Planning under periodic observations: bounds and bounding-based solutions

Federico Rossi[1] and Dylan A. Shell[2]

Abstract— **We study planning problems faced by robots operating in uncertain environments with incomplete knowledge of state, and actions that are noisy and/or imprecise. This paper identifies a new problem sub-class that models settings in which information is revealed only intermittently through some exogenous process that provides state information periodically. Several practical domains fit this model, including the specific scenario that motivates our research: autonomous navigation of a planetary exploration rover augmented by remote imaging. With an eye to efficient specialized solution methods, we examine the structure of instances of this sub-class. They lead to Markov Decision Processes with exponentially large action-spaces but for which, as those actions comprise sequences of more atomic elements, one may establish performance bounds by comparing policies under different information assumptions. This provides a way in which to construct performance bounds systematically. Such bounds are useful because, in conjunction with the insights they confer, they can be employed in bounding-based methods to obtain high-quality solutions efficiently; the empirical results we present demonstrate their effectiveness for the considered problems. The foregoing has also alluded to the distinctive role that time plays for these problems —more specifically: time until information is revealed— and we uncover and discuss several interesting subtleties in this regard.**

I. Introduction

Autonomous robots are compelled to cope with uncertainty. The inherent imperfections of sensing and actuation, as well as the inevitable shortfalls of world models, mean that robots must select actions despite having only imprecise state information. Unfortunately, as is well known, the problem of planning under uncertainty in full generality remains out of practical reach—except in problem instances that are tiny or where planning horizons are short. In light of this predicament, this paper represents a campaign of attack focused on specialization: it aims at uncovering opportunities for development of efficient methods that produce high quality solutions, even if only for a restricted sub-class of planning problems. So long as the sub-class includes problems of practical value, such methods will have obvious utility. As motivation, we begin with a specific instance of signal interest to us.

Consider the autonomous rover in Figure 1 that is navigating across the surface of some remote asteroid, moon, or planetoid. Its objective is to reach a goal region efficiently in order to collect samples at that location for detailed analysis later. Even if it departs from a known position, the rover's knowledge of its pose rapidly becomes unreliable unless sensors can help circumscribe probable locations. Suppose

[1]Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109, USA. `federico.rossi@jpl.nasa.gov`

[2]Dept. of Comp. Sci. & Eng., Texas A&M University, College Station, TX 77843, USA. `dshell@tamu.edu`

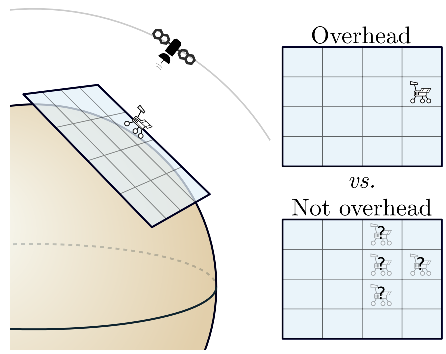


Fig. 1: Motivating example: a satellite orbiting a planetary body helps localize an autonomous rover that is tasked with operating on the body's surface. The rover executes a sequence of actions, but only obtains its state (shown as a definitive location within a specific cell) when the satellite is overhead. The rover must plan and act despite receiving observations that only arrive periodically.

that, along with the rover, a separate orbital device had also been deployed. This satellite carries surface-directed sensors that include a detector capable of picking up the rover. From its extrinsic perspective, as the satellite circles, it acquires information (e.g., imagery and ranging data) providing the rover's position. When the two are in communication range, there is the possibility of a *check-in* to provide the rover with its location.

In this scenario, the rover's knowledge of its state is sporadic: the data providing its pose are sparse, though regular, and when the check-ins do occur they resolve the rover's position. From the point of view of the rover, the process that generates observations is exogenous. The process's periodicity is known, which means that, even though the rover may not know the information it will receive (since it does not know where it is, precisely), it can be certain *when* the data will be received.

The traits present in the rover example —*viz.* infrequent but periodic observations of state— form a special sub-class of partially-observable planning problems. These same properties also appear in other robotic domains. For instance, marine robots operating in tidal regions may find their sensors inhibited by periodic phenomena (e.g., those driven by diurnal factors). Quite different instances arise when, to reduce the energy expended on radio transmission, a team of multiple robots employs a pre-determined synchronization and communication schedule. In fact, intermittency can have multiple advantages, such as in facilitating stealthy operation, desirable for robot operation in clandestine conditions.

One significant source of complexity in dealing with general partially-observable problems is that they involve balancing information gathering with reward-realizing activities. Being planning problems, there are (state- or beliefmediated) correlations across time which complicate the process of choosing actions. Any state-revealing observation process that is exogenous weakens what otherwise would be a tightly-coupled causal cycle. In problems like that in Figure 1, the robot's actions affect neither when it will receive an observation nor the quality of its estimate when it does; none of its actions can be said to gather more information than any other. But the cycle is not entirely severed either, because the actions do affect what is observed, i.e., what state the robot finds itself in when a check-in occurs. Critically, when the observation process is periodic, the robot may select actions —knowing when observations will arrive— so that it is at some juncture where what is discerned will be of most value.

A. Contribution and Organization

Our contribution is threefold. First, in Section II, we formally define the novel problem of decision-making under uncertainty with periodic information check-ins as a stochastic decision problem with the usual assumption of an underlying Markovian process. Second, in Sections III and IV, we derive upper and lower bounds for the state-action values of the problem that can be computed efficiently. Along the way, Section III-B presents an important example illustrating that more frequent observations are not always better. Third, in Section V we propose a branch-and-bound method that makes use of bounds to compute exact solutions to the PSO-MDP; Section VI presents numerical results that testify to the effectiveness of this algorithm, showing that it is significantly faster compared to a naive MDP approach. The final section, Section VII, presents our conclusions.

B. Related work

The framework of Markov Decision Processes (MDPs) is a useful basis for optimal control, planning, and learning in robots [2], [12]. Classic fully-observable problems have a rich history with a variety of effective solution techniques [1]; recent work has sought extensions to the basic MDP formulation to capture additional features including time-varying models [3] and more complex representations [4], as well as exploring various means to improve performance, especially in solving very large instances [14]. The settings we will consider are not fully-observable (except in the degenerate case with unit period), and it so might be better considered partially observable.

The general framework of Partially-Observable Markov Decision Processes (POMDPs) has been explored as a solution to robotics problems for the last two decades [**?**]; early attempts to apply the techniques of the day ended up highlighting the twin curses of history and dimensionality [8] as obstructions to the tractable solution of POMDPs. A subsequent and popular line of work then pursued policy-based approaches [7]. More recent work has employed online sampling-based methods to great effect, most notably [10] and [11], along with their descendants. These methods explore only those belief states that can be reached from the circumstances actually facing the robot, which helps increase the scale of problems that can be effectively attacked.

Still other techniques improve scalability further by treating what might be termed "intermediate" formulations, imposing constraints derived from other insights. In terms of observations: for instance, the locally observable MDPs of [6] consider observations derived from the readings of realistic sensors. There, when something is sensed, it is sensed well; when it is not observed, no data are obtained. In some ways this is akin to the periodic observations we treat, as the challenge is sparseness rather than degradation or corruption through random noise. In terms of actions: the options/macro-actions framework [13] considers aggregate actions, permitting a notion of hierarchical solution. They treat policies as macro-actions but, for periodic observations, no observations occur between the atomic actions, so we consider just simple sequences. Finally, unlike semi-Markov processes, no different mathematical machinery will be needed for problems with periodic observations, other than some few complexities raised with regard to discounting.

II. Preliminaries and basic definitions

We formally define the periodically state-observed Markov Decision Process (PSO-MDP) as follows.

Definition 1 (PSO-MDP)**.** *A periodically state-observed Markov Decision Process is a 5-tuple $\langle S, A, T, R, \kappa \rangle$ where*

- $S = \{s_0, s_1, \ldots, s_{|S|}\}$ *is the finite set of states;*
- $A = \{a_0, a_1, \ldots, a_{|A|}\}$ *is the finite set of actions;*
- $T : S \times A \times S \to [0,1]$ *is the transition dynamics, or transition model, describing the stochastic state transitions of the system, assumed to be Markovian in the states, where* $\forall t, P(s^{t+1} = s' | s^t = s, a^t = a) = T(s', a, s)$*;*
- $R : S \times A \to \mathbb{R}$ *is the function which prescribes that reward* $R(s,a)$ *is obtained for taking action a in state s;*
- $\kappa \in \mathbb{N}_{>0}$ *is the check-in period.*

The optimization objective is to maximize the expected discounted cumulative reward

$$U(s^0) = \mathbb{E}\left(\sum_{t=0}^{\infty} \gamma^t R\left(s^t, a^t\right)\right) \tag{1}$$

via selection of actions $a^0, a^1, a^2, \ldots$.

The key difference with respect to standard MDPs is that, when an agent's planning problem is modeled via a PSO-MDP, it must take actions at every time, but with the current state being disclosed only every κ steps: $t \in \{0, \kappa, 2\kappa, 3\kappa, \cdots,\}$. Between check-ins, the agent cannot directly observe its own state, and it must maintain a belief over its state and plan based on this belief. Following standard notation, in what follows we write $U^*(\cdot)$ for the value function that gives the maximal expected discounted cumulative reward at each state.

The PSO-MDP problem can be cast both as a Markov Decision Process with composite (or macro) actions, and as a POMDP with many uninformative observations.

A. Equivalent MDP formulation

To rigorously define a solution concept for an PSO-MDP (i.e., to show the concept of a policy is appropriate), we first need two definitions, which we shall re-use later too.

Definition 2 (transition composition)**.** *For some* $\kappa \in \mathbb{N}_{>0}$ *and transitions* $T : S \times A \times S \to [0,1]$ *the* κ-composed transition model *is the function* $T^{\kappa} : S \times A^{\kappa} \times S \to [0,1]$ *defined as*

$$T^{\kappa}\left(s', (a_0, \ldots, a_{\kappa-1}), s\right) = \sum_{\substack{(s_0, \ldots, s_{\kappa-1}) \in S^{\kappa} \\ \text{where } s_0 = s \\ \text{and } s_{\kappa-1} = s'}} \prod_{i=0}^{\kappa-1} T(s_{i+1}, a_i, s_i). \quad (2)$$

The transition model T describes the distribution of states reached after a single step, conditioned on a single action being issued. By unfurling copies of T, the κ-composed version, T^{κ}, describes the distribution of states reached after κ steps, now conditioned on a sequence of κ actions; we will write $\vec{a}$ for such sequences. By definition, $T = T^1$.

Definition 3 (Reward Composition)**.** *For some* $\kappa \in \mathbb{N}_{>0}$, *discounting factor* $\gamma \in [0,1)$, *and reward function* $R : S \times A \to \mathbb{R}$ *the* κ-composed γ-discounted reward *is the function* $R^{\kappa,\gamma} : S \times A^{\kappa} \to \mathbb{R}$ *defined as*

$$R^{\kappa,\gamma}(s, \vec{a}) = R^{\kappa,\gamma}(s, (a_0, a_1, \ldots a_{\kappa-1})) = \sum_{d=0}^{\kappa-1} \gamma^d \left(\sum_{\substack{(s_0, \ldots, s_d) \in S^{d+1} \\ \text{where } s_0 = s}} R(s_d, a_d) \prod_{i=0}^{d-1} T(s_{i+1}, a_i, s_i) \right). \quad (3)$$

The κ-composed version of the reward function is analogous to the transition composition, but with the additional complexity that the discount is incorporated as one runs along the length of the sequence. Note that, by definition $R = R^{1,\gamma}$. In circumstances, like this one here, where γ plays no role it will be elided and we will write R^1 only.

The κ-composed transition model (2) and the κ-composed rewards (3) can be computed recursively; one can show that the resulting computation time grows exponentially with the check-in period κ as $O((|\mathrm{S}| \cdot |\mathrm{A}|)^{\kappa})$.

We are now in a position to define an MDP equivalent to any PSO-MDP:

Definition 4 (Composite Action Process)**.** *Given* PSO-MDP $\mathscr{M} = \langle S, A, T, R, \kappa \rangle$, *its associated* composite action decision process *is* $\mathscr{M}_{cmp} = \langle S, A^{\kappa}, T^{\kappa}, R^{\kappa,\gamma}, 1 \rangle$.

The composite action process is an MDP because the Markov property is preserved when state sequences are gathered together, indicating that it has solution in the form of a mapping from states to κ-length sequences of actions, *viz.* a policy. By "solution" here, we mean actions that yield an optimal cumulative reward in expectation over the stochastic transition dynamics. Since $\mathscr{M}$ and $\mathscr{M}_{\text{cmp}}$ are really identical problems on the same Markov process, every PSO-MDP has a solution in the form of a policy. The optimal state-action values, or Q-values, can be computed as

$$Q^*(s, \vec{a}) = R^{\kappa,\gamma}(s, \vec{a}) + \gamma^{\kappa} \sum_{s' \in \mathrm{S}} T^{\kappa}(s', \vec{a}, s) \max_{\vec{a}' \in \mathrm{A}^{\kappa}} Q^*(s', \vec{a}'), \quad (4)$$

and the corresponding optimal policy can be computed as

$$\pi^*(s) = \arg\max_{\vec{a} \in \mathrm{A}^{\kappa}} Q^*(s, \vec{a}). \quad (5)$$

B. Equivalent POMDP formulation

The PSO-MDP can also be cast as a POMDP where the state $S_{\text{POMDP}} = \{s_0, s_1, \ldots, s_{|S|}\} \times \{[0, \ldots, \kappa - 1\}$ captures the PSO-MDP state and the time until the next check-in; the actions, transitions, and rewards are identical to the PSO-MDP's (with two minor exceptions: the transitions also update the time until the next check-in, and the rewards ignore the temporal portion of the state); and the observation function O_{POMDP} returns the current state at times corresponding to check-ins, and is uninformative otherwise.

C. Discussion

Note how, in the preceding, the POMDP treatment is a poor fit for the PSO-MDP sub-class of problems. We are required to inflate the state space to account for the check-in period because the observations depend on the time since the last check-in, but must be conditioned on state. Also, the expressive freedom which the POMDP does provide, a distribution in O_{POMDP}, can't be turned to advantage.

The composite action MDP suffers from problems too. Its action space is exponential in the size of the PSO-MDP's; indeed, as κ grows, the possibility of obtaining any solution in this form looks increasingly implausible. Part of the problem is that standard MDP solution techniques treat the action set as an opaque collection. The fact that these particular actions are sequences of more atomic actions suggests that it could be useful to consider interrelationships between solutions with differing actions. This motivates the search for upper and lower bounds, which follows next. However, to do so we find that it aids the intuition to adopt an information-oriented interpretation.

III. Upper Bounds

In this section, we explore how additional information check-ins can provide upper bounds on the value of PSO-MDP problems. First, we introduce an auxiliary definition.

Notation (Action sequence subset)**.** *Consider an action sequence* $\vec{a} = (a_0, a_1, \ldots, a_{\kappa-1})$ *of length* κ. *We denote as* $\vec{a}_{\ell:m}$ *the subsequence* $(a_{\ell}, a_{\ell+1}, \ldots, a_{m-1})$ *of length* $m - \ell$.

A. Bonus and extra check-ins

We start by assessing the value of receiving supererogatory check-ins in addition to the periodic check-ins that occur with period κ. We distinguish two situations: *announced extra check-ins*, when the availability of a future additional check-in is known in advance, and *unannounced bonus check-ins*, where the occurrence of the check-in is not anticipated.

Definition 5 (Unannounced bonus check-in)**.** *Consider an agent following an optimal* PSO-MDP *policy. Suppose that* τ *time steps after the last check-in (with* $\tau < \kappa$*), the agent receives an unanticipated check-in, which reveals its state; the agent can use this newly-disclosed bonus information to optimize the expected discounted reward.*

Definition 6 (Announced extra check-in)**.** *Suppose an agent is following an optimal* PSO-MDP *policy. At the time of a check-in, the agent is informed that it will receive an extra check-in after* $\tau < \kappa$ *time steps,* in addition *to the regularly scheduled check-ins. The agent can use this newly-disclosed information to optimize the expected discounted reward.*

The next two lemmas show that, perhaps unsurprisingly, both unannounced and announced check-ins do not decrease the expected reward, and announced check-ins never result in a lower reward compared to unannounced ones.

Lemma 1 (Unannounced bonus check-ins bound PSO-MDP values from above)**.** *Consider a* PSO-MDP *with an unannounced bonus check-in τ time steps after a regular check-in. The optimal policy that uses the information provided by the unannounced bonus check-in has an expected discounted reward no lower than the original* PSO-MDP *policy.*

Proof. The optimal reward for an agent in state s, τ time steps after the last check-in, when the bonus check-in occurs, can be computed as

$$U^*_{U,[\tau]}(s) = \max_{\vec{a}_{\tau:\kappa}\in \mathrm{A}^{\kappa-\tau}} \Bigg(R^{\kappa-\tau,\gamma}(s,\vec{a}_{\tau:\kappa}) + \gamma^{\kappa-\tau}\sum_{s'\in \mathrm{S}} T^{\kappa-\tau}(s',\vec{a}_{\tau:\kappa},s)U^*(s')\Bigg). \tag{6}$$

In contrast, in absence of the bonus check-in, the agent executes the tail of the action $\hat{\vec{a}} = \pi^*_{\text{PSO-MDP}}(s)$ computed at the last check-in, τ time steps before. The reward that results is just

$$R^{\kappa-\tau,\gamma}(s,\hat{\vec{a}}_{\tau:\kappa}) + \gamma^{\kappa-\tau}\sum_{s'\in \mathrm{S}} T^{\kappa-\tau}(s',\hat{\vec{a}}_{\tau:\kappa},s)U^*(s'). \tag{7}$$

Since $\hat{\vec{a}}_{\tau:\kappa} \in \mathrm{A}^{\kappa-\tau}$, $\hat{\vec{a}}_{\tau:\kappa}$ is an admissible solution to (6), and hence, the claim follows. □

Lemma 1 focuses on state values. In contrast, for announced extra check-ins, we start by providing a bound on Q-values as follows.

Lemma 2 (Announced extra check-ins bound PSO-MDP Q-values from above)**.** *Consider a* PSO-MDP *with an announced extra check-in τ time steps after a regular check-in. Denote the Q-value of a state-action pair $(s,\vec{a})$ under the optimal* PSO-MDP *policy that ignores the additional check-in as $Q^*(s,\vec{a})$; and denote the optimal Q-value of the state-action pair $(s,\vec{a}_{0:\tau})$ that uses the extra check-in information as $Q^*_1(s,\vec{a}_{0:\tau})$ (where the subscript refers to the fact that a single extra check-in is provided). Then, $Q^*_1(s,\vec{a}_{0:\tau}) \geq Q^*(s,\vec{a})$ for all $s \in \mathrm{S}, \vec{a} \in \mathrm{A}^\kappa$.*

Proof. The optimal Q-value of a state-action pair under a policy that uses the extra check-in information is

$$\begin{aligned} Q^*_1(s,\vec{a}_{0:\tau}) &= R^{\tau,\gamma}(s,\vec{a}_{0:\tau}) + \gamma^\tau \sum_{s'\in \mathrm{S}} T^\tau(s',\vec{a}_{0:\tau},s)U^*_{U,[\tau]}(s') = \\ & R^{\tau,\gamma}(s,\vec{a}_{0:\tau}) + \gamma^\tau \sum_{s'\in \mathrm{S}} T^\tau(s',\vec{a}_{0:\tau},s) \times \\ & \max_{\vec{a}_{\tau:\kappa}\in \mathrm{A}^{\kappa-\tau}} \Bigg[R^{\kappa-\tau,\gamma}(s',\vec{a}_{\tau:\kappa}) + \\ & \gamma^{\kappa-\tau}\sum_{s''\in \mathrm{S}} T^{\kappa-\tau}(s'',\vec{a}_{\tau:\kappa},s')\Big(\max_{\vec{a}''\in \mathrm{A}^\kappa} Q^*(s'',\vec{a}'')\Big)\Bigg]. \end{aligned} \tag{8}$$

Recall that Equation (4) captures the Q-value $Q^*(s,\vec{a})$ of a state-action pair under the optimal policy that ignores the extra check-in. Using Definition 3, rewrite the reward $R^{\kappa,\gamma}(s,\vec{a})$ in (4) as

$$R^{\kappa,\gamma}(s,\vec{a}) = R^{\tau,\gamma}(s,\vec{a}_{0:\tau}) + \gamma^\tau \sum_{s'\in \mathrm{S}} T^\tau(s',\vec{a}_{0:\tau},s)R^{\kappa-\tau,\gamma}(s',\vec{a}_{\tau:\kappa}), \tag{9}$$

and use Definition 2 to rewrite the reward-to-go as

$$\begin{aligned} &\sum_{s''\in \mathrm{S}} T^\kappa(s'',\vec{a},s)\max_{\vec{a}''\in \mathrm{A}^\kappa} Q^*(s'',\vec{a}'') = \\ &\sum_{s'\in \mathrm{S}} T^\tau(s',\vec{a}_{0:\tau},s)\sum_{s''\in \mathrm{S}} T^{\kappa-\tau}(s'',\vec{a}_{\tau:\kappa},s')\Big(\max_{\vec{a}''\in \mathrm{A}^\kappa} Q^*(s'',\vec{a}'')\Big). \end{aligned} \tag{10}$$

Replacing (9) and (10) in (4), we obtain

$$\begin{aligned} Q^*(s,\vec{a}) &= R^{\tau,\gamma}(s,\vec{a}_{0:\tau}) + \gamma^\tau \sum_{s'\in \mathrm{S}} T^\tau(s',\vec{a}_{0:\tau},s)R^{\kappa-\tau,\gamma}(s',\vec{a}_{\tau:\kappa}) + \\ &\gamma^\tau\gamma^{\kappa-\tau}\sum_{s'\in \mathrm{S}} T^\tau(s',\vec{a}_{0:\tau},s)\sum_{s''\in \mathrm{S}} T^{\kappa-\tau}(s'',\vec{a}_{\tau:\kappa},s')\Big(\max_{\vec{a}''\in \mathrm{A}^\kappa} Q^*(s'',\vec{a}'')\Big) \\ &= R^{\tau,\gamma}(s,\vec{a}_{0:\tau}) + \gamma^\tau \sum_{s'\in \mathrm{S}} T^\tau(s',\vec{a}_{0:\tau},s) \times \\ &\Bigg(R^{\kappa-\tau,\gamma}(s',\vec{a}_{\tau:\kappa}) + \gamma^{\kappa-\tau}\sum_{s''\in \mathrm{S}} T^{\kappa-\tau}(s'',\vec{a}_{\tau:\kappa},s')\Big(\max_{\vec{a}''\in \mathrm{A}^\kappa} Q^*(s'',\vec{a}'')\Big)\Bigg). \end{aligned} \tag{11}$$

Comparing (11) with (8), one can see that former is an admissible solution to the maximization problem in the latter: therefore, $Q^*_1(s,\vec{a}_{0:\tau}) \geq Q^*(s,\vec{a})$. □

We then use Lemma 2 to provide a bound on state values.

Lemma 3 (Announced extra check-ins bound PSO-MDP state values from above)**.** *Consider a* PSO-MDP *with an announced extra check-in τ time steps after a regular check-in. The optimal policy that uses the information provided by the announced extra check-in has an expected discounted reward no lower than the original* PSO-MDP *policy, and no lower than the reward from an unannounced bonus check-in at time τ.*

Proof. The optimal reward for an agent in state s when an announced extra check-in is revealed is

$$U^*_{[\tau]}(s) = \max_{\vec{a}_{0:\tau}\in \mathrm{A}^\tau} Q_1(s,\vec{a}_{0:\tau}). \tag{12}$$

In contrast, the expected discounted reward if no extra check-ins are available can be written as $U^*(s) = \max_{\vec{a}\in \mathrm{A}^\kappa} Q(s,\vec{a})$. Lemma 2 shows that $Q_1(s,\vec{a}_{0:\tau}) \geq Q(s,\vec{a}), \forall s \in \mathrm{S}, \vec{a} \in \mathrm{A}^\kappa$; the claim follows.

We also show that the reward (8) is no lower than the corresponding reward for an unannounced bonus. The reward in state s for an agent that will receive a bonus check-in after τ time steps (but does not know it yet) is

$$R^{\tau,\gamma}(s,\hat{\vec{a}}_{0:\tau}) + \gamma^\tau \sum_{s'\in \mathrm{S}} T^\tau(s',\hat{\vec{a}}_{0:\tau},s)U^*_{U,\tau}(s') = Q^*_1(s,\hat{\vec{a}}_{0:\tau})$$

where $\hat{\vec{a}} = \pi^*_{\text{PSO-MDP}}(s)$ follows the optimal policy in absence of check-ins. The action prefix $\hat{\vec{a}}_{0:\tau}$ is an admissible solution to the maximization problem in (12); the claim follows. □

Next, we consider the effect of adding announced extra check-ins after *every* regular check-in.

First, we provide an auxiliary definition.

Definition 7 (PSO-MDP with additional check-ins)**.** *Consider a* PSO-MDP *$\mathscr{M}$ with check-in period κ. We define a* PSO-MDP *with additional check-ins $\hat{\mathscr{M}}^\tau$ as a modification of* PSO-MDP *$\mathscr{M}$ where, after an action is taken, an* additional *check-in occurs after $\tau < \kappa$ steps. That is, a policy for $\hat{\mathscr{M}}^\tau$ specifies an action of length τ (to be taken if the previous action was of length $\kappa-\tau$), and an action of length $\kappa-\tau$ (to be taken if the previous action was of length τ) for every state.*

We denote as $Q(s,\vec{a}_{0:\tau})$ the Q-value of the state-action pair $s,\vec{a}_{0:\tau}$, i.e. the set of values that satisfy

$$Q^*(s,\vec{a}_{0:\tau}) = R^{\tau,\gamma}(s,\vec{a}_{0:\tau}) + \gamma^\tau \sum_{s' \in S} T^\tau(s',\vec{a}_{0:\tau},s) \max_{\vec{a}'_{\tau:\kappa} \in A^{\kappa-\tau}} Q^*(s',\vec{a}'_{\tau:\kappa}),$$

$$Q^*(s,\vec{a}_{\tau:\kappa}) = R^{\kappa-\tau,\gamma}(s,\vec{a}_{\tau:\kappa}) + \gamma^{\kappa-\tau} \sum_{s' \in S} T^{\kappa-\tau}(s',\vec{a}_{\tau:\kappa},s) \max_{\vec{a}'_{0:\tau} \in A^\tau} Q^*(s',\vec{a}'_{0:\tau}).$$

Theorem 4 (extra check-ins bound PSO-MDP Q-values from above)**.** *Suppose $\mathscr{M}$ is a* PSO-MDP *with check-in period κ, and denote the associated optimal Q-values as $Q^*(s,\vec{a})$. Also consider a modified* PSO-MDP *$\hat{\mathscr{M}}^\tau$ with additional check-ins at τ, as per Definition 7, and denote the associated state values as $\hat{Q}^*(s,\vec{a}_{0:\tau})$. Then, $Q^*(s,\vec{a}) \leq \hat{Q}^*(s,\vec{a}_{0:\tau})$ for all $s \in \mathrm{S}, \vec{a} \in \mathrm{A}^\kappa$.*

Proof. The proof is by induction on the number of extra check-ins. We define as $\hat{Q}^*_\ell(s,\vec{a})$ the optimal Q-value when ℓ consecutive extra check-ins are provided, with $\hat{Q}_0(s,\vec{a}_{0:\tau}) = Q(s,\vec{a})$, and $\hat{Q}_1(s,\vec{a}_{0:\tau})$ defined in Equation (8). We show that, for all $\ell \in \mathbb{N}$, $\hat{Q}^*_{\ell+1}(s,\vec{a}_{0:\tau}) \geq \hat{Q}^*_\ell(s,\vec{a}_{0:\tau})$, which implies that $\hat{Q}^*_{\ell+1}(s,\vec{a}_{0:\tau}) \geq \hat{Q}^*_0(s,\vec{a}_{0:\tau}) = Q^*(s,\vec{a})$.

Base case: One extra check-in, is Lemma 2.

Inductive case: We wish to show that, if $\hat{Q}^*_{\ell-1}(s,\vec{a}_{0:\tau}) \leq \hat{Q}^*_\ell(s,\vec{a}_{0:\tau})$, then $\hat{Q}^*_\ell(s,\vec{a}_{0:\tau}) \leq Q^*_{\ell+1}(s,\vec{a}_{0:\tau})$. We can express $\hat{Q}^*_{\ell+1}(s,\vec{a}_{0:\tau})$ as

$$\begin{aligned}\hat{Q}^*_{\ell+1}(s,\vec{a}_{0:\tau}) &= R^{\tau,\gamma}(s,\vec{a}_{0:\tau}) + \gamma^\tau \sum_{s' \in \mathrm{S}} T^\tau(s',\vec{a}_{0:\tau},s) \times \\ &\max_{\vec{a}'_{\tau:\kappa} \in \mathrm{A}^{\kappa-\tau}} \Bigg[R^{\kappa-\tau,\gamma}(s',\vec{a}_{\tau:\kappa}) + \\ &\gamma^{\kappa-\tau} \sum_{s'' \in \mathrm{S}} T^{\kappa-\tau}(s'',\vec{a}_{\tau:\kappa},s') \times \left(\max_{\vec{a}''_{0:\tau} \in \mathrm{A}^\tau} Q^*_\ell(s'',\vec{a}''_{0:\tau}) \right) \Bigg].\end{aligned} \tag{13}$$

Analogously, we can express $\hat{Q}^*_\ell(s,\vec{a}_{o:\tau})$ as

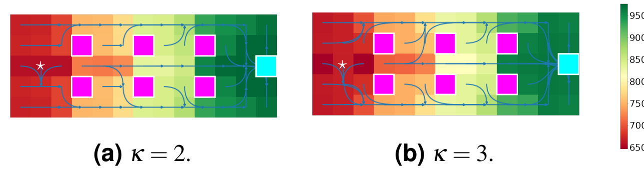
$$\begin{aligned}\hat{Q}^*_\ell(s,\vec{a}_{0:\tau}) &= R^{\tau,\gamma}(s,\vec{a}_{0:\tau}) + \gamma^\tau \sum_{s' \in \mathrm{S}} T(s',\vec{a}_{0:\tau},s) \times \\ &\max_{\vec{a}'_{\tau:\kappa} \in \mathrm{A}^{\kappa-\tau}} \Bigg[R^{\kappa-\tau,\gamma}(s',\vec{a}_{\tau:\kappa}) + \\ &\gamma^{\kappa-\tau} \sum_{s'' \in \mathrm{S}} T^{\kappa-\tau}(s'',\vec{a}_{\tau:\kappa},s') \times \left(\max_{\vec{a}''_{0:\tau} \in \mathrm{A}^\tau} Q^*_{\ell-1}(s'',\vec{a}''_{0:\tau}) \right) \Bigg].\end{aligned} \tag{14}$$

The expressions for $\hat{Q}^*_{\ell+1}$ and $\hat{Q}^*_\ell$ present the same nested maximization problems and have identical arguments except for $\hat{Q}^*_\ell$ in (13) and $\hat{Q}^*_{\ell-1}$ in (14); by the inductive assumption, $\hat{Q}^*_\ell \geq \hat{Q}^*_{\ell-1}$, therefore $\hat{Q}^*_{\ell+1} \geq \hat{Q}^*_\ell$ and the claim follows. □

The difference in expressions for values when check-ins are unannounced (Lemma 1) versus announced (Lemma 3), and indeed the numerical difference in their respective value functions, expresses the value of knowing beforehand that a check-in will occur. Like most information, this has value; but notice that announcements are second-order statements: they are statements *about* subsequent disclosures.

B. More frequent check-ins are not always beneficial

Next, in a perhaps unintuitive result, we show that increasing the frequency of check-ins is *not* guaranteed to bound state values from above. To show this, we provide a counterexample in Figure 2. In it, an agent is tasked with navigating a simple grid world with multiple ranks of obstacles (shown in purple) set three steps away from each other; the goal is to reach the cyan cell on the right. The agent's control is imprecise: when commanded to drive in a given direction, the agent also drifts to the left or to the right of the desired direction with 5% probability each.



(a) $\kappa = 2$. **(b)** $\kappa = 3$.

Fig. 2: State values and policy for a grid-world PSO-MDP. Increasing the check-in frequency can be detrimental to state values. The expected reward for the starred location in (b) is higher than that same location in (a).

We compare two PSO-MDPs with $\kappa = 2$ (Figure 2a) and $\kappa = 3$ (Figure 2b), respectively. When the time between check-ins is $\kappa = 3$, corresponding to the stride between obstacles, the state values (and therefore the Q-values) for many cells farthest from the goal are *higher* compared to the case with more frequent check-ins. (The star helps indicate one especially clear example.) Intuitively, more infrequent check-ins provide information at times that are well-attuned with the environment, allowing the agent to recognize its position just before traversing each rank of obstacles.

C. Extended Q-values

We generalize the upper bounds identified in Section III-A by introducing the notion of *extended Q-values*. Intuitively, extended Q-values generalize the notion of announced extra check-ins in two ways: they allow extra check-ins to occur periodically, as opposed to once, and also allow multiple extra check-ins to occur between pairs of regularly-scheduled ones. By changing the times at which extra check-ins occur, extended Q-values provide a way of generating families of upper bounds for a given PSO-MDP problem. Adding additional extra check-ins to a given instance results in lower computational complexity, at the price of a looser upper bound; however we note that, as discussed in the example above, simply increasing the frequency of extra check-ins does not necessarily result in looser upper bounds.

To start with, we consider κ separate state value functions, $Q^*_{[m]} : S \times \mathrm{A} \to \mathbb{R}, m \in \{0,\ldots,\kappa-1\}$, each corresponding to the addition of an unannounced bonus check-in at offset $\kappa - m$ from the regular check-ins, defined as follows:

$$Q^*_{[0]}(s,\vec{a}) = Q^*(s,\vec{a}), \tag{15a}$$

$$Q^*_{[m]}(s,\vec{a}) = R^{m,\gamma}(s,\vec{a}_{0:m}) + \gamma^m \sum_{s' \in S} T^m(s',\vec{a}_{0:m},s) \times \max_{\vec{a}' \in A^\kappa} Q^*(s',\vec{a}'). \tag{15b}$$

Note that Equation (15b) is identical to the argument of the minimizer in Equation (6). Next, we rewrite (15), by admitting the possibility of receiving, additionally, an announced extra check-in, but selecting its placement so that it results in the *smallest* Q-value:

$$Q^*_{[0]}(s,\vec{a}) = \min_{\ell\in\{1,\ldots,\kappa\}} \Bigg(R^{\ell,\gamma}(s,\vec{a}_{0:\ell}) + \gamma^\ell \sum_{s'\in S} T^\ell(s',\vec{a}_{0:\ell},s) \max_{\vec{a}'\in \mathrm{A}^\kappa} Q^*_{[\kappa-\ell]}(s',\vec{a}') \Bigg), \tag{16a}$$

$$Q^*_{[m]}(s,\vec{a}) = \min_{\ell\in\{1,\ldots,m\}} \Bigg(R^{\ell,\gamma}(s,\vec{a}_{0:\ell}) + \gamma^\ell \sum_{s'\in S} T^\ell(s',\vec{a}_{0:\ell},s) \max_{\vec{a}'\in \mathrm{A}^\kappa} Q^*_{[m-\ell]}(s',\vec{a}') \Bigg). \tag{16b}$$

As shown in Theorem 4, the minimum state value is achieved when $\ell = m$, and no extra check-ins are provided, thus Equations (16) are *equivalent* to (15). But, importantly, Equations (16) can be manipulated to create families of upper bounds by removing selected entries from the argument of the min operator. We pick extra check-in times so each can form some subset summing to κ, i.e., $B \subset \{1,\ldots,\kappa\}$ satisfying the following:

$$\forall \ell \in B,\ \exists \hat{B}_\ell \subseteq B \text{ such that } \left(\ell + \sum_{\hat{\ell}\in\hat{B}_\ell} \hat{\ell}\right) = \kappa.$$

Intuitively, the property ensures that the extra check-in times can be "composed" to achieve a stride of κ. In practice this is easily achieved, e.g., by selecting B to contain divisors of κ, or by ensuring that $(\ell \in B) \iff ((\kappa-\ell)\in B)$.

Next, we modify (16) as follows:

$$Q^*_{[0]}(s,\vec{a}) \le \overline{Q}^*_{[0]}(s,\vec{a}) = \min_{\ell\in B} \Bigg(R^{\ell,\gamma}(s,\vec{a}_{0:\ell}) + \gamma^\ell \sum_{s'\in S} T^\ell(s',\vec{a}_{0:\ell},s) \max_{\vec{a}'\in \mathrm{A}^\kappa} \overline{Q}^*_{[\kappa-\ell]}(s',\vec{a}') \Bigg), \tag{17a}$$

$$Q^*_{[m]}(s,\vec{a}) \le \overline{Q}^*_{[m]}(s,\vec{a}) = \min_{\ell\in B, \ell\le m} \Bigg(R^{\ell,\gamma}(s,\vec{a}_{0:\ell}) + \gamma^\ell \sum_{s'\in S} T^\ell(s',\vec{a}_{0:\ell},s) \max_{\vec{a}'\in \mathrm{A}^\kappa} \overline{Q}^*_{[m-\ell]}(s',\vec{a}') \Bigg), \tag{17b}$$

where the *in*equality here (unlike the equality in (16)) derives from the fact that the minimum is taken over only a subset of all possible check-in times. In (17a) the "less than" requirement of the minimizer is omitted because $Q^*_{[0]}$ and $\overline{Q}^*_{[0]}$ correspond to $m=\kappa$, and $\ell \in B$ is less than κ by definition.

Evaluating the upper bounds in (17) requires solving a set of up to κ coupled MDPs with actions of length $\ell \in B$. A standard procedure for solving MDPs is Value Iteration [2], which has time complexity $O(|A||S|^2)$ per iteration [9]. Hence, the complexity of solving a PSO-MDP as an MDP (following Definition 4) grows exponentially with the time between check-ins κ as $|A|^\kappa$. Actions in (17) have length $\ell < \kappa$; therefore, solving (17) for carefully-selected values of $\ell \in B$ can provide upper bounds at much lower cost compared to solving the original PSO-MDP problem as an MDP.

D. Selecting the set of check-in times B

Selection of the set B is critical to achieve a good balance between computational complexity and tightness of the bounds $\overline{Q}^*(s,\vec{a}_{0:\overline{\ell}})$. In this section, we show that selecting B to contain a single divisor of κ, i.e. $B=\{\ell\}$ for $\kappa = n\ell, n\in\mathbb{N}$, results in significant computational savings.

Equation (17) becomes

$$\overline{Q}^*_{[0]}(s,\vec{a}) = R^{\ell,\gamma}(s,\vec{a}_{0:\ell}) + \gamma^\ell \sum_{s'\in S} T^\ell(s',\vec{a}_{0:\ell},s) \max_{\vec{a}'\in \mathrm{A}^\kappa} \overline{Q}^*_{[\kappa-\ell]}(s',\vec{a}'), \tag{18a}$$

$$\overline{Q}^*_{[m]}(s,\vec{a}) = R^{\ell,\gamma}(s,\vec{a}_{0:\ell}) + \gamma^\ell \sum_{s'\in S} T^\ell(s',\vec{a}_{0:\ell},s) \max_{\vec{a}'\in \mathrm{A}^\kappa} \overline{Q}^*_{[m-\ell]}(s',\vec{a}'), \quad \text{for those } m\in\{\ell, 2\cdot\ell,\ldots,\kappa-\ell\}. \tag{18b}$$

By writing out the Equations (18) for ℓ, and $2\cdot\ell$, and then $3\cdot\ell$, and so on, we observe two things. Firstly, each is only concerned with the choice of action sequences of length exactly ℓ. Secondly, they are all, actually, posing precisely the same optimization problem. The separate degrees-of-freedom offered by having multiple functions, like both $\overline{Q}^*_{[\ell]}(s,\vec{a})$ and $\overline{Q}^*_{[2\cdot\ell]}(s,\vec{a})$, is unnecessary at the optimum. The $\frac{\kappa}{\ell}$ copies of the function are redundant because, in all these cases, the agent begins at a known state, and solving over a sequence of ℓ steps, arrives then at a state which will be observed. Therefore, we drop the bracketed subscript and the admissible solution with these extra check-ins has just:

$$\overline{Q}^*(s,\vec{a}_{0:\ell}) = R^{\ell,\gamma}(s,\vec{a}_{0:\ell}) + \gamma^\ell \sum_{s'\in S} T^\ell(s',\vec{a}_{0:\ell},s) \max_{\vec{a}'\in \mathrm{A}^\ell} \overline{Q}^*(s,\vec{a}'_{0:\ell}). \tag{19}$$

(We have been especially explicit in our notation to emphasize that all of these actions have length only ℓ.) Thus, this is equivalent to a MDP with $|S|$ states and $|A|^\ell$ actions.

A special case of the selection above is $B=\{1\}$, corresponding to an *omniscient relaxation* where the policy has access to state information at all time steps.

IV. Lower Bounds

Next, we turn our attention to establishing lower bounds that provide suboptimal, feasible policies for the PSO-MDP that are computationally efficient to compute.

To achieve these lower bound, we reduce the action space and only search through *action prefixes* of length $\tau < \kappa$. Consider an arbitrary, fixed action suffix $\vec{u} = (a_{\tau+1}, a_{\tau+2},\ldots,a_\kappa)$ of length $\kappa-\tau$. We consider the set of all actions with suffix $\vec{u}$, that is,

$$\underline{\mathrm{A}}^{\vec{u}}_\tau = \{\vec{a}\in \mathrm{A}^\kappa \mid \vec{a}_{\tau:\kappa} = \vec{u}\} = \{(\vec{a}\vec{u}) \mid \vec{a}\in \mathrm{A}^\tau\}.$$

In order to achieve a lower bound, we solve a restricted PSO-MDP where the action space is limited to $\underline{\mathrm{A}}^{\vec{u}}_\tau$. The corresponding Q-values can be computed as

$$\underline{Q}^*(s,\vec{a}) = R^{\kappa,\gamma}(s,\vec{a}) + \gamma^\kappa \sum_{s'\in \mathrm{S}} T^\kappa(s',\vec{a},s) \max_{\vec{a}'\in\underline{\mathrm{A}}^{\vec{u}}_\tau} \underline{Q}^*(s',\vec{a}'), \tag{20}$$

for all $\vec{a}\in\underline{\mathrm{A}}^{\vec{u}}_\tau$.

Solving Equation (20) through value iteration incurs a computational complexity of $O(|S|^2|A|^\ell)$, which is significantly smaller than the complexity of solving the full PSO-MDP as a MDP, i.e., $O(|S|^2|A|^\kappa)$.

The following lemma shows that $\underline{Q}^*(s,\vec{a})$ is indeed a lower bound on $Q^*(s,\vec{a})$.

Lemma 5 (Restricting the action set lower-bounds the Q-values of the selected actions)**.** *Consider a* PSO-MDP $\langle S,A,T,R,\kappa\rangle$ *with Q-values* $Q^*(s,\vec{a})$*. Also consider the restriction of the* PSO-MDP *to action sequences* $\underline{\mathrm{A}}^{\vec{u}}_\tau$*, with Q-values* $\underline{Q}^*(s,\vec{a})$*. Then,* $\underline{Q}^*(s,\vec{a}) \le Q^*(s,\vec{a}) \forall s\in \mathrm{S}, \vec{a}\in\underline{\mathrm{A}}^{\vec{u}}_\tau$.

Proof Sketch. The claim follows from the observation that (20) and (4) have the same structure, and the maximization in (20) is on a smaller set of actions. □

The bound in Lemma 5 only applies to the Q-values corresponding to actions with suffix $\vec{u}$. Next, we extend the bound to all actions with a given prefix, and to state values.

Theorem 6 (Restricting the action set lower-bounds the Q-values of actions sharing the same prefix)**.** *Consider a* PSO-MDP $\langle S, A, T, R, \kappa \rangle$ *with Q-values* $Q^*(s, \vec{a})$*. Also consider the restriction of the* PSO-MDP *to action sequences* $\underline{\mathrm{A}}_{\tau}^{\vec{u}}$ *with suffix* $\vec{u}$*, with Q-values* $\underline{Q}^*(s, \vec{a})$*. Define* $Q^*(s, \vec{a}_{0:\tau})$ *as*

$$Q^*(s, \vec{a}_{0:\tau}) = \max_{\substack{\hat{\vec{a}} \in \mathrm{A}^{\kappa} \\ \text{with} \\ \hat{\vec{a}}_{0:\tau} = \vec{a}_{0:\tau}}} Q^*(s, \hat{\vec{a}}). \tag{21}$$

Then $Q^*(s, \vec{a}_{0:\tau}) \geq \underline{Q}^*(s, (\vec{a}_{0:\tau}\vec{u}))$*.*

Proof. According to Lemma 5, $\underline{Q}^*(s, \vec{a}) \leq Q^*(s, \vec{a}), \forall s \in \mathrm{S}, \vec{a} \in \underline{\mathrm{A}}_{\tau}^{\vec{u}}$. In particular, $\forall s \in \mathrm{S}$, $\underline{Q}^*(s, (\vec{a}_{0:\tau}\vec{u})) \leq Q^*(s, (\vec{a}_{0:\tau}\vec{u})) \leq \max_{\hat{\vec{a}} \in \mathrm{A}^{\kappa} \text{ with } \hat{\vec{a}}_{0:\tau} = \vec{a}_{0:\tau}} Q^*(s, \hat{\vec{a}}) = Q^*(s, \vec{a}_{0:\tau})$, the last inequality follows because $(\vec{a}_{0:\tau}\vec{u}) \in \mathrm{A}^{\kappa}$ with $(\vec{a}_{0:\tau}\vec{u})_{0:\tau} = \vec{a}_{0:\tau}$. □

Lemma 7 (Restricting the set of admissible actions lower-bounds the state values)**.** *Consider a* PSO-MDP $\langle S, A, T, R, \kappa \rangle$ *with Q-values* $Q^*(s, \vec{a})$ *and state values* $U^*(s)$*. Also consider the restriction of the* PSO-MDP *to action sequences* $\underline{\mathrm{A}}_{\tau}^{\vec{u}}$*, with Q-values* $\underline{Q}^*(s, \vec{a})$*. Then,*

$$U^*(s) \geq \underline{U}^*(s) = \max_{\vec{a} \in \underline{\mathrm{A}}_{\tau}^{\vec{u}}} \underline{Q}^*(s, \vec{a}). \tag{22}$$

Proof. The proof follows from Theorem 6 and from the definition of $U^*(s) = \max_{\vec{a} \in \mathrm{A}^{\kappa}} Q^*(s, \vec{a})$. □

A. Dilatory Process as a lower bound

A PSO-MDP is termed a *non-drift* PSO-MDP, if it admits NO-OP actions that leave the agent in the same state with probability one, and provides zero reward. We shall denote such actions by "$a_{\perp}$". For a non-drift PSO-MDP, the natural choice for the suffix $\vec{u}$ is a sequence of NO-OP actions $\vec{u} = (a_{\perp}, \ldots, a_{\perp})$. This offers an intuitive interpretation of the lower bounds in Theorem 6 and Lemma 7 as the outcome of a *dilatory process* where the agent follows the optimal policy of length τ, and then stops taking actions until it receives the information delivered by the next check-in.

The interpretation of the lower bounds as a dilatory process is of interest because it provides a connection between the upper bounds in Section III-A (which have the agent replanning with new information after τ steps) and the lower bounds in this section (which, in the non-drift case, assume the agent pauses after τ steps to wait for new information). However, note that the optimal policy for the upper bound may differ from the optimal policy for the lower bound due to the discounting factor.

The use of NO-OP actions for the action suffix $\vec{u}$ also offers a computational advantage: since NO-OP actions result in no transition state and no reward, the transition and reward functions can be computed as $T^{\kappa}(s', \vec{a}, s) = T^{\tau}(s', \vec{a}, s)$ and $R^{\kappa,\gamma}(s, \vec{a}) = R^{\tau,\gamma}(s, \vec{a}), \forall \vec{a} \in \underline{\mathrm{A}}_{\tau}^{\vec{u}}$, where $\vec{u} = (a_{\perp}, \ldots, a_{\perp})$. Since the cost of computing T^{κ} and R^{κ} scales exponentially with κ, this can result in significant computational savings.

V. A BRANCH-AND-BOUND ALGORITHM

We are now in a position to use the upper and lower bounds described in the previous sections to efficiently solve PSO-MDPs. Specifically, we propose a *branch-and-bound* algorithm that builds a sequence of increasingly tight upper and lower bounds, and uses the bounds to prune suboptimal actions. The proposed approach is described in Algorithm 1. The algorithm iteratively builds upper and lower bounds for the Q-values of action prefixes for each state, and uses the bounds to discard actions whose prefix's upper bound is smaller than another action prefix's lower bound. The key insight is to keep track of non-dominated actions for each state through the set $\mathsf{A}(s)$; whenever the upper bound $\overline{Q}^*(s, \vec{a}_{0:\tau})$ for a given action prefix $\vec{a}_{0:\tau}$ (computed by using the extended Q-values presented in Section III-C) is lower than the lower bound $\underline{U}^*(s)$, as obtained via Equations (20) and (22), the actions with the prefix $\vec{a}_{0:\tau}$ are discarded for that state.

Algorithm 1 Branch-and-bound algorithm for PSO-MDPs

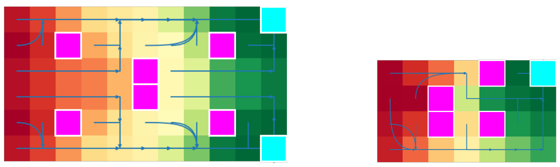
for $s \in \mathrm{S}$ **do**
  $\mathsf{A}(s) \leftarrow \mathrm{A}$ ▷ Keep track of non-dominated actions
end for
for $\tau \in \{1, \ldots, \kappa\}$ **do** ▷ Consider increasingly long action prefixes
  $T^{\tau}(s', \vec{a}, s) \leftarrow$ Extend $T^{\tau-1}(s', \vec{a}, s)$ ▷ Update transitions
  $R^{\tau,\gamma}(s, \vec{a}) \leftarrow$ Extend $R^{\tau-1,\gamma}(s, \vec{a})$ ▷ Update rewards
  if τ is a divisor of κ **then**
    $\overline{Q}^*(s, \vec{a}_{0:\tau}) \leftarrow$ Solve (19) with $B = \{\tau\}$ but with the action prefix set restricted to be $\{\vec{a}_{0:\tau} : \vec{a} \in \mathsf{A}(s)\}$ ▷ Update the upper bound
  else
    $\overline{Q}^*(s, \vec{a}_{0:\tau}) \leftarrow \overline{Q}^*(s, \vec{a}_{0:\tau-1})$ ▷ Adapt previous upper bound
  end if
  $\underline{U}^*(s) \leftarrow$ Solve (20), (22) with action prefix set restricted to $\{\vec{a}_{0:\tau} : \vec{a} \in \mathsf{A}(s)\}$ ▷ Update the lower bound
  for $s \in \mathrm{S}, \vec{a} \in \mathsf{A}(s)$ **do**
    if $\overline{Q}^*(s, \vec{a}_{0:\tau}) \leq \underline{U}^*(s)$ **then** ▷ Prune actions
      Remove all actions with prefix $\vec{a}_{0:\tau}$ from $\mathsf{A}(s)$
    end if
  end for
end for
$Q^*(s, \vec{a}), \pi^*(s) \leftarrow$ Solve (4), (5) with only $\mathsf{A}(s)$ actions

Compared to naively solving the MDP version of the PSO-MDP with Equations (4) and (5), Algorithm 1 requires solving up to $2(\kappa - 1)$ additional MDPs with action sets of size upper-bounded by $A, A^2, \ldots, A^{\kappa-1}$. However, the pruning procedure can greatly reduce the size of the action set, which can result in significantly reduced computation times in practical applications, as shown next.

VI. NUMERICAL EXPERIMENTS

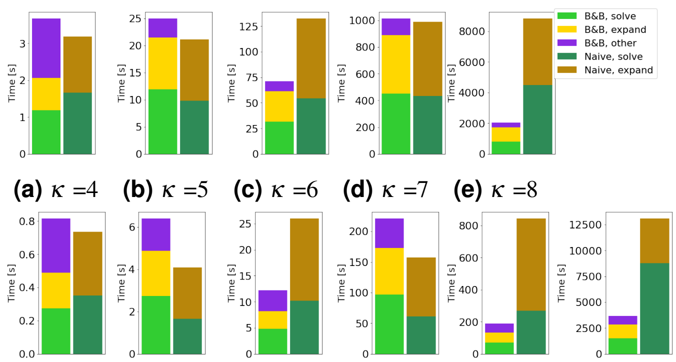
We assess the performance of the proposed branch-and-bound algorithm on robot navigation problems. We consider two grid-world PSO-MDPs, shown in Figure 3. An agent must navigate to rewarding states (shown in cyan) while avoiding obstacles (shown in purple). The agent's navigation is imperfect: when trying to drive in a given direction, the agent remains in place with 5% probability, and drifts left or right of the desired direction with 7.5% probability each.

We compare the time required to solve the PSO-MDPs with Algorithm 1 with a naive approach where we formulate the PSO-MDP as a MDP, and then solve it via value iteration. Figure 4 shows the time required to formulate and solve the problem with both approaches, for both problems. Due to memory limitations, the larger grid is only solved for $\kappa \leq 8$.



(a) 6×11 grid **(b)** 4×7 grid

Fig. 3: Grid world PSO-MDP problem, state values, and optimal policy. Rewarding terminal states are cyan; obstacles are purple.



(a) $\kappa = 4$ **(b)** $\kappa = 5$ **(c)** $\kappa = 6$ **(d)** $\kappa = 7$ **(e)** $\kappa = 8$

(f) $\kappa = 4$ **(g)** $\kappa = 5$ **(h)** $\kappa = 6$ **(i)** $\kappa = 7$ **(j)** $\kappa = 8$ **(k)** $\kappa = 9$

Fig. 4: Time required to formulate and solve a navigation PSO-MDP problem. Top (a-e): 6×11 grid. Bottom (f-k): 4×7 grid.

The proposed approach significantly outperforms the naive approach for $\kappa = 6$, $\kappa = 8$, and (for the smaller grid) $\kappa = 9$, offering a twofold to fourfold reduction in computation time—an encouraging result that points to the branch-and-bound approach, informed by upper and lower bounds, as a highly promising technique to make PSO-MDPs with large check-in periods tractable. We note that, for $\kappa = 5$ and $\kappa = 7$, the performance of the branch-and-bound approach is on par or slightly worse compared to the naive approach. This is not unexpected: the upper bound is only updated for ℓ that are divisors of κ, which results in modest bounding and pruning when the check-in period is a prime number. This is not a fundamental limitation of the algorithmic approach, but rather a byproduct the simple technique used to select the set B in Algorithm 1. Extending the approach to accommodate generic sets of check-in times B, and devising techniques to select sets B that result in tight upper bounds for general check-in periods κ, are critical directions for future research.

VII. Conclusions

Planning under uncertainty —the crucial problem faced by robots— is computationally intractable to solve in the form of completely general POMDPs. One approach to handle this impasse is to add simplifying assumptions, or to impose constraints, that afford opportunities for efficient specialized solution methods. This is, broadly, the approach employed in the present paper. We have identified and examined a novel class of decision-making problems in between MDPs and POMDPs, the former not accounting for observation uncertainty, while the latter being generally computationally intractable. The class of PSO-MDPs model situations where the state is only observed periodically. We establish a collection of bounds for these problems by, quite intuitively, considering cases that vary when state information is made available to the agent. These bounds are then turned to gains in computational efficiency via a branch-and-bound algorithm. The paper also uncovers some intriguing nuances. For instance, that receiving data more frequently is not always better. Also, knowledge of *when* uncertainty will be quashed can be exploited and thus be understood to have specific value.

Acknowledgements

Part of this work was carried out at the Jet Propulsion Laboratory (JPL), California Institute of Technology, under a contract with the National Aeronautics and Space Administration (80NM0018D0004). The work at TAMU was supported in part by NASA/Jet Propulsion Lab R&TD Innovative Spontaneous Concept Award #1652187, and in part by NSF Award IIS-2034097.

References

[1] R. Bellman and E. Lee, "History and development of dynamic programming," *IEEE Control Systems Magazine*, vol. 4, no. 4, pp. 24–28, 1984.

[2] D. P. Bertsekas, *Reinforcement Learning and Optimal Control*. Belmont, M.A., U.S.A: Athena Scientific, 2019.

[3] J. Boyan and M. Littman, "Exact solutions to time-dependent MDPs," *Advances in Neural Information Processing Systems*, vol. 13, 2000.

[4] C. Diuk, A. Cohen, and M. L. Littman, "An object-oriented representation for efficient reinforcement learning," in *Proc. Int. Conf. on Machine Learning (ICML)*, 2008, pp. 240–247.

[5] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[6] M. Merlin, N. Parikh, E. Rosen, and G. Konidaris, "Locally observable Markov decision processes," in *ICRA 2020 Workshop on Perception, Action, Learning*, 2020.

[7] N. Meuleau, L. Peshkin, K. Kim, and L. P. Kaelbling, "Learning Finite-State Controllers for Partially Observable Environments," in *Proc. Conf. on Uncertainty in Artificial Intelligence (UAI)*, Stockholm, Sweden, 1999, pp. 427–436.

[8] J. Pineau, G. Gordon, and S. Thrun, "Point-based value iteration: An anytime algorithm for POMDPs," in *Proc. Int. Joint Conference on AI (IJCAI)*, Acapulco, Mexico, 2003, pp. 1025–1032.

[9] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ, U.S.A.: Prentice-Hall, Inc., 2009.

[10] D. Silver and J. Veness, "Monte-Carlo planning in large POMDPs," *Advances in Neural Information Processing Systems*, vol. 23, 2010.

[11] A. Somani, N. Ye, D. Hsu, and W. S. Lee, "DESPOT: Online POMDP planning with regularization," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, M.A., U.S.A.: MIT Press, 2018.

[13] R. S. Sutton, D. Precup, and S. Singh, "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[14] M. Świechowski, K. Godlewski, B. Sawicki, and J. Mańdziuk, "Monte Carlo Tree Search: A Review of Recent Modifications and Applications," *arXiv:abs/2103.04931*, 2021. [Online]. Available: https://arxiv.org/abs/2103.04931