Online Policy Improvement in Large POMDPs via an Error Minimization Search

Stéphane Ross STEPHANE.ROSS@MAIL.MCGILL.CA
Joelle Pineau JPINEAU@CS.MCGILL.CA
School of Computer Science
McGill University, Montreal, Canada, H3A 2A7

Brahim Chaib-draa CHAIB@DAMAS.IFT.ULAVAL.CA
Department of Computer Science and Software Engineering
Laval University, Quebec, Canada, G1K 7P4

Abstract

Partially Observable Markov Decision Processes (POMDPs) provide a rich mathematical framework for planning under uncertainty. However, most real world systems are modelled by huge POMDPs that cannot be solved due to their high complexity. To palliate to this difficulty, we propose combining existing offline approaches with an online search process, called AEMS, that can improve locally an approximate policy computed offline, by reducing its error and providing better performance guarantees. We propose different heuristics to guide this search process, and provide theoretical guarantees on the convergence to ϵ-optimal solutions. Our experimental results show that our approach can provide better solution quality within a smaller overall time than state-of-the-art algorithms and allow for interesting online/offline computation tradeoff.

1. Introduction

Partially Observable Markov Decision Processes (POMDPs) provide a powerful model for sequential decision making under uncertainty. A POMDP allows one to model uncertainty related to both the agent's actions and observations about the state of the world. However, most real world applications have huge state space and observation space, such that exact solving approaches are completely intractable (finite-horizon POMDPs are PSPACE-complete (Papadimitriou & Tsitsiklis, 1987) and infinite-horizon POMDPs are undecidable (Madani et al., 1999)).

Most of the recent research in the area has focused on developing approximate offline algorithms that can find approximate policies for larger POMDPs (Braziunas & Boutilier, 2004; Pineau et al., 2003; Poupart, 2005; Smith & Simmons, 2005; Spaan & Vlassis, 2005). Still, successful application of POMDPs to real world problems has been limited due to the fact that even these approximate algorithms are intractable in the huge state space of real world systems. One of the main drawback of these offline approaches is that they need to compute a policy over the whole belief state space. In fact, a lot of these computations are generally not necessary since the agent will only visit a small subset of belief states when acting in the environment. This is the strategy online POMDP algorithms tries to exploit (Satia & Lave, 1973; Washington, 1997; Geffner & Bonet, 1998; McAllester & Singh, 1999; Paquet et al., 2006; Ross & Chaib-draa, 2007). Online algorithms proceed by planning only for the current belief state of the agent during the execution. Consequently, one needs only to compute the best action to do in this belief state, considering the subset of belief states that can be reached over some finite planning horizon.

In this paper, we propose a new anytime online search algorithm which aims to reduce, as efficiently as possible, the error made by approximate offline value iteration algorithms. Our algorithm can be combined with any approximate offline value iteration algorithm to refine and improve locally the approximate policies computed by such algorithm. We propose various heuristics, based on an error analysis of lookahead

Presented at *North East Student Colloquium on Artificial Intelligence (NESCAI)*, 2007.

search, that aim to guide the search toward reachable belief states that can most reduce the error. We also provide theoretical guarantees that these heuristics are admissible, in the sense that they lead to ϵ-optimal solutions within finite time. Finally, we compare empirically our approach with different online and offline approaches in different domains and show that our approach provides good solution quality within a smaller overall time.

2. Background: POMDP

A POMDP is generally defined by a tuple $(S, A, Z, T, R, O, b_0, \gamma)$ where S is a set of states, representing the possible states of the environment, A is the action set, representing the possible actions the agent can take in the environment, Z is the observation set, representing the possible observations the agent can make in the environment, $T : S \times A \times S \rightarrow [0, 1]$ is the transition function where $T(s, a, s')$ specifies the probability of moving to a certain state s', given that we were in state s and did action a, $R : S \times A \rightarrow \Re$ is the reward function where $R(s, a)$ specifies the immediate reward obtained by doing action a in state s, $O : S \times A \times Z \rightarrow [0, 1]$ is the observation function where $O(s', a, z)$ specifies the probability of observing a certain observation z, given that we did action a and moved to state s', b_0 is the initial belief state of the agent, which represents its belief about the initial state of the environment, and γ is the discount factor.

In a POMDP, the agent does not know exactly in which state it currently is, since its observations on the current state are uncertain. Instead the agent maintains a belief state b which is a probability distribution over all states that specifies the probability that the agent is in each state, given the complete sequence of action and observation it has made so far. After the agent performs an action a and perceives an observation z, the agent can update its current belief state b using the belief update function $\tau : \Delta S \times A \times Z \rightarrow \Delta S$ specified by the following equation.

$$b^{a,z}(s') = \eta O(s', a, z) \sum_{s \in S} T(s, a, s') b(s) \qquad (1)$$

Here, $b^{a,z}$ is the new belief state returned by $\tau(b, a, z)$ and η is a normalization constant such that the new probability distribution over all states sums to 1.

Solving a POMDP consists in finding an optimal policy π^* which specifies the best action to do in every belief state b. This optimal policy depends on the planning horizon and on the discount factor used. In order to find this optimal policy, we need to compute the optimal value of a belief state over the planning horizon. For the infinite horizon, the optimal value function is the fixed point of the following equation.

$$V^*(b) = \max_{a \in A} \left[R(b, a) + \gamma \sum_{z \in Z} P(z|b, a) V^*(\tau(b, a, z)) \right] \qquad (2)$$

In this equation, $R(b, a) = \sum_{s \in S} R(s, a) b(s)$ is the expected immediate reward of doing action a in belief state b and $P(z|b, a)$ is the probability of observing z after doing action a in belief state b. This probability can be computed according to.

$$P(z|b, a) = \sum_{s' \in S} O(s', a, z) \sum_{s \in S} T(s, a, s') b(s) \qquad (3)$$

We also denote the value $Q^*(b, a)$ of a particular action a in belief state b, as the long term reward we will obtain if we perform a in b and then follow the optimal policy:

$$Q^*(b, a) = R(b, a) + \gamma \sum_{z \in Z} P(z|b, a) V^*(\tau(b, a, z)) \qquad (4)$$

Using this definition, we can define the optimal policy π^* as follows.

$$\pi^*(b) = \arg\max_{a \in A} Q^*(b, a) \qquad (5)$$

However, since there is an infinite number of belief states, it would be impossible to compute such a policy for all belief states. But, since it has been shown that the optimal value function of a finite-horizon POMDP is piecewise linear and convex, we can define the optimal value function and policy of a finite-horizon POMDP using a finite set of $|S|$-dimensional hyper plan, called α-vector, over the belief state space. This is how exact offline value iteration algorithms are able to compute V^* in a finite amount of time. However, exact value iteration algorithms can only be applied to small problems of 10 to 20 states due to their high complexity. For more detail, refer to Littman and Cassandra (Littman, 1996; Cassandra et al., 1997).

Contrary to exact value iteration algorithms, approximate value iteration algorithms try to keep only a subset of α-vectors after each iteration of the algorithm in order to limit the complexity of the algorithm. This

is generally achieved by optimizing the value function only for a small finite subset of chosen belief points (Pineau et al., 2003; Spaan & Vlassis, 2005; Smith & Simmons, 2005), thus limiting the number of α-vectors to the number of belief points. The precision of these algorithms depend on the number of belief points and their location in the belief state space.

3. Online Search in POMDPs

Contrary to offline approaches, which compute a complete policy determining an action for every belief state, an online algorithm takes as input the current belief state and returns the single action that seems to be the best for this particular belief state. The advantage of such an approach is that it only needs to consider the most important reachable belief states from the current belief state to plan the next action. However, it may take a lot of time choosing an action to ensure some optimality guaranties.

Lookahead search algorithms in POMDP generally proceed by exploring the tree of reachable belief states from the current belief state, by considering the different sequence of actions and observations. In this tree, belief states are represented as OR-nodes (we must choose an action) and actions are represented as AND-nodes (we must consider all possible observations). This tree structure is used to determine the value of the current belief state and the best action to do in this belief state. The values of the actions and belief states in the tree are evaluated by propagating the fringe belief state values to their ancestors, according to equation 2. An approximate value function is generally used at the fringe of the tree to approximate the infinite-horizon value of these fringe belief states. In our case, we will be interested in using a lower bound and an upper bound on the value of these fringe belief states, as it will allow us to conduct branch & bound pruning in the tree. In this case, the lower and upper bounds are propagated to parent nodes according to the following equations:

$$U_T(b) = \begin{cases} U(b) & \text{if } b \text{ is a leaf in } T \\ \max_{a \in A} U_T(b,a) & \text{otherwise} \end{cases} \quad (6)$$

$$U_T(b,a) = R(b,a) + \gamma \sum_{z \in Z} P(z|b,a) U_T(\tau(b,a,z)) \quad (7)$$

$$L_T(b) = \begin{cases} L(b) & \text{if } b \text{ is a leaf in } T \\ \max_{a \in A} L_T(b,a) & \text{otherwise} \end{cases} \quad (8)$$

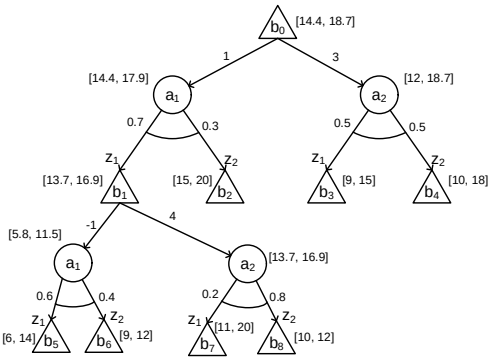

Figure 1. Example of an AND-OR tree constructed by a lookahead search algorithm.

$$L_T(b,a) = R(b,a) + \gamma \sum_{z \in Z} P(z|b,a) L_T(\tau(b,a,z)) \quad (9)$$

Here, $U_T(b)$ and $L_T(b)$ represent the upper and lower bound on $V^*(b)$ associated to belief state b in the tree T and $U_T(b,a)$ and $L_T(b,a)$ represent the upper and lower bound on $Q^*(b,a)$ for the corresponding action node in the tree T. We use $L(b)$ and $U(b)$ to represent the lower bound and upper bound computed offline.

An example AND-OR tree constructed by such search process is presented in figure 1.

In this figure, the belief states OR-nodes are represented by triangular nodes and the action AND-nodes by circular nodes. The rewards $R(b,a)$ are represented by values on the outgoing arcs from OR-nodes and probabilities $P(z|b,a)$ by probabilities on the outgoing arcs from AND-nodes. The values inside brackets represent the lower and upper bounds that have been computed according to equations 6, 7, 8, 9 and, using a discount factor $\gamma = 0.95$.

One motivation for doing an online search is that it can reduce the error made by the approximate value function used at the fringe. It has been shown by Puterman (Puterman, 1994) that a complete k-step lookahead multiplies the error bound on the approximate value function used at the fringe by γ^k, and therefore reduces the error bound for $\gamma \in [0,1)$ (this is also a consequence of theorem 1 presented in the next section). However, since a k-step lookahead search has a complexity exponential in k, it motivates the need for more efficient online algorithms that can guarantee similar or better error bounds.

One intuitive reason why a k-step lookahead may be inefficient is that it may explore belief states that have

very small probabilities to occur and that therefore, does not influence a lot the value function (see equation 2). Furthermore, a k-step lookahead may also explore nodes that have very small error, which will not bring much more precision to our approximation. Finally, a k-step lookahead will also explore nodes that are reached by suboptimal actions that do not influence V^* (suboptimal actions can be detected using lower and upper bounds, i.e. a is guaranteed to be suboptimal when $U_T(b,a) \leq L_T(b)$). Hence, in all these cases, it would be more efficient to guide the search toward nodes that contributes more to the value function in order to reduce their error.

These observations lead us to the fact that, if we want a more efficient online search algorithm, then it should optimize the error reduction on the current belief state, i.e. it should try to minimize the error on the current belief state as quickly as possible, such as to guarantee performance as close as possible to the optimal policy. In the next section, we will present an error analysis of lookahead search that will allow us to derive heuristics to achieve this goal.

4. Anytime Error Minimization Search

As we have mentioned before, our new approach, called AEMS, for Anytime Error Minimization Search, adopts an error minimization point of view by trying to reduce as quickly as possible the error on the current belief state, in order to guarantee performance as close as possible to the optimal within the allowed online planning time. In order to achieve this objective, we adopt a heuristic search approach where the search is guided toward the fringe node that contributes the most to the error on the current belief state. To this end, we need a way to express the error on the current belief states in terms of the error on the fringe belief state. Theorem 1 states that more formally.

We will denote $\mathcal{F}(T)$ the set of fringe nodes of a tree T, $h^b_{b_0}$ the unique sequence of action and observation that leads from b_0 to node b in tree T, $d(b,b_0)$ the depth of belief node b in T from the current belief state b_0, $P(h^b_{b_0}|b_0,\pi^*) = \prod_{i=0}^{d(b,b_0)-1} P(z_i(h^b_{b_0})|b_i(h^b_{b_0}),a_i(h^b_{b_0}))\pi^*(b_i(h^b_{b_0}),a_i(h^b_{b_0}))$ the probability of reaching belief b via the sequence $h^b_{b_0}$ if we follow the optimal policy π^* from the current belief b_0, where $z_i(h^b_{b_0})$, $b_i(h^b_{b_0})$ and $a_i(h^b_{b_0})$ refer to the observation, belief state and action encountered at depth i in the sequence $h^b_{b_0}$ and $\pi^*(b,a)$ is the probability that the optimal policy executes action a in belief state b. Notice here that in the context of online algorithms, we will refer to b_0 as the current belief state during the execution, which may differ from the initial belief state of the environment. Hence b_0 will always refer to the root of the search tree.

Theorem 1. *In any tree T where values are computed according to equations 8 and 9 using a lower bound value function L with error $e(b) = V^*(b) - L(b)$, the error on the root belief state b_0 is bounded by: $e_T(b_0) = V^*(b_0) - L_T(b_0) \leq \sum_{b\in\mathcal{F}(T)} \gamma^{d(b,b_0)} P(h^b_{b_0}|b_0,\pi^*)e(b)$.*

Proof. Proof provided in the appendix. □

Here, e_T is the error function for nodes in the tree T: $e_T(b) = V^*(b) - L_T(b)$ ($e_T(b) = e(b)$ for fringe nodes).

4.1. Heuristics

From theorem 1, we see that the contribution of each fringe node to the error in b_0 is simply the term $\gamma^{d(b,b_0)} P(h^b_{b_0}|b_0,\pi^*)e(b)$. Consequently, if we want to minimize $e_T(b_0)$ as quickly as possible, we should explore leaves reached by the optimal policy π^* that maximize the term $\gamma^{d(b,b_0)} P(h^b_{b_0}|b_0,\pi^*)e(b)$ as they offer the greatest potential to reduce $e_T(b_0)$. Therefore, this gives us a sound heuristic to explore the tree in a best-first-search way.

However, we do not know V^* nor π^*, which are required to compute the terms $e(b)$ and $P(h^b_{b_0}|b_0,\pi^*)$; nevertheless, we can approximate them.

First, the term $e(b)$ can be estimated by the difference between the lower and upper bound. That is, we will define $\hat{e}(b)$ as an estimation of the error introduced by our bounds at fringe nodes:

$$\hat{e}(b) = U(b) - L(b) \tag{10}$$

Clearly, $\hat{e}(b)$ is an upper bound on $e(b)$ since $U(b) \geq V^*(b)$.

To approximate the term $P(h^b_{b_0}|b_0,\pi^*)$, we can view the terms $\pi^*(b,a)$ as the probability that action a is optimal in belief b. In this context, we will approximate $\pi^*(b,a)$ by using an approximate policy $\hat{\pi}_T$, such that $\hat{\pi}_T(b,a)$ will represent the probability that action a is optimal in belief state b, given the lower and upper bounds $L_T(b,a)$ and $U_T(b,a)$ that we have on $Q^*(b,a)$ in tree T. In particular, if for all actions $a' \neq a$, $U_T(b,a') \leq L_T(b,a)$, then we would know for sure that action a is optimal in belief b, i.e. $\hat{\pi}_T(b,a)$ should be 1. However in general there will be many actions that could be optimal, such that, in order to compute the probability $\hat{\pi}_T(b,a)$, we will need to consider the Q-value $Q^*(b,a)$ as a random variable and make some

assumptions about its underlying probability distribution. Once density functions f_a^b and cumulative distribution functions F_a^b are determined for each (b, a), we can compute the probability $\hat{\pi}_T(b, a)$ that action a is optimal in belief b, given the tree T, as follows:

$$\hat{\pi}_T(b,a) = \int_{-\infty}^{\infty} f_a^b(x) \prod_{a' \neq a} F_{a'}^b(x) dx \qquad (11)$$

This comes from the fact that $P(\forall a' \neq a, Q^*(b,a) > Q^*(b,a')) = \sum_x P(Q^*(b,a) = x) \prod_{a' \neq a} P(Q^*(b,a') < x)$ if $Q^*(b,a)$ would be discrete. The last integral is just a generalization of this term for continuous random variables. However, computing this integral may not be computationally efficient depending on how we define the functions f_a^b. One possible approximation is to simply compute the probability that the Q-value of a certain action is higher than its parent belief state value instead of all actions' Q-value. In this case, the integral reduces to $\hat{\pi}_T(b,a) = \int_{-\infty}^{\infty} f_a^b(x) F^b(x) dx$, where F^b is the cumulative distribution function for the value $V^*(b)$. Henceforth, by considering both $Q^*(b,a)$ and $V^*(b)$ as random variables that follow uniform distributions between their respective lower and upper bounds, we get that:

$$\hat{\pi}_T(b,a) = \begin{cases} \eta \frac{(U_T(b,a) - L_T(b))^2}{U_T(b,a) - L_T(b,a)} & \text{if } U_T(b,a) > L_T(b) \\ 0 & \text{otherwise} \end{cases} \qquad (12)$$

η is a normalization constant such that the probabilities $\hat{\pi}_T(b,a)$ over all actions $a \in A$ will sum to 1. Notice also that if the density function is 0 outside the interval between the lower and upper bound, then $\hat{\pi}_T(b,a) = 0$ for dominated actions and we will never explore fringe nodes that are reached by taking these dominated actions. Consequently, the term $\hat{\pi}_T(b,a)$ implicitly prunes the dominated actions.

Another practical approximation that we can use is to simply consider the action that maximizes the upper bound as the optimal action:

$$\hat{\pi}_T(b,a) = \begin{cases} 1 & \text{if } a = \arg\max_{a' \in A} U_T(b,a') \\ 0 & \text{otherwise} \end{cases} \qquad (13)$$

Consequently, this will restrict the exploration to fringe nodes that are reached by sequence of actions that maximize the upper bound of their parent belief state, as done in the AO^* algorithm (Nilsson, 1980). The nice property of this approximation is that these fringe nodes are the only nodes that can potentially reduce the upper bound in b_0.

Using these two approximations, we can evaluate an approximate error contribution $E(b, b_0, T)$ of a fringe node b on the value of the root belief b_0 in tree T as:

$$E(b, b_0, T) = \gamma^{d(b,b_0)} P(h_{b_0}^b | b_0, \hat{\pi}_T) \hat{e}(b) \qquad (14)$$

Using this approximate error contribution as an heuristic for a best-first-search algorithm, the next fringe node $\widetilde{b}(T)$ to expand in tree T is defined as follows:

$$\widetilde{b}(T) = \arg\max_{b \in \mathcal{F}(T)} \gamma^{d(b,b_0)} P(h_{b_0}^b | b_0, \hat{\pi}_T) \hat{e}(b) \qquad (15)$$

Since we proposed 2 different approximations for $\hat{\pi}_T$, we will refer to this heuristic as AEMS1 when using $\hat{\pi}_T$ as defined in equation 12 and AEMS2 when using equation 13.

Intuitively, $\widetilde{b}(T)$ seems a sound heuristic to guide the search since it has several desired properties, i.e. it will favor the exploration of belief states that have high probabilities to be reached by promising actions and that have a lot of error.

4.2. Algorithm

A detailed description of our algorithm is presented in algorithm 1.

Algorithm 1 *AEMS* : Anytime Error Minimization Search

```
Function AEMS(t, ε)
Static : T: an AND-OR tree representing the current search tree.
Initialize T with initial belief state b_0
while not EnvironmentTerminated() do
  t_0 ← Time()
  while Time() − t_0 ≤ t and not Solved(Root(T), ε) do
    b* ← b̃(T)
    Expand(b*)
    UpdateAncestors(b*)
  end while
  â ← arg max_{a∈A} L_T(Root(T), a)
  DoAction(â)
  z ← GetObservation()
  T ← SubTree(Child(Root(T), â, z))
end while
```

Here the parameter t represents the online search time allowed per action and ϵ is the desired precision on the policy. The EXPAND function simply does a one-step lookahead under the node b^* by adding the next action and belief nodes to the tree T and computing their lower and upper bounds according to equations 6, 7,

8 and 9. Notice that if we reach a belief state that is already somewhere else in the tree, it will be duplicated, since our current algorithm does not handle graph structure. We could possibly try to use a technique proposed for AO* (LAO* algorithm (Hansen & Zilberstein, 2001)) to handle cycle, but we have not investigated this further and how it affects the heuristic value. After a node is expanded, the UPDATEANCESTORS function simply recomputes the bounds of its ancestors according to equations 6, 7, 8 and 9. It also recomputes the the probabilities $\hat{\pi}_T(b,a)$ and the best actions of each ancestor nodes. The search will terminate whenever there is no more time available or we have found an ϵ-optimal solution (verified by the SOLVED function). Then the algorithm executes the action that maximizes the lower bound and updates the tree T such that our new belief state will be the root of T; all the nodes under this new root can be reused at the next time step.

4.3. Theoretical Results

We now provide some sufficient conditions under which AEMS is guaranteed to converge to an ϵ-optimal policy after a finite number of expansions. We will show that the heuristics we have presented satisfy those conditions and therefore, are admissible.

Theorem 2. *Given $U(b) \geq V^*(b)$, $L(b) \leq V^*(b)$ and $\hat{e}(b) = U(b) - L(b)$ for all belief b and U is bounded above and L is bounded below, if $\gamma \in [0,1)$ and $\inf_{b,T|\hat{e}_T(b)>\epsilon} \hat{\pi}_T(b, \hat{a}_b^T) > 0$ for $\hat{a}_b^T = \arg\max_{a \in A} U_T(b,a)$, then the AEMS algorithm using heuristic $\widetilde{b}(T)$ is complete and ϵ-optimal.*

Proof. Proof provided in the appendix. □

From this last theorem, we can see that the main sufficient property for convergence if we use $\hat{e}(b) = U(b) - L(b)$ is to just guarantee that $\hat{\pi}_T(b,a)$ must be greater than 0 for the action a that maximizes $U_T(b,a)$. Hence we can potentially develop many different admissible heuristics for the AEMS algorithm. It also follows from this theorem that the 2 heuristics we have proposed, AEMS1 and AEMS2, are admissible. This is demonstrated formally in the following corollaries:

Corollary 1. *Given $U(b) \geq V^*(b)$ and $L(b) \leq V^*(b)$ for all belief b and U is bounded above and L is bounded below, if $\gamma \in [0,1)$, the AEMS algorithm using heuristic $\widetilde{b}(T)$, with approximations $\hat{\pi}_T$ as defined in equation 12 and $\hat{e}(b) = U(b) - L(b)$, is complete and ϵ-optimal.*

Proof. Proof provided in the appendix. □

Corollary 2. *Given $U(b) \geq V^*(b)$ and $L(b) \leq V^*(b)$ for all belief b and U is bounded above and L is bounded below, if $\gamma \in [0,1)$, the AEMS algorithm using heuristic $\widetilde{b}(T)$, with approximations $\hat{\pi}_T$ as defined in equation 13 and $\hat{e}(b) = U(b) - L(b)$, is complete and ϵ-optimal.*

Proof. Proof provided in the appendix. □

5. Experiments

We have tested our algorithm in the Tag (Pineau et al., 2003), RockSample (Smith & Simmons, 2005) and the FieldVisionRockSample (FVRS) environments (Ross & Chaib-draa, 2007). In each of these environments, we used factored POMDP representation (Poupart, 2005) to lower the complexity of computing $\tau(b,a,z)$ and $P(z|b,a)$; and we first computed the Blind policy[1] to obtain a lower bound and used the FIB algorithm (Hauskrecht, 2000) to obtain an upper bound. We then compared the performance yielded by different online approaches (Satia (Satia & Lave, 1973), BI-POMDP (Washington, 1997), RTBSS (Paquet et al., 2006), AEMS1, AEMS2) using these lower and upper bounds, under a real-time constraint of 1 second/action, according to different metrics: average reward, average error reduction[2] (ER) and average lower bound improvement[3] (LBI). Satia, BI-POMDP, AEMS1 and AEMS2 were all implemented in the same algorithm since they differ only by the heuristic used to guide the search, which allowed us to measure directly the performance of the specific heuristics. RTBSS serves as a base line for a complete k-step lookahead search using branch & bound pruning.

In table 1, we present 95% confidence intervals on the different statistics. The Belief Nodes column represents the average number of belief nodes explored at each time step and the Reuse column contains the average percentage of belief nodes that were reused in the next timestep. The time column contains the average online time per action taken by the algorithm (results are slightly lower than 1 second because sometimes the algorithms found ϵ-optimal action). For RTBSS, the depth used is presented in parenthesis and corresponds to the highest depth that had an average running time under 1 second. We also present the performance obtained by the Blind Policy, which is the policy we are trying to improve with the online algorithm.

[1] The best policy performing the same action in every belief state

[2] The error reduction is defined as $1 - \frac{U_T(b_0) - L_T(b_0)}{U(b_0) - L(b_0)}$, when the search process terminates on b_0

[3] The lower bound improvement is defined as $L_T(b_0) - L(b_0)$, when the search process terminates on b_0

Table 1. Comparison of different online search algorithm in different environments.

Heuristic / Algorithm	Reward ± 0.01	ER (%) ± 0.1	LBI ± 0.01	Belief Nodes -	Reuse (%) ±0.1	Time (ms) ±1
Tag ($\|S\| = 870, \|A\| = 5, \|Z\| = 30$)						
Blind	-19.21	-	-	-	-	-
RTBSS(5)	-10.30	22.3	3.03	45067	0	580
Satia & Lave	-8.35	22.9	2.47	36908	10.0	856
AEMS1	**-6.73**	**49.0**	**3.92**	**43693**	**25.1**	**814**
BI-POMDP	-6.22	76.2	7.81	79508	54.6	622
AEMS2	**-6.19**	**76.3**	**7.81**	**80250**	**54.8**	**623**
RockSample[7,8] ($\|S\| = 12545, \|A\| = 13, \|Z\| = 2$)						
Blind	7.35	-	-	-	-	-
Satia & Lave	7.35	3.6	0	509	8.9	900
AEMS1	**10.30**	**9.5**	**0.90**	**579**	**5.3**	**916**
RTBSS(2)	10.30	9.7	1.00	439	0	896
BI-POMDP	18.43	33.3	4.33	2152	29.9	953
AEMS2	**20.75**	**52.4**	**5.30**	**3145**	**36.4**	**859**
FVRS[5,7] ($\|S\| = 3201, \|A\| = 5, \|Z\| = 128$)						
Blind	8.15	-	-	-	-	-
RTBSS(1)	20.57	7.7	2.07	516	0	254
BI-POMDP	22.75	11.1	2.08	4457	0.4	923
Satia & Lave	22.79	11.1	2.05	3683	0.4	947
AEMS1	**23.31**	**12.4**	**2.24**	**3856**	**1.4**	**942**
AEMS2	**23.39**	**13.3**	**2.35**	**4070**	**1.6**	**944**

As we can see from these results, AEMS2 provide the best average reward, average error reduction and average lower bound improvement in all these environments. The higher error reduction and lower bound improvement obtained by AEMS2 indicates that it can guarantee performance closer to the optimal. Furthermore, we also observe that AEMS2 has the best average reuse percentage, which indicates that AEMS2 is able to guide the search toward the most probable nodes and allows it to generally maintain a higher number of belief nodes in the tree. On the other hand, AEMS1 did not perform very well, except in FVRS[5,7]. This could be explained by the fact that our assumption that the values of the actions are uniformly distributed between the lower and upper bounds is not valid in these environments. AEMS2 might also be more efficient than AEMS1 due to the fact that it always explore nodes that will reduce the upper bound in b_0, which is not the case for AEMS1.

Compared to offline algorithms, AEMS2 fares pretty well. In Tag, the best result in terms of rewards comes from the Perseus algorithm which has an average reward of -6.17 but requires 1670 seconds of offline time (Spaan & Vlassis, 2005), while our approach obtained -6.19 but only required 1 second of offline time to compute the lower and upper bounds. Again in RockSample[7,8], HSVI, the best offline approach, obtains an average reward of 20.6 and requires 1003 seconds of offline time (Smith & Simmons, 2005), while our approach obtained 20.75 but only required 25 seconds of offline time to compute the lower and upper bounds.

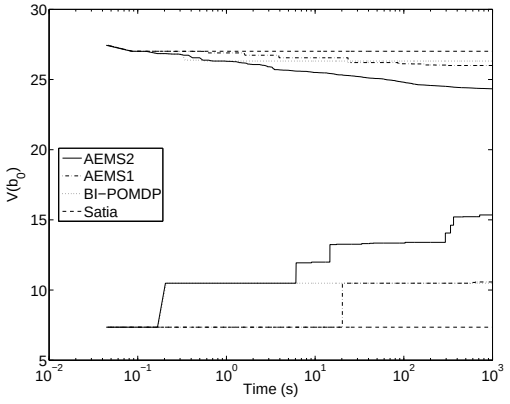

Figure 2. Evolution of the upper and lower bounds on *RockSample[7,8] with the different heuristics.*

In previous work, we had also compared AEMS2 to other online approaches using PBVI as a lower bound (Ross & Chaib-draa, 2007). Again, AEMS2 showed better performance, which indicates that AEMS2 tends to be better than the other approaches, no matter which bound we use.

We also looked at how fast the lower and upper bound converge if we just let the algorithm run up to 1000 seconds on the initial belief state, to give us an idea of which heuristic would be the best given more online planning time is available. The results on RockSample[7,8] are presented in figure 2.

As we can see, the bounds converge much more quickly for the AEMS2 heuristic.

6. Conclusion

We have proposed a new online heuristic search algorithm which seeks to minimize the error on approximate policies computed offline. Our approach has shown the best overall performance in terms of rewards, error reduction and lower bound improvement compared to the other existing online algorithm in large POMDPs. It also offers an interesting offline/online computation tradeoff and obtains similar performance to the best offline approach at a fraction of the total computation time. We have also showed that the heuristics we proposed are admissible and lead to ϵ-optimal solutions within finite time. For future work, we would like to investigate further improvement to the algorithm. As we mentioned, our current algorithm does not handle cycle which makes it do redundant computations when belief states are duplicated. We would also like to experiment with other variants of the term $\hat{\pi}_T(b, a)$ and see whether we can come up with better approximations.

7. Acknowledgements

This research was supported in part by the Natural Science and Engineering Council of Canada (NSERC).

8. Appendix

In this section we present the proofs of the different theorems introduced in this paper.

Theorem 1. *In any tree T where values are computed according to equations 8 and 9 using a lower bound value function L with error $e(b) = V^*(b) - L(b)$, the error on the root belief state b_0 is bounded by: $e_T(b_0) = V^*(b_0) - L_T(b_0) \leq \sum_{b \in \mathcal{F}(T)} \gamma^{d(b,b_0)} P(h^b_{b_0}|b_0, \pi^*) e(b)$.*

Proof. Consider an arbitrary parent node b in tree T and let's denote $a^*_b = \arg\max_{a \in A} Q^*(b,a)$ and $\hat{a}^T_b = \arg\max_{a \in A} L_T(b,a)$. We have $e_T(b) = V^*(b) - L_T(b)$. If $\hat{a}^T_b = a^*_b$, then $e_T(b) = \gamma \sum_{z \in Z} P(z|b, a^*_b) e_T(\tau(b, a^*_b, z))$. On the other hand, when $\hat{a}^T_b \neq a^*_b$, then we know that $L_T(b, a^*_b) \leq L_T(b, \hat{a}^T_b)$ and therefore $e_T(b) \leq \gamma \sum_{z \in Z} P(z|b, a^*_b) e_T(\tau(b, a^*_b, z))$. Consequently, the following recurrence is an upper bound on $e_T(b)$:

$$e_T(b) = \begin{cases} e(b) & \text{if } b \in \mathcal{F}(T) \\ \gamma \sum_{z \in Z} P(z|b, a^*_b) e_T(\tau(b, a^*_b, z)) & \text{otherwise} \end{cases}$$

Solving this recurrence for b_0 proves the theorem. □

Lemma 1. *In any tree T, the approximate error contribution $E(b_d, b_0, T)$ of a belief node b_d at depth d is bounded by $E(b_d, b_0, T) \leq \gamma^d \sup_b \hat{e}(b)$.*

Proof. Since $P(h^b_{b_0}|b_0, \hat{\pi}_T) \leq 1$ and $\hat{e}(b) \leq \sup_{b'} \hat{e}(b')$ for all belief b, then it follows that $E(b_d, b_0, T) \leq \gamma^d \sup_b \hat{e}(b)$. □

For the following lemma and theorem, we will denote $P(h^b_{b_0}|b_0) = \prod_{i=0}^{d(b,b_0)-1} P(z_i(h^b_{b_0})|a_i(h^b_{b_0}), b_i(h^b_{b_0}))$ the product of the observation probabilities along the path $h^b_{b_0}$ from current belief b_0 to belief b and $\hat{\beta}(b_0, T) \subseteq \mathcal{F}(T)$ as the set of all fringe nodes in T such that $P(h^b_{b_0}|b_0, \hat{\pi}_T) > 0$, for $\hat{\pi}_T$ defined as in equation 13, i.e. the set of all fringe nodes reached by a sequence of actions that maximize the upper bound $U_T(b,a)$ in their respective belief state.

Lemma 2. *Given $U(b) \geq V^*(b)$, $L(b) \leq V^*(b)$, and $\hat{e}(b) = U(b) - L(b)$ for all belief b, U bounded above and L bounded below, then for any tree T, $\epsilon > 0$ and D such that $\gamma^D \sup_b \hat{e}(b) \leq \epsilon$, if for all $b \in \hat{\beta}(b_0, T)$, either $d(b, b_0) > D$ or there exists an ancestor b' of b such that $\hat{e}_T(b') \leq \epsilon$, then $\hat{e}_T(b_0) \leq \epsilon$.*

Proof. Let's denote $\hat{a}^T_b = \arg\max_{a \in A} U_T(b,a)$. We first notice that for any tree T, and parent belief $b \in T$, $\hat{e}_T(b) = U_T(b) - L_T(b) \leq U_T(b, \hat{a}^T_b) - L_T(b, \hat{a}^T_b) = \gamma \sum_{z \in Z} P(z|b, \hat{a}^T_b) \hat{e}_T(\tau(b, \hat{a}^T_b, z))$. Consequently, the following recurrence is an upper bound on $\hat{e}_T(b)$:

$$\hat{e}_T(b) = \begin{cases} \hat{e}(b) & \text{if } b \in \mathcal{F}(T) \\ \epsilon & \text{if } \hat{e}_T(b) \leq \epsilon \\ \gamma \sum_{z \in Z} P(z|b, \hat{a}^T_b) \hat{e}_T(\tau(b, \hat{a}^T_b, z)) & \text{otherwise} \end{cases}$$

By unfolding the recurrence for b_0, we get $\hat{e}_T(b_0) = \sum_{b \in A(T)} \gamma^{d(b,b_0)} P(h^b_{b_0}|b_0) \hat{e}(b) + \epsilon \sum_{b \in B(T)} \gamma^{d(b,b_0)} P(h^b_{b_0}|b_0)$, where $B(T)$ is the set of parent nodes b' having a descendant in $\hat{\beta}(b_0, T)$ such that $\hat{e}_T(b') \leq \epsilon$ and $A(T)$ is the set of fringe nodes b'' in $\hat{\beta}(b_0, T)$ not having an ancestor in $B(T)$. Hence if for all $b \in \hat{\beta}(b_0, T)$, $d(b, b_0) > D$ or there exists an ancestor b' of b such that $\hat{e}_T(b') \leq \epsilon$, then this means that for all b in $A(T)$, $d(b, b_0) > D$, and therefore, $\hat{e}_T(b_0) \leq \gamma^D \sup_b \hat{e}(b) \sum_{b \in A(T)} P(h^b_{b_0}|b_0) + \epsilon \sum_{b \in B(T)} P(h^b_{b_0}|b_0) \leq \epsilon \sum_{b \in A(T) \cup B(T)} P(h^b_{b_0}|b_0) = \epsilon$. □

Theorem 2. *Given $U(b) \geq V^*(b)$, $L(b) \leq V^*(b)$ and $\hat{e}(b) = U(b) - L(b)$ for all belief b and U is bounded above and L is bounded below, if $\gamma \in [0, 1)$ and $\inf_{b, T|\hat{e}_T(b) > \epsilon} \hat{\pi}_T(b, \hat{a}^T_b) > 0$ for $\hat{a}^T_b = \arg\max_{a \in A} U_T(b,a)$, then the AEMS algorithm using heuristic $\widetilde{b}(T)$ is complete and ϵ-optimal.*

Proof. Consider an arbitrary $\epsilon > 0$ and current belief state b_0. If $\gamma = 0$, then obviously, as soon as AEMS will have expanded b_0, $\hat{e}_T(b_0) = 0$ since $U_T(b_0) = L_T(b_0) = max_{a \in A} R(b_0, a)$ and therefore AEMS is complete and ϵ-optimal. Therefore let's concentrate on the case where $\gamma \in (0, 1)$. Clearly, since U is bounded above and L is bounded below, $\hat{e}$ is bounded above, and therefore, since $\gamma \in (0, 1)$, there exists a positive integer D such that $\gamma^D \sup_b \hat{e}(b) < \epsilon$. Let's denote $\mathcal{A}^T_b$ the set of ancestor belief states of b in the tree T, and given a finite set A of belief nodes, let's define $\hat{e}^{min}_T(A) = \min_{b \in A} \hat{e}_T(b)$. Now let's define $\mathcal{T}_b = \{T | T finite, b \in \hat{\beta}(b_0, T), \hat{e}^{min}_T(\mathcal{A}^T_b) > \epsilon\}$ and $\mathcal{B} = \{b | \hat{e}(b) \inf_{T \in \mathcal{T}_b} P(h^b_{b_0}|b_0, \hat{\pi}_T) > 0, d(b, b_0) \leq D\}$. Clearly, by the assumption that $\inf_{b, T|\hat{e}_T(b) > \epsilon} \hat{\pi}_T(b, \hat{a}^T_b) > 0$, then $\mathcal{B}$ contains all belief states b within depth D such that $\hat{e}(b) > 0$, $P(h^b_{b_0}|b_0) > 0$ and there exists a finite tree T where $b \in \hat{\beta}(b_0, T)$ and all ancestors b' of b have $\hat{e}_T(b') > \epsilon$. Furthermore, $\mathcal{B}$ is finite since there are only finitely many belief states within depth D. Hence there exists a $E_{min} = \min_{b \in \mathcal{B}} \gamma^{d(b,b_0)} \hat{e}(b) \inf_{T \in \mathcal{T}_b} P(h^b_{b_0}|b_0, \hat{\pi}_T)$.

Clearly, $E_{min} > 0$ and we know that for any tree T, all beliefs b in $\mathcal{B} \cap \hat{\beta}(b_0, T)$ have an approximate error contribution $E(b, b_0, T) \geq E_{min}$. Since $E_{min} > 0$ and $\gamma \in (0, 1)$, there exists a positive integer D' such that $\gamma^{D'} \sup_b \hat{e}(b) < E_{min}$. Hence by lemma 1, this means that AEMS cannot expand any node at depth D' or beyond before expanding a tree T where $\mathcal{B} \cap \hat{\beta}(b_0, T) = \emptyset$. Because there are only finitely many nodes below depth D', then it is clear that AEMS will reach such tree T after a finite number of expansions. Furthermore, for this tree T, since $\mathcal{B} \cap \hat{\beta}(b_0, T) = \emptyset$, we have that for all beliefs $b \in \hat{\beta}(b_0, T)$, either $d(b, b_0) \geq D$ or $\hat{e}_T^{min}(\mathcal{A}_b^T) \leq \epsilon$. Hence by lemma 2, this implies that $\hat{e}_T(b_0) < \epsilon$, and consequently AEMS will terminate (SOLVED(b_0, ϵ) will evaluate to true) with an ϵ-optimal solution (since $e_T(b_0) \leq \hat{e}_T(b_0)$) after a finite number of expansions. □

Corollary 1. *Given $U(b) \geq V^*(b)$ and $L(b) \leq V^*(b)$ for all belief b and U is bounded above and L is bounded below, if $\gamma \in [0, 1)$, the AEMS algorithm using heuristic $\widetilde{b}(T)$, with approximations $\hat{\pi}_T$ as defined in equation 12 and $\hat{e}(b) = U(b) - L(b)$, is complete and ϵ-optimal.*

Proof. We first notice that $(U_T(b, a) - L_T(b))^2 / (U_T(b, a) - L_T(b, a)) \leq \hat{e}_T(b, a)$, since $L_T(b) \geq L_T(b, a)$ for all a. Furthermore, $\hat{e}_T(b, a) \leq \sup_{b'} \hat{e}(b')$. Therefore the normalization constant $\eta \geq (|A| \sup_b \hat{e}(b))^{-1}$. For $\hat{a}_b^T = \arg\max_{a \in A} U_T(b, a)$, we have $U_T(b, \hat{a}_b^T) = U_T(b)$, and therefore $U_T(b, \hat{a}_b^T) - L_T(b) = \hat{e}_T(b)$. Hence this means that $\hat{\pi}_T(b, \hat{a}_b^T) = \eta(\hat{e}_T(b))^2 / \hat{e}_T(b, \hat{a}_b^T) \geq (|A|(\sup_{b'} \hat{e}(b'))^2)^{-1} (\hat{e}_T(b))^2$ for all T, b. Hence, for any $\epsilon > 0$, $\inf_{b, T | \hat{e}_T(b) > \epsilon} \hat{\pi}_T(b, \hat{a}_b^T) \geq (|A|(\sup_b \hat{e}(b))^2)^{-1} \epsilon^2 > 0$. Therefore it immediatly follows from theorem 2 that AEMS is complete and ϵ-optimal. □

Corollary 2. *Given $U(b) \geq V^*(b)$ and $L(b) \leq V^*(b)$ for all belief b and U is bounded above and L is bounded below, if $\gamma \in [0, 1)$, the AEMS algorithm using heuristic $\widetilde{b}(T)$, with approximations $\hat{\pi}_T$ as defined in equation 13 and $\hat{e}(b) = U(b) - L(b)$, is complete and ϵ-optimal.*

Proof. By definition of $\hat{\pi}_T$, $\hat{\pi}_T(b, \hat{a}_b^T) = 1$ for all b,T. Hence, $\inf_{b, T | \hat{e}_T(b) > \epsilon} \hat{\pi}_T(b, \hat{a}_b^T) = 1$ and therefore it immediatly follows from theorem 2 that AEMS is complete and ϵ-optimal. □

References

Braziunas, D., & Boutilier, C. (2004). Stochastic local search for POMDP controllers. *AAAI-04*.

Cassandra, A., Littman, M. L., & Zhang, N. L. (1997). Incremental pruning: a simple, fast, exact method for partially observable Markov decision processes. *UAI-97* (pp. 54–61).

Geffner, H., & Bonet, B. (1998). Solving large POMDPs using real time dynamic programming. *Fall AAAI symp. on POMDPs*.

Hansen, E. A., & Zilberstein, S. (2001). LAO * : A heuristic search algorithm that finds solutions with loops. *Artificial Intelligence*, *129*, 35–62.

Hauskrecht, M. (2000). Value-function approximations for partially observable Markov decision processes. *Journal of Artificial Intelligence Research*, *13*, 33–94.

Littman, M. L. (1996). *Algorithms for sequential decision making*. Doctoral dissertation, Brown University.

Madani, O., Hanks, S., & Condon, A. (1999). On the undecidability of probabilistic planning and infinite-horizon partially observable markov decision problems. *AAAI-99* (pp. 541–548).

McAllester, D., & Singh, S. (1999). Approximate Planning for Factored POMDPs using Belief State Simplification. *UAI-99* (pp. 409–416).

Nilsson, N. (1980). *Principles of Artificial Intelligence*. Tioga Publishing.

Papadimitriou, C., & Tsitsiklis, J. N. (1987). The complexity of Markov decision processes. *Mathematics of Operations Research*, *12*, 441–450.

Paquet, S., Chaib-draa, B., & Ross, S. (2006). Hybrid POMDP algorithms. *Proceedings of The Workshop on Multi-Agent Sequential Decision Making in Uncertain Domains (MSDM-2006)*. Hakodate, Hokkaido, Japan.

Pineau, J., Gordon, G., & Thrun, S. (2003). Point-based value iteration: an anytime algorithm for POMDPs. *IJCAI-03*.

Poupart, P. (2005). *Exploiting structure to efficiently solve large scale partially observable Markov decision processes*. Doctoral dissertation, University of Toronto.

Puterman, M. L. (1994). *Markov decision processes: Discrete stochastic dynamic programming*. New York, NY, USA: John Wiley & Sons, Inc.

Ross, S., & Chaib-draa, B. (2007). AEMS: an anytime online search algorithm for approximate policy refinement in large pomdps. *IJCAI'07* (pp. 2592–2598). Hyderabad, India.

Satia, J. K., & Lave, R. E. (1973). Markovian decision processes with probabilistic observation of states. *Management Science*, *20*, 1–13.

Smith, T., & Simmons, R. (2005). Point-based POMDP algorithms: improved analysis and implementation. *UAI-05*. Edinburgh, Scotland.

Spaan, M. T. J., & Vlassis, N. (2005). Perseus: randomized point-based value iteration for POMDPs. *JAIR*, *24*, 195–220.

Washington, R. (1997). BI-POMDP: bounded, incremental partially observable Markov model planning. *4th Eur. Conf. on Planning* (pp. 440–451).