Online Planning in POMDPs with State-Requests

Raphaël Avalos[1,2]* **Eugenio Bargiacchi**[1] **Ann Nowé**[1]
Diederik M. Roijers[1,3] **Frans A. Oliehoek**[2]
[1] AI Lab, Vrije Universiteit Brussel [2] TU Delft [3] City of Amsterdam
raphael.avalos@vub.be

Abstract

In key real-world problems, full state information is sometimes available but only at a high cost, like activating precise yet energy-intensive sensors or consulting humans, thereby compelling the agent to operate under partial observability. For this scenario, we propose AEMS-SR (Anytime Error Minimization Search with State Requests), a principled online planning algorithm tailored for POMDPs with state requests. By representing the search space as a graph instead of a tree, AEMS-SR avoids the exponential growth of the search space originating from state requests. Theoretical analysis demonstrates AEMS-SR's ε-optimality, ensuring solution quality, while empirical evaluations illustrate its effectiveness compared with AEMS and POMCP, two SOTA online planning algorithms. AEMS-SR enables efficient planning in domains characterized by partial observability and costly state requests offering practical benefits across various applications.

1 Introduction

The *Partially Observable Markov Decision Process (POMDP)* is a powerful framework that models sequential decision-making in scenarios where the environment's true state is inaccessible. Often, this partial observability is an inherent characteristic of the environment, perhaps due to noise or the unavailability of suitable sensors. However, in numerous instances, determining the true state of the system is feasible but entails a considerable cost. For example, consider a scenario involving a battery-powered robot that lacks the necessary power to employ highly accurate sensors continuously, and thus, is also equipped with power-efficient yet less precise sensors. Additionally, the concept of requesting state information can extend to scenarios with privacy implications, such as the use of surveillance cameras in public spaces for crowd control. In these cases, the decision to activate cameras involves weighing the benefits of state access against potential privacy costs.

We can think of these settings as situations where the agent has the option to consult an oracle (like a precise sensor or a human expert) at every step to obtain the state against a cost. In the context of our battery-powered robot, the cost could represent the electricity cost of activating the accurate sensor. We refer to this setting as *POMDPs with State Requests (POMDP-SR)*, where the agent, for a cost, can eliminate all uncertainty regarding its current state before selecting each action.

A naive approach to handling POMDP-SRs is converting them to equivalent POMDPs, as detailed in Section 3. However, such a method overlooks the unique characteristics of POMDP-SRs, potentially leading to suboptimal performance of conventional POMDP planning techniques. This is largely because in the conversion to an equivalent POMDP, the number of time-steps effectively doubles, and the observation space expands considerably. For methods relying on tree search, such as POMCP (Silver & Veness, 2010) and AEMS (Ross & Chaib-Draa, 2007), the transformation into an equivalent POMDP introduces an exponential increase in the search tree, significantly impeding their efficiency. Extensions that build on sparse samplings, such as DESPOT (Somani et al., 2013),

*Work done during a research visit at TU Delft.

17th European Workshop on Reinforcement Learning (EWRL 2024).

theoretically can visit only a small part of the tree, but to obtain good results, this small part is in practice still large.

In this paper, we introduce a novel online planning algorithm, AEMS-SR (Anytime Error Minimization Search with State Requests), tailored for POMDPs with state requests. While traditional approaches use trees, AEMS-SR can leverage a cyclic graph, significantly reducing the search space by avoiding redundant expansions and improving computational efficiency.

Our contributions are threefold: 1) We formalize the request the state framework; 2) We introduce a new algorithm, AEMS-SR, and theoretically demonstrate its ε-optimality; 3) We conduct experiments on RobotDelivery, our newly developed benchmark, and Tag demonstrating AEMS-SR's superiority over AEMS and POMCP. Our results highlight AEMS-SR's efficiency in circumventing the exponential growth of the search tree, highlighting its potential in this challenging setting.

2 Background

POMDPs Partially Observable Markov Decision Processes (Åström, 1965) are defined as a tuple $\mathcal{P} = \langle \mathcal{S}, \Omega, \mathcal{A}, \mathbf{P}, \mathcal{O}, \mathcal{R}, \gamma \rangle$ where $\mathcal{S}$ is the set of states, Ω is the sets of observations, $\mathcal{A}$ is the set of actions, $\mathbf{P}\colon \mathcal{S} \times \mathcal{A} \to \Delta_{\mathcal{S}}$ is probability transition function with $\Delta_{\mathcal{S}}$ being the simplex over the state space, $\mathcal{O}\colon \mathcal{S} \times \mathcal{A} \to \Delta_{\Omega}$ is the probability observation function, $\mathcal{R}\colon \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor. At each time step the agent selects an action based on its observation-action history $h \in (\mathcal{A} \cdot \Omega)^*$. Due to the exponential growth of the history space in the number of time-steps dealing with histories might not be practical. Beliefs, defined as the probability distribution over current states $b \in \mathcal{B} \equiv \Delta_{\mathcal{S}}$, are sufficient statistics of the history for control (Åström, 1965) and a more compact alternative. Beliefs are computed recursively: after taking an action a in belief b and receiving an observation o the next belief is defined for any next state $s' \in \mathcal{S}$ as follows, with η being the normalizing factor.

$$b'(s') = \eta \mathop{\mathbb{E}}_{s \sim b} \mathbf{P}\big(s' \mid s, a\big) \cdot \mathcal{O}\big(o \mid s', a\big)$$

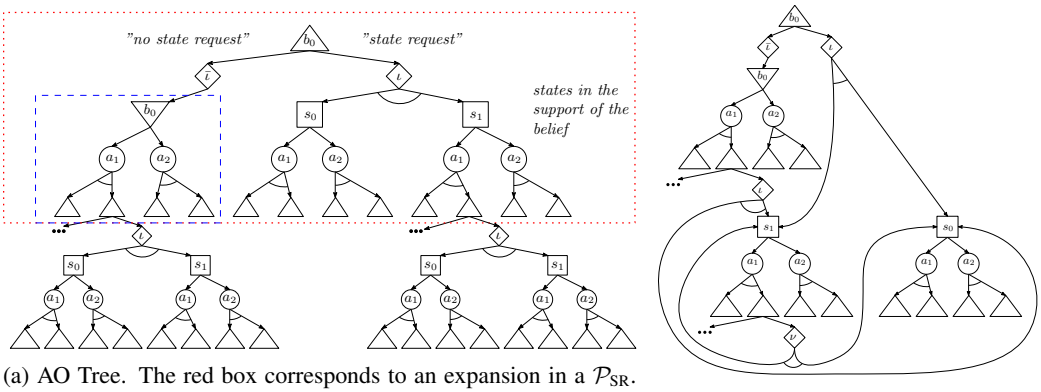
A policy $\pi\colon \mathcal{B} \to \mathcal{A}$ is a mapping from beliefs to actions and is associated with a Value $V^{\pi}(b)$. We denote as π^* and V^* the optimal policy and its value. For finite horizon, V is a piece-wise linear and convex (PWLC) function of the belief (Sondik, 1971), and can therefore be represented as a set Γ of α-vectors which corresponds to the slopes of the PWLC function.

We refer to beliefs as *corner beliefs* when the probability of being in a state s is 1 and 0 for the other states. In clear contexts, we directly use the state s to reference such beliefs. The support of a belief, $\text{supp}(b)$, is the set of states with non-zero probability. POMDP planning methods generally fall into two categories: offline and online approaches. Offline methods precompute comprehensive plans for all scenarios but suffer from computational demands and scalability issues. In contrast, online methods provide real-time computational capabilities for determining optimal actions within time constraints.

AEMS Anytime Error Minimization Search (AEMS) (Ross & Chaib-Draa, 2007) is an online algorithm that, following the stochastic shortest path approach of AO* (Nilsson, 1982), builds a tree $\mathcal{T}$ from the current belief b_0. The algorithm maintains an upper bound $U_{\mathcal{T}}(b)$ and a lower bound $L_{\mathcal{T}}(b)$ of the value $V^*(b)$. At each step, AEMS expands the node that is believed to have the highest reduction potential for the error at the root. Let $\mathcal{F}(\mathcal{T})$ be the set of fringe nodes (nodes without children) in $\mathcal{T}$, $\hat{e}(b) = U(b) - L(b)$ be the gap between the upper and lower bounds of the value, $d_{\mathcal{T}}(b, b_0)$ be the number of actions that separate b and b_0 in the tree $\mathcal{T}$, $h^b_{b_0}$ be the history from b_0 to b, and $P(h^b_{b_0} \mid b_0, \hat{\pi}_{\mathcal{T}})$ be the probability of reaching b from b_0 by following the policy $\hat{\pi}_{\mathcal{T}}$ that selects the action maximizing the upper bound. AEMS expands the fringe node that maximizes the heuristic of Eq. 1.

$$\tilde{b}(\mathcal{T}) = \mathop{\arg\max}_{b \in \mathcal{F}(\mathcal{T})} \gamma^{d_{\mathcal{T}}(b, b_0)} P(h^b_{b_0} \mid b_0, \hat{\pi}_{\mathcal{T}}) \hat{e}(b) \tag{1}$$

POMCP (Silver & Veness, 2010) extends MCTS to POMDPs. The algorithm relies on rollouts and removes the need to compute belief updates allowing it to scale to large state spaces.



(a) AO Tree. The red box corresponds to an expansion in a $\mathcal{P}_{\text{SR}}$. For comparison, the blue box would match the expansion in a $\mathcal{P}$.

(b) AO Graph

Figure 1: Tree and Graph representation after three successive expansions (the expanded beliefs are in green). Beliefs before selecting state request depicted by upward triangles, beliefs before environmental action by downward triangles, (not-)request state actions by diamonds, environmental actions by circles, and corner beliefs by rectangles. Some nodes are hidden for readability.

3 Framework

POMDP-SR We define the POMDP with State Request as a tuple $\mathcal{P}_{\text{SR}} = \langle \mathcal{P}, c \rangle$ where $\mathcal{P}$ is a POMDP and $c > 0$ is the associated cost to request the state. At each timestep, the agent first decides whether to request the state, which is immediately revealed if requested and then selects an action. The decision is binary: ι to request and $\bar{\iota}$ to not request the state.

A property of POMDP-SR that may not be immediately apparent is that even in cases where the optimal actions in an MDP and a POMDP align, the POMDP-SR's optimal action might differ. This arises from the fact that the agent operates under the anticipation of potential future state requests. In such contexts, a suboptimal action in the MDP and the POMDP can become the optimal one in the POMDP-SR when combined with a future request the state, achieving a return that is sub-optimal for the MDP but significantly better than the one of a POMDP. An illustrative example demonstrating this aspect of POMDP-SR is elaborated in the Appendix. This highlights how the integration of state requests fundamentally shifts the dynamics of decision-making in POMDPs.

Equivalent POMDP A POMDP-SR $\mathcal{P}_{\text{SR}} = \langle \mathcal{P}, c \rangle$ can be transformed into an equivalent POMDP $\mathcal{P}' = \langle \mathcal{S}', \Omega', \mathcal{A}', \mathbf{A}', \mathbf{P}', \mathcal{O}', \mathcal{R}', \gamma' \rangle$ with variable action space and with $\mathbf{P}', \mathcal{O}', \mathcal{R}'$ only defined over legal actions. While the comprehensive technicalities of this transformation are detailed in the Appendix, the core concept is to separate the state request action from the environmental action doubling the number of timesteps. Additionally, the state space is expanded by integrating a binary indicator, which functions to signal the phase in which the agent is operating. [2] For any state $s \in \mathcal{S}$, s^0 indicates the request the state phase, and s^1 is the environment action phase. We denote similarly the beliefs containing only states of one type (i.e. b^0, b^1).

Equivalent POMDP Complexity Transforming a POMDP-SR into its equivalent POMDP enables the use of classic POMDP planning algorithms, but this approach may prove inefficient. One inefficiency arises from the lack of support for variable action spaces in classic implementations, necessitating the use of deterrent penalties for illegal action which impacts the algorithm's complexity. For offline methods like PBVI, doubling the state space and augmenting the observation space to include the state space lead to a steep increase in complexity. Heuristic search algorithms like AEMS face an even more challenging situation as doubling the horizon in POMDP-SR and requiring a new sub-tree for every state in the support of the belief result in an exponential expansion of the search tree (Fig. 1a). This growth underscores the fundamental issue: current planning algorithms are ill-suited to effectively handle the unique complexities introduced by POMDP-SR scenarios.

[2] Including the state request action ι as an additional action at every step avoids doubling timesteps and state space, but does not yield an equivalent model to a POMDP-SR due to the discounting factor.

4 Online planning: AEMS-SR

In this section, we present our new method Anytime Error Minimization Search for POMDP-SRs (AEMS-SR) which adapts AEMS to our framework. As outlined in section 3, the introduction of state requests leads to an exponential increase in the size of the search tree. This growth is primarily due to two factors: the doubling of timesteps and the generation of a new subtree for each state in the belief's support. Consequently, each expansion of belief in a POMDP-SR, illustrated by the red box in Fig. 1a, adds $(1 + |\text{supp}(b)|) \cdot |\mathcal{A}| \cdot |\Omega|$ nodes. This is in stark contrast to classic POMDPs, where only $|\mathcal{A}| \cdot |\Omega|$ nodes are added per expansion, as illustrated by the blue box in Fig. 1a.

Upon examining the search tree in POMDP-SR scenarios, illustrated in Figure 1a, we observe that many nodes are similar. This redundancy is particularly pronounced in cases involving position uncertainty and potential action failure, leading to a significant overlap in subsequent beliefs. As a result, the tree often contains identical subtrees that are redundantly expanded, impairing search efficiency. The challenge of repetitive subtree expansions is not unique to POMDP-SR; it is a known issue in both POMDPs and MDPs. Techniques like transposition tables (Childs et al., 2008) have been used to address this problem, offering computational trade-offs that can be beneficial in certain environments but are less practical for continuous spaces such as beliefs. To deal with these, AEMS-SR employs a rooted cyclic graph, denoted as $\mathcal{G}$, with the current belief b_0 as its root, for the search replacing the conventional tree structure. A rooted cyclic graph is defined as a regular cyclic graph where every node can be reached from its root, and where the root does not have any parents. This shift to a cyclic graph necessitates the development of novel heuristic and algorithmic solutions.

4.1 AEMS-Loop

We first introduce AEMS-Loop, the extension of AEMS to cyclic graphs, and theoretically prove its completeness and $\varepsilon-$optimiality, meaning that the algorithm will always return a solution that is $\varepsilon-$close to the optimal solution given enough time. Similar to other online tree search algorithms, we rely on upper and lower bounds, denoted as $U(b)$ and $L(b)$, of the optimal value function $V^*(b)$ that are computed offline. These values are propagated in the graph $\mathcal{G}$ to the parents using the following equations, allowing expansions to reduce the error gap at the root: $e_{\mathcal{G}}(b_0) = U_{\mathcal{G}}(b_0) - L_{\mathcal{G}}(b_0)$.

$$U_{\mathcal{G}}(b,a) = R(b,a) + \gamma \sum_{o \in \Omega} P(o \,|\, b,a) U_{\mathcal{G}}(\tau(b,a,o)) \quad U_{\mathcal{G}}(b) = \begin{cases} U(b) & \text{if } b \in \mathcal{F}(\mathcal{G}) \\ \max\limits_{a \in \mathcal{A}} U_{\mathcal{G}}(b,a) & \text{otherwise} \end{cases} \tag{2}$$

$$L_{\mathcal{G}}(b,a) = R(b,a) + \gamma \sum_{o \in \Omega} P(o \,|\, b,a) L_{\mathcal{G}}(\tau(b,a,o)) \quad L_{\mathcal{G}}(b) = \begin{cases} L(b) & \text{if } b \in \mathcal{F}(\mathcal{G}) \\ \max\limits_{a \in \mathcal{A}} L_{\mathcal{G}}(b,a) & \text{otherwise} \end{cases} \tag{3}$$

Working with a cyclic graph introduces the possibility of multiple paths, and potentially an infinite number, between the root b_0 and any fringe node $b \in \mathcal{F}(\mathcal{G})$. We define as $\Phi_{\mathcal{G}}(b_0, b)$ the set of paths in $\mathcal{G}$ that start on b_0 and end up on b. A path $h \in \Phi_{\mathcal{G}}(b_0, b)$ is a sequence of beliefs, action and observation $(b_i, a_i, o_{i+1})_{i<T}$. Based on a policy π, each path has an associated probability $P(h|b_0, \pi) = \Pi_{i=0}^{T-1} P(o_{i+1}|b_i, a_i)\pi(a_i|b_i)$ corresponding to the probability of observing the path h while starting from b_0 and following π. Additionally, we define $\Psi_{\pi}^{\mathcal{G}}(b_0, b)$ as the sum over all possible paths between the root b_0 and a fringe $b \in \mathcal{F}(\mathcal{G})$ of the probability of observing the path discounted by the length of the path $d(h)$. We drop the subscript $\mathcal{G}$ when the dependency is clear.

$$\Psi_{\pi}^{\mathcal{G}}(b_0, b) = \sum_{h \in \Phi_{\mathcal{G}}(b_0, b)} \gamma^{d(h)} \mathbf{P}(h|b_0, \pi) \tag{4}$$

Theorem 1. *In any rooted graph $\mathcal{G}$ with root b_0 where values are computed according to Eq. 3 using a lower bound value function L with error $e(b) = V^*(b) - L(b)$, the error on the root belief state is bounded by: $e_{\mathcal{G}}(b_0) = V^*(b_0) - L_{\mathcal{G}}(b_0) \leqslant \sum_{b \in \mathcal{F}(\mathcal{G})} \Psi_{\pi^*}(b_0, b) e(b)$ where $e(b) = V^*(b) - L(b)$.*

Proof sketch. We use a similar proof as AEMS on an enrolling of the tree of size n to obtain an upper bound composed of two elements: (i) the discounted probabilities of observing a path of a size at most n from the root to one of the replicas of an element in $\mathcal{F}(\mathcal{G})$; (ii) the discounted probabilities of other paths which are of size n. By making $n \to +\infty$, the first part of the upper-bound converges to the term in the theorem and the second to 0. □

Theorem 1 gives an upper bound on the contribution of each fringe node to the error at the root, extending AEMS's Theorem 1 (Ross & Chaib-Draa, 2007) to cyclic graphs. Assuming a tree structure, which implies $|\Phi(b_0, b)| = 1$ for any fringe belief $b \in \mathcal{F}(\mathcal{T})$, recovers the original theorem.

Similar to AEMS, this theorem provides a robust method for choosing the next belief to expand to rapidly minimize root error: prioritize expanding the belief with the greatest estimated contribution $\arg\max_{b \in \mathcal{F}(\mathcal{G})} \Psi_{\pi*}(b_0, b)e(b)$. However, as the optimal policy π^* and value function V^* are unknown, we need to approximate them to compute $\Psi_{\pi*}(b_0, b)$ and $e(b)$. We denote the approximation of π^* as $\hat{\pi}_{\mathcal{G}}$. As in AEMS, we employ the following two approximations:

$$\hat{\pi}_{\mathcal{G}}(b, a) = \mathbb{1}\{a = \arg\max_{a'} U_{\mathcal{G}}(b, a)\} \quad (5) \qquad \hat{e}(b) = U(b) - L(b) \geqslant e(b) \quad (6)$$

While other approximations $\hat{\pi}_{\mathcal{G}}$ are possible, we selected the one presented in Equation 5 because of its empirical performance in AEMS and its simplicity. Using those two approximations, we can leverage Theorem 1 to define the following heuristic for selecting the next belief to expand $\tilde{b}(\mathcal{G})$:

$$\tilde{b}(\mathcal{G}) = \arg\max_{b \in \mathcal{F}(\mathcal{G})} \Psi_{\hat{\pi}_{\mathcal{G}}}(b_0, b)\hat{e}(b) \quad (7)$$

Theorem 2. *Given U bounded above, L bounded below such as $\forall b \in \mathcal{B}$, $U(b) \geqslant V^*(b) \geqslant L(b)$, and $\hat{e}(b) = U(b) - L(b)$, if $\gamma \in [0, 1)$ and $\inf_{b,\mathcal{G}|\hat{e}_{\mathcal{G}}(b)>\varepsilon} \hat{\pi}_{\mathcal{G}}\left(b, \hat{a}_b^{\mathcal{G}}\right) > 0$ for $\hat{a}_b^{\mathcal{G}} = \arg\max_{a \in \mathcal{A}} U_{\mathcal{G}}(b, a)$, then the AEMS-Loop algorithm using heuristic $\tilde{b}(\mathcal{G})$ is complete and $\varepsilon-$optimal.*

Theorem 2 establishes the completeness and $\varepsilon-$optimality of AEMS-Loop for any policy $\hat{\pi}_{\mathcal{G}}$ assigning non-zero probability to the upper-bound maximizing action, such as the one defined in Eq.5.

4.2 Algorithm

This subsection explains how AEMS-SR (Alg. 1), a practical implementation of AEMS-Loop adapted to POMDP-SR, works in practice. While the graph structure enables consolidating identical belief nodes to prevent redundant work, fully implementing this strategy would require comparing each new belief against all existing beliefs in the current graph. Such an approach would lead to scalability issues similar to those that render graphs less practical in general MDPs and POMDPs. To maintain tractability while still achieving our main goal of mitigating the exponential growth in the search tree, we restrict the capacity for multiple parents to corner beliefs.

AEMS-SR starts with a graph $\mathcal{G}$ containing only the root belief b_0. The algorithm then iterates through the following steps until the time limit is reached: a) find the fringe belief $\tilde{b}(\mathcal{G}) \in \mathcal{F}(\mathcal{G})$ that maximizes Eq. 7, this corresponds to Alg. 2; b) update the graph $\mathcal{G}$ by expanding the belief $\tilde{b}(\mathcal{G})$, for the request the state action ι we reuse the existing corner beliefs creating a graph similar to Fig. 1b; c) update the upper and lower bound value to match Eq.2-3.

For all fringe beliefs $b \in \mathcal{F}(\mathcal{G})$, we define the *origin* function $\xi_{\mathcal{G}}(b)$, which returns the first ancestor among the corner beliefs, or b_0 if such an ancestor does not exist. Additionally, we define h_b^{ξ} as the unique path between the origin $\xi(b)$ and the belief b which does not contain a state request action ι. The fact that $b \in \mathcal{F}(\mathcal{G})$ ensures the existence of the path, while the uniqueness is guaranteed by construction as only the corner beliefs can have multiple parents.

Computing Ψ Our heuristic (Eq. 7) requires to compute $\Psi(b_0, b)$ (Eq. 4) for all fringe beliefs $b \in \mathcal{F}(\mathcal{G})$. While straightforward in a tree structure, it becomes complex in a graph due to potentially infinite paths between b_0 and b. We address this challenge by leveraging the fact that only corner beliefs can have multiple parents to obtain Eq. 8. As the second part of the equation is straightforward to compute, our focus shifts to calculating $\Psi(b_0, s)$ for all corner beliefs s.

$$\Psi(b_0, b) = \Psi(b_0, \xi(b))P\left(h_b^{\xi}|\xi(b), \hat{\pi}_{\mathcal{G}}\right) \quad (8)$$

We start by reducing the graph $\mathcal{G}$, such as the one in Figure 1b, to $\bar{\mathcal{G}}$ containing only the root belief and corner beliefs. The set of nodes of $\bar{\mathcal{G}}$ is defined as $N' = \{b_0\} \cup \mathcal{S}$. An edge connects a node $b \in N'$ to a corner belief $s \in \mathcal{S}$ if there is at least one *direct path* h between the two nodes in the original graph $\mathcal{G}$, with a direct path defined as a path where only the last action is a request the state

ι. This edge is weighted by the sum of the discounted cumulative probabilities over direct paths:

$$\text{edge}(b, s) = \sum_{\substack{h \in \Phi(b,s) \\ h \text{ is direct}}} \gamma^{d(h)} \mathbf{P}(h|b, \pi_{\mathcal{G}})$$

For nodes $n, n' \in N'$, we define $\bar{\Psi}(n, n')$ as 0 if there is no edge between n and n', and as the weight of the edge otherwise. As the paths starting in b_0 and ending in a corner belief s are either direct or pass through another corner belief s', we can write for all corner beliefs s:

$$\Psi(b_0, s) = \sum_{s' \in \mathcal{S}} \Psi\big(b_0, s'\big) \bar{\Psi}\big(s', s\big) + \bar{\Psi}(b_0, s) \tag{9}$$

We can rewrite Eq. 9 more compactly into Eq.10 using matrix notations by defining $\bar{\Psi}_{s,s'} = \bar{\Psi}(s, s')$, $\bar{\Psi}_{b_0}$ as the vector with components equal to $\bar{\Psi}(b_0, s)$, and Ψ_{b_0} the vector with components equal to $\Psi(b_0, s)$. From which we obtain the solution given in Eq. 11.

$$\Psi_{b_0} = \bar{\Psi}\Psi_{b_0} + \bar{\Psi}_{b_0} \tag{10}$$

$$\Psi_{b_0} = (I - \bar{\Psi})^{-1}\bar{\Psi}_{b_0} \tag{11}$$

We note that this solution could be seen as constructing the Markov chain corresponding to the corner beliefs and finding the stationary distribution of the policy $\hat{\pi}_{\mathcal{G}}$. We can now compute $\Psi(b_0, b)$ for all fringes $b \in \mathcal{F}(\mathcal{G})$ (Eq. 8), and expand $\tilde{b}(\mathcal{G})$ (Eq. 7).

Algorithm 2 includes the pseudo-code, where line 3 calls Algorithm 3 (see Appendix) to obtain $\bar{\Psi}$ and $\bar{\Psi}_{b_0}$. This computation enables the calculation of Ψ_{b_0} (line 4) and returns $\tilde{b}(\mathcal{G})$ for expansion.

Algorithm 1: AEMS-SR: Anytime Error Minimization Search with State Requests

input: t: Maximum time, ε: Error threshold

1 **while** *not `EnvironmentTerminated()`* **do**
2 Initialize $\mathcal{G}$ with initial belief state b_0^0 as root
3 $t_0 \leftarrow$ `Time()`
4 **while** *`Time()` - $t_0 \leqslant t$ and not `Solved`*(b_0^0, ε) **do**
   `// Calls Algorithm 2`
5 $b^* \leftarrow$ `GetBeliefToExpand`$(\mathcal{G})$
6 `Expand`(b^*)
7 `UpdateAncestors`(b^*)
8 $\hat{a}_0 \leftarrow \arg\max_{a \in \{\bar{\iota}, \iota\}} L_{\mathcal{G}}(b_0^0, a)$
9 **if** $\hat{a}_0 = \iota$ **then**
10 $s \leftarrow$ `GetState()`
11 $\hat{a}_1 \leftarrow \arg\max_{a \in \mathcal{A}} L_{\mathcal{G}}(s, a)$
12 **else**
13 $\hat{a}_1 \leftarrow \arg\max_{a \in \mathcal{A}} L_{\mathcal{G}}(b_0^1, a)$
14 `DoAction`$(\hat{a}_1)$; $o \leftarrow$ `GetObservation()`
15 $b_0^0 \leftarrow \tau(s, \hat{a}_1, o)$ `if` $\hat{a}_0 = \iota$ `else` $\tau\big(b_0^0, \bar{\iota}, \hat{a}_1, o\big)$
   `// Potentially improve the bounds`

Algorithm 2: getBeliefNodeToExpand

input: Graph $\mathcal{G}$, root belief b_0

1 **if** $b_0 \in \mathcal{F}(\mathcal{G})$ **then**
2 **return** b_0
`/* Call to Algorithm 3 (Appendix) that traverse the graph to compute` $\bar{\Psi}, \bar{\Psi}_{b_0}, \mathcal{F}(\mathcal{G})$ `*/`
3 $\mathcal{V}, \bar{\Psi}, \bar{\Psi}_{b_0}, \mathcal{F}(\mathcal{G}) \leftarrow$ `GWalk`$(b_0, \{\}, 0_{|\mathcal{S}|\times|\mathcal{S}|}, 0_{|\mathcal{S}|}, \{\})$
4 $\Psi_{b_0} = (I_{|\mathcal{S}|} - \bar{\Psi})^{-1}\bar{\Psi}_{b_0}$ (Eq. 11)
5 bestE = $-\infty$
6 **for** $b \in \hat{\mathcal{F}}$ **do**
7 E = $(U(b) - L(b)) \cdot \Psi(b_0, b)$ *(Eq. 8)*
8 **if** *E* > *bestE* **then**
9 bestE = E
10 $b_{\text{best}} = b$
11 **return** b_{best}

Update ancestors consists of leveraging the knowledge gained through a belief expansion to update the lower and upper bound in the graph $L_{\mathcal{G}}$ and $U_{\mathcal{G}}$ by enforcing Equations 2 and 3. As explained in LAO* (Hansen & Zilberstein, 2001), this requires running dynamic programming. When working with stochastic trees, such as AEMS, the dynamic programming results in updating the parents sequentially until reaching the root, guaranteeing convergence in a number of steps equal to the depth of the tree. In contrast, when dealing with cyclic graphs, a full dynamic programming update is needed, which can be computationally expensive. To address this challenge, LAO* proposes a method to limit the number of nodes to update by focusing on a subset of nodes. In our implementation, instead of traversing the graph to determine the set of nodes to consider for dynamic programming, we update the parents recursively until the updates become smaller than a threshold (10^{-6} in our experiments), and we maintain a queue to deal with the cyclic structure.

The cycle of identifying the next belief to expand, expanding it, and backtracking the lower and upper bounds continues until the predefined time limit is reached or the error gap at the root, $\hat{e}_{\mathcal{G}}(b_0)$,

becomes less than ε. Subsequently, the agent determines whether to request the state and selects an environmental action, by maximizing the lower bound $L_{\mathcal{G}}$. Then the agent obtains a new observation, triggering the reinitialization of the graph with the updated belief.

5 Bounds

AEMS-SR requires a lower and upper bound L, U for Eq. 2, 3 and 6. Those bounds, computed offline, are represented as a set Γ of $|\mathcal{A}|$ α-vectors, one for each action a and denoted as α_a. Evaluation of the bounds on a belief b is given by $\max_{\alpha\in\Gamma} \langle b, \alpha\rangle$. Typically, the lower bound is derived from Blind policies (Hauskrecht, 1997) that consistently select the same action. For the upper bound, the two main algorithms are QMDP (Cassandra et al., 1997) and FIB (Hauskrecht, 2000). As the agent retains the option not to request the state, the ability to request the state cannot reduce the expected return but it may potentially increase it. Therefore, ensuring the validity of the upper bounds in POMDP-SRs is crucial.

Q-MDP is constructed by assuming that the uncertainty about the state will disappear after one step and corresponds to solving the underlying MDP. The $\alpha-$vector α_a is the fixed point of Eq. 12, and $\alpha_a(s)$ corresponds to the Q-Value $Q(s, a)$ of the underlying MDP.

$$\alpha_a(s) = \mathcal{R}(s, a) + \gamma \sum_{s'\in\mathcal{S}} \mathbf{P}\big(s' \mid s, a\big) \max_{\alpha_{a'}\in\Gamma} \alpha_{a'}\big(s'\big) \tag{12}$$

Lemma 3. *The Q-MDP upper bound of a standard POMDP $\mathcal{P}$ is an upper bound for the equivalent POMDP $\mathcal{P}'$ of $\mathcal{P}_{SR}$.*

Proof. In the underlying MDP $\mathcal{M}'$ of the Equivalent POMDP, the optimal policy will avoid state requests due to the penalty and the full observability. Consequently, the optimal policies of both MDPs only differ by the inclusion of non-request actions, which carry zero reward and thus do not affect the expected return. Hence, for any $s \in \mathcal{S}$, the value of the optimal policy in $\mathcal{M}$ at s matches the one in $\mathcal{M}'$ at s^0. Therefore QMDP is an upper bound for the values of $\mathcal{P}_{\text{SR}}$. □

Fast Informed Bound (FIB) considers the partial observability at the next step which provides a tighter upper bound compared to Q-MDP. The associated α-vectors are constructed by iteratively applying the operator associated to Eq. 13.

$$\alpha_a(s) = \mathcal{R}(s, a) + \gamma \sum_{o\in\Omega} \max_{\alpha_{a'}\in\Gamma} \sum_{s'\in\mathcal{S}} P\big(s', o \mid s, a\big)\alpha_{a'}\big(s'\big) \tag{13}$$

Lemma 4. *The Fast Informed Bound upper bound of a standard POMDP $\mathcal{P}$ is not guaranteed to be an upper bound for $\mathcal{P}_{SR}$'s equivalent POMDP $\mathcal{P}'$.*

Proof. Consider a POMDP $\mathcal{P}$ with two states, a uniform probability transition function, a unique observation, and two actions such that $\mathcal{R}(s_1, a_1) = \mathcal{R}(s_2, a_2) = 1$, $\mathcal{R}(s_1, a_2) = \mathcal{R}(s_2, a_1) = -1$. Q-MDP returns $(\frac{\gamma}{1-\gamma} + 1, \frac{\gamma}{1-\gamma} - 1)$ and $(\frac{\gamma}{1-\gamma} - 1, \frac{\gamma}{1-\gamma} + 1)$ as α-vectors, which correspond to the uncertainty of the initial state ± 1 and observing to the following states. Conversely, FIB returns $(1, -1)$ and $(-1, 1)$ as α-vectors corresponding to the first reward followed by an expected future return of 0. Let us now consider the POMDP-SR $\mathcal{P}_{\text{SR}} = \langle \mathcal{P}, c\rangle$ and the policy that always request the state to then select the optimal action. This policy has an expected discounted return equal to $\frac{1-c}{1-\gamma}$. Setting the cost c to 0.1 proves that FIB is not an upper bound for POMDP-SR. □

FIB-SR Adapting FIB to POMDP-SR involves introducing an additional α-vector, α_c, corresponding to the action of requesting the state. FIB-SR alternates between updating α-vectors for environmental actions using Eq. 13 with $\Gamma = \{\alpha_a, \forall a \in \mathcal{A}\} \cup \alpha_c$ and updating α_c using Eq. 14. This process reflects paying the cost c to observe the state and then selecting the environmental action.

$$\alpha_c(s) = -c + \max_{a\in\mathcal{A}} \alpha_a(s) \tag{14}$$

Table 1: Experiment results comparing POMCP, AEMS and AEMS-SR with a time limit of 0.1s, 0.5s and 1s on Robot Delivery (with 3, 5 and 7 corridors and cost $c = 0.1$) and Tag (cost $c = 1$). AEMS and AEMS-SR use the FIB-SR upper bound and **do not improve the offline bounds during planning**. We report the mean and standard error for the return and the Error Reduction (ER), and the mean for the Number of Expansions (NE).

		Return			Number of Expansions		Error Reduction (%)	
	T	POMCP	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR
RobotDelivery 3	0.1	0.97±0.00	0.98±0.00	**1.38±0.01**	55	**2525**	3.8±0.0	**24.1±0.4**
	0.5	0.97±0.00	0.98±0.00	**2.49±0.03**	111	**11150**	4.4±0.0	**42.4±0.1**
	1.0	0.97±0.00	0.98±0.00	**2.50±0.03**	152	**20035**	4.6±0.0	**43.7±0.1**
RobotDelivery 5	0.1	0.95±0.00	0.96±0.00	**1.00±0.01**	29	**1292**	3.2±0.0	**7.6±0.2**
	0.5	0.95±0.00	0.96±0.00	**1.01±0.01**	59	**6668**	3.7±0.0	**8.6±0.2**
	1.0	0.95±0.00	0.96±0.00	**1.45±0.01**	80	**13347**	4.0±0.0	**32.3±0.3**
RobotDelivery 7	0.1	**0.93±0.00**	**0.93±0.00**	**0.93±0.00**	17	**696**	2.5±0.0	**6.2±0.0**
	0.5	0.93±0.00	**0.94±0.00**	**0.94±0.00**	37	**4024**	3.4±0.0	**6.9±0.0**
	1.0	0.93±0.00	**0.94±0.00**	**0.94±0.00**	51	**8071**	3.6±0.0	**7.1±0.0**
Tag	0.1	-17.43±0.06	-6.30±0.06	**-4.66±0.06**	20	**67**	54.7±0.1	**58.8±0.1**
	0.5	-17.45±0.06	-5.35±0.08	**-4.56±0.09**	56	**566**	**67.8±0.1**	64.2±0.1
	1.0	-17.47±0.06	-5.35±0.09	**-4.50±0.09**	79	**1124**	**69.6±0.1**	64.9±0.1

Improving the bounds during learning

In traditional POMDPs, maintaining offline-computed bounds unchanged during online phases is standard practice. While updating these bounds could enhance the algorithm's efficiency over successive time steps and episodes, this approach is generally not feasible. The primary obstacle is the need to store additional alpha vectors, which would diminish the efficiency of computing bounds for new beliefs and increase memory demands. Conversely, in POMDP-SRs, corner beliefs present an opportunity to update bounds efficiently. For every corner belief s in the graph $\mathcal{G}$ and action a, $U_{\mathcal{G}}(s, a)$ provides a tighter bound than $\alpha_a(s)$ computed offline. Therefore, by replacing $\alpha_a(s)$ with the value of $U_{\mathcal{G}}(s, a)$, we can update the upper bound without requiring additional memory. The same approach applies to lower bounds, resulting in a practical and efficient solution.

6 Experiments

Many existing POMDP benchmarks, like RockSample (Smith & Simmons, 2004), feature partial observability that can be permanently eliminated with a single state request, thereby rendering the problem trivial. We evaluate AEMS-SR on Tag, where partial observability is restored at the next timestep, and on RobotDelivery, a new benchmark tailored to POMDP-SR.

RobotDelivery is a grid-world (Appendix F), featuring a main room (3, $2n + 1$) with, at the top, n corridors, each two units long and leading to a package pickup point. At the beginning of each episode, the agent starts at (A) and a package is in one of the pickup-points, with equal probability. Its mission is to collect the package and deliver it to point (D), receiving a reward of 1 for each successful delivery. After each delivery, there is a probability e that no new packages will spawn. Package spawning occurs with probability $1 - t$ into the waiting area (W) and with probability t in one of the pickup points. If a package is in the waiting area, it has a probability t of being transferred to a pickup point. The waiting area forces the agent to time its state request as requesting the state when the package is in (W) would force the agent to request it again. The agent has four possible actions (up, down, left, right). Except when moving into a package location or delivery location, actions have a failure probability f causing the agent to remain stationary. There are 5 observations, the first four indicate the number of walls surrounding the agent (0-3), and the fifth occurs when going up to a pickup point with a package or going down into the delivery zone.

Tag is a grid-world ($|\mathcal{S}| = 842, |\Omega| = 30$) where the agent chases a moving prey (Pineau et al., 2003). The agent observes its own position if not in the same tile as the prey and a special observation otherwise. There are 5 actions, 4 corresponding to cardinal directions, each resulting in a reward of -1, and one action to tag the prey yielding -10 if not in the same tile and $+10$ otherwise.

Table 2: Experiment results comparing AEMS and AEMS-SR with a time limit of 0.1s, 0.5s and 1s on Robot Delivery (with 3, 5 and 7 corridors and cost $c = 0.1$) and Tag (cost $c = 1$). AEMS and AEMS-SR use the FIB-SR upper bound and **improve the offline bounds during planning**. We report the mean and standard error for the return and the Error Reduction (ER), and the mean for the Number of Expansions (NE).

		Return		Number of Expansions		Error Reduction (%)	
	T	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR
RobotDelivery 3	0.1	0.98±0.00	**2.33±0.02**	54	**2170**	3.8±0.0	**72.6±0.5**
	0.5	0.98±0.00	**2.11±0.01**	111	**8106**	4.4±0.0	**77.2±0.5**
	1.0	0.98±0.00	**2.03±0.01**	151	**13279**	5.7±0.2	**79.2±0.5**
RobotDelivery 5	0.1	0.96±0.00	**2.17±0.02**	29	**1252**	3.2±0.0	**52.2±0.4**
	0.5	0.96±0.00	**2.25±0.02**	59	**5853**	3.7±0.0	**64.8±0.6**
	1.0	0.96±0.00	**2.21±0.02**	80	**10581**	4.0±0.0	**67.4±0.6**
RobotDelivery 7	0.1	0.94±0.00	**1.84±0.02**	17	**713**	2.5±0.0	**41.1±0.5**
	0.5	0.94±0.00	**2.14±0.02**	37	**3729**	3.4±0.0	**53.2±0.5**
	1.0	0.94±0.00	**2.16±0.02**	52	**7227**	3.6±0.0	**56.5±0.5**
Tag	0.1	-6.11±0.06	**-4.83±0.07**	20	**68**	60.8±0.1	**70.6±0.1**
	0.5	-5.41±0.09	**-4.26±0.09**	55	**571**	73.7±0.1	**80.8±0.1**
	1.0	-5.23±0.09	**-4.53±0.09**	78	**1144**	74.6±0.1	**82.2±0.1**

Successfully tagging the prey terminates the episode. The prey observes both positions, staying in place with a 0.2 probability, and only moves to increase its distance from the agent.

Setup We evaluate AEMS-SR on RobotDelivery environments with 3, 5, and 7 corridors, resulting in state spaces of 133, 305, and 541, respectively, along with Tag. For RobotDelivery, we set $\gamma = 0.99$, $f = 0.1$, $t = 0.8$, and $e = 1/3$ to achieve an expected total number of packages as 3. For Tag, we use $\gamma = 0.95$. We tested 0.1s, 0.5s, and 1s as the time per action T. We compare against AEMS and POMCP, both running on the Equivalent POMDP, and modified to ensure a fair evaluation by considering the structure of the Equivalent POMDP. AEMS and AEMS-SR both utilize the FIB-SR upper bound. Additional implementation details and experiments, including the Q-MDP upper bound, can be found in the Appendix. We use R-n and R-n-T to refer to RobotDelivery with n corridors and T seconds of compute time, and T-T for Tag.

Metrics For AEMS and AEMS-SR, in addition of the discounted return, we report the Number of Expansion (NE) and the Error Reduction (ER): $1 - (U_{\mathcal{G}}(b_0) - L_{\mathcal{G}}(b_0))/(U(b_0) - L(b_0))$. We note that the return is not necessarily correlated with the error reduction.

Tables 1 and 2 report the results for POMCP, AEMS, and AEMS-SR, without and with bound improvements during planning, respectively.

In RobotDelivery, POMCP consistently yields an average return below 1, opting to exit the room immediately without delivering any packages or making state requests. This poor performance is attributed to the sparse rewards. In Tag, POMCP generally fails to tag the prey and obtains an average return inferior to -17.

Results without Improving the Bounds In RobotDelivery, AEMS exhibits a strategy similar to POMCP's. In contrast, AEMS-SR achieves an average return of at least 1 in R-3 and R-5 environments, indicating successful package delivery before exiting. Notably, in R-3, with a minimum of 0.5s per step, AEMS-SR's return increases to 2.49, reflecting multiple deliveries. However, in the more complex R-7 scenario, AEMS-SR reverts to an exit-immediately strategy, suggesting potential areas for further enhancement. In Tag, AEMS performs better than POMCP, as the agent manages to tag the prey, but it is still outperformed by AEMS-SR.

The ER metric, which improves as T grows, aligns with the strategy of expanding the belief deemed to have the greatest impact on reducing the error gap at the root. Interestingly, in Tag, AEMS exhibits a higher ER than AEMS-SR. This is attributed to AEMS episodes being longer, and the initial steps having a very low ER.

To understand the superior performance of AEMS-SR, we can compare its NE against AEMS. AEMS-SR conducts up to two orders of magnitude more updates. The reason of this difference

is that AEMS's belief expansion spawns a new sub-tree for each state in the support (Fig. 1a). This requires computing many belief updates which is time-consuming. AEMS-SR by leveraging the graph structure reuses the already expanded corner beliefs, avoiding unnecessary computations. This empirically proves the advantage of representing the search space as a graph instead of a tree.

Results with Improving the Bounds As detailed in Section 5, the offline bounds can be easily improved by using the corner beliefs. For AEMS, however, such refinement has minimal impact on the average return due to the limited number of update steps available for substantial bound improvement. In contrast, AEMS-SR displays notable performance gains in RobotDelivery. Particularly in the R-7 environment, moving beyond the exit-immediately strategy it delivers packages and obtains an average return over 2 if given at least 0.5s and 1.84 otherwise. This improvement highlights the efficacy of AEMS-SR combined with online bounds enhancement. In some instances of AEMS-SR, we observe a significant decrease in NE compared to the non-bounds-improving variant, suggesting that AEMS-SR reaches an ε solution before the allocated time ends. Additionally, the ER metric, based on the original offline bounds, is higher in scenarios with bounds improvement. Both observations are positive indicators of the algorithm's efficiency.

Overall, the experiments demonstrate AEMS-SR's superior performance over AEMS and POMCP in POMDP-SRs, emphasizing the potential for bounds improvement during the learning phase.

7 Related Work

Heuristic search Other heuristic search algorithms are notable in the realm of online planning for POMDPs. The approach by Satia & Lave (1973) employs a branch and bound strategy and utilizes a heuristic similar to AEMS, wherein fringe beliefs are weighted by their likelihood of observation. A key distinction, however, is that all non-dominated actions are deemed equally probable. The BI-POMDP algorithm (Washington, 1997) aligns more closely with AO*, and therefore AEMS(-SR), focusing only on fringe nodes accessible with a greedy policy which selects the action that maximizes the upper bound, akin to our Equation 5. Unlike AEMS, BI-POMDP does not impose additional weighting on the probability of reaching a particular fringe node and instead prioritizes node expansion based on maximizing the error gap. In this work, we decided to extend over AEMS because it was shown to be more efficient (Ross et al., 2008). However, our approach is not limited to AEMS and could be applied to other heuristic search algorithms.

State requests in POMDPs have seen growing research interest. Bellinger et al. (2021) developed the AMRL framework, where agents incur a cost to request *the next state*. This framework, unlike our POMDP-SR, delays state access and doubles the action space instead of separating the two decision steps. Their AMRL-Q algorithm, based on Q-learning (Watkins & Dayan, 1992), focuses on state-conditioned policies rather than histories or beliefs, making it sub-optimal. ACNO-MDPs (Nam et al., 2021) differ from AMRL by not providing observations without state requests, thus simplifying belief updates. They propose two RL methods: 'observe-before-planning', combining initial MDP learning with subsequent POMCP application, and 'observe-while-planning', where POMCP or DVRL (Igl et al., 2018) in its deep learning variant, make decisions on state requests and environmental actions. Krale et al. (2023) further investigate ACNO-MDPs, focusing on timing state requests through heuristics. While these approaches offer valuable insights, they contrast with our methodology of planning with a pre-known model and striving for an ε-optimal solution.

8 Conclusion

To address environments where the agent can obtain full state information before each action at a cost, we introduce the POMDP with State Requests framework. Within this framework, we present AEMS-SR, a principled algorithm that effectively tackles the exponential growth challenge in POMDP-SR tree-based search by employing a cyclic graph structure. Our theoretical analysis proved that AEMS-SR is complete and ε-optimal. Empirical evaluation in RobotDelivery — a novel benchmark designed for POMDP-SR — and Tag demonstrates AEMS-SR's superior performance compared to established algorithms, AEMS and POMCP, in POMDP-SR settings. In future work, we aim to develop policies $\hat{\pi}_{\mathcal{G}}$ tuned to the specificity of POMDP-SR, and we plan to investigate the potential application of AEMS-Loop to other subclasses of POMDPs.

Acknowledgments

Raphaël Avalos is supported by the Research Foundation – Flanders (FWO), under grant number 11F5721N, and under grant number V418823N for the research visit at TU Delft. This research was supported by funding from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" program. We would like to thank all the members of TU Delft's Interactive Intelligence group for making this research visit enjoyable and fruitful. Finally, we would like to thank Willem Röpke for his help in proofreading the paper.

References

Karl Johan Åström. Optimal control of markov processes with incomplete state information i. *Journal of mathematical analysis and applications*, 10:174–205, 1965.

Colin Bellinger, Rory Coles, Mark Crowley, and Isaac Tamblyn. Active measure reinforcement learning for observation cost minimization. In *Canadian Conference on AI*, 2021.

Anthony Cassandra, Michael L. Littman, and Nevin L. Zhang. Incremental pruning: A simple, fast, exact method for partially observable markov decision processes. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, UAI'97, pp. 54–61, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc. ISBN 1558604855.

Benjamin E Childs, James H Brodeur, and Levente Kocsis. Transpositions and move groups in monte carlo tree search. In *2008 IEEE Symposium On Computational Intelligence and Games*, pp. 389–395. IEEE, 2008.

Eric A. Hansen and Shlomo Zilberstein. Lao*—: A heuristic search algorithm that finds solutions with loops. *Artificial Intelligence*, 129(1):35–62, 2001. ISSN 0004-3702. doi: https://doi.org/10.1016/S0004-3702(01)00106-0. URL `https://www.sciencedirect.com/science/article/pii/S0004370201001060`.

M. Hauskrecht. Value-function approximations for partially observable markov decision processes. *Journal of Artificial Intelligence Research*, 13:33–94, aug 2000. doi: 10.1613/jair.678. URL `https://doi.org/10.1613%2Fjair.678`.

Milos Hauskrecht. Incremental methods for computing bounds in partially observable markov decision processes. In *AAAI/IAAI*, pp. 734–739, 1997.

Maximilian Igl, Luisa Zintgraf, Tuan Anh Le, Frank Wood, and Shimon Whiteson. Deep variational reinforcement learning for pomdps. In *International Conference on Machine Learning*, pp. 2117–2126. PMLR, 2018.

Merlijn Krale, Thiago D. Simao, and Nils Jansen. Act-then-measure: Reinforcement learning for partially observable environments with active measuring. *Proceedings of the International Conference on Automated Planning and Scheduling*, 33(1):212–220, Jul. 2023. doi: 10.1609/icaps.v33i1.27197. URL `https://ojs.aaai.org/index.php/ICAPS/article/view/27197`.

HyunJi Nam, Scott L Fleming, and Emma Brunskill. Reinforcement learning with state observation costs in action-contingent noiselessly observable markov decision processes. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL `https://openreview.net/forum?id=jgze2dDL9y8`.

Nils J Nilsson. *Principles of artificial intelligence*. Springer Science & Business Media, 1982.

Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Point-based value iteration: An anytime algorithm for pomdps. In *Proceedings of 18th International Joint Conference on Artificial Intelligence (IJCAI '03)*, pp. 1025 – 1032, August 2003.

S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32:663–704, jul 2008. doi: 10.1613/jair.2567. URL `https://doi.org/10.1613%2Fjair.2567`.

Stéphane Ross and Brahim Chaib-Draa. Aems: An anytime online search algorithm for approximate policy refinement in large pomdps. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pp. 2592–2598, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

JK Satia and RE Lave. Markovian decision processes with probabilistic observation of states. *Management Science*, 20(1):1–13, 1973.

David Silver and Joel Veness. Monte-carlo planning in large pomdps. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta (eds.), *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010. URL `https://proceedings.neurips.cc/paper/2010/file/edfbe1afcf9246bb0d40eb4d8027d90f-Paper.pdf`.

Trey Smith and Reid Simmons. Heuristic search value iteration for pomdps. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pp. 520–527, 2004.

Adhiraj Somani, Nan Ye, David Hsu, and Wee Sun Lee. Despot: Online pomdp planning with regularization. *Advances in neural information processing systems*, 26, 2013.

Edward Jay Sondik. *The optimal control of partially observable Markov processes*. Stanford University, 1971.

Richard Washington. Bi-pomdp: Bounded, incremental partially-observable markov-model planning. In *European Conference on Planning*, pp. 440–451. Springer, 1997.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8:279–292, 1992.

A Equivalent POMDP

In this section, we explain the technicalities of the transformation of a POMDP-SR $\mathcal{P}_{\text{SR}} = \langle \mathcal{P}, c \rangle$ into an equivalent POMDP $\mathcal{P}' = \langle \mathcal{S}', \Omega', \mathcal{A}', \mathbf{A}', \mathbf{P}', \mathcal{O}', \mathcal{R}', \gamma' \rangle$ with variable action space and with $\mathbf{P}', \mathcal{O}', \mathcal{R}'$ defined only over legal actions.

- The state space $\mathcal{S}' \equiv \{0, 1\} \times \mathcal{S}$ augments the original state space with a binary indicator i, which equals 0 when the agent needs to decide whether to request the state and 1 for selecting the environmental action. To ease notations, we add the binary indicator in superscript s^i.
- The observations space $\Omega' = \Omega \cup \mathcal{S} \cup \{o^*\}$; o^* is a special observation associated with not requesting the state.
- The action space $\mathcal{A}' \equiv \{\bar{\iota}, \iota\} \cup \mathcal{A}$ includes additional actions: ι for requesting the state, and $\bar{\iota}$ not to request it. The set of legal actions is returned by $\mathbf{A}$ which is defined as follows $\mathbf{A}\left(s^0\right) = \{\bar{\iota}, \iota\}$, $\mathbf{A}\left(s^1\right) = \mathcal{A}$. We note that, since at each time step the states in the support of the belief have always the same binary indicator i, $\mathbf{A}$ can be extended to beliefs.
- The transition function $\mathbf{P}'$ is defined for all $s, s' \in \mathcal{S}$ as follows, $\mathbf{P}'\left(s'^0 | s^1, a\right) = \mathbf{P}(s' | s, a)$ and $\mathbf{P}'\left(s'^1 | s^0, a\right) = 1_s(s')$, $\mathbf{P}'\left(s'^i | s^i, a\right) = 0$, with 1 the indicator function,
- The observation function $\mathcal{O}' : \mathcal{S} \times \mathcal{A} \to \Delta_o$ is defined for legal actions as follows, $\mathcal{O}'\left(s'^0, a\right) = \mathcal{O}(s', a)$, $\mathcal{O}'\left(s'^1, \iota\right)$ is a dirac distribution centered in s', and $\mathcal{O}'\left(s'^1, \bar{\iota}\right)$is a dirac distribution centered in o^*
- The reward function $\mathcal{R}'$ is defined for legal actions as follows: $\mathcal{R}'\left(s^0, a\right) = -1_\iota(a)c/\sqrt{\gamma}$, $\mathcal{R}'\left(s^1, a\right) = \mathcal{R}(s, a)$
- $\gamma' = \sqrt{\gamma}$

B Example of Optimal Action Divergence between MDP, POMDP and POMDP-SR

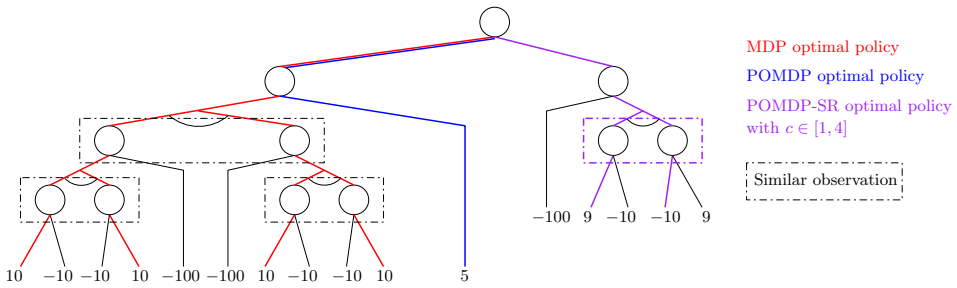


Figure 2: Tree representing an environment where the MDP and POMDP optimal action are the same but if the agent can request the state for a cost $1 \leqslant c \leqslant 4$ the optimal action changes. Circles represent states, actions are left a_1 and right a_2,

As discussed in the Framework section, the optimal action in a POMDP-SR scenario may deviate from that in an MDP or POMDP, even when their optimal actions are aligned. This divergence is attributed to the capability of requesting state information in future steps. To illustrate this concept, we present a toy environment, depicted in Figure 2, where such a divergence occurs.

In our example environment, states producing the same observations are enclosed within dashed boxes. Initially, at state s_0, both MDP and POMDP strategies suggest executing action 1, which leads to expected returns of 10 and 5, respectively. However, in the POMDP-SR, the calculation of Q-values yields $\max(5, 10 - 2c)$ for action 1 and $9 - c$ for action 2. Consequently, action 2 emerges as the optimal choice when the cost values c lie within the range of $[1, 4]$. This example demonstrates how POMDP-SR compels the agent to operate with a forward-looking approach, considering potential future state requests. This adds a layer of complexity to the decision-making process, differing significantly from situations in classic POMDPs where state requests are either limited to the current timestep or entirely absent.

C Notations

We starts by restating some definitions and notations.

- We denote trees as $\mathcal{T}$ and graphs as $\mathcal{G}$.
- $\mathcal{F}(\mathcal{G})$ is the set of nodes in graph $\mathcal{G}$ that does not have any children. We use the same notation for trees $\mathcal{F}(\mathcal{T})$.
- Corner beliefs corresponds to beliefs where one of the state has probability 1. We use the notation s to refer both to the state and the associated corner belief.
- $\Phi_{\mathcal{G}}(b_0, b)$ is the set of paths in the graph $\mathcal{G}$ that starts on b_0 and finishes on b.
- Paths h are sequences of beliefs, action and observations $(b_i, a_i, o_{i+1})_{i<T}$ with $T = d(h)$ being its length. We write paths that start in b_1 and ends in b_2 as $h_{b_1}^{b_2}$.
- $P(h|b_0, \pi) = \Pi_{i=0}^{T-1} P(o_{i+1}|b_i, a_i)\pi(a_i|b_i)$ corresponds to the probability of observing the path h while starting at the root belief b_0 and following the policy π.
- $\Psi_{\pi}^{\mathcal{G}}(b_0, b) = \sum_{h \in \Phi_{\mathcal{G}}(b_0,b)} \gamma^{d(h)} P(h|b_0, \pi)$ corresponds to the sum over all possible paths between the root belief b_0 and a belief b of the probability of observing the path discounted by its length.
- For any belief $b \in \mathcal{B}$ and lower bound L, we defined the error gap as $e(b) = V^*(b) - L(b)$
- For any belief $b \in \mathcal{B}$, lower bound L and upper bound U, we defined the approximate error gap as $\hat{e}(b) = U(b) - L(b) \geqslant e(b)$

D Proofs

Theorem 5. *In any rooted graph $\mathcal{G}$ with root b_0 where values are computed according to Equation 3 using a lower bound value function L, bounded bellow, with error $e(b) = V^*(b) - L(b)$, the error on the root belief state is bounded by: $e_{\mathcal{G}}(b_0) = V^*(b_0) - L_{\mathcal{G}}(b_0) \leqslant \sum_{b \in \mathcal{F}(\mathcal{G})} \Psi_{\pi *}(b_0, b) e(b)$. where $e(b) = V^*(b) - L(b)$.*

Proof. Consider an arbitrary node $b \in \mathcal{G} \backslash \mathcal{F}(\mathcal{G})$ in a graph $\mathcal{G}$ that is not a fringe, and $a_b^* = \arg\max_a Q^*(b, a)$ the optimal action. By definition $\gamma \in [0, 1)$. If $\gamma = 0$, then $L(b) = V^*(b) = r(b, a_b^*)$ and $e_{\mathcal{G}}(b) = 0$ which conclude the proof. Therefore let us focus on $\gamma \in (0, 1)$. By definition of $L_{\mathcal{G}}$ we have $L_{\mathcal{G}}(b, a_b^*) \leqslant L_{\mathcal{G}}(b)$.

$$
\begin{aligned}
e_{\mathcal{G}}(b) &= V^*(b_0) - L_{\mathcal{G}}(b_0) \\
&\leqslant V^*(b_0) - L_{\mathcal{G}}(b_0, a_b^*) \\
&\leqslant r(b, a_b^*) + \gamma \sum_{o \in \Omega} P(o|b, a_b^*) V^*(\tau(b, a_b^*, o)) - \left(r(b, a_b^*) + \gamma \sum_{o \in \Omega} P(o|b, a_b^*) L_{\mathcal{G}}(\tau(b, a_b^*, o)) \right) \\
&\leqslant \gamma \sum_{o \in \Omega} P(o|b, a_b^*) e_{\mathcal{G}}(\tau(b, a_b^*, o))
\end{aligned}
$$

This result in the following inequality:

$$
e_{\mathcal{G}}(b) \leqslant \begin{cases} e(b) & \text{if } b \in \mathcal{F}(\mathcal{G}) \\ \gamma \sum_{o \in \Omega} P(o|b, a_b^*) e_{\mathcal{G}}(\tau(b, a_b^*, o)) & \text{otherwise} \end{cases} \tag{15}
$$

Let us now consider $n \in \mathbb{N}^*$, and unroll the rooted graph $\mathcal{G}$ into a tree $\mathcal{T}_n$ with root b_0 and with maximum depth n. We define the following elements:

- Similar to our definition of Φ, we define, for all $b' \in \mathcal{G}$, $\Phi_n(b_0, b')$ the set of paths starting from b_0 and finishing in any of the replicas of b' in $\mathcal{T}_n$. It is important to note that the length of the paths in Φ_n are bounded by n. We have $\lim_{n \to +\infty} \Phi_n(b_0, b') = \Phi(b_0, b)$ the (possibly infinite) set of paths between b_0 and b' in G.

- For any $b' \in \mathcal{G}$ and any policy π, $\Psi^n_\pi(b_0, b') = \sum_{h \in \Phi_n(b_0,b')} \gamma^{d(h)} \mathbf{P}(h|b_0, \pi)$
- $\mathcal{F}_n$ as the fringes nodes of $\mathcal{T}_n$
- $\mathcal{F}_n^{\mathcal{G}}$ is the set of nodes of $\mathcal{F}_n$ that are replicas of a node in $\mathcal{F}(\mathcal{G})$
- $\bar{\mathcal{F}}_n^{\mathcal{G}} = \mathcal{F}_n \backslash \mathcal{F}_n^{\mathcal{G}}$
- $e_n = e_{\mathcal{T}_n}$

We note that Equation 15 hold for any graph including $\mathcal{T}_n$. Therefore, by solving the recurrence in $\mathcal{T}_n$ for b_0 we obtain (as in the proof of Theorem 1 of AEMS):

$$\begin{aligned} e_n(b_0) &\leqslant \sum_{b \in \mathcal{F}_n} \gamma^{d(b_0,b)} P\big(h^b_{b_0}|b_0, \pi^*\big) e(b) \\ &\leqslant \sum_{b \in \mathcal{F}_n^{\mathcal{G}}} \gamma^{d(b_0,b)} P\big(h^b_{b_0}|b_0, \pi^*\big) e(b) + \sum_{b \in \bar{\mathcal{F}}_n^{\mathcal{G}}} \gamma^{d(b_0,b)} P\big(h^b_{b_0}|b_0, \pi^*\big) e(b) \\ &\leqslant \sum_{b \in \mathcal{F}(\mathcal{G})} \Psi^n_{\pi^*}(b_0, b) e(b) + \sum_{b \in \bar{\mathcal{F}}_n^{\mathcal{G}}} \gamma^{d(b_0,b)} P\big(h^b_{b_0}|b_0, \pi^*\big) e(b) \end{aligned}$$

We note that for all $b \in \bar{\mathcal{F}}_n^{\mathcal{G}}$ the associated history h in $\mathcal{T}_n$ is of length n. Indeed, if the history size was shorter than n, b would also be a fringe in $\mathcal{G}$ which is impossible by definition of $\bar{\mathcal{F}}_n^{\mathcal{G}}$. It follows that:

$$\begin{aligned} e_n(b_0) &\leqslant \sum_{b \in \mathcal{F}(\mathcal{G})} \Psi^n_{\pi^*}(b_0, b) e(b) + \gamma^n \sum_{b \in \bar{\mathcal{F}}_n^{\mathcal{G}}} P\big(h^b_{b_0}|b_0, \pi^*\big) e(b) \\ &\leqslant \sum_{b \in \mathcal{F}(\mathcal{G})} \Psi^n_{\pi^*}(b_0, b) e(b) + \gamma^n \sup_{b'} e\big(b'\big) \\ &\qquad \text{(the sup exists because } V^* \text{ is bounded above and } L \text{ bellow)} \end{aligned}$$

In the limit, when $n \to +\infty$ we have $e_n \to e_{\mathcal{G}}$, $\Psi^n \to \Psi$, and $\gamma^n \to 0$ as $\gamma \in (0, 1)$, leading to

$$e_{\mathcal{G}}(b_0) \leqslant \sum_{b \in \mathcal{F}(\mathcal{G})} \Psi_{\pi^*}(b_0, b) e(b)$$

□

Definition 6. *We define the approximate error contribution of a fringe node $b \in \mathcal{F}(\mathcal{G})$ on the value at the root b_0 as*

$$E(b, b_0, \mathcal{G}) = \Psi_{\hat{\pi}_{\mathcal{G}}}(b_0, b) \hat{e}(b)$$

Lemma 7. *In any graph $\mathcal{G}$, the approximate error contribution $E(b, b_0, \mathcal{G})$ of a belief node b is bounded by $E(b, b_0, \mathcal{G}) \leqslant \gamma^{d_b} \sup_{b'} \hat{e}(b')$ with $d_b = \min_{h \in \Phi_{\mathcal{G}}(b_0,b)} d(h)$.*

Proof.

$$\begin{aligned} E(b, b_0, \mathcal{G}) &= \Psi_{\hat{\pi}_{\mathcal{G}}}(b_0, b) \hat{e}(b) \\ &= \sum_{h \in \Phi(b_0,b)} \gamma^{d(h)} \mathbf{P}(h|b_0, \hat{\pi}_{\mathcal{G}}) \hat{e}(b) \\ &\leqslant \gamma^{d_b} \sum_{h \in \Phi(b_0,b)} \mathbf{P}(h|b_0, \hat{\pi}_{\mathcal{G}}) \hat{e}(b) \\ &\leqslant \gamma^{d_b} \sup_{b'} \hat{e}\big(b'\big) \sum_{h \in \Phi(b_0,b)} \mathbf{P}(h|b_0, \hat{\pi}_{\mathcal{G}}) \\ &\leqslant \gamma^{d_b} \sup_{b'} \hat{e}\big(b'\big) \end{aligned}$$

□

Definition 8. *We define the set of accessible fringe nodes of a graph $\mathcal{G}$ under $\hat{\pi}_{\mathcal{G}}$ as $\hat{\beta}(\mathcal{G}) = \{b | b \in \mathcal{F}(\mathcal{G})$ and $\exists h \in \Phi_{\mathcal{G}}(b_0, b), P(h|b_0, \hat{\pi}_{\mathcal{G}}) > 0\}$. And the set of possible histories $\hat{\zeta}(b_0, \mathcal{G}) = \{h | P(h|b_0, \hat{\pi}_{\mathcal{G}}) > 0\}$. We define, for any $b \in \mathcal{G}$, the set of possible histories between the root belief b_0 and b as $\Phi^p_{\mathcal{G}}(b_0, b) = \{h^b_{b_0} | h^b_{b_0} \in \Phi_{\mathcal{G}}(b_0, b) \bigcap \hat{\zeta}(b_0, \mathcal{G})\}$*

Definition 9. *For all histories $h = (b_i, a_i, o_{i+1})_{i<T}$, where $T = d(h)$ is the length of the history, we define the observation probability $P(h|b_0) = \Pi_{i=0}^{T-1} P(o_{i+1}|b_i, a_i)$*

Lemma 10. *Given U bounded above and L bounded bellow such as $U(b) \geqslant V^*(b) \geqslant L(b)$, and $\hat{e}(b) = U(b) - L(b)$ for all $b \in \mathcal{B}$, then for any graph $\mathcal{G}$, $\varepsilon > 0$ and $D \in \mathbb{N}^*$ such that $\gamma^D \sup_b \hat{e}(b) \leqslant \varepsilon$, if for all $b \in \hat{\beta}(\mathcal{G})$ and for all $h \in \Phi^p_{\mathcal{G}}(b_0, b)$, either $d(h) > D$ or there exists an ancestor $b' \in h$ such that $\hat{e}_{\mathcal{G}}(b') \leqslant \varepsilon$, then $\hat{e}_{\mathcal{G}}(b_0) \leqslant \varepsilon$.*

Proof. For any graph $\mathcal{G}$, and any belief b that is not a fringe belief node $b \in \mathcal{G} \backslash \mathcal{F}(\mathcal{G})$. We define as $\hat{a}^{\mathcal{G}}_b = \arg\max_{a \in \mathcal{A}} U_{\mathcal{G}}(b, a)$.

$$
\begin{aligned}
\hat{e}_{\mathcal{G}}(b) &= U_{\mathcal{G}}(b) - L_{\mathcal{G}}(b) \\
&\leqslant U_{\mathcal{G}}\big(b, \hat{a}^{\mathcal{G}}_b\big) - L_{\mathcal{G}}\big(b, \hat{a}^{\mathcal{G}}_b\big) \\
&\leqslant r\big(b, \hat{a}^{\mathcal{G}}_b\big) + \gamma \sum_{o \in \Omega} P\big(o|b, \hat{a}^{\mathcal{G}}_b\big) U_{\mathcal{G}}\big(\tau\big(b, \hat{a}^{\mathcal{G}}_b, o\big)\big) - \left(r\big(b, \hat{a}^{\mathcal{G}}_b\big) + \gamma \sum_{o \in \Omega} P\big(o|b, \hat{a}^{\mathcal{G}}_b\big) L_{\mathcal{G}}\big(\tau\big(b, \hat{a}^{\mathcal{G}}_b, o\big)\big)\right) \\
&\leqslant \gamma \sum_{o \in \Omega} P\big(o|b, \hat{a}^{\mathcal{G}}_b\big) \hat{e}_{\mathcal{G}}\big(\tau\big(b, \hat{a}^{\mathcal{G}}_b, o\big)\big)
\end{aligned}
$$

We obtain the following upper bound for $b \in \mathcal{G}$ on $\hat{e}_{\mathcal{G}}(b)$:

$$
\hat{e}_{\mathcal{G}}(b) \leqslant \begin{cases} \hat{e}(b) & \text{if } b \in \mathcal{F}(\mathcal{G}) \\ \varepsilon & \text{if } \hat{e}_{\mathcal{G}}(b) \leqslant \varepsilon \\ \gamma \sum_{o \in \Omega} P(o|b, a^*_b) \hat{e}_{\mathcal{G}}\big(\tau\big(b, \hat{a}^{\mathcal{G}}_b, o\big)\big) & \text{otherwise} \end{cases} \tag{16}
$$

We define :

- $\hat{e}_{\mathcal{G}}\big(h^b_{b_0}\big) = \hat{e}_{\mathcal{G}}(b)$
- $A(\mathcal{G})$ is the set of possible histories $h^b_{b_0} \in \hat{\zeta}(b_0, \mathcal{G})$ of length $d\big(h^b_{b_0}\big) < D$ such that $\hat{e}_{\mathcal{G}}(b) \leqslant \varepsilon$.
- $B(\mathcal{G})$ is the set of possible histories $h^b_{b_0} \in \hat{\zeta}(b_0, \mathcal{G})$ of length $d\big(h^b_{b_0}\big) < D$ such that $b \in \mathcal{F}(\mathcal{G})$, and that for all intermediary $b' \in h^b_{b_0}$, the partial history $h^{b'}_{b_0} \notin A(\mathcal{G})$.
- $C(\mathcal{G})$ is the set of all possible histories $h^b_{b_0} \in \hat{\zeta}(b_0, \mathcal{G})$ of size at least D, that do not belong to $B(\mathcal{G})$ and for all intermediary $b' \in h^b_{b_0}$, the partial history $h^{b'}_{b_0} \notin A(\mathcal{G})$.

As for all $b \in \hat{\beta}(\mathcal{G})$ and for all $h \in \Phi^p_{\mathcal{G}}(b_0, b)$ either $d(h) > D$ or there exists an ancestor $b' \in h$ such that $\hat{e}_{\mathcal{G}}(b') \leqslant \varepsilon$, $B(\mathcal{G})$ is empty.

By unfolding the recurrence above we obtain:

$$
\begin{aligned}
\hat{e}_{\mathcal{G}}(b_0) &= \sum_{h^b_{b_0} \in A(\mathcal{G})} \gamma^{d\left(h^b_{b_0}\right)} P\big(h^b_{b_0}|b_0\big) \hat{e}_{\mathcal{G}}(b) + \sum_{h^b_{b_0} \in C(\mathcal{G})} \gamma^{d\left(h^b_{b_0}\right)} P\big(h^b_{b_0}|b_0\big) \hat{e}_{\mathcal{G}}(b) \\
&\leqslant \varepsilon \sum_{h^b_{b_0} \in A(\mathcal{G})} P\big(h^b_{b_0}|b_0\big) + \sum_{h^b_{b_0} \in C(\mathcal{G})} \gamma^{d\left(h^b_{b_0}\right)} P\big(h^b_{b_0}|b_0\big) \hat{e}_{\mathcal{G}}(b) \\
&\leqslant \varepsilon \sum_{h^b_{b_0} \in A(\mathcal{G})} P\big(h^b_{b_0}|b_0\big) + \gamma^D \sup_b \hat{e}(b) \sum_{h^b_{b_0} \in C(\mathcal{G})} P\big(h^b_{b_0}|b_0\big)
\end{aligned}
$$

$$
\begin{aligned}
&\leqslant \varepsilon \sum_{h^b_{b_0} \in A(\mathcal{G})} P\big(h^b_{b_0}|b_0\big) + \varepsilon \sum_{h^b_{b_0} \in C(\mathcal{G})} P\big(h^b_{b_0}|b_0\big) \\
&\leqslant \varepsilon \sum_{h^b_{b_0} \in A(\mathcal{G}) \cup C(\mathcal{G})} P\big(h^b_{b_0}|b_0\big) \\
&\leqslant \varepsilon
\end{aligned}
$$

□

Theorem 11. *Given U bounded above and L bounded bellow such as $U(b) \geqslant V^*(b) \geqslant L(b)$, and $\hat{e}(b) = U(b) - L(b)$ for all $b \in \mathcal{B}$, if $\gamma \in [0,1)$ and $\inf_{b,\mathcal{G}|\hat{e}_\mathcal{G}(b)>\varepsilon} \hat{\pi}_\mathcal{G}\big(b, \hat{a}^\mathcal{G}_b\big) > 0$ for $\hat{a}^\mathcal{G}_b = \arg\max_{a\in\mathcal{A}} U_\mathcal{G}(b,a)$, then the AEMS-Loop algorithm using heuristic $\tilde{b}(\mathcal{G})$ is complete and $\varepsilon-$ optimal.*

Proof. Consider an arbitrary $\varepsilon > 0$ and the current root belief b_0. If $\gamma = 0$, then after one expansion $\hat{e}_\mathcal{G}(b_0) = 0$ since $U_\mathcal{G}(b_0) = L_\mathcal{G}(b_0) = \max_{a\in\mathcal{A}} r(b_0, a)$. And therefore AEMS-Loop is complete and $\varepsilon-$optimal.

Lets focus on $\gamma \in (0,1)$. Because U is bounded above and L bellow, $\sup_b \hat{e}(b)$ exists and there exists $D \in \mathbb{N}$ such that $\gamma^D \sup_b \hat{e}(b) < \varepsilon$. We define the following elements:

- $\mathcal{A}^\mathcal{G}\big(h^b_{b_0}\big)$ the set of ancestors beliefs of b in the history $h^b_{b_0}$
- $\mathcal{A}^\mathcal{G}(b) = \bigcup_{h^b_{b_0} \in \Phi^p_\mathcal{G}(b_0,b)} \mathcal{A}^\mathcal{G}\big(h^b_{b_0}\big)$ the set of ancestors beliefs of b across possible histories.
- $\hat{e}^{min}_\mathcal{G}(A) = \min_{b\in A} \hat{e}_\mathcal{G}(b)$ for any finite set of beliefs A.
- $\mathcal{G}_b = \{\mathcal{G} | \mathcal{G} \text{ is finite}, b \in \hat{\beta}(b_0, \mathcal{G}), \max_{h\in\Phi^p_\mathcal{G}(b_0,b)} \hat{e}^{min}_\mathcal{G}\big(\mathcal{A}^\mathcal{G}(h)\big) > \varepsilon\}$ Intuitively, $\mathcal{G}_b$ is the set of finite graphs $\mathcal{G}'$ with root b_0 for which b is an accessible fringe node, i.e. that can be attained with non zero probability under the policy $\hat{\pi}_{\mathcal{G}'}$, and for which there exist an history $h^b_{b_0}$ such that all the ancestors beliefs b' in that history have an approximate error gap $\hat{e}_{\mathcal{G}'}(b') > \varepsilon$. The existence of the max relies on the graph being finite.
- $\mathcal{B} = \{b | \hat{e}(b) \inf_{\mathcal{G}\in\mathcal{G}_b} \sum_{h\in\Phi_\mathcal{G}(b_0,b)|d(h)\leqslant D} P\big(h^b_{b_0}|b_0, \hat{\pi}_\mathcal{G}\big) > 0\}$

The assumption $\inf_{b,\mathcal{G}|\hat{e}_\mathcal{G}(b)>\varepsilon} \hat{\pi}_\mathcal{G}\big(b, \hat{a}^\mathcal{G}_b\big) > 0$ ensures that $\mathcal{B}$ contains all the beliefs states b within depth D such that (i) $\hat{e}(b) > 0$, (ii) there exists a finite graph $\mathcal{G}$ where $b \in \hat{\beta}(b_0, \mathcal{G})$ and for which there exists an history $h^b_{b_0} \in \Phi^p_\mathcal{G}(b_0, b)$ such that all ancestors $b' \in \mathcal{A}\big(h^b_{b_0}\big)$ have $\hat{e}_\mathcal{G}(b') > \varepsilon$.

As there are only a finite number of beliefs for which there exists an history of size smaller than D, $\mathcal{B}$ is finite. This allows us to define $E_{min} = \min_{b\in\mathcal{B}} \hat{e}(b) \inf_{\mathcal{G}\in\mathcal{G}_b} \sum_{h^b_{b_0}\in\Phi_\mathcal{G}(b_0,b)} \gamma^{d\left(h^b_{b_0}\right)} P\big(h^b_{b_0}|b_0, \hat{\pi}_\mathcal{G}\big)$.

By construction $E_{min} > 0$. We also know that for any graph $\mathcal{G}$, all beliefs $b \in \mathcal{B} \cap \hat{\beta}(b_0, \mathcal{G})$ have an approximate error contribution $E(b, b_0, \mathcal{G}) \geqslant E_{min}$

$$
E(b, b_0, \mathcal{G}) = \Psi_{\hat{\pi}_\mathcal{G}}(b_0, b)\hat{e}(b) = \sum_{h^b_{b_0}\in\Phi(b_0,b)} \gamma^{d\left(h^b_{b_0}\right)} P\big(h^b_{b_0}|b_0, \hat{\pi}_\mathcal{G}\big)\hat{e}(b) \geqslant E_{\min}
$$

As $\gamma \in (0,1)$ and $E_{min} > 0$, there exist $D' \in \mathbb{N}^+$ such that $\gamma^{D'} \sup_b \hat{e}(b) < E_{min}$. Therefore, we know from Lemma 7 that AEMS-Loop cannot expand any node of depth D' or more before expanding a graph $\mathcal{G}$ where $\mathcal{B} \cap \hat{\beta}(b_0, \mathcal{G}) = \varnothing$.

As there exist a finite number of belief nodes for which an history starting from b_0 of length at most D' exists, it is clear that AEMS-Loop will reach such a graph $\mathcal{G}$ after a finite number of expansions.

Since, for this graph $\mathcal{G}$, $\mathcal{B} \cap \hat{\beta}(b_0, \mathcal{G}) = \varnothing$ we have that for all beliefs $b \in \hat{\beta}(b_0, \mathcal{G})$ the possible histories $h^b_{b_0} \in \Phi^p_\mathcal{G}(b_0, b)$ with length $d\big(h^b_{b_0}\big) \leqslant D$ have $\hat{e}^{min}_\mathcal{G}\big(\mathcal{A}^\mathcal{G}\big(h^b_{b_0}\big)\big) < \varepsilon$. Therefore, Lemma 10 ensures that $\hat{e}_\mathcal{G}(b_0) < \varepsilon$ and consequently AEMS-Loop will terminate with an ε-optimal solution in a finite number of expansions as $\hat{e}_\mathcal{G}$ is an upper bound of $e_\mathcal{G}$. □

E Algorithm to compute $\bar{\Psi}$ and $\bar{\Psi}_{b_0}$

Algorithm 3 presents the pseudo code to compute $\bar{\Psi}$ and $\bar{\Psi}_{b_0}$. The algorithm recursively traverse the graph by following $\hat{\pi}_{\mathcal{G}}$ and by maintaining two sets, one for the visited corner beliefs and one for the fringe beliefs.

Algorithm 3: GWalk: computing $\bar{\Psi}$ and $\bar{\Psi}_{b_0}$

input: belief b, visitedStates $\mathcal{V}$, stateMatrix M, $\bar{p}$, fringeNodes $\hat{\mathcal{F}}$, Graph $\mathcal{G}$, root belief b_0

if $b \in \mathcal{F}(\mathcal{G})$ **then**
  $\hat{\mathcal{F}} = \hat{\mathcal{F}} \cup \{b\}$
else if $\pi(b) = \iota$ **then**
  for $s \in supp(b)$ **do**
    if $\xi(b) = b_0$ **then**
      $\bar{p}[s]$ += $b[s] * \bar{\Psi}(\xi(b), b)$
    else
      $M_{\xi(b),s} += b[s] * \bar{\Psi}(\xi(b), b)$
    if $s \notin \mathcal{V}$ **then**
      $\mathcal{V} = \mathcal{V} \cup \{s\}$
      for $o \in \mathcal{O}(s, \pi(s))$ **do**
        $\mathcal{V}, M, \bar{p}, \hat{\mathcal{F}} \leftarrow$ `GWalk`$(\tau(s, \pi(s), o), \mathcal{V}, M, \bar{p}, \hat{\mathcal{F}})$
else
  for $o \in \mathcal{O}(b, \pi(b))$ **do**
    $\mathcal{V}, M, \bar{p}, \hat{\mathcal{F}} \leftarrow$ `GWalk`$(\tau(b, \pi(b), o), \mathcal{V}, M, \bar{p}, \hat{\mathcal{F}})$
return $\mathcal{V}, M, \bar{p}, \hat{\mathcal{F}}$

F Robot Delivery Environment

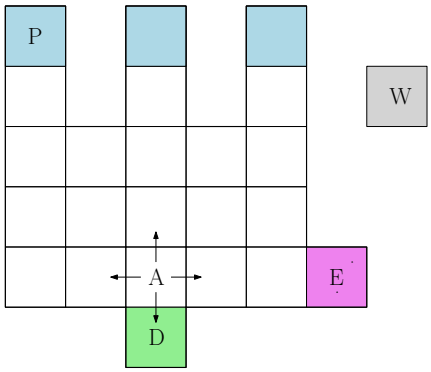


Figure 3: RobotDelivery (n=3): The agent (A) needs to pickup the package (P) that is in one of the tree possible locations (blue) and bring it to the delivery cell (D, green). Whenever a package is delivered a new one is put in the waiting area (W, grey) with probability w. It is then transferred to one of the pickup points with probability t. The agent can exit through the violet cell (E) at any time.

Table 3: Results for RobotDelivery for POMCP, AEMS and AEMS-SR with 3, 5 and 7 corridors, a time limit of 0.1s, 0.5 and 1s, a cost $c = 0.1$, probability of failure of movement $f = 0.1$, probability transfer from the waiting area $t = 0.8$, expected number of packages $e = 3$. POMCP results are averaged on 400 runs. AEMS and AEMS-SR use the Q-MDP upper bound and their results are averaged over 800 runs. We report the mean and standard error to the mean.

			Not Improve Bounds						Improve Bounds					
		Return	Return		NU		ER (%)		Return		NU		ER (%)	
		POMCP	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR
	T	±0.00	±0.00		±0		±0.0				±0			
R-3	0.1	0.97	0.98	**1.37±0.01**	56	**2473±3**	3.9	**22.6±0.4**	0.98±0.00	**2.33±0.02**	56	**2209±7**	4.0±0.0	**71.6±0.5**
	0.5	0.97	0.98	**2.48±0.03**	113	**10624±26**	4.5	**42.1±0.1**	0.98±0.00	**2.12±0.02**	113	**7994±86**	4.5±0.0	**78.1±0.5**
	1.0	0.97	0.98	**2.47±0.03**	156	**19142±49**	4.8	**41.7±0.1**	1.04±0.01	**2.00±0.01**	152	**13671±188**	15.8±0.7	**79.7±0.5**
R-5	0.1	0.95	0.96	**1.00±0.01**	28	**1287±2**	3.2	**7.7±0.2**	0.96±0.00	**2.21±0.02**	29	**1251±4**	3.3±0.0	**53.0±0.4**
	0.5	0.95	0.96	**1.01±0.01**	58	**6637±8**	3.8	**8.7±0.2**	0.96±0.00	**2.26±0.02**	59	**5918±28**	3.8±0.0	**65.9±0.6**
	1.0	0.95	0.96	**1.54±0.01**	79	**13348±14**	4.0	**35.7±0.2**	0.96±0.00	**2.21±0.02**	80	**10652±72**	4.1±0.0	**67.4±0.6**
R-7	0.1	0.93	**0.94**	**0.94±0.00**	17	**704±1**	2.5	**6.3±0.0**	0.94±0.00	**1.98±0.02**	17	**713±2**	2.6±0.0	**46.3±0.4**
	0.5	0.93	**0.94**	**0.94±0.00**	37	**4029±4**	3.4	**7.0±0.0**	0.93±0.00	**2.17±0.02**	37	**3785±13**	3.4±0.0	**53.9±0.5**
	1.0	0.93	**0.94**	**0.94±0.00**	51	**8042±8**	3.7	**7.2±0.0**	0.94±0.00	**2.16±0.02**	51	**7313±32**	3.7±0.0	**57.2±0.5**

G Experiment Supplementary Details

POMCP Our implementation of POMCP is an adaptation of the one provided by `https://github.com/Svalorzen/AI-Toolbox` to handle the variable action space of the extended POMDP. This allows us to avoid resorting to deterrent penalties for illegal actions, which can hinder learning.

AEMS Our implementation of AEMS is also tailored for the POMDP-SR structure. This prevents the need to double the state space, as done in the extended POMDP formulation, ensuring a fair evaluation as doubling the state space would slow the belief update computation.

Tag The original Tag environment has 870 states, while our implementation has 842. The difference arises from the number of terminal states; in the original implementation, they differentiate based on which tile the prey was successfully tagged, while we reduced them to a unique state. The evaluation is conducted by performing 400 runs on 10 initial states (for a total of 4000 runs). The initial states are kept identical for all algorithms.

H Additional experiments

Table 3 shows results for the same RobotDelivery instances as in the main paper, but using the Q-MDP upper bound. Generally, both upper bounds yield similar average results. A notable exception is observed in R-3-1, where AEMS with improved bounds achieves an average return of 1.04, suggesting a shift away from the exit-directly strategy in certain instances. This further underscores the benefit of improving bounds during the online phase.

Table 4 presents the results for a modified RobotDelivery version with a movement failure probability $f = 0.2$, also using the Q-MDP upper bound for ease of comparison. AEMS's performance remains unaffected by this change, with variations in return due to the longer expected exit time. AEMS-SR shows lower returns compared to the $f = 0.1$ scenario. This outcome was expected given that the increasing f results in expending the support of the beliefs and on increasing the necessary time to pickup and deliver packages. Nonetheless, AEMS-SR still outperforms AEMS and POMCP, both consistently adopting the exit-immediately strategy.

Table 5 details results for the RobotDelivery environment with a state request cost of $c = 0.25$, maintaining other parameters as in the main paper. Given our use of a greedy policy to approximate π^*, increasing the cost requires a more significant reduction in the no-request action's upper bound for AEMS-SR to consider state requests. This makes the problem more difficult. In contrast to POMCP and AEMS, which consistently opt for immediate exits, AEMS-SR still manages to deliver packages in R-3-1 without bounds improvement and in all R-3 instances if updating the offline bounds.

We note that EMS-SR's performance does not uniformly improve with increased compute time per step. Upon examining the results, we identified instances where AEMS-SR, while carrying a package in the corridor, consistently chooses the right action over the down action, resulting in the agent staying in place. This behavior occurs because both actions yield nearly identical optimal values,

Table 4: Results for RobotDelivery for POMCP, AEMS and AEMS-SR with 3, 5 and 7 corridors, a time limit of 0.1s, 0.5 and 1s, a cost $c = 0.1$, probability of failure of movement $\boldsymbol{f} = \mathbf{0.2}$, probability transfer from the waiting area $t = 0.8$, expected number of packages $e = 3$. POMCP results are averaged on 400 runs. AEMS and AEMS-SR use the Q-MDP upper bound and their results are averaged over 800 runs. We report the mean and standard error to the mean.

			Not Improve Bounds						Improve Bounds					
		Return	Return		NU		ER (%)		Return		NU		ER (%)	
		POMCP	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR
	T	±0.00	±0.00		±0		±0.0		±0.00		±0		±0.0	
R-3	0.1	0.97	0.98	**1.15±0.01**	55	**2289±5**	3.6	**13.6±0.4**	0.98	**2.43±0.03**	55	**2240±7**	3.6	**59.6±0.5**
	0.5	0.97	0.98	**2.45±0.03**	111	**10410±29**	4.1	**39.5±0.1**	0.97	**2.33±0.02**	112	**8627±66**	4.1	**70.4±0.5**
	1.0	0.97	0.98	**2.48±0.03**	151	**18972±63**	4.3	**41.8±0.1**	0.97	**2.23±0.02**	152	**14465±149**	4.3	**73.0±0.5**
R-5	0.1	0.94	0.95	**0.95±0.00**	28	**1302±2**	3.0	**6.3±0.0**	0.95	**0.99±0.01**	28	**1337±2**	3.1	**38.5±0.7**
	0.5	0.95	0.95	**0.97±0.00**	57	**6121±8**	3.6	**7.7±0.1**	0.95	**2.10±0.03**	58	**6105±22**	3.7	**50.4±0.6**
	1.0	0.95	0.95	**1.04±0.01**	78	**11433±27**	3.8	**11.0±0.3**	0.95	**2.19±0.02**	79	**11608±50**	3.9	**58.9±0.5**
R-7	0.1	0.92	**0.93**	**0.93±0.00**	16	**725±1**	2.4	**6.0±0.0**	**0.93**	**0.93±0.00**	16	**732±1**	2.5	**6.3±0.0**
	0.5	0.92	**0.93**	**0.93±0.00**	35	**4025±4**	3.2	**7.3±0.0**	0.93	**1.55±0.02**	36	**4023±11**	3.3	**39.3±0.7**
	1.0	0.92	**0.93**	**0.93±0.00**	49	**7711±7**	3.4	**7.7±0.0**	0.93	**1.89±0.02**	50	**7553±27**	3.5	**43.4±0.6**

Table 5: Results for RobotDelivery for POMCP, AEMS and AEMS-SR with 3, 5 and 7 corridors, a time limit of 0.1s, 0.5 and 1s, a cost $\boldsymbol{c} = \mathbf{0.25}$, probability of failure of movement $f = 0.2$, probability transfer from the waiting area $t = 0.8$, expected number of packages $e = 3$. POMCP results are averaged on 400 runs. AEMS and AEMS-SR use the Q-MDP upper bound and their results are averaged over 800 runs. We report the mean and standard error to the mean.

			Not Improve Bounds						Improve Bounds					
		Return	Return		NU		ER (%)		Return		NU		ER (%)	
		POMCP	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR	AEMS	AEMS-SR
	T	±0.00	±0.00	±0.00	±0		±0.0		±0.00		±0		±0.0	
R-3	0.1	0.93	**0.98**	**0.98**	54	**2416±3**	3.9	**6.9±0.0**	0.98	**2.38±0.03**	55	**2308±5**	3.9	**75.7±0.2**
	0.5	0.94	**0.98**	**0.98**	110	**12043±14**	4.5	**7.9±0.0**	0.98	**2.31±0.03**	111	**11220±29**	4.5	**78.7±0.2**
	1.0	0.94	0.98	**1.65**	150	**24867±27**	4.8	**37.6±0.1**	0.98	**2.42±0.03**	151	**22689±52**	4.8	**80.8±0.2**
R-5	0.1	0.93	**0.96**	**0.96**	28	**1317±2**	3.2	**6.7±0.0**	**0.96**	**0.96±0.00**	28	**1331±2**	3.2	**6.7±0.0**
	0.5	0.93	**0.96**	**0.96**	57	**6973±8**	3.8	**7.8±0.0**	**0.96**	**0.96±0.00**	59	**7051±7**	3.8	**7.9±0.0**
	1.0	0.93	**0.96**	**0.96**	79	**14020±15**	4.0	**8.3±0.0**	**0.96**	**0.96±0.00**	80	**14185±13**	4.1	**8.3±0.0**
R-7	0.1	0.92	**0.93**	**0.93**	17	**721±1**	2.5	**6.3±0.0**	**0.94**	0.93±0.00	17	**726±1**	2.6	**6.4±0.0**
	0.5	0.92	**0.94**	0.93	37	**4198±4**	3.4	**7.7±0.0**	**0.94**	**0.94±0.00**	37	**4231±4**	3.4	**7.7±0.0**
	1.0	0.92	**0.94**	**0.94**	50	**8567±7**	3.7	**8.2±0.0**	**0.94**	**0.94±0.00**	51	**8638±8**	3.7	**8.2±0.0**

differing only by a factor of γ, and that the dynamic programming approach used in backtracking converges to an ε-solution, potentially introducing minor errors that can persist over time due to bound updates.

Remark 1. All AEMS-SR and AEMS experiments were conducted on an Intel Xeon Gold CPU, utilizing a single core and less than 300Mb of RAM.