizing a cost function, which is a sum of sensor costs and the state estimation error costs over time.

The information state \( \pi(k \mid k-1) \) is a sufficient statistic to describe the current state of an HMM [2]. It can be also interpreted as the probability of the hidden state \( x(k) \) given the information, \( \{Y^{k-1},U^{k-1}\} \), i.e.

\[
\pi(k \mid k-1) = P(x(k) \mid Y^{k-1},U^{k-1}) \tag{4}
\]

The scheduling cost from time epoch 1 to \( K \) is denoted by \( J_{\mu} \)

\[
J_{\mu} = E\left[ f_{\mu}(\pi(K \mid K)) + \sum_{k=1}^{K-1} \left[ f_{\mu}(\pi(k \mid k)) + g_{\mu}(u(k),\pi(k \mid k-1)) \right] \right] \tag{5}
\]

where \( \pi(k \mid k) \) is the updated information state, \( f_{\mu}(\pi(k \mid k)) \) is the state estimation error, and \( g_{\mu}(u(k),\pi(k \mid k-1)) \) is the sensor cost. Typical cost functions for the state estimation error are as follows:

\[
f_{\mu}(\pi(k \mid k)) = 1 - \pi^2(k \mid k) \pi(k \mid k), \tag{6}
\]
$$f_k(\pi(k | k)) = 1 - \max_i \pi_i(k | k),$$  \hspace{1cm} (7)  

$$f_k(\pi(k | k)) = \min_{i \in \mathcal{A}} \sum_{t=0}^{\infty} \pi_f(k | k) \lambda_{ij}.$$  \hspace{1cm} (8)  

The first criterion can be interpreted as an $L_2$ norm of the state estimation error, the second as the error probability of a maximum a posteriori probability-based decision rule, and the third as the expected cost of errors in estimating the information state, respectively. In (8), $\lambda_{ij}$ represents the cost of erroneously estimating the hidden state as $s_j$ when the true state is $s_i$. The sensor cost is sum of sensor usage cost and sensor moving cost i.e.

$$g_k(u(k), \pi(k | k - 1)) = h_k(u(k), \pi(k | k - 1)) + c_m(u(k))$$  \hspace{1cm} (9)  

where the sensor usage cost is,

$$h_k(u(k), \pi(k | k - 1)) = \sum_{t=1}^{T} \sum_{i=1}^{L} c_i(s_i, u(k)) \pi_i(k | k - 1)$$  \hspace{1cm} (10)  

and $u^T_k(u(k)) = (c(s_1, u(k)), c(s_2, u(k)), ..., c(s_n, u(k)))$ is the usage cost of sensor $u(k)$ associated with each HMM state $s_i$. We also considered the cost of moving a sensor $c_m(u(k))$, which is computed for sensor $u(k)$ as:

$$c_m(u(k)) = \frac{\| (a_{u_0}, a_{u_1}, a_{u_2}) - (a_{u_0}, a_{u_1}, a_{u_2}) \|_2}{v(u(k))}$$  \hspace{1cm} (11)  

where $(a_{u_0}, a_{u_1}, a_{u_2})$ and $(a_{u_0}, a_{u_1}, a_{u_2})$ denotes the cartesian coordinates of the sensor monitoring place and the selected sensor respectively. Here, $w(u(k))$ indicates the priority and $v(u(k))$ denotes the velocity of the sensor.

The optimum solution to the sensor scheduling problem is the sensor sequence $q^* = U^* = \min_{\{u(k) \mid u(k) \in \mathcal{A} \}} J_u$. Formally,  

$$q^* = U^* = \min_{\{u(k) \mid u(k) \in \mathcal{A} \}} \{ J_u \}$$  \hspace{1cm} (12)  

The optimal solution for (5) can be obtained using the dynamic programming (DP) technique; however, the computational complexity is $O(D^{n-1}mnLK)$ , which is exponential in $D$, the number of quantization levels used to discretize the continuous-valued information state. Here, $n$ is the number of states in a HMM, $m$ is the number of sensors, $L$ is the maximum number of observation symbols, and $K$ is the number of time epochs. Hence, the DP technique is impractical for an HMM containing more than around 15 states [2]. This motivates us to investigate suboptimal algorithms designed to solve the sensor scheduling problem containing hundreds of HMM states. We propose the rollout information gain (RIG) algorithm with computational complexity of $O(mnLK)$ per rollout, which is significantly lower than the DP technique.

![Fig. 2. Rollout information gain (RIG) algorithm](image)

**Step 1:** Predict the information state using (15) as described in Section IV.

**Step 2:** Compute the information gain per unit cost of a sensor given in (13) for all $m$ sensors at time $k$.

**Step 3:** Select top $\kappa$ sensors which maximize the information gain per unit cost given in (13). Usually, $\kappa$ is in the range $[2,5]$ to avoid excessive rollouts.

**Step 4:** Compute the total scheduling cost as defined in (5) for the top $\kappa$ sensors. Select the sensor which minimizes the cost as given by (12).

**Step 5:** Update time epoch $k \leftarrow k + 1$ and go back to Step 1 until $k = K$.

**Fig. 3. Pseudo code for the rollout information gain (RIG) algorithm**

**III. ROLLOUT ALGORITHM**

Rollout algorithms are a class of suboptimal methods which are capable of improving the effectiveness of any given heuristic through sequential application. This is due to the policy improvement mechanism of the underlying policy iteration process [9]. They can be viewed as a single step of the classical policy iteration method, wherein we start from a given easily implementable and computationally tractable policy, and then try to improve on that policy using online learning and simulation. The attractive aspects of rollout algorithms are simplicity, broad applicability, and suitability for online implementation. The details of the rollout algorithms are provided in [7]-[9].

To solve the sensor scheduling problem, we combine the rollout method with the information gain heuristic. Figure 2 graphically illustrates the RIG algorithm with two rollouts at each time epoch. Figure 3 shows the pseudo code of the rollout information gain (RIG) algorithm.

**IV. COMBINING ROLLOUT ALGORITHM WITH INFORMATION GAIN HEURISTIC**

The information heuristic selects a sensor $q^* = u^*(k)$ in the predicted information state $\pi_i(k | k - 1)$ that maximizes
the information gain (IG) per unit cost of a sensor:

\[ q^* = u^*(k) = \arg \max_{q \in q(k)} \frac{I(\pi(k | k-1), u(k) = q)}{g_s(u(k), \pi(k | k-1))} \]  \hspace{1cm} (13)

where \( g_s(u(k), \pi(k | k-1)) \) is the sensor cost as defined in (9) and \( I(\pi(k | k-1), u(k) = q) \) is the information gain given by:

\[ I(\pi(k | k-1), u(k) = q) = \sum_{i=1}^n \pi_i(k | k-1) \sum_{j=1}^n b_{ij}(k) \log_2 b_{ij}(k) - \sum_{i=1}^n \left( \sum_{j=1}^n b_{ij}(k) \pi_i(k | k-1) \log_2 \left( \frac{b_{ij}(k)}{\pi_i(k | k-1)} \right) \right). \]  \hspace{1cm} (14)

The derivation of information gain is provided in the Appendix. The information heuristic algorithm is a terminating and sequentially improving one step lookahead procedure [9]. In the context of sensor scheduling problem, the information gain (IG) heuristic algorithm consists of the following four steps (Figure 4):

**Step 1 (State Prediction):** In this step, we predict the information state vector \( \pi(k | k-1) \) for the next sampling epoch \( k \) using the current information state update \( \pi(k-1 | k-1) \) and the transition matrix \( A(k) \) as follows:

\[ \pi(k | k-1) = A^T(k) \pi(k-1 | k-1). \]  \hspace{1cm} (15)

**Step 2 (Sensor Selection):** We compute the information gain \( I(\pi(k | k-1), u(k) = q) \) for all the \( m \) sensors using (14). Select a sensor \( q^* = u^*(k) \) that maximizes the information gain per unit cost as given in (13).

**Step 3 (Observation):** The new observation \( y(k) = O(k) \) is obtained using the sensor \( u^*(k) \) according to the emission probability matrix given in (2).

\[ \pi(k | k-1) = \sum_{j=1}^n b_{ij}(k) \pi_i(k | k-1). \]  \hspace{1cm} (16)

**Step 4 (Observation Update):** The updated state at the next time \( \pi(k | k-1) \) is obtained using the forward algorithm [12] as follows:

\[ \pi(k | k) = \frac{b_{ij}(k) \pi_i(k | k-1)}{\sum_{j=1}^n b_{ij}(k) \pi_j(k | k-1)}. \]

V. SIMULATIONS AND RESULTS

We consider a scenario where an ISR officer needs to dynamically allocate sensors to monitor asymmetric threat activities in a notional area that involves primarily the fictitious countries of Asiland and Bartola [3]. The northern shore of Asiland was struck by a tsunami that caused massive losses to her civilian populace. Large numbers of Asiland refugees fled and crossed the strait to fishing villages and refugee camps in Bartola seeking help and assistance. Insurgents and terrorist factions in Asiland began to take advantage of the situation, move their operations into Bartola by intermingling with the refugees and smuggling weapons onboard fishing boats and merchant ships.

The problem of monitoring for the presence of terrorists and weapons in a fishing village of Bartola is modeled using a four state HMM. Activities such as the presence of terrorists and weapons, and to learn the crowd demeanor (normal or protesters or terrorists) are continually monitored using the following six sensors: SH-60, E2-C, UAV, P3, AV-8B and MEU. As shown in Figure 5, state 1 represents the normal state of crowd in the fishing village. In state 2, refugees move into the village. Terrorists disguised as refugees arrive in the village and they also smuggle weapons into the village; this is modeled as state 3. In state 4, the weapons and terrorists are prosecuted/pacified and the village resumes normalcy, which is modeled by a transition to state 1.

In this scenario, there are several sensors capable of performing ISR and they have different speeds and capabilities. The scenario assumes 4 observation symbols for each sensor; the symbols provide information on the state of threat activity in the fishing village. We use nominal values of the transition probabilities for the sake of illustration, which are shown in Figure 5. The emission probabilities were also set to their nominal values and varied as a function of sensor accuracy variable \( p \) to analyze the robustness of the algorithm for various sensor accuracies. The cost function in (5) employed the state estimation error criterion in (6). The sensor scheduling cost considered in the fishing village scenario is:
The per unit cost is 0.2 and the costs associated with \( K \in \mathbb{R}^+ \) are given by:

\[
J = E[\beta(1 - \pi(K | K))^T \pi(K | K)]
\]

\[
+ \sum_{t=0}^{k-1} \beta(1 - \pi(k | k))^T \pi(k | k)
\]

\[
+ \sum_{k=0}^{K-1} \{c_T^u(u(k)) + c_T^m(u(k))\} \pi(k | k-1) \tag{17}
\]

where \( c_T^u(u(k)) \) and \( c_T^m(u(k)) \) are the costs associated with sensor usage cost and moving the sensor as given in (10) and (11), respectively. For simulations, we set the weighted priority vector \( w^T(u(k)) = [1, 2, 3, 4, 1, 1] \), \( \beta = 10 \), and the total number of time epochs \( K = 5 \). The velocities of the six sensors are set as \( v^T(u(k)) = [200, 450, 300, 100, 500, 80] \). The sensor usage costs are selected as \( c_T^u(u(k)) = [4, 11, 8, 5, 6, 5, 2] \).

Figure 6 shows the total scheduling cost averaged over 1000 Monte Carlo runs. To check the robustness of the algorithm, the cost was obtained by varying the observation probability using the variable \( p \). Here, Sensor 5 or Sensor 6 curves indicate the static scheduling i.e. the cost was obtained by using sensor 5 or sensor 6 for all time epochs. Sensor scheduling via the rollout information gain algorithm (RIG) has approximately 9–10% lower cost as compared to one using only the information gain heuristic (IG) and 42% lower cost as compared to static scheduling (sensor 6). The dual advantage of using rollout information gain (RIG) algorithm is that it significantly reduces the complexity of dynamic programming, while improving the accuracy over the base heuristic, information gain.

VI. SENSOR SCHEDULING FOR MULTIPLE HMMs

In this section, we will briefly explain how the single HMM sensor scheduling problem can be extended to multiple HMMs. A simple example of sensor scheduling with multiple HMMs involves a scenario where we need to simultaneously monitor the threat activities that are dispersed geographically in a region. Figure 7 shows the sensor scheduling problem with \( m \) sensors and \( N \) surveillance tasks modeled using \( N \) HMMs. The information gain for each sensor-HMM pair is computed, and the sensors are assigned to HMMs using the \( \kappa \)-best version of the auction algorithm [11], [13]. The auction algorithm maximizes the sum of information gains at each time epoch with the constraint that at most one sensor is assigned to each HMM and each sensor is assigned to at most one HMM. We plan to implement this algorithm in our future work.

VII. CONCLUSION

This paper formulated the sensor scheduling problem using the HMM formalism. The optimal solution of the sensor scheduling problem via dynamic programming (DP) is intractable. To overcome the computational intractability of DP, we proposed a rollout information gain (RIG) algorithm, and illustrated its application on a realistic mission scenario involving the monitoring of the threat activities in a fishing village to support an ISR officer. An extension of the RIG algorithm is proposed to solve the sensor scheduling problem involving multiple HMMs by combining it with the auction algorithm.

APPENDIX

Let \( H(x) = -\sum_i p_i(x) \log_2 p_i(x) \) denote the entropy of the state with a probability mass function \( \{p_i(x)\} \). The information state is defined in (4). The entropy of the information state is:

\[
H(\pi(k | k-1)) = -\sum_{i=1}^N \pi_i(k | k-1) \log_2 \pi_i(k | k-1).
\]

The sensor scheduling problem is to select \( u(k) \) such that the information gain \( H(\pi(k | k-1)) - H(\pi(k | k-1) | u(k)) \) per unit cost is maximized. Recall that the joint entropy is given by:
and mutual information (or information gain) is:

$$I(x, u) = H(x) - H(x | u) = H(u) - H(u | x).$$

Using (18) and (19),

$$I(\pi(k | k-1), u(k)) = H(\pi(k | k-1)) - H(\pi(k | k-1) | u(k))$$

and mutual information (or information gain) is:

$$I(\pi(k | k-1), u(k)) = H(u(k)) - H(u(k) | \pi(k | k-1)).$$

where,

$$H(u(k) = q) = -\sum_{i=1}^{L(q)} P(y(k) = O_i(k) | Y^{k-1}, U^{k-1}, u(k) = q) \log_2 P(y(k) | O_i(k), Y^{k-1}, U^{k-1}, u(k) = q),$$

and

$$P(y(k) = O_i(k) | Y^{k-1}, U^{k-1}, u(k) = q) = \sum_{j=1}^{n} P(y(k) = O_i(k) | x(k) = s_j, Y^{k-1}, U^{k-1}, u(k) = q) \pi_j(k | -1),$$

Recall that the conditional entropy is:

$$H(X | Y) = \sum_{y \in Y} P(y) H(X | y).$$

Using (24):

$$H(u(k) | \pi(k | k-1)) = -\sum_{i=1}^{n} \pi_i(k | k-1) \sum_{j=1}^{L(q)} b_{ijq}(k) \log_2 b_{ijq}(k).$$

Using (22) and (25), the information gain is:

$$I(\pi(k | k-1), u(k) = q) = \sum_{i=1}^{n} \pi_i(k | k-1) \sum_{j=1}^{L(q)} b_{ijq}(k) \log_2 b_{ijq}(q)$$

and

$$-\sum_{i=1}^{n} \sum_{j=1}^{L(q)} b_{ijq}(k) \sigma_i(k | k-1)) \log_2 \left( \sum_{j=1}^{L(q)} b_{ijq}(k) \sigma_i(k | k-1) \right).$$

We can also derive the information gain using (20). In this case,

$$H(\pi(k | k-1)) = -\sum_{i=1}^{n} \pi_i(k | k-1) \log_2 \pi_i(k | k-1)$$

and

$$H(\pi(k | k-1) | u(k)) = -\sum_{i=1}^{L(q)} P(y(k) = O_i(k) | Y^{k-1}, u(k) = q) \log_2 P(y(k) = O_i(k) | Y^{k-1}, u(k) = q)$$

Using the forward algorithm [12],

$$P(x(k) = s_j | Y^{k-1}, u(k) = q, y(k) = O_i(k))$$

and

$$P(y(k) = O_i(k) | Y^{k-1}, u(k) = q) = \sum_{j=1}^{L(q)} b_{ijq}(k) \pi_j(k | k-1).$$

Inserting (29) and (30) in (28), we get:

$$H(\pi(k | k-1) | u(k) = q) = -\sum_{j=1}^{L(q)} \sum_{i=1}^{n} b_{ijq}(k) \pi_j(k | k-1) \log_2 b_{ijq}(k)$$

and

$$-\sum_{i=1}^{n} \pi_i(k | k-1) \log_2 \pi_i(k | k-1)$$

and

$$\frac{\log_2 \left( \sum_{j=1}^{L(q)} b_{ijq}(k) \pi_j(k | k-1) \right)}{\sum_{j=1}^{n} b_{ijq}(k) \pi_j(k | k-1)}.$$