Abstract

In reinforcement learning, agents have successfully used environments modeled with Markov decision processes (MDPs). However, in many problem domains, an agent may suffer from noisy observations or random times until its subsequent decision. While partially observable Markov decision processes (POMDPs) have dealt with noisy observations, they have yet to deal with the unknown time aspect. Of course, one could discretize the time, but this leads to Bellman’s Curse of Dimensionality. To incorporate continuous sojourn-time distributions in the agent’s decision making, we propose that partially observable semi-Markov decision processes (POSMDPs) can be helpful in this regard. We extend Spaan and Vlassis (2005) randomized point-based value iteration (PBVI) Perseus algorithm used for POMDP to POSMDP by incorporating continuous sojourn time distributions and using importance sampling to reduce the solver complexity. We call this new PBVI algorithm with importance sampling for POSMDPs—ChronosPerseus. This further allows for compressed complex POMDPs requiring temporal state information by moving this information into state sojourn time of a POMSDP. The second insight is that keeping a set of sampled times and weighting it by its likelihood can be used in a single backup; this helps further reduce the algorithm complexity. The solver also works on episodic and non-episodic problems. We conclude our paper with two examples, an episodic bus problem and a non-episodic maintenance problem.

Keywords: partially observable semi-Markov decision process (POSMDP), point-based value iteration (PBVI), Perseus, timing, importance sampling

1. Introduction

Time is critical to cognitive agents making decisions (Maniadakis and Trahanias, 2011). Planning not only requires what to do but also when to do it (Rivest and Kohar, 2020). For instance, when should we give up waiting for a bus and start walking? When should we perform some maintenance on a particular piece of machinery? When should an autonomous
robotic vacuum cleaner start to return to its base before it runs out of battery life? In each of these mentioned situations, the agent requires a time model to aid in the decision process.

In artificial intelligence, a common model for planning in a stochastic environment is the Markov decision process (MDP) (Russell and Norvig, 2010; Sutton and Barto, 2018). It is a formal framework of sequential decision making that not only considers which sequence of actions, or a policy, gains the most immediate reward but also future rewards (Bellman, 1957b). Time is considered by discounting the reward, but to consider the issue of when to act, we could augment the state space with time steps leading to what Bellman calls the Curse of Dimensionality.

Two well-known extensions to MDPs are semi-Markov MDP (SMDP) (Howard, 1963) and partially observable MDP (POMDP) (Aoki, 1965; Åström, 1965). The former allows for transitions to have sojourn time distributions, while the latter allows for partially observable states (incomplete information). In reinforcement learning, SMDPs are primarily used with options (Sutton et al., 1999; Precup, 2000), where the underlying MDP of the micro-actions defines the transitions of macro-actions. The SMDP, in that case, represents a temporal compression of the underlying MDP. In contrast, POMDPs are computationally expensive (Papadimitriou and Tsitsiklis, 1987; Madani et al., 1999), as the solvers usually construct a policy over a belief state space that must cover all the combinations of states.

POMDPs are not widely used in practice due to their computational complexity (Papadimitriou and Tsitsiklis, 1987; Madani et al., 1999; Sutton and Barto, 2018). However, with the development of approximate solutions methods such as point-based value iteration by Pineau et al. (2003) leading to PERSEUS by Spaan and Vlassis (2005).

Cognitive agents, as well as many scheduling and maintenance problems, could naturally benefit from the ability to solve problems that mix both partial observability and rich temporal transitions. The partially observable semi-Markov decision process (POSMDP) extends POMDP by including stochastic sojourn time between transitions. As we will show, this sojourn time can further be used in the belief state update. White (1976) first proposed the finite-horizon discrete-time POSMDP model, and his algorithm for computing the optimal cost and an optimal policy were based on procedures for POMDPs by Sondik (1971). Half a decade later, Wakuta defined the infinite-horizon continuous-time POSMDP model with the average-cost criterion (Wakuta, 1981) and the discounted-cost criterion (Wakuta, 1982). The definition and notation of POSMDP that we present in Section 2 is influenced by Wakuta (1982), Hernández-Lerma (1989), and Yu (2006).

While Zhang and Revie (2017) developed a POSMDP solver based on PERSEUS, we introduce importance sampling to use the collected samples more efficiently. This leads us to the significant contribution of this paper, which is our importance sampling point-based POSMDP solver called CHRONOSPERSEUS that we will present in Section 3. This allows us to include temporal properties to POMDPs without the added complexity that would be generated by extending the temporal information through a large expansion of the state space dimensions.

The remainder of this paper is organized as follows. First, in Section 2 we formally introduce the framework, the notation, and the dynamics for the POSMDP model. Then, we discuss how we handle concepts of time in Section 2.1 and partial observability in Section 2.2. We discuss the difference between the process’s history and the observable history in Section 2.3. In Section 2.4, we give belief update rules that do incorporate and do not
incorporate sojourn time. We believe that the sojourn time is important information that can be incorporated into the belief update, which in turn affects the agent's optimal decision. Section 2.5 discusses the reward and the assumptions required to ensure the reward function is well-defined. For the value function of a POSMDP in Section 3.1, we show that depending on the approach taken, either extending the POMDP model with the addition of continuous time. Finally, we describe in Section 3 the main contribution of this paper—the ChronosPerseus algorithm. In Section 4, we conclude this paper with some solved POSMDP examples: the bus problem of an agent deciding to stay on the bus or ride its bike, and maintenance of water filters in the real world based on parameters from Zhang and Revie (2017). The first one includes a mixture of random continuous and fixed sojourn times as well as mixed-observability on an episodic task, while the second includes continuous sojourn times and observation space on a non-episodic problem.

The code to the ChronosPerseus algorithm and the examples are available online (https://github.com/rkohar/ChronosPerseus).

2. Partially Observable Semi-Markov Decision Processes

A partially observable semi-Markov decision process (POSMDP) is an 11-tuple

\[ \langle S, A, \mathcal{K}, O, Q, G, G_0, R, \xi_0, \beta, N \rangle \]

where

- \( S \) is the Borel state space, and the elements of \( S \) are called states.
- \( A \) is the Borel action space, and the elements of \( A \) are called actions. To each \( s \in S \), we associate a nonempty Borel-measurable subset \( \mathcal{A}(s) \subseteq A \), whose elements are the admissible actions for the agent in state \( s \).
- \( \mathcal{K} \) is the set of admissible state-action pairs, and it is assumed to be a Borel subset \( \mathcal{K} \subseteq S \times A \).
- \( O \) is the Borel observation space, and the elements of \( O \) are called observations.
- \( Q(d\tau, ds' | s, a) \) denotes the sojourn time-state (Borel-measurable) stochastic kernel on \( \mathbb{R}_{>0} \times S \) given \( S \times A \).
- \( G(da | a, s') \) is the observation (Borel-measurable) stochastic kernel on \( O \) given \( A \times S \).
- \( G_0(da | s') \) denotes the initial observation (Borel-measurable) stochastic kernel on \( O \) given \( S \).
- \( R(s, a) \) is the per-stage (bounded Borel-measurable) reward function given \( S \times A \).
  \( (R(s, a) \) could be further broken down into a leap sum and a continuous reward, see Section 2.5.)
- \( \xi_0 \) is the (a priori) initial (belief) state distribution \( (\xi_0 \in P(S)) \).
- \( \beta \) is the discounting rate where \( \beta \in [0, 1) \).
$N$ is the planning horizon. It could be finite, or $N = \infty$.

Given the model $\langle S, A, K, O, Q, G_0, R, \xi_0, \beta, N \rangle$, the dynamics of the POSMDP proceed according to Algorithm 1. This involves at each decision epoch $n$ choosing an action $a_n$, accruing reward $R(s_n, a_n)$ as the state changes from $s_n$ to $s_{n+1}$ in $\tau_n$ amount of time, and partially observing $s_{n+1}$ as $o_{n+1}$. The agent uses all the information available up to decision epoch $n$, namely the observable history $h_n$, to choose action $a_n = \pi_n(h_n)$ using policy $\pi_n$. With the dynamics of the POSMDP specified by Algorithm 1 and illustrated in Fig. 1, we denote the sequence of policies that the agent uses from decision epoch 0 to $n-1$ as $\pi = (\pi_0, \pi_1, \ldots, \pi_{n-1})$. The set of all admissible policies is denoted $\Pi$.

**Algorithm 1** Dynamics of POSMDP

In the beginning $n = 0$, the state $s_0$ is simulated from an initial (belief) state distribution $\xi_0$.

For each decision epoch $n = 1, 2, \ldots, N$:

1. Based on the observable history
   
   $h_0 = (\xi_0)$
   
   $h_n = (\xi_0, a_1, t_1, o_1, \ldots, a_n, t_n, o_n)$,

   the agent performs action

   $a_n = \pi_n(h_n) \in A \quad n = 1, 2, \ldots, N$.

   Here $\pi_n$ denotes a policy that the agent uses at decision epoch $n$.

2. The agent obtains a reward $R(s_n, a_n)$ for choosing action $a_n$ at decision epoch $n$.

3. The state evolves randomly with sojourn time-state transition probability

   $Q(\tau, s' \mid s, a) = P(T_{n+1} - T_n \leq \tau, S_{n+1} = s' \mid S_n = s, A_n = a)$  \hspace{1cm} (1)

   to the next state $s_{n+1}$ that decision epoch $n + 1$.

4. The agent records a noisy observation $O_n \in O$ of the state $S_{n+1}$ according to

   $G(o \mid a, s') = P(O_n = o \mid A_n = a, S_{n+1} = s')$.

5. The agent updates its history as

   $h_{n+1} = (h_n, a_{n+1}, t_{n+1}, s_{n+1})$.

   If $n < N$, then set $n$ to $n + 1$, and go back to step (a).

   If $n = N$, then the agent receives the last reward and the process terminates.
Figure 1: Graphical model of POSMDP. The states $S_i$ are hidden from the agent, while the observations $O_i$ and times $T_i$ are observable. The sojourn times are $\tau_i$ and the agent’s actions are $A_i$. The solid arrows $\rightarrow$ represent the direct influence from one element to another, while the dashed arrows $\xrightarrow{}$ represent indirect influence. For example, action $A_0$ is directly influenced by time $T_0$ and observation $O_0$ because this is the information that is available to the agent, while state $S_0$ is indirectly influencing action $A_0$ since the agent has to infer in which state it is in currently from time $T_0$ and observation $O_0$.

2.1 Time

Unlike POMDPs where the sojourn times are constant, the sojourn times are random in a POSMDP; see Fig. 2.

Figure 2: The difference between POMDPs and POSMDPs for sojourn times. The sojourn time for a POMDP is constant whereas the sojourn time for a POSMDP is a random amount of time that follows some probability distribution.

The time of the $n^{th}$ decision epoch is denoted by the random variable

$$T_n = \begin{cases} 0 & n = 0 \\ T_{n-1} + \tau_{n-1} & n = 1, 2, 3, \ldots, \end{cases}$$

(2)
in which the observed time of $T_n$ is denoted by $t_n$. The sojourn time from state $s$ to state $s'$ is a nonnegative real-valued random variable $\tau_n = T_{n+1} - T_n$ distributed by $F(\tau \mid s, a, s')$.
conditional on \((s, a, s')\) such that
\[
Q(\tau, s' \mid s, a) = P(s' \mid s, a)F(\tau \mid s, a, s').
\] (3)

From (3), we assume implicitly that a sojourn time \(\tau_n\) does not affect the next sojourn time \(\tau_{n+1}\).

2.2 Partial observability

For partial observability, we follow the same structure from POMDPs. We do not change the notion of partial observability with the introduction of random sojourn time between decision epochs. In our definition of the POSMDP in Section 2, the observation is independent of the previous state. If the observation depends on more information, such as \((s, a, \tau, s')\), then we could define \(G(o \mid s, a, \tau, s')\) as
\[
G(o \mid a, s') = \sum_{s \in S} Q(\tau, s' \mid s, a)G(o \mid s, a, \tau, s').
\] (4)

2.3 History

A realization of this process is \(s_0, o_0, a_0, \tau_0, s_1, o_1, a_1, \tau_1, s_2, o_2, a_2, \tau_2, \ldots \) \(\in (S \times O \times A \times R_{> 0})^\infty \triangleq \Omega\).

However, we are not able to rely on the unobservable states \(s_0, s_1, s_2, \ldots\), so we introduce the concept of observable histories. We define the space of observable histories until the \(n^\text{th}\) decision epoch by
\[
\mathcal{H}_n = \begin{cases} 
\mathcal{P}(S) \times O, & n = 0; \\
\mathcal{H}_{n-1} \times A \times R_{> 0} \times O, & n = 1, 2, 3, \ldots.
\end{cases}
\] (5)

Thus, the element \(h_0 = (\xi_0, o_0) \in \mathcal{H}_0\) is an initial observed history, and an element \(h_n \in \mathcal{H}_n\) is called an observed history up to \(n\), and it is denoted by
\[
h_n = (\xi_0, o_0, a_0, \tau_0, o_1, a_1, \tau_1, \ldots, o_{n-1}, a_{n-1}, \tau_{n-1}, a_n, \tau_n, o_n)
\] (6)
\[
= (h_{n-1}, a_{n-1}, \tau_n, o_n).
\] (7)

2.4 Belief State Representation

The agent can only partially observe the state \(s \in S\) by inferring from its past and current observations. Thus, a memory of past observations is required for an agent to optimally choose its actions in a partially observable environment.

A na"ive approach would allow the agent to remember its sequence of sojourn times, observations, and actions. However, this sequence can grow unbounded over time, which is not practical with finite memory. Instead, we can summarize this information with sufficient statistics (Aoki, 1965; Åström, 1965). A statistic is considered sufficient when no other statistic calculated from the same sample provides any additional information to the
estimated parameter value (Fisher, 1922). With these sufficient statistics that summarize
the observable history, we can construct a probability distribution of where the agent is in
the state space.

**Definition 1 (Belief state)** A belief state \( \xi \) is a probability distribution over the state
space \( S \).

If the state space \( S = \{s_1, s_2, \ldots, s_{|S|}\} \) is a finite set, then the belief state \( \xi \) is defined
by

\[
\xi = \begin{bmatrix}
\xi(s_1) \\
\xi(s_2) \\
\vdots \\
\xi(s_{|S|})
\end{bmatrix} \triangleq \begin{bmatrix}
P(S = s_1) \\
P(S = s_2) \\
\vdots \\
P(S = s_{|S|})
\end{bmatrix}.
\]

To ensure that \( \xi \) is a probability distribution,

\[
\xi(s_i) \geq 0 \quad \forall s_i \in S, \quad \text{and} \quad \sum_{i=1}^{|S|} \xi(s_i) = 1.
\]

This probability distribution encodes the agent’s subjective probability of its location
in the environment’s state space from its history.

\[
\xi_n(s_i) = \mathbb{P}(S_n = s_i \mid \xi_0, o_0, a_0, o_1, a_1, \ldots, o_{n-1}, a_{n-1}, o_n). \tag{8}
\]

The belief state is a sufficient statistic of the observable history. In other words, the belief
state carries all the information the agent requires to make its decision from the present
state; hence, the process retains the Markov property.

**Proposition 2** A belief state \( \xi \) is a sufficient statistic of the agent’s history, and given an
action \( a \), sojourn time \( \tau \), and observation \( o \), the next belief state is

\[
\xi_n(s_i) = \mathbb{P}(S_n = s_i \mid \xi_{n-1}, a_{n-1}, \tau_n, o_n)
\]

We can update the belief state in two ways: with or without observing sojourn time \( \tau \).

**2.4.1 The Belief Update without Sojourn Time**

The belief update for POMDP is usually given by

\[
\xi(s' \mid a, o) = \frac{G(o \mid a, s') \sum_{s \in S} \xi(s) P(s' \mid s, a)}{P(o \mid \xi, a)} \quad \forall s' \in S, \tag{9}
\]

where the denominator

\[
P(o \mid \xi, a) = \sum_{s' \in S} G(o \mid a, s') \sum_{s \in S} \xi(s) P(s' \mid s, a) \tag{10}
\]

is a normalization factor (Spaan, 2012).
By replacing the state transition probability distribution $P$ with the sojourn-time state transition probability distribution $Q$, the belief update for POSMDP, if the transition time $\tau$ is not observed, would be given by

$$
\xi(s' | a, o) = \frac{G(o | a, s') \sum_{s \in S} \xi(s) \int_0^\infty Q(d\tau | s', s, a)}{P(o | \xi, a)}
$$

(11)

$$
G(o | a, s') \sum_{s \in S} \xi(s) P(s' | s, a) \int_0^\infty F(d\tau | s, a, s')
$$

(12)

$$
= \frac{G(o | a, s') \sum_{s \in S} \xi(s) P(s' | s, a) \sum_{s'\prime \in S} G(o | a, s'\prime) \sum_{s \in S} Q(\tau, s'\prime | s, a)}{P(o | \xi, a)}
$$

(13)

since the integral of $F$ is one. Therefore, the belief update (including its denominator, the normalization factor) would remain unchanged in this case. However, it would make sense for the agent to use this information since different transitions may have distinguishable sojourn time distributions, indicative of the current state.

### 2.4.2 The Belief Update with Sojourn Time

**Theorem 3** Given an action $a$ the agent performed, a sojourn time $\tau$, and an observation $o$ seen, the belief update for a particular state $s'$ is

$$
\xi(s' | a, \tau, o) = \frac{G(o | a, s') \sum_{s \in S} Q(\tau, s' | s, a)\xi(s)}{\sum_{s'\prime \in S} G(o | a, s'\prime) \sum_{s \in S} Q(\tau, s'\prime | s, a)\xi(s)}.
$$

(14)

**Proof** By definition,

$$
\xi(s' | a, \tau, o) \triangleq P(s' | a, \tau, o).
$$

Applying the conditional rule

$$
P(A \mid B) = \frac{P(A, B)}{P(B)},
$$

we have

$$
\xi(s' | a, \tau, o) = P(s' | a, \tau, o) = \frac{P(a, \tau, o, s')}{P(a, \tau, o)} = \frac{P(a, \tau, o, s')}{\sum_{s'' \in S} P(a, \tau, o, s'')}.
$$

(14)

Using the multiplication rule $P(A, B) = P(A \mid B)P(B)$, we have

$$
P(s, a, \tau, o, s') = P(o \mid s, a, \tau, s')P(s, a, \tau, s')
$$

and applying the multiplication rule again to the second term,

$$
P(s, a, \tau, o, s') = \overbrace{P(o \mid s, a, \tau, s')}^{G(o | a, s')} \underbrace{P(\tau, s' | s, a)}_{Q(\tau, s' | s, a)} \xi(s) \overbrace{P(s, a)}^{P(o | \xi, a)}.
$$

8
Since the observation $o$ only depends on action $a$ and the next state $s'$,

$$P(o \mid s, a, \tau, s') = G(o \mid a, s'),$$

and by definition $P(\tau, s' \mid s, a) = Q(\tau, s' \mid s, a)$ (see Eq. (1)) and $P(s, a) = P(a \mid s)P(s) = \xi(s)$ since $P(a \mid s) = 1$ for the selected action $a$ (the policy is deterministic) and $P(s) = \xi(s)$ (our Bayesian estimate). Thus,

$$P(s, a, \tau, o, s') = G(o \mid a, s')Q(\tau, s' \mid s, a)\xi(s).$$

and

$$P(a, \tau, o, s') = \sum_{s \in S} P(s, a, \tau, o, s') = G(o \mid a, s')\sum_{s \in S} Q(\tau, s' \mid s, a)\xi(s).$$  \hspace{1cm} (15)

Substituting Eq. (15) into Eq. (14) yields the required result.

We can further decomposed the belief update function when $Q(\tau, s' \mid s, a)$ is given as $P(s' \mid s, a)$ and $F(\tau \mid s, a, s')$ by

$$\xi(s' \mid a, \tau, o) = \frac{G(o \mid a, s')\sum_{s \in S} Q(\tau, s' \mid s, a)\xi(s)}{P(o \mid \xi, a, \tau)} \hspace{1cm} \forall s' \in S$$  \hspace{1cm} (16)

and

$$= \frac{G(o \mid a, s')\sum_{s \in S} P(s' \mid s, a)F(\tau \mid s, a, s')\xi(s)}{P(o \mid \xi, a, \tau)} \hspace{1cm} \forall s' \in S,$$  \hspace{1cm} (17)

where the denominator

$$P(o \mid \xi, a, \tau) = \sum_{s' \in S} G(o \mid a, s')\sum_{s \in S} Q(\tau, s' \mid s, a)\xi(s)$$  \hspace{1cm} (18)

$$= \sum_{s' \in S} G(o \mid a, s')\sum_{s \in S} P(s' \mid s, a)F(\tau \mid s, a, s')\xi(s)$$  \hspace{1cm} (19)

is a normalization factor similar to Wakuta (1982).

Note that this is similar to the belief update for POMDP; for instance, Eq. (17) is the same as Eq. (9), with the addition of the sojourn-time cumulative probability distribution $F(t \mid s, a, s')$ in the numerator and the denominator. This is important as Eq. (17) allows the agent to use the observed transition time to better estimate its real underlying state. Our solver is using this more complete approach.

### 2.5 Reward

#### 2.5.1 Discount reward criterion

Let $R(s, a)$ denote the expected total discounted reward between two decision epochs, given that the agent is in state $s$ and it chooses to perform action $a$, which can be expressed by

$$R(s, a) = r_1(s, a) + \int_0^\infty \left( \int_0^\infty e^{-\beta t}r_2(t \mid s, a, s') \, dt \right) Q(d\tau, ds' \mid s, a).$$  \hspace{1cm} (20)
Note that the discount rate $\beta$ is given as part of the model. The reward function can be thought of two parts:

1. the lump sum reward $r_1(s, a)$ that is received immediately upon the agent performing action $a$ in state $s$; and

2. the continuous reward rate $r_2(\tau \mid s, a, s')$ that is received by the agent continuously over the sojourn time $\tau$ from state $s$ to $s'$ under action $a$.

Since the agent cannot observe the state $s \in S$ directly, we can rewrite Eq. (20) for a given belief $\xi \in \Delta$ and action $a \in A$ as

$$R(\xi, a) = \int_{s \in S} \xi(ds) R(s, a),$$

or if it is a discrete set of states, it can be written as

$$R(\xi, a) = \sum_{s \in S} \xi(s) R(s, a).$$

We need a few assumptions to ensure that the expected rewards are well-defined. Time is the safeguard that prevents everything from happening at once. So, we borrow the following assumption for SMDPs by Ross (1970) that allows only a finite number of decision epochs during a finite sojourn time.

**Assumption 1** There exists a sojourn time $\tau > 0$ and $\epsilon > 0$ such that

$$\int_{s' \in S} Q(\tau, ds' \mid s, a) \leq 1 - \epsilon \quad \forall (s, a) \in \mathcal{K}.$$

In other words, Assumption 1 says that for every (admissible) state-action pair $(s, a) \in \mathcal{K}$, there is a positive probability of at least $\epsilon$ that the transition time to state $s'$ will be greater than $\tau$.

We also impose the condition that the per-stage reward $R(s, a)$ is bounded for every state-action pair $(s, a) \in \mathcal{K}$.

**Assumption 2** There exists a constant $M$ such that for $\beta = 0$,

$$\sup_{(s, a) \in \mathcal{K}} |R(s, a)| < M.$$

This assumption is satisfied if $r_1$ and $r_2$ in Eq. (20) are bounded, and

$$\mathbb{E}(\tau_n \mid S_n = s, A_n = a) < \infty \quad \forall (s, a) \in \mathcal{K}.$$

In other words, we would like each expected sojourn time $\mathbb{E}(\tau_n)$ to be bounded. Therefore, we make the following assumption.

**Assumption 3**

$$\sup_{(s, a) \in \mathcal{K}} \mathbb{E}(\tau_n \mid S_n = s, A_n = a) < \infty.$$
3. ChronosPerseus

In this paper, we present ChronosPerseus, an approximate point-based value iteration algorithm with importance sampling developed to solve POSMDPs. It follows other point-based value iteration methods in that it implements a belief state collection by simulating the environment (Pineau et al., 2003, 2006; Spaan and Vlassis, 2005; Shani et al., 2013), but additionally, it also collects sample sojourn times that are part of the belief update. Once the collection is complete, an approximate value function is constructed for which a policy is found. First, we will describe below the exact value function of a POSMDP followed by its construction with linear vectors (known as α-vectors in the POMDP literature). Then, we will construct an approximate value function by sampling beliefs and sojourn times, which will lead to our summary of the ChronosPerseus algorithm.

3.1 The Value Function of POSMDP

The optimal value function—otherwise known as the Bellman (1957a) equation—for a finite state, action, and observation POMDP is

\[ V^*(\xi) = \max_{a \in A(\xi)} \left[ \sum_{s \in S} \xi(s) R(s, a) + \sum_{o \in O} \sum_{s' \in S} G(o \mid a, s') \sum_{s \in S} \xi(s) P(s' \mid s, a) \gamma V^\pi(\xi(\cdot \mid a, o)) \right] \tag{23} \]

for all belief states \( \xi \) in the belief simplex \( \Delta \). When Eq. (23) holds for every belief state in the belief simplex, we are ensured the solution is optimal. If we were to compute the value function over the continuous belief space, belief by belief, it may seem intractable. However, Sondik (1971) shows that in this case, the POMDP optimal value function is convex, and that it can be parameterized by a finite set of hyperplanes.

In order to adapt Eq. (23) for a POSMDP, the discount factor must take the sojourn time probability \( F(d \tau \mid s, a, s') \) into account. It thus becomes

\[ \gamma(s, a, s') = \int_0^\infty e^{-\beta \tau} F(d \tau \mid s, a, s'). \tag{24} \]

Then, the Bellman equation for POSMDP is

\[ V^*(\xi) = \max_{a \in A(\xi)} \left[ \sum_{s \in S} \xi(s) R(s, a) + \sum_{o \in O} \sum_{s' \in S} G(o \mid a, s') \sum_{s \in S} \xi(s) P(s' \mid s, a) \int_0^\infty e^{-\beta \tau} F(d \tau \mid s, a, s') V^*(\xi(\cdot \mid a, \tau, o)) \right] \tag{25} \]

or equivalently using Eq. (4),

\[ V^*(\xi) = \max_{a \in A(\xi)} \left[ \sum_{s \in S} \xi(s) R(s, a) + \sum_{o \in O} \sum_{s' \in S} G(o \mid a, s') \sum_{s \in S} \xi(s) \int_0^\infty e^{-\beta d \tau} G(d \tau, s' \mid s, a) V^*(\xi(\cdot \mid a, d \tau, o)) \right] \tag{26} \]
3.2 Alpha vectors

Since the belief simplex $\Delta$ is continuous, the optimal value function $V^*(\xi)$ in Eq. (23) is computationally intractable to calculate pointwise for every belief state $\xi \in \Delta$. To circumvent this, Sondik (1971) made the following assumption. Consider a two-state $S = \{s_1, s_2\}$ POMDP. In Fig. 3(a), we represent the belief state on the horizontal axis, while on the vertical axis will be the value function. The agent knows the value with absolute certainty when with absolute certainty it knows in which state it is; see Fig. 3(b).

**Assumption 4** The value is linearly propositional to how certain the agent believes in which state it is.\(^1\)

In other words, linear interpolation fills in the values between the two known values; see Fig. 3(c). The line segment that connects the two values in Fig. 3(c) is an example of what Sondik (1971) called an $\alpha$-vector. In general, we give the following definition.

![Figure 3: An example of an $\alpha$-vector for a two-state $S = \{s_1, s_2\}$ POMDP](image)

**Definition 4 ($\alpha$-vector)** An $\alpha$-vector represents an $|S|$-dimensional hyperplane that defines the value function for a particular action $a \in A$ over a bounded region of the belief simplex. In this work, we represent $\alpha$-vectors in matrix form:

$$\alpha = \begin{bmatrix} \alpha(s_1) \\ \alpha(s_2) \\ \vdots \\ \alpha(s_{|S|}) \end{bmatrix} = \begin{bmatrix} V(s_1) \\ V(s_2) \\ \vdots \\ V(s_{|S|}) \end{bmatrix}. \tag{27}$$

While we have generally referred to $V$ as the value function, we can also think of $V$ as a set of $\alpha$ vectors:

$$V = \{\alpha_1, \alpha_2, \ldots, \alpha_m\}. \tag{27}$$

To calculate the optimal value function\(^2\) at a particular belief state is

$$V^*(\xi) = \max_{\alpha \in V} \langle \xi, \alpha \rangle \tag{28}$$

\(^1\) While Sondik (1971) did not explicitly state this in his thesis, this assumption is apparent and required to show that the finite state, finite action, finite observation POMDP is piecewise linear and convex.

\(^2\) We will denote the optimal value function by $V^*$ and the set of $\alpha$ vectors by $V$. 

12
\[= \max_{\alpha \in \mathcal{V}} \sum_{s \in \mathcal{S}} \xi(s)\alpha(s), \quad (29)\]
and the corresponding optimal policy \(\pi^* : \xi \rightarrow a\) is
\[
\pi^* = \arg \max_{i \in \{1, 2, \ldots, |V|\}} \langle \xi, \alpha_i \rangle. \quad (30)
\]

### 3.3 Approximate Value Iteration

A critical element to solve a POSMDP is to approximate the integral
\[
\mu(\xi, s, o, a, s') = \int_0^\infty e^{-\beta \tau} f(\tau | s, a, s') V^*(\xi(\cdot | a, \tau, o)) \, d\tau. \quad (31)
\]
We solve this problem using the Monte Carlo method with importance sampling described in Hammersley and Handscomb (1964).

Suppose we have a finite set \(C\) whose elements \(\tau_1, \tau_2, \ldots, \tau_{|C|}\) are independent sojourn time samples from the probability density function \(D\). An unbiased estimator of \(\mu(s, o, a, s')\) is
\[
\hat{\mu}(\xi, s, o, a, s') = \frac{1}{|C|} \sum_{n=1}^{|C|} e^{-\beta \tau_n} \frac{f(\tau_n | s, a, s')}{D(\tau_n)} V^*(\xi(\cdot | a, \tau_n, o)) \quad (32)
\]
where the function \(D(\tau_n)\) given by
\[
D(\tau_n) = \sum_{s \in \mathcal{S}} \sum_{a \in A} \sum_{s' \in \mathcal{S}} w(s, a, s') f(\tau_n | s, a, s')
\]
is a mixture distribution of each sojourn-time distribution for each \(s, a, s'\) and the weights \(w(s, a, s')\) reflect the proportion of samples that came from each distribution. Note that the weights must sum to one; that is,
\[
\sum_{s \in \mathcal{S}} \sum_{a \in A} \sum_{s' \in \mathcal{S}} w(s, a, s') = 1.
\]

By replacing the integral (31) with the approximation (32) in the Bellman equation for POSMDP, we get
\[
V^*(\xi) = \max_{a \in A(\xi)} \left[ R(\xi, a) + \sum_{o \in O} \sum_{s' \in \mathcal{S}} G(o | a, s') \right.

\sum_{s \in \mathcal{S}} \xi(s) P(s' | s, a) \frac{1}{|C|} \sum_{n=1}^{|C|} e^{-\beta \tau_n} \frac{f(\tau_n | s, a, s')}{D(\tau_n)} V^*(\xi(\cdot | a, \tau_n, o)) \left. \right]. \quad (33)
\]

We bring \(\frac{1}{|C|} \sum_{n=1}^{|C|} e^{-\beta \tau_n} D(\tau_n)\) to the beginning of the second term since it does not depend on state \(s\) or observation \(o\), yielding
\[
V^*(\xi) = \max_{a \in A(\xi)} \left[ R(\xi, a) + \frac{1}{|C|} \sum_{n=1}^{|C|} e^{-\beta \tau_n} D(\tau_n) \sum_{o \in O} \sum_{s' \in \mathcal{S}} G(o | a, s') \right.

\sum_{s \in \mathcal{S}} \xi(s) P(s' | s, a) \left. \right]. \quad (33)
\]
\[ \sum_{s \in S} \xi(s) P(s' | s, a) f(\tau_n | s, a, s') V^*(\xi(|a, \tau_n, o)) \]  

(34) 

and by Eq. (19),

\[ V^*(\xi) = \max_{a \in A(\xi)} \left[ R(\xi, a) + \frac{1}{|C|} \sum_{n=1}^{|C|} \frac{e^{-\beta \tau_n}}{D(\tau_n)} \sum_{o \in O} P(o | \xi, a, \tau_n) V^*(\xi(|a, \tau_n, o)) \right]. \]  

(35) 

Next, we follow the procedure similar to POMDPs in Spaan and Vlassis (2005) to express the value function as a finite set of \( \alpha \) vectors. Using Eq. (28), we can express the value of the next belief as an inner product as

\[ V^*(\xi) = \max_{a \in A(\xi)} \left[ R(\xi, a) + \frac{1}{|C|} \sum_{n=1}^{|C|} \frac{e^{-\beta \tau_n}}{D(\tau_n)} \sum_{o \in O} P(o | \xi, a, \tau_n) \max_{\alpha \in \mathcal{V}} \langle \xi | a, \tau_n, o, \alpha \rangle \right]. \] 

Expressing the inner product as a sum by Eq. (29), we have

\[ V^*(\xi) = \max_{a \in A(\xi)} \left[ R(\xi, a) + \frac{1}{|C|} \sum_{n=1}^{|C|} \frac{e^{-\beta \tau_n}}{D(\tau_n)} \sum_{o \in O} P(o | \xi, a, \tau_n) \max_{\alpha \in \mathcal{V}} \sum_{s' \in \mathcal{S}} \alpha(s') \xi(s' | a, \tau_n, o) \right] \]

and using Eq. (17) for the value of the next belief,

\[ V^*(\xi) = \max_{a \in A(\xi)} \left[ R(\xi, a) + \frac{1}{|C|} \sum_{n=1}^{|C|} \frac{e^{-\beta \tau_n}}{D(\tau_n)} \sum_{o \in O} P(o | \xi, a, \tau_n) \max_{\alpha \in \mathcal{V}} \sum_{s' \in \mathcal{S}} \alpha(s') \xi(s' | a, \tau_n, o) \right] \]

\[ \max_{\alpha \in \mathcal{V}} \sum_{s' \in \mathcal{S}} \alpha(s') \sum_{s \in \mathcal{S}} \xi(s) P(s' | s, a) f(\tau_n | s, a, s') \]

\[ G(o | a, s') \sum_{s \in \mathcal{S}} \xi(s) P(s' | s, a) f(\tau_n | s, a, s') \]

\[ = \max_{a \in A(\xi)} \left[ R(\xi, a) + \frac{1}{|C|} \sum_{n=1}^{|C|} \frac{e^{-\beta \tau_n}}{D(\tau_n)} \sum_{o \in O} \max_{\alpha \in \mathcal{V}} \sum_{s' \in \mathcal{S}} \alpha(s') G(o | a, s') \right] \]

Note how this allows the elimination of the computationally expensive term \( P(o | \xi, a, \tau_n) \).

Rearranging the order of the summations and the terms so that we have an \( \alpha \)-vector yields,

\[ V^*(\xi) = \max_{a \in A(\xi)} \left[ R(\xi, a) + \frac{1}{|C|} \sum_{n=1}^{|C|} \frac{e^{-\beta \tau_n}}{D(\tau_n)} \sum_{o \in O} \max_{\alpha \in \mathcal{V}} \sum_{s \in \mathcal{S}} \xi(s) \sum_{s' \in \mathcal{S}} \alpha(s') G(o | a, s') P(s' | s, a) f(\tau_n | s, a, s') \right] \]

\[ \alpha(s(a, \tau_n, o)) \]
\[
= \max_{a \in A(\xi)} \left[ R(\xi, a) + \frac{1}{|C|} \sum_{n=1}^{|C|} e^{-\beta \tau_n} \sum_{o \in O} \max_{\alpha \in V} \sum_{s \in S} \xi(s) \alpha(s | a, \tau_n, o) \right],
\]
and finally by Eq. (28),
\[
V^*(\xi) = \max_{a \in A(\xi)} \left[ R(\xi, a) + \frac{1}{|C|} \sum_{n=1}^{|C|} e^{-\beta \tau_n} \sum_{o \in O} \max_{\alpha \in V} \langle \xi, \alpha(\cdot | a, \tau_n, o) \rangle \right],
\]
which is the approximate form of the POSMDP value function that we will use for the CHRONOSPERSEUS algorithm.

We can now write a concise backup operation that generates a new \( \alpha \) vector for a specific belief \( \xi \); that is,
\[
\text{BACKUP}(V, C, w, \xi) = \arg \max_{\alpha(\cdot | a, \tau_n, o)} \langle \xi, \alpha(\cdot | a, \tau_n, o) \rangle
\]
where for all \( s \in S \),
\[
\alpha(s \mid \xi, a) = R(s, a) + \frac{1}{|C|} \sum_{n=0}^{|C|} e^{-\beta \tau_n} \sum_{o \in O} \max_{\alpha \in V} \langle \xi, \alpha(\cdot | a, \tau_n, o) \rangle,
\]
and
\[
\alpha(s \mid a, \tau_n, o) = \sum_{s' \in S} G(o \mid a, s') P(s' \mid s, a) f(\tau_n | s, a, s') \alpha(s').
\]

### 3.4 The Algorithm

We present now the resulting contribution, the CHRONOSPERSEUS algorithm, which is an approximate point-based value iteration with importance sampling for POSMDPs. It proceeds in two steps as specified in Algorithm 2: first, collect beliefs and sojourn time samples, and then approximate the value function using the backup operator.

The first step, as outlined in Algorithm 3, is to let the agent explore the environment and collect a finite set of beliefs \( B \), a finite set of sojourn times \( C \), and the weights \( w(s, a, s') \) that reflect the proportion of samples that came from each state-action-state transition; after these are collected, they remain fixed for the rest of the algorithm. In Step 2, as outlined in Algorithm 4, from the initial value function \( V_0 \), the algorithm will continue to perform backups until a convergence criterion is achieved.

**Theorem 5** Let \( M = \min_{(s, a) \in K} R(s, a) \). The initial value function \( V_0 \) is a single \( \alpha \)-vector with
\[
V_0(s) = \frac{M}{1 - \lambda} \quad \forall s \in S,
\]
where
\[
\lambda = \begin{cases} 
\min_{1 \leq n \leq |C|} \frac{e^{-\beta \tau_n}}{D(\tau_n)}, & \text{if } M \geq 0; \\
\max_{1 \leq n \leq |C|} \frac{e^{-\beta \tau_n}}{D(\tau_n)}, & \text{if } M < 0.
\end{cases}
\]
Proof Since $\mathcal{S}$ and $\mathcal{A}$ are finite sets, the per-stage minimum reward $M$ exists by Assumption 2.

If $M = 0$, then this is trivial: set $\alpha(s) = 0$ for all $s \in \mathcal{S}$.

For $M > 0$, we start with Eq. (35), for any belief state $\xi$,

$$V(\xi) = \max_{a \in A(\xi)} \left[ R(\xi, a) + \frac{1}{|C|} \sum_{n=1}^{|C|} e^{-\beta \tau_n} \sum_{o \in O} P(o \mid \xi, a, \tau_n) V(\xi(\cdot \mid a, \tau_n, o)) \right].$$

Let $\lambda_n = e^{-\beta \tau_n} / D(\tau_n)$, and $0 < \lambda < 1$. Since $M$ is positive, we would like the discount to be as small as possible. So, we define

$$\lambda = \min_{1 \leq n \leq |C|} \lambda_n, \quad n^* = \arg \min_{1 \leq n \leq |C|} \lambda_n, \quad \text{and} \quad \tau = \tau_{n^*},$$

so that

$$V(\xi) \geq \max_{a \in A(\xi)} \left[ R(\xi, a) + \frac{1}{|C|} \sum_{n=1}^{|C|} \lambda \sum_{o \in O} P(o \mid \xi, a, \tau)V(\xi(\cdot \mid a, \tau, o)) \right]$$

$$= \max_{a \in A(\xi)} \left[ R(\xi, a) + \lambda \sum_{o \in O} P(o \mid \xi, a, \tau)V(\xi(\cdot \mid a, \tau, o)) \right]$$

$$\geq \max_{a \in A(\xi)} \left[ M + \lambda \left( \frac{M}{1 - \lambda} \right) \right]$$

$$= M + \frac{\lambda M}{1 - \lambda}$$

$$= M + \frac{\lambda M}{1 - \lambda} = V_0.$$

If $M < 0$, then it is a negative reward, and so the discount should be as large as possible, so

$$\lambda = \max_{1 \leq n \leq |C|} \lambda_n, \quad n^* = \arg \max_{1 \leq n \leq |C|} \lambda_n, \quad \text{and} \quad \tau = \tau_{n^*}.$$  

Algorithm 2 CHRONOSPERSEUS

1: function CHRONOSPERSEUS($n, \xi_0, V_0, \epsilon$)
2: \hspace{1cm} $(B, C, w) \leftarrow \text{collectBeliefs}(n, \xi_0)$
3: \hspace{1cm} $V' \leftarrow V_0$
4: \hspace{1cm} repeat
5: \hspace{1.5cm} $V' \leftarrow V'$
6: \hspace{1.5cm} $V' \leftarrow \text{UPDATE}(B, C, w, V')$
7: \hspace{1.5cm} until $\|V - V'\|_{\infty, B} < \epsilon$
8: \hspace{1cm} return $V$
9: end function
Algorithm 3 CollectBelief

1: function collectBeliefs(n, ξ₀)
2:     B ← {ξ₀}, C ← ∅, w ← 0  ▷ Initialization
3:     repeat
4:         Randomly select the belief state ξ ∈ B
5:         Generate state s according to the multinomial distribution with weights ξ
6:         Randomly select an action a ∈ A
7:         Generate a random sojourn-time τ according to f(τ | s, a, s') ▷ Add τ to set C and update counts
8:         Randomly select o ∈ O according to Eq. (19)
9:         Update ξ(· | a, τ, o) according to Eq. (17)
10:        B ← B ∪ {ξ(· | a, τ, o)}  ▷ Add new belief to set B
11:        until |B| = n
12:     w ← \frac{w}{||w||}  ▷ Normalize w; ||w|| = \sum_{s \in S} \sum_{a \in A} \sum_{s' \in S} w(s, a, s').
13:     return B, C, and w
14: end function

Algorithm 4 Update

1: function update(B, C, w, V)
2:     B' ← B, V' ← ∅  ▷ Initialization
3:     while B' ≠ ∅ do
4:         Randomly select a belief ξ ∈ B'
5:         α ← backup(V, C, w, ξ)  ▷ Eq (38)
6:         if ⟨ξ, α⟩ < V(ξ) then
7:             α ← arg max_{α' ∈ V} ⟨ξ, α'⟩
8:         end if
9:         B' ← B' \ {ς ∈ B' | ⟨ς, α⟩ ≥ V(ς)}
10:        V' ← V' ∪ {α}
11:     end while
12:     V ← V'
13:     return V
14: end function

The complexity of computing Eq. (40) is $O(|S|^2)$ since it needs to be calculated for every $(s, s')$ tuple, and it is done for every $α ∈ V$, hence computing all $α(· | a, τ, o)$ for every $a ∈ A$, $τ ∈ C$, and $o ∈ O$ requires $O(|V| \times |S|^2 \times |A| \times |C| \times |O|)$. The complexity of computing Eq. (39) requires the computation of all relevant $α(· | a, τ, o)$, but then the summation and inner products require only $O(|S| \times |C| \times |O|)$ operations and another $O(|S|)$ operations to add the reward (vector). Lastly, the Backup (Eq. (38)) requires for all $α(· | ξ, a)$ another $O(|S|)$ operations for the inner product. Therefore, the complexity of the Backup operation requires

$$O(|V| \times |S|^2 \times |A| \times |C| \times |O|).$$  (41)
Given there are no more beliefs than observed sojourns time, $|B| \leq |C|$, the complexity of an Update iteration is therefore

$$O(|V| \times |S|^2 \times |A| \times |C|^2 \times |O|).$$

(42)

4. Applications

We now apply ChronosPerseus to two interesting problems. The first is an episodic toy problem related to the often encountered situation in our daily life in which our waiting time influences our belief and, in turn, our decision. In this simple scenario, an agent is riding a bus at an unknown traffic intensity with the ability to give-up and ride its bike instead at any given bus stop. The second problem is a real application to a routine maintenance problem first presented with a continuous observation space by Zhang and Revie (2017).

These two examples will show how ChronosPerseus can fully solve real POSMDP problems with mixed observability, a mixture of observable continuous and discrete random sojourn times, discrete or continuous observation space, and episodic or non-episodic tasks.

4.1 The Bus Problem

We introduce the bus problem, an episodic toy example, that demonstrates ChronosPerseus follows our human intuition.

Suppose the agent boarded a bus and brought along a bicycle, which has been attached to the bus’ rack. There are four bus stops until the agent reaches the final destination. There are three levels of traffic intensity $i$ (low $i = 1$, medium $i = 2$, and high $i = 3$) that remains the same throughout the agent’s bus journey, but the traffic intensity $i$ is unknown to the agent. When the bus is at stop $s$, the agent can either elect to stay on the bus, or to exit the bus and ride its bike directly to the destination (the last stop). Reaching the destination leads to a (lump sum) reward, but the longer it takes to reach it, the more discounted it will be. Suppose the agent decides to continue. In that case, the next state will be $s + 1$ with sojourn time $\tau$ according to the probability density function $f(\tau, (s + 1, i) \mid (s, i))$. If the agent chooses to take the bike, it takes a fixed time to reach the destination depending on the current stop $s$.

![Figure 4: The agent waiting for the bus.](image)

Intuitively, we would expect that the agent should ride the bus long enough to deduce which traffic intensity level:

- if the traffic intensity is low, then the agent would ride the bus to the end;
- if the traffic intensity is high, then the agent would get off at the next stop and ride the bike to the end; and
• if the traffic intensity is medium, then the agent would have to balance the advantage of taking the bus versus riding the bike.

With this intuition in mind, we can model this problem formally using the POSMDP framework. The state space consists of the bus stops and the three levels of traffic intensity,

$$\mathcal{S} = \{0, 1, 2, 3, 4\} \times \{1, 2, 3\},$$

which means the number of states is

$$|\mathcal{S}| = 5 \times 3 = 15.$$  \hspace{1cm} (43)

A typical state would be the pair \( (s, i) \in \mathcal{S} \), where the bus is at stop \( s \) and the traffic intensity is \( i \). This state is mixed observable since the agent can observe perfectly at which bus stop \( s \) it is, while it cannot observe directly the traffic intensity \( i \). We will assume that the traffic intensity remains \( i \) throughout the entire bus ride; that is,

\[(0, i) \rightarrow (1, i) \rightarrow (2, i) \rightarrow (3, i) \rightarrow (4, i).\]

The action space consists of two actions

$$\mathcal{A} = \{\text{bus, bike}\}.$$  \hspace{1cm} (45)

The observation space consists of the bus stops that the agent observes,

$$\mathcal{O} = \{0, 1, 2, 3, 4\}.\hspace{1cm} (46)$$

Please note that the observation space is not the same as the state space; in other words, \( \mathcal{O} \neq \mathcal{S} \). Since only a component of the state is fully known to the agent, this problem is called mixed observable (Ong et al., 2010). We will discuss this when we create the observation transition function.

If the agent decides to continue riding the bus, the traffic intensity \( i \) does not change, but the bus stop changes from \( s \) to \( s + 1 \). When the last stop is reached \( (s = 4) \), the problem resets probabilistically by placing the agent at bus stop \( s = 0 \) independently of the action selected, and randomly selecting a new traffic intensity \( i \) with equal probability; this is reflected in the probability \( P((0, i') \mid (4, i), \text{bus}) = \frac{1}{3} \).

To illustrate this visually, a state transition diagram for the bus action is given in Fig. 5. Thus, the transition probability matrices for continuing to ride the bus are

\[
P((\cdot, i) \mid (\cdot, i), \text{bus}) = \begin{bmatrix}
\frac{1}{3} & 0 & \ldots & 0 \\
0 & 1 & 0 & \ldots \\
\vdots & \ddots & \ddots & \ddots \\
\ldots & \ldots & \ldots & \frac{1}{3} \\
0 & \ldots & 0 & 1
\end{bmatrix}
\]  \hspace{1cm} (47)
and

$$P((\cdot, i') \neq i) \mid (\cdot, i), \text{bus}) = \begin{bmatrix}
{s'}=0 & s'=1 & s'=2 & s'=3 & s'=4 \\
s=0 & 0 & \cdots & \cdots & 0 \\
s=1 & \cdots & \cdots & \cdots & \cdots \\
s=2 & \cdots & \cdots & \cdots & \cdots \\
s=3 & 0 & \cdots & \cdots & 0 \\
s=4 & \frac{1}{3} & 0 & \cdots & \cdots
\end{bmatrix}$$  \hspace{1cm} (48)$$

for all traffic intensities $$i \in \{1, 2, 3\}$$. If at stop $$s \neq 4$$ the agent decides to stop riding the bus and ride the bike to the final stop, the traffic intensity $$i$$ does not change, but the bus stop changes directly from $$s$$ to $$s = 4$$. Again, when the last stop is reached, the problem resets probabilistically by placing the agent at bus stop 0 independently of the selected action, and randomly selecting a new traffic intensity $$i$$ with equal probability. Visually, this can be illustrated with a state diagram for the bike action, which is given in Fig. 6. Thus, the transition probability matrices for getting off the bus and riding the bike are

$$P((\cdot, i) \mid (\cdot, i), \text{bike}) = \begin{bmatrix}
{s'}=0 & s'=1 & s'=2 & s'=3 & s'=4 \\
s=0 & 0 & \cdots & \cdots & 1 \\
s=1 & \cdots & \cdots & \cdots & \cdots \\
s=2 & \cdots & \cdots & \cdots & \cdots \\
s=3 & 0 & \cdots & \cdots & 0 \\
s=4 & \frac{1}{3} & 0 & \cdots & \cdots
\end{bmatrix}$$  \hspace{1cm} (49)$$

and

$$P((\cdot, i') \neq i) \mid (\cdot, i), \text{bike}) = \begin{bmatrix}
{s'}=0 & s'=1 & s'=2 & s'=3 & s'=4 \\
s=0 & 0 & \cdots & \cdots & 0 \\
s=1 & \cdots & \cdots & \cdots & \cdots \\
s=2 & \cdots & \cdots & \cdots & \cdots \\
s=3 & 0 & \cdots & \cdots & 0 \\
s=4 & \frac{1}{3} & 0 & \cdots & \cdots
\end{bmatrix}$$  \hspace{1cm} (50)$$

for all traffic intensities $$i \in \{1, 2, 3\}$$.

If the agent continues to ride the bus, the sojourn time distributions $$f(\tau \mid (s, i), \text{bus}, (s', i))$$ follow an inverse Gaussian probability density distribution (Tweedie, 1957)

$$f(x \mid \mu, \lambda) = \sqrt{\frac{\lambda}{2\pi x^3}} \exp \left[ -\frac{\lambda(x - \mu)^2}{2\mu^2 x} \right], \quad \forall x > 0, \quad \mu > 0, \quad \lambda > 0,$$  \hspace{1cm} (51)$$

where $$\mu$$ is the expected value and the variance is $$\mu^3/\lambda$$, the parameters conditional on the bus stops and traffic intensity given by

$$\mu((\cdot, 1), \text{bus}, (\cdot, 1)) = \begin{bmatrix}
{s'}=0 & s'=1 & s'=2 & s'=3 & s'=4 \\
s=0 & 0 & 5 & 0 & 0 \\
s=1 & 0 & 0 & 5 & 0 \\
s=2 & 0 & 0 & 0 & 5 \\
s=3 & 0 & 0 & 0 & 5 \\
s=4 & 455 & 0 & 0 & 0
\end{bmatrix}$$  \hspace{1cm} (52)$$

and

$$\mu((\cdot, i), \text{bus}, (\cdot, i)) = \begin{bmatrix}
{s'}=0 & s'=1 & s'=2 & s'=3 & s'=4 \\
s=0 & 0 & \cdots & \cdots & 0 \\
s=1 & \cdots & \cdots & \cdots & \cdots \\
s=2 & \cdots & \cdots & \cdots & \cdots \\
s=3 & 0 & \cdots & \cdots & 0 \\
s=4 & \frac{1}{3} & 0 & \cdots & \cdots
\end{bmatrix}$$  \hspace{1cm} (53)$$

for all traffic intensities $$i \in \{1, 2, 3\}$$. If at stop $$s \neq 4$$ the agent decides to stop riding the bus and ride the bike to the final stop, the traffic intensity $$i$$ does not change, but the bus stop changes directly from $$s$$ to $$s = 4$$. Again, when the last stop is reached, the problem resets probabilistically by placing the agent at bus stop 0 independently of the selected action, and randomly selecting a new traffic intensity $$i$$ with equal probability. Visually, this can be illustrated with a state diagram for the bike action, which is given in Fig. 6. Thus, the transition probability matrices for getting off the bus and riding the bike are

$$P((\cdot, i) \mid (\cdot, i), \text{bike}) = \begin{bmatrix}
{s'}=0 & s'=1 & s'=2 & s'=3 & s'=4 \\
s=0 & 0 & \cdots & \cdots & 1 \\
s=1 & \cdots & \cdots & \cdots & \cdots \\
s=2 & \cdots & \cdots & \cdots & \cdots \\
s=3 & 0 & \cdots & \cdots & 0 \\
s=4 & \frac{1}{3} & 0 & \cdots & \cdots
\end{bmatrix}$$  \hspace{1cm} (49)$$

and

$$P((\cdot, i') \neq i) \mid (\cdot, i), \text{bike}) = \begin{bmatrix}
{s'}=0 & s'=1 & s'=2 & s'=3 & s'=4 \\
s=0 & 0 & \cdots & \cdots & 0 \\
s=1 & \cdots & \cdots & \cdots & \cdots \\
s=2 & \cdots & \cdots & \cdots & \cdots \\
s=3 & 0 & \cdots & \cdots & 0 \\
s=4 & \frac{1}{3} & 0 & \cdots & \cdots
\end{bmatrix}$$  \hspace{1cm} (50)$$

for all traffic intensities $$i \in \{1, 2, 3\}$$.

If the agent continues to ride the bus, the sojourn time distributions $$f(\tau \mid (s, i), \text{bus}, (s', i))$$ follow an inverse Gaussian probability density distribution (Tweedie, 1957)

$$f(x \mid \mu, \lambda) = \sqrt{\frac{\lambda}{2\pi x^3}} \exp \left[ -\frac{\lambda(x - \mu)^2}{2\mu^2 x} \right], \quad \forall x > 0, \quad \mu > 0, \quad \lambda > 0.$$  \hspace{1cm} (51)$$

where $$\mu$$ is the expected value and the variance is $$\mu^3/\lambda$$, the parameters conditional on the bus stops and traffic intensity given by

$$\mu((\cdot, 1), \text{bus}, (\cdot, 1)) = \begin{bmatrix}
{s'}=0 & s'=1 & s'=2 & s'=3 & s'=4 \\
s=0 & 0 & 5 & 0 & 0 \\
s=1 & 0 & 0 & 5 & 0 \\
s=2 & 0 & 0 & 0 & 5 \\
s=3 & 0 & 0 & 0 & 5 \\
s=4 & 455 & 0 & 0 & 0
\end{bmatrix}$$  \hspace{1cm} (52)$$

and

$$\mu((\cdot, i), \text{bus}, (\cdot, i)) = \begin{bmatrix}
{s'}=0 & s'=1 & s'=2 & s'=3 & s'=4 \\
s=0 & 0 & \cdots & \cdots & 0 \\
s=1 & \cdots & \cdots & \cdots & \cdots \\
s=2 & \cdots & \cdots & \cdots & \cdots \\
s=3 & 0 & \cdots & \cdots & 0 \\
s=4 & \frac{1}{3} & 0 & \cdots & \cdots
\end{bmatrix}$$  \hspace{1cm} (53)$$

for all traffic intensities $$i \in \{1, 2, 3\}$$.
Figure 5: Bus problem state transition diagram when the agent selects action $a = \text{bus}$. The problem begins at stop $s = 0$, with one of the randomly selected initial intensity $i$. The bus action allows the agent to go from one bus stop to the next while traffic intensity remains constant, until it reaches the end of the bus line. At the end, it is returned to the beginning of the bus line for a new randomly selected intensity.
Figure 6: Bus problem state transition diagram when the agent selects action $a = \text{bike}$. Riding the bike leads directly to the last stop, at which point, it is returned to the beginning of the bus line for a new randomly selected intensity.
Figure 7: The sojourn time distributions for the action \( a = \text{bus} \).

\[
\mu((\cdot, 2), \text{bus}, (\cdot, 2)) = \begin{bmatrix}
0 & 5 & 0 & 0 & 0 \\
0 & 0 & 10 & 0 & 0 \\
0 & 0 & 0 & 10 & 0 \\
0 & 0 & 0 & 0 & 20 \\
455 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

medium intensity \ (53)

\[
\mu((\cdot, 3), \text{bus}, (\cdot, 3)) = \begin{bmatrix}
0 & 10 & 0 & 0 & 0 \\
0 & 0 & 25 & 0 & 0 \\
0 & 0 & 0 & 25 & 0 \\
0 & 0 & 0 & 45 & 0 \\
455 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

high intensity \ (54)

\[
\mu((s, i), \text{bus}, (s', i' \neq i)) = 0
\]

and

\[
\lambda((s, i), \text{bus}, (s', i)) = 10 \cdot \mu((s, i), \text{bus}, (s', i))^2.
\]

The graphs of the sojourn time distributions for riding the bus from stop to stop at each traffic intensity are shown in Fig. 7. We can see that the time it takes from stop to stop in low traffic is relatively short compared to the long times between stops in high traffic; the mean total time is 20 in low traffic whereas the mean total time in high traffic is 105. Note that technically we cannot have \( \mu = 0 \) in the matrices because the inverse Gaussian distribution is defined for \( \tau > 0 \). This is not an issue since the corresponding probability \( P(\tau \mid s, a) = 0 \), hence, \( Q(\tau, s' \mid s, a) = 0 \).

If the agent decides to stop riding the bus and use its bicycle, then the sojourn time distribution follows a deterministic function \( f(\tau \mid (s, i), \text{bike}, (s', i)) \). If \( c_0 \) is a constant, then the probability mass function is

\[
f(\tau \mid c_0((s, i), \text{bike}, (s', i))) = \begin{cases} 1, & \text{if } \tau = c_0((s, i), \text{bike}, (s', i)); \\ 0, & \text{otherwise,} \end{cases}
\]

(57)
where
\[
  c_0((s, i), \text{bike}, (s', i)) = \begin{bmatrix}
    s' = 0 & s' = 1 & s' = 2 & s' = 3 & s' = 4 \\
    s = 0 & 0 \cdots \cdots & 0 & 30 \\
    s = 1 & \cdot & \cdot & 25 \\
    s = 2 & \cdot & \cdot & 20 \\
    s = 3 & 0 \cdots \cdots & 0 & 12 \\
    s = 4 & 455 & 0 \cdots \cdots & 0 \\
  \end{bmatrix}
\] (58)

and
\[
  c_0((s, i), \text{bike}, (s', i) \neq i)) = 0. \tag{59}
\]

Since ChronosPerseus uses an infinite planning horizon, we need to ensure rewards from one trip (episode) from \(s = 0\) to \(s = 4\) does not affect the value for the next trip (episode) by creating a large discount when the environment resets probabilistically. This technique turns this episodic task into a non-episodic task, allowing ChronosPerseus to work seamlessly for episodic and non episodic tasks. We assume continuous-time discounting at a rate of \(\beta = 0.02\). This means that the present value of one reward unit received at time \(\tau\) units in the future equals \(e^{-0.02\tau} = \gamma\). To get a discount factor of \(\gamma = 0.0001\), we set \(\tau = 455\) between \(s = 4\) to \(s = 0\); this is the entry for \(\mu((4, \cdot), \text{bus}, (0, \cdot))\) in (52)–(54) and \(c_0((4, \cdot), \text{bike}, (0, \cdot))\) in (58).

The cumulative distribution function is
\[
  F(\tau | c_0((s, i), \text{bike}, (s', i))) = \begin{cases} 
    0, & \text{if } \tau < c_0((s, i), \text{bike}, (s', i)); \\
    1, & \text{if } \tau \geq c_0((s, i), \text{bike}, (s', i)).
  \end{cases} \tag{60}
\]

The sojourn time-state transition function is defined by
\[
  Q(\tau, (s', i) | (s, i), a) = P((s', i) | (s, i), a) F(\tau | (s, i), a, (s', i))
  = P((s', i) | (s, i), a) \int_0^\tau f(t | (s, i), a, (s', i)) \, dt \tag{61}
\]

Given the landing state \((s', i)\), regardless of which action \(a\) was selected, the agent with certainty would observe bus stop \(s'\). Remember, it is the traffic intensity \(i\) that is hidden from the agent, but this does not influence what bus stop \(s'\) is observed; it is deterministic what bus stop is observed depending on whether the agent decides to continue to ride the bus or take the bicycle. This leads to the observation probability matrix given by
\[
  G(\cdot | a, (\cdot, i)) = \begin{bmatrix}
    o = 0 & o = 1 & o = 2 & o = 3 & o = 4 \\
    s' = 0 & 1 & 0 \cdots \cdots & 0 \\
    s' = 1 & 0 & \cdots \cdots & 0 \\
    s' = 2 & \cdots \cdots & \cdots \cdots & 0 \\
    s' = 3 & \cdots \cdots & \cdots \cdots & 0 \\
    s' = 4 & 0 \cdots \cdots & 0 & 1 \\
  \end{bmatrix} \quad \forall i \in \{1, 2, 3\}. \tag{63}
\]

Now, we will setup the reward function. Set the lump sum reward \(r_1((s, i), a) = 0\) and continuous reward rate as \(r_2((s, i), a, (s', i')) = -1\). Using the Laplace transform of the
inverse Gaussian distribution

\[
\int_0^{\infty} e^{-\beta \tau} f(\tau \mid (s, i), a, (s', i')) \, d\tau,
\]

the reward function is

\[
R((s, i), a) = -\frac{1}{\beta} \sum_{(s', i') \in S} P((s', i') \mid (s, i), a) \left( 1 - M((s, i), a, (s', i')) \right)
\]

where

\[
M((s, i), a, (s', i')) = \begin{cases} 
\exp \left[ -\beta c_0((s, i), \text{bike}, (s', i)) \right], & \text{if } a = \text{bike}; \\
\exp \left[ \frac{\lambda(s, \text{bus}, s')}{\mu(s, \text{bus}, s')} \left( 1 - \sqrt{1 + \frac{2\mu(s, \text{bus}, s')^2 \beta}{\lambda(s, \text{bus}, s')}} \right) \right], & \text{if } a = \text{bus}.
\end{cases}
\]

The agent begins at bus stop 0, and since this is fully observable, but the traffic level intensity is not, the agent assumes that the three traffic intensities are equally likely; that is,

\[
\xi_0((\cdot, i)) = \begin{bmatrix} \frac{1}{3} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \forall i \in \{1, 2, 3\}.
\]

The lump sum reward is

\[
r_1((4, i), a) = \begin{cases} 
100, & \forall i \in \{1, 2, 3\}, a \in A; \\
0, & \text{otherwise},
\end{cases}
\]

and the continuous reward rate is

\[
r_2(\cdot, \cdot, \cdot) = 0.
\]

**Results.** The resulting policy is shown on Fig. 8. Fig. 8(a) shows that if the agent had enough evidence to believe the traffic was medium or high, it should ride its bike (expected time 30) rather than staying on the bus (expected time of 45 and 105 respectively). But in lack of any information initially, the best option is to stay on the bus for one stop (for 5 time units) and then decide (the exact center of the simplex would be a red dot for the bus action). After one stop (Fig. 8(b), medium and high traffic already have significant differences in remaining expected time, making the bike a better option only in the high traffic condition. The expected remaining time are 25 for the bike, and 15, 40 and 95 for the low, medium, and high traffic respectively. But the bike action must be compared against the averaged of the 3 traffic intensity’s expected discounting weighted by the agent’s belief. In Fig. 8(c) and Fig. 8(d) similarly shows the optimal action at stops 2 and 3 given the current belief about the traffic intensities. Note that because we are looking at belief-weighted exponential discounting over stochastic inverse-Gaussian distributed sojourn time, the action decision boundary may not look as it would if we instead assumed a negative reward per time unit and no discounting.
Figure 8: The optimal policy for the bus problem at each bus stop with a regular mesh of belief states in which intensity the agent believes. A red dot • represents a belief whose optimal action is to continue riding the bus, and a blue dot • represents a belief whose optimal action is to stop riding the bus and take the bicycle.
In this problem, we showed the ability of ChronosPerseus to quickly solve POMDPs that involve time without extending the problem state space to represent elapsed time. This would have required the state space with an elapsed time dimension of at least $5 \times 3 \times T$ where $T \geq 125$ for 1 state per time unit. Instead, we naturally place the various durations into sojourn time distributions, and allowed the belief to be updated directly based on the likelihood of the distribution given the observed transitions. Using ChronosPerseus, we see that POSMDPs are not more complicated to solve than their POMDP counterparts. Instead, like using SMDPs in options, it allows for the compression of the temporal properties of the problem. This simplifies not only the creation of the model, but also its evaluation by avoiding the explosion of the state space. This approach also allows us to easily mix various temporal distributions and solve episodic problems as efficiently as non-episodic problems.

4.2 A Machine Maintenance Problem

Zhang and Revie (2017) studied an industrial application of POSMDPs to maintain water (rapid gravity) filters. Filters are essential components in municipal drinking water treatment plants, where the filters can be classified into four states

$$S = \{1 = \text{good}, 2 = \text{acceptable}, 3 = \text{poor}, 4 = \text{awful}\}.$$ 

Unlike the bus problem in Section 4.1, the state of the filter is (fully) hidden and it must be inferred with uncertain observations. To extend the longevity and reliability of the filters, there are four maintenance actions, which are

$$A = \{1 = \text{do nothing}, 2 = \text{backwash}, 3 = \text{dose chemicals}, 4 = \text{replace}\}.$$ 

An observation to infer the state of the filter is to measure turbidity. Turbidity is the amount of cloudiness or haziness of a fluid caused by suspended particles: this is measured by a nephelometer, which shines a light beam through the water sample and measures how much light is reflected into a detector (often located at $90^\circ$ from the source beam). The turbidity of the incoming water into the filter and the turbidity of the outgoing water from the filter are measured. If the outgoing to incoming water turbidity ratio is close to zero, then the filter is in a good state. However, if the ratio is closer to one, the filter is likely to be in a poor or awful state. Hence, the observation space is given by the continuous interval $O = [0, 1]$.

The probability transition matrices are

$$P(\cdot | \cdot, 1) = P(\cdot | \cdot, 2) =
\begin{bmatrix}
  s' = 1 & s' = 2 & s' = 3 & s' = 4 \\
  s = 1 & 0.1043 & 0.7413 & 0.1493 & 0.0051 \\
  s = 2 & 0 & 0.1043 & 0.7413 & 0.1544 \\
  s = 3 & 0 & 0 & 0.1043 & 0.8957 \\
  s = 4 & 0 & 0 & 0 & 1
\end{bmatrix},$$

$$P(\cdot | \cdot, 3) =
\begin{bmatrix}
  s' = 1 & s' = 2 & s' = 3 & s' = 4 \\
  s = 1 & 1 & 0 & 0 & 0 \\
  s = 2 & 0.50 & 0.50 & 0 & 0 \\
  s = 3 & 0.25 & 0.70 & 0.05 & 0 \\
  s = 4 & 0.20 & 0.55 & 0.20 & 0.05
\end{bmatrix},$$
and

\[
P(\cdot , 4) = \begin{bmatrix} s' = 1 & s' = 2 & s' = 3 & s' = 4 \\ s = 1 & 1 & 0 & 0 & 0 \\ s = 2 & 1 & 0 & 0 & 0 \\ s = 3 & 1 & 0 & 0 & 0 \\ s = 4 & 1 & 0 & 0 & 0 \end{bmatrix}.
\]

The sojourn times for actions 1, 2, and 3 are fixed, whereas \( \tau(s' \mid s, 1) = 78.7433, \)
\( \tau(s' \mid s, 2) = 85.3052, \) and \( \tau(s' \mid s, 3) = 3. \) The sojourn time for action 4 follows a truncated
Gaussian distribution with mean \( \mu = 10, \) standard deviation \( \sigma = 1.5, \)
and \( \tau(s' \mid s, 4) > 0; \)
in other words,

\[
f(\tau \mid s, a = 4, s') = \begin{cases} 
\frac{1}{1.5} \cdot \frac{\varphi\left(\frac{\tau - 10}{1.5}\right)}{1 - \Phi\left(\frac{10}{1.5}\right)}, & \text{if } \tau > 0; \\
0, & \text{otherwise,}
\end{cases}
\]

where

\[
\varphi(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}x^2\right)
\]

is the standard normal probability density function, and \( \Phi \) is its corresponding cumulative
distribution function.

To deal with continuous observations, we discretize the interval \([0, 1]\) into \( j \) evenly spaced
numbers, which yields a finite set of observations \( O (|O| = j). \) The observation transition
probability is

\[
G(o \mid a, s') = G(o \mid a) = \frac{f_B(o \mid \varphi(a), \eta(a))}{\sum_{o' \in O} f_B(o' \mid \varphi(a), \eta(a))}
\]

where

\[
f_B(o \mid \varphi(a), \eta(a)) = \frac{\Gamma(\varphi + \eta)}{\Gamma(\varphi)\Gamma(\eta)} o^{\varphi - 1}(1 - o)^{\eta - 1}, \quad \Gamma(z) = \int_0^\infty x^{z-1}e^{-x} \, dx, \quad \Re(z) > 0
\]
is the beta probability density function and the parameters

\[
\varphi(\cdot) = \begin{bmatrix} 2 \\ 6 \\ 18 \\ 18 \end{bmatrix} \quad \text{and} \quad \eta(\cdot) = \begin{bmatrix} 18 \\ 18 \\ 6 \end{bmatrix}.
\]

The beta probability density functions are shown in Fig. 9.

The lump sum reward is

\[
r_1(s, \cdot) = \begin{bmatrix} a = 1 & a = 2 & a = 3 & a = 4 \\ 0 & -100 & -200 & -500 \end{bmatrix} \quad \forall s \in \mathcal{S},
\]
and the continuous reward rate is

\[
    r_2(\cdot, \cdot, s') = \begin{bmatrix}
        a=1 & a=2 & a=3 & a=4 \\
        s=1 & 500 & 500 & -100 & -100 \\
        s=2 & 250 & 250 & -100 & -100 \\
        s=3 & -300 & -300 & -100 & -100 \\
        s=4 & -500 & -500 & -100 & -100 
    \end{bmatrix} \quad \forall s' \in S.
\]

The initial \( \alpha \)-vector is

\[
    \alpha(\cdot) = \begin{bmatrix}
        s=1 & -10^6 \\
        s=2 & -10^6 \\
        s=3 & -10^6 \\
        s=4 & -10^6 
    \end{bmatrix}.
\]

The initial belief \( \xi_0 \) is that the filter is good; that is,

\[
    \xi_0(\cdot) = \begin{bmatrix}
        s=1 & 1 \\
        s=2 & 0 \\
        s=3 & 0 \\
        s=4 & 0 
    \end{bmatrix}.
\]

**Results.** Running ChronosPerseus on the problem with discount rate \( \beta = 0.01 \) generates the set

\[
    V = \left\{ \begin{array}{cccccc}
        \text{a=backwash} & 46351.3242 & 43257.2109 & 43404.0859 & 43690.0625 & 43987.1328 \\
        \text{a=dose} & 31551.5723 & 41467.2891 & 41505.2148 & 41519.4531 & 41352.2109 \\
        \text{V(good)} & -71.0176 & 40529.1953 & 40516.1875 & 40054.4766 & 40036.6328 \\
        \text{V(acceptable)} & -11550.9883 & 40172.0430 & 40159.6211 & 40057.0312 & 39764.9141 \\
        \text{V(poor)} & & & & & \\
        \text{V(awful)} & & & & & \\
        \text{V(good)} & 43567.4844 & 42355.8906 & 43890.7227 & 43800.3984 & 44187.7969 \\
        \text{V(acceptable)} & 41526.0000 & 40804.3320 & 41438.7500 & 41149.0391 & 40967.9141 \\
        \text{V(poor)} & 40470.5047 & 39155.7461 & 40201.9414 & 40313.0742 & 39406.7695 \\
        \text{V(awful)} & 40112.9766 & 39067.2930 & 39896.9844 & 39984.2852 & 39265.6484 \\
    \end{array} \right\}
\]
where $V$ contains 13 $\alpha$-vectors with their corresponding actions labeled at the top. With $|B| = 5000$ sampled belief points and 40 iterations, it took around 16 hours for Zhang and Revie (2017) (Intel Core i5-4590 CPU @ 3.30 GHz) to complete their calculations, while ChronosPerseus took less than 40 seconds (Intel Core Xeon Silver 4210 CPU @ 2.20 Ghz, Nvidia GeForce RTX 2080 Super 8 GB). Their paper did not state the number of bins of observations, so we selected 100 for our simulation.

Table 1 lists some belief points with their corresponding optimal value and optimal action from ChronosPerseus alongside the results of Zhang and Revie (2017). The beliefs in Table 1 were selected by Zhang and Revie, and these values were included in the sampled belief set $B$ of ChronosPerseus as well. At the top of the column, these beliefs correspond to a filter believed to be in good condition and descend to acceptable, poor, and finally awful. We notice that the ChronosPerseus has selected to backwash for a filter believed to be in good condition compared to Zhang and Revie whose action is do-nothing. The accumulated reward for backwash is more than the accumulated reward for doing nothing: $R(1, 1) = 27249.43$ while $R(1, 2) = 28594.38$. There is more time during the backwash, which allows accumulation of continuous reward overcoming the initial lump negative reward of $-100$ from backwash compared to the time allowed for the do-nothing action. The accumulated reward has to compensate for the risk of getting bad water (such as ending up in the poor or awful state). Hence, we believe that ChronosPerseus has selected the correct action.

Table 1: Sample of belief points, their corresponding optimal values and actions

| Belief state, $\xi$ | ChronosPerseus | Zhang and Revie (2017) |
|---------------------|----------------|-----------------------|
|                     | Optimal Value  | Optimal Action        | Optimal Value  | Optimal Action        |
| $[0.9972, 0.0028]_T$ | 46309.8867     | 2                     | 46316.40       | 1                     |
| $[0.9965, 0.0035]_T$ | 46299.5234     | 2                     | 46306.04       | 1                     |
| $[0.8714, 0.1286]_T$ | 44448.0742     | 2                     | 44545.43       | 2                     |
| $[0.8160, 0.1840]_T$ | 43628.1680     | 2                     | 43634.51       | 2                     |
| $[0.0031, 0.6803]_T$ | 41197.9805     | 3                     | 41215.31       | 3                     |
| $[0.0001, 0.0390]_T$ | 40560.6250     | 3                     | 40574.81       | 3                     |
| $[0.0000, 0.0003]_T$ | 40504.4453     | 4                     | 40498.43       | 4                     |
| $[0.0000, 0.0000]_T$ | 40504.4414     | 4                     | 40385.84       | 4                     |

In this section, we showed that ChronosPerseus could solve real-world POSMDP problems with continuous observation spaces and that importance sampling combined with GPU implementation provides few orders of magnitude speed up the computations allowing for a higher number of iterations and thus a longer horizon.
5. Summary and Conclusion

The partially observable semi-Markov decision process (POSMDP) model has been used for decades in the operations research community to provide planning under uncertainty. While it has been used for tasks such as maintenance scheduling, it can be equally valuable in the artificial intelligence community by modeling not only the interaction between an agent and its environment but also dealing with the issue of time—that is, how long it takes for the agent to transition from one hidden state to another. Moreover, solving a POSMDP rather than an equivalent time-extended state-space POMDP using importance sampling is faster by avoiding the huge state-space explosion that adding time-steps would require, in a way similar to the use of SMDPs and options in reinforcement learning.

In this paper, we presented a new POSMDP solver—**ChronosPerseus**—combining *Perseus* and importance sampling to solve a POSMDP where the transition time is observable. We showed that the solver works on episodic and non-episodic problems, with mixed-observably, discrete or continuous observation space, and a mixture of fixed and stochastic continuous sojourn times. Furthermore, we demonstrate how it could learn a policy on a POMDP where the time is the only available information to resolve the partially observable state, as this may happen in many real-world problems.

Acknowledgments

This research has been supported by CD-ARP research grants from Canadian Defence Academy and the Royal Military College of Canada to Dr. F. Rivest. Icons in Figure 4 are provided by The Noun Project: *Bus* by Rainbow Designs, *Bus Stop Square* by Rihards Gromuls, *man and bicycle* by Gan Khoon Lay, and *clock* and *Time* by kiddo.

References

K. J. Åström. Optimal control of Markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*, 10(1):174–205, February 1965. doi:10.1016/0022-247X(65)90154-X

Masanao Aoki. Optimal control of partially observable Markovian systems. *Journal of the Franklin Institute*, 280(5):367–386, November 1965. doi:10.1016/0016-0032(65)90528-4

Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957a.

Richard Bellman. A Markovian decision process. *Indiana University Mathematics Journal*, 6(4):679–684, 1957b. doi:10.1512/iumj.1957.6.56038

Ronald Aylmer Fisher. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 222:309–368, January 1922. doi:10.1098/rsta.1922.0009

J. M. Hammersley and D. C. Handscomb. *Monte Carlo Methods*. Monographs on Applied Probability and Statistics. John Wiley and Sons, New York, 1964. doi:10.1007/978-94-009-5819-7.
Onésimo Hernández-Lerma. *Adaptive Markov Control Processes*, volume 79 of *Applied Mathematical Sciences*. Springer-Verlag, New York, 1989. doi:10.1007/978-1-4419-8714-3.

Ronald A. Howard. Semi-Markovian decision processes. *Bulletin of the International Statistical Institute*, 40(2):625–652, 1963. From the Proceedings of the 34th Session of the International Statistical Institute, Ottawa, Canada, August 21-29, 1963.

Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, pages 541–548, 1999.

Michail Maniadakis and Panos Trahanias. Temporal cognition: A key ingredient of intelligent systems. *Frontiers in Neurorobotics*, 5:2, 2011. doi:10.3389/fnbot.2011.00002.

Sylvie C. W. Ong, Shao Wei Png, David Hsu, and Wee Sun Lee. Planning under uncertainty for robotic tasks with mixed observability. *International Journal of Robotics Research*, 29(8):1053–1068, 2010. doi:10.1177/0278364910369861.

Christos H. Papadimitriou and John N. Tsitsiklis. The complexity of Markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, August 1987. doi:10.1287/moor.12.3.441.

Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1025–1030, August 2003.

Joelle Pineau, Geoffrey Gordon, and Sebastian Thrun. Anytime point-based approximations for large POMDPs. *Journal of Artificial Intelligence Research*, 27:335–380, November 2006. doi:10.1613/jair.2078.

Doina Precup. *Temporal Abstraction in Reinforcement Learning*. PhD thesis, University of Massachusetts Amherst, 2000.

François Rivest and Richard Kohar. A new timing error cost function for binary time series prediction. *IEEE Transactions on Neural Networks and Learning Systems*, 31(1):174–185, Jan 2020. doi:10.1109/tnnls.2019.2900046.

Sheldon M. Ross. *Applied Probability Models with Optimization Applications*. Holden-Day, 1970.

Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, New Jersey, 3rd edition, 2010.

Guy Shani, Joelle Pineau, and Robert Kaplow. A survey of point-based POMDP solvers. *Autonomous Agents and Multi-Agent Systems*, 27(1):1–51, July 2013. doi:10.1007/s10458-012-9200-2.

Edward Jay Sondik. *The Optimal Control of Partially Observable Markov Processes*. PhD thesis, Standford University, 1971.
Matthijs T. J. Spaan. Partially observable Markov decision processes. In Marco Wiering and Martijn van Otterlo, editors, *Reinforcement Learning: State-of-the-Art*, volume 12 of *Adaptation, Learning, and Optimization*, chapter 12, pages 387–414. Springer Berlin Heidelberg, 2012. doi:10.1007/978-3-642-27645-3_12.

Matthijs T. J. Spaan and Nikos Vlassis. Perseus: Randomized point-based value iteration for POMDPs. *Journal of Artificial Intelligence Research*, 24:195–220, 2005. doi:10.1613/jair.1659.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. Adaptive Computation and Machine Learning. MIT Press, 2nd edition, 2018.

Richard S. Sutton, Doina Precup, and Satinder Singh. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112 (1-2):181–211, August 1999. doi:10.1016/S0004-3702(99)00052-1

M. C. K. Tweedie. Statistical properties of inverse Gaussian distributions. I. *The Annals of Mathematical Statistics*, 28:362–377, 1957. doi:10.1214/aoms/1177706964.

Kazuyoshi Wakuta. Semi-Markov decision processes with incomplete state observation: average cost criterion. *Journal of the Operations Research Society of Japan*, 24(2):95–108, June 1981.

Kazuyoshi Wakuta. Semi-Markov decision processes with incomplete state observation: discounted cost criterion. *Journal of the Operations Research Society of Japan*, 25(4):351–361, December 1982.

Chelsea C. White. Procedures for the solution of a finite-horizon, partially observed, semi-Markov optimization problem. *Operations Research*, 24(2):348–358, April 1976. doi:10.1287/opre.24.2.348.

Huizhen Yu. *Approximate Solutions for Partially Observable Markov and Semi-Markov Decision Processes*. PhD thesis, Massachusetts Institute of Technology, 2006.

Mimi Zhang and Matthew Revie. Continuous-observation partially observable semi-Markov decision processes for machine maintenance. *IEEE Transactions on Reliability*, 66(1):202–218, March 2017. doi:10.1109/tr.2016.2626477.