Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling (ICAPS 2020)

Multiple-Environment Markov Decision Processes: Efficient Analysis and Applications

Krishnendu Chatterjee,[1] **Martin Chmelík,**[2] **Deep Karkhanis,**[3] **Petr Novotný,**[4] **Amélie Royer**[1]
[1]Institute of Science and Technology Austria, Klosterneuburg, Austria [2]Google LLC, Zürich, Switzerland
[3]IIT Bombay, India [4]Faculty of Informatics, Masaryk University, Brno, Czech Republic
{krishnendu.chatterjee, amelie.royer}@ist.ac.at, {martin.chmelik.cz, deepkarkhanis}@gmail.com,
petr.novotny@fi.muni.cz

Abstract

Multiple-environment Markov decision processes (MEMDPs) are MDPs equipped with not one, but multiple probabilistic transition functions, which represent the various possible unknown environments. While the previous research on MEMDPs focused on theoretical properties for long-run average payoff, we study them with discounted-sum payoff and focus on their practical advantages and applications. MEMDPs can be viewed as a special case of Partially observable and Mixed observability MDPs: the state of the system is perfectly observable, but not the environment. We show that the specific structure of MEMDPs allows for more efficient algorithmic analysis, in particular for faster belief updates. We demonstrate the applicability of MEMDPs in several domains. In particular, we formalize the sequential decision-making approach to contextual recommendation systems as MEMDPs and substantially improve over the previous MDP approach.

1 Introduction

Planning under probabilistic uncertainty is one of the core problems in artificial intelligence, and Markov decision processes (MDPs) are the standard model for it. MDPs consist of a finite state space and finite action space along with a probabilistic transition function that, given a state and action, gives the probability distribution of the next state (Howard 1960). Every transition is also assigned a reward value. A history is a sequence of state and action pairs, and a policy specifies the choice of the next action for every history. The classical objective for MDPs is to maximize the expected discounted sum of the rewards (Puterman 1994).

To further model uncertainty in the planning process, the MDP model is extended with partial observation, such that state space is not perfectly observable. This leads to partially-observable MDPs (POMDPs) (Sondik 1971; Papadimitriou and Tsitsiklis 1987; Littman 1996). While significant progress has been made for point-based solvers for POMDPs (Kurniawati, Hsu, and Lee 2008; Pineau et al. 2003; Smith and Simmons 2012; Shani, Brafman, and Shimony 2007; Spaan 2004), solving large POMDPs is still a challenging problem and an active research area.

Given the challenge to solve large POMDPs, an attractive research direction is to identify a special subclass of POMDPs allowing more efficient analysis. In particular, there are two properties or challenges that must be satisfied: first, this special class must allow for more efficient algorithmic analysis, and second, it must be able to model interesting applications. For example, to model robotics applications, *mixed observability MDPs* (MOMDPs) have been considered; in this case, the state space is a product space where the first component is perfectly observable and the second component is partially observable (Ong et al. 2010).

In this work, we consider multi-environment MDPs (MEMDPs) (Raskin and Sankur 2014). MEMDPs are MDPs equipped with not one, but multiple probabilistic transition functions. These multiple transition functions represent the various possible unknown environments. We study the problem of computing policies in MEMDPs with discounted-sum payoff that maximizes the expected payoff, as well as their practical advantages and applications.

Our contributions are as follows:

1. *Efficient analysis.* We show that MEMDPs are a special class of MOMDPs (hence also POMDPs): there is no information about the unobservable component and the unobservable component does not change state. We show that this specific structure of MEMDPs allows for more efficient algorithmic analysis; in particular, they have sparse transitions, which allows for faster belief updates (linear as opposed to quadratic). We also prove another property of MEMDPs: they have monotone average belief entropy, i.e. the uncertainty about the current state never increases.
2. *Applications.* We show that MEMDPs with discounted-sum payoff provide adequate model for several applications, in particular for recommendation systems and parametric MDPs with unknown parameter values. In a recommendation system, MDPs represent the behavior of customers, and hence the different probabilistic transition function represents different types of customers, and the decision making proceeds without knowing the precise type. Hence recommendation systems are naturally modeled as MEMDPs. Parametric MDPs consist of MDPs where certain transition probabilities depend on some parameter values. When the parameter values are unknown, but each parameter has a discrete range, then every combination of parameter values represents an environment, and thus they are naturally modeled as MEMDPs.

3. *Implementation and experimental results.* We propose an implementation[1] of our efficient solutions for MEMDPs, and report experimental results for stochastic maze exploration, recommendation systems, and parametric MDPs; and show that our approach outperforms the existing approaches.

Related Work. We discuss relevant related works in the items below.

- *MDPs and POMDPs.* MDPs are a standard model for planning under uncertainty (Puterman 1994). Their partially-observable extensions, the POMDPs, have also been widely studied, both for theoretical results (Littman 1996; Papadimitriou and Tsitsiklis 1987) as well as practical tools (Kurniawati, Hsu, and Lee 2008; Silver and Veness 2010), including point-based methods (Pineau et al. 2003; Smith and Simmons 2012; Spaan 2004) and simulation techniques (Geffner and Bonet 1998; Silver and Veness 2010).
- *MOMDPs.* MOMDPs and efficient algorithms for them have been considered in literature for robotics applications (Ong et al. 2010). MEMDPs are a special class of MOMDPs that allows for much simpler belief updates, and yet can model interesting applications.
- *MEMDPs.* MEMDPs have been studied with the focus on theoretical properties and for long-run average payoff in (Raskin and Sankur 2014). However, this is quite different from the discounted-sum payoff scenario. Since long-run average payoffs are independent of finite prefixes, policies can play long to learn (explore) in MEMDPs with such payoff, and then play optimally with respect to the environment (exploit). In MEMDPs with discounted-sum payoff there is an exploration-vs-exploitation tradeoff. Moreover, in contrast to the theoretical study of MEMDPs with long-run average payoff, we focus on practical approaches to solve MEMDPs with discounted-sum payoff. We show how several important problems can be modelled in this framework, and present experimental results.
- *Parametric MDPs.* The control problem for parametric Markov chains has been considered in (Ceska et al. 2019), however, this work does not consider MDPs. The control problem for parametric MDPs has been studied in (Arming et al. 2018); however, general POMDPs are used for algorithmic analysis, whereas we propose MEMDPs to model them and present efficient algorithm for MEMDPs.
- *MDPs with Imprecise Parameters (MDPIPs).* In MDPIPs (White III and Eldeib 1994; Delgado, Sanner, and de Barros 2011; Itoh and Nakamura 2007), there is a continuum of possible transition functions and a "nature"player can select any transition probability function for each action in every decision epoch. In contrast, in MEMDPs the environment is fixed throughout the duration of the interaction, but it is not known precisely.
- *Related models.* In Interval MDPs (Sen, Viswanathan, and Agha 2006), the transition probabilities are not known but belong to known intervals. Hence, they can be seen as continuous counterparts of MEMDPs. In (Fern and Tadepalli 2010) they study Hidden Goal MDPs, which are analogous to MEMDPs. However, the authors actually study a restricted version of the model suitable for modeling interactive assistants (HAMDPs), and the focus is on theoretical aspects (theoretical complexity and regret analysis). In (Doshi-Velez and Konidaris 2013) they study Hidden parameter MDPs, where Hidden parameters influence transition dynamics of the MDP. Their focus is on defining a suitable generative model of how the hidden parameters influence the transition dynamics and on learning an instance of such a model from observational data. In (Poupart et al. 2006) they employ a sophisticated representation of the optimal value function to adapt the classical value iteration to models where the hidden parameter space is uncountable. While the approach is very generic and elegant, the presented experiments run on systems with at most 10 observable states. Finally, in (Wang et al. 2012) they focus on proving that the uncountable-space model of (Poupart et al. 2006) can be approximated by suitable Monte-Carlo based discretization, which produces models analogous to MEMDPs. The focus is on defining this discretization and proving that the resulting discrete system approximates the original one, but the discrete system is solved using a point-based solver SARSOP for general POMDPs. In contrast, we focus on two main aspects: First, we present efficient algorithmic analysis of MEMDPs with discrete environment space and arbitrary transition dynamics within each environment. We show that both point-based and simulation-based general POMDP solvers can be significantly optimized to exploit the structure of MEMDPs. One particular novelty of our paper is proving that MEMDPs admit linear-time belief updates. Second, we show that MEMDPs together with our improved POMDP solvers are effective for solving real-life problems with large number of states (with focus on recommender systems). To summarize, our paper complements the previous works on analogous models by focusing on efficient optimization in discrete-space MEMDPs and showing the applicability of these models on a real-life case-study.
- *Recommendation systems.* In previous work, recommendation systems have been modelled as MDPs (Shani, Heckerman, and Brafman 2005; Brafman, Heckerman, and Shani 2003). However, these approaches ignore the aspects of unknown environments, i.e. they aggregate the behaviour of all customers into a single probabilistic environment. We show that modelling as MEMDPs, to take the variability of customer preferences into account, can improve the performance of contextual recommendation systems.

[1] https://github.com/ameroyer/ReCA

2 Preliminaries

We present the definitions of POMDPs, MEMDPs, strategies, objectives, and other basic notions required for our results. Throughout this work, we follow standard POMDP notations from (Puterman 1994; Littman 1996).

2.1 Partially Observable Markov Decision Processes

We denote by $\mathcal{D}(X)$ the set of all probability distributions on a finite set X, i.e. all functions $f : X \to [0,1]$ s.t. $\sum_{x \in X} f(x) = 1$. For $f \in \mathcal{D}(X)$ we denote by $\mathrm{Supp}(f)$ the *support* of f, i.e. the set $\{x \in X \mid f(x) > 0\}$.

Definition 1 ***POMDPs.*** *A* Partially Observable Markov Decision Process (POMDP) *is defined as a tuple* $P = (Q, \mathcal{A}, \delta, r, \mathcal{Z}, \mathcal{O}, \lambda)$ *where* Q *is a finite set of* states*;* $\mathcal{A}$ *is a finite alphabet of* actions*;* $\delta : Q \times \mathcal{A} \to \mathcal{D}(Q)$ *is a* probabilistic transition function *that given a state* s *and an action* $a \in \mathcal{A}$ *gives the probability distribution over the successor states, i.e.,* $\delta(q,a)(q')$ *denotes the transition probability from* q *to* q' *given action* a*;* $r : Q \times \mathcal{A} \to \mathbb{R}$ *is a* reward *function;* $\mathcal{Z}$ *is a finite set of* observations*;* $\mathcal{O} : Q \to \mathcal{D}(\mathcal{Z})$ *is a probabilistic* observation function *that maps every state to an distribution over observations; and* $\lambda \in \mathcal{D}(Q)$ *is the* initial belief.

We use $\delta(q'|q,a)$ as a shorthand for $\delta(q,a)(q')$, $\mathcal{O}(o|q)$ as a shorthand for $\mathcal{O}(q)(o)$, etc.

Perfectly observable Markov Decision Processes (MDPs) can be seen as a special case of POMDPs where $\mathcal{Z} = Q$ and for each state q the distribution $\mathcal{O}(q)$ assigns 1 to q.

Plays and Histories. A *play* (or a path) in a POMDP is an infinite sequence $\rho = (q_0, a_0, q_1, a_1, q_2, a_2, \ldots)$ of states and actions such that $q_0 \in \mathrm{Supp}(\lambda)$ and for all $i \geq 0$ we have $\delta(q_i, a_i)(q_{i+1}) > 0$. We write Ω for the set of all plays. A *finite path* (or just *path*) is a finite prefix of a play ending with a state, i.e. a sequence from $(S \cdot \mathcal{A})^* \cdot S$. A *history* is a finite sequence of actions and observations $h = a_1 o_1 \ldots a_{i-1} o_i \in (\mathcal{A} \cdot \mathcal{Z})^*$ s.t. there is a path $w = s_0 a_1 s_1 \ldots a_{i-1} s_i$ with $o_j \in \mathrm{Supp}(\mathcal{O}(s_j))$ for each $1 \leq j \leq i$.

Beliefs. A belief b is a probability distribution on the set of states, i.e., $b \in \mathcal{D}(Q)$. We denote by $\mathcal{B}$ the set of all beliefs. Informally, a belief gives the probability of being in a particular state of the POMDP given the past history of observations and actions. At each point of a play, the current belief can be exactly determined by looking at the prefix of the play up to this point: the initial belief b_0 is given as a part of the POMDP; in i-th step, when the current belief is b_i, an action $a_i \in \mathcal{A}$ is played and an observation $o \in \mathcal{Z}$ is received, it is straightforward to compute the updated belief b_{i+1} for the new situation (Cassandra 1998).

Infinite-horizon Discounted Payoff. Given a play $\rho = (q_0, a_0, q_1, a_1, q_2, a_2, \ldots)$ and a discount factor $0 \leq \gamma < 1$, the *infinite-horizon discounted payoff* Disc_γ of the play ρ is $\mathsf{Disc}_\gamma(\rho) = \sum_{i=0}^{\infty} \gamma^i r(q_i, a_i)$.

Strategies (or Policies). A *strategy (or a policy)* is a recipe to extend prefixes of plays. Formally, it is a function $\sigma : (\mathcal{Z} \cdot \mathcal{A})^* \cdot \mathcal{Z} \to \mathcal{D}(\mathcal{A})$ that given a finite history of observations and actions selects a probability distribution over the actions.

Values of Strategies. Given a POMDP P, a strategy σ, and a discount factor γ, the value ν of σ is the expected value of the infinite-horizon discounted payoff under strategy σ, i.e. the quantity $\nu(\sigma) = \mathbb{E}^\sigma_\lambda[\mathsf{Disc}_\gamma]$.

Remark 1 *For maximizing (or minimizing) the expected infinite-horizon discounted sum, deterministic belief-based strategies (i.e. strategies* σ *such that for each history* h *the distribution* $\sigma(h)$ *is determined solely by the belief after observing* h*) are known to be sufficient (Sondik 1971).*

2.2 Multi-Environment Markov Decision Processes

We define *Multi-environment Markov Decision Processes (MEMDPs)* analogously to the definition in (Raskin and Sankur 2014). Informally, an MEMDP consists of a finite collection of perfectly observable MDPs, all of which have an identical state space but differ in their transition functions (i.e. in the probabilistic environments). We start in one of these MDPs, selected at random. During the play we can observe the current state, but we cannot observe in which of the MDPs the state lies (the knowledge of the state does not yield any knowledge about the MDP, as the MDPs have the same state set). During the course of the play it is not possible to switch to a different MDP, i.e. we play against a randomly chosen but then fixed environment. A formal definition of an MEMDP is given below.

Definition 2 ***MEMDPs.*** *An MEMDP is a tuple* $M = (\mathcal{I}, S, \mathcal{A}, \{\delta_i\}_{i \in \mathcal{I}}, \{r_i\}_{i \in \mathcal{I}}, s_0, \lambda)$ *where* $\mathcal{I}$ *is a finite set of* environments*;* S *is a finite set of* control states*;* $\mathcal{A}$ *is a finite alphabet of actions;* $\{\delta_i\}_{i \in \mathcal{I}}$ *is a collection of probabilistic transition functions, one for every environment* $i \in \mathcal{I}$*, each being of type* $\delta_i : S \times \mathcal{A} \to \mathcal{D}(S)$*;* $\{r_i\}_{i \in \mathcal{I}}$ *: is a set of reward functions, each of type* $r_i : S \times \mathcal{A} \to \mathbb{R}$*;* $s_0 \in S$ *is the initial state; and* $\lambda \in \mathcal{D}(\mathcal{I})$ *is the initial distribution over the environments. For simplicity and without loss of generality we assume that all actions are enabled in all states and all environments, i.e., for every state* $s \in S$*, environment* $i \in \mathcal{I}$*, and every action* a *the support of the distribution* $\delta_i(s,a)$ *is non-empty.*

Each MEMDP M can be translated, in a natural way, into a POMDP P_M whose state set is $Q = S \times \mathcal{I}$, observation set is $\mathcal{Z} = S$, transition function and rewards are derived from M, and where observation received in state (s,i) equals s with probability 1. Formally, for an MEMDP $M = (\mathcal{I}, S, \mathcal{A}, \{\delta_i\}_{i \in \mathcal{I}}, \{r_i\}_{i \in \mathcal{I}}, s_0, \lambda)$ we construct a POMDP $P_M = (Q_M, \mathcal{A}_M, \delta_M, r_M, \mathcal{Z}_M, \mathcal{O}_M, \lambda_M)$ where (i) the set of states is $Q_M = S \times \mathcal{I}$; (ii) the set of actions remains unchanged, i.e., $\mathcal{A}_M = \mathcal{A}$; (iii) the transition function is defined as $\delta_M((s,i),a)((s',i')) = \delta_i(s,a)(s')$ if $i = i'$ and 0 otherwise; (iv) the reward function is defined as $r_M((s,i),a) = r_i(s,a)$; (v) the set of observations is $\mathcal{Z}_M = S$; (vi) the observation function is defined as $\mathcal{O}_M((s,i))(s') = 1$ if $s = s'$ and 0 otherwise; (vii) the initial belief is defined as $\lambda_M((s,i)) = \lambda(i)$ if $s = s_0$ and 0 otherwise.

Example 1 *Consider an MEMDP* M *with* $\mathcal{I} = \{1,2\}$*,* $S = \{s,t\}$*,* $\mathcal{A} = \{a,b\}$ *and* δ_1, δ_2 *as follows:* $\delta_1(s,a)$ *and* $\delta_1(t,b)$ *both assign probability* 1 *to state* s*,* $\delta_1(s,b)$ *and* $\delta_1(t,a)$ *both assign probability* 1 *to* t*, and* $\delta_2(s,a), \delta_2(s,b), \delta_2(t,a), \delta_2(t,b)$ *are all identical, assigning probability* $\frac{1}{2}$ *to each of the two states. Assume* s *is the*

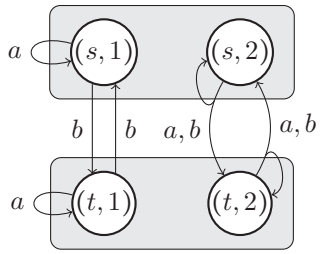

Figure 1: A POMDP associated to the MEMDP from Example 1. Action names are pictured next to corresponding transitions, grey rectangles indicate observations (note that the observation function is deterministic, i.e. each state produces one concrete observation with probability 1). For succinctness, rewards (all of them equal to 1) and branching probabilities are not pictured. The initial belief assigns probability $\frac{1}{2}$ to $(s, 1)$ and $(s, 2)$.

initial state, the initial distribution over $\mathcal{I}$ is uniform, and all rewards equal 1. The POMDP P_M is pictured in Figure 1.

3 Advantages of MEMDPS and Relationship to MOMDPS

In this section, we show that MEMDPs are a special case of mixed-observability MDPs (MOMDPs), while offering additional advantages: sparse transition representation, faster belief updates and monotonic average belief entropy.

MOMDPs. Mixed-observability MDPs were introduced in (Ong et al. 2010) as a model of systems in which certain features of the state space are perfectly observable while the others are not and can be observed only through imprecise sensors. Formally, MOMDPs can be viewed as POMDPs whose states are tuples $(x, y) \in \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are sets of perfectly and imperfectly observable features, respectively. The transition function δ is given by two functions $\delta_{\mathcal{X}}: \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \to \mathcal{D}(\mathcal{X})$ and $\delta_{\mathcal{Y}}: \mathcal{X} \times \mathcal{Y} \times \mathcal{A} \times \mathcal{X} \to \mathcal{D}(\mathcal{Y})$ such that $\delta((x', y')|x, y, a) = \delta_{\mathcal{X}}(x'|x, y, a) \cdot \delta_{\mathcal{Y}}(y'|x, y, a, x')$. Since the features in $\mathcal{X}$ are perfectly observable, we can view the observations in MOMDPs as tuples (x, z), where $x \in X$, and z belongs to a fixed set of observations $\mathcal{Z}_{\mathcal{Y}}$. The observation function $\mathcal{O}$ then satisfies, for each $(x, y) \in \mathcal{X} \times \mathcal{Y}$, the condition $\mathrm{Supp}(\mathcal{O}(x, y)) \subseteq \{x\} \times \mathcal{Z}_{\mathcal{Y}}$. This shows that MOMDPs can be viewed as a sub-case of POMDPs.

MOMDPs vs. MEMDPs. We observe that MEMDPs can be in turn viewed as a special case of MOMDPs: a MEMDP $(\mathcal{I}, S, \mathcal{A}, \{\delta_i\}_{i \in \mathcal{I}}, \{r_i\}_{i \in \mathcal{I}}, s_0, \lambda)$ can be framed as a MOMDP by putting $\mathcal{X} = S$, $\mathcal{Y} = \mathcal{I}$, and $\mathcal{Z}_{\mathcal{Y}}$ being a singleton set (since in MEMDPs we do not observe, even partially, any aspect of the environment apart from how it affects the probabilistic outcomes of actions), and by setting $\delta_{\mathcal{X}}(x'|x, y, a) = \delta_y(x'|x, a)$ (note that y is an environment) and $\delta_{\mathcal{Y}}(y'|x, y, a, x') = \Delta_y(y')$, where Δ_y is a *Dirac* distribution concentrated on y: $\Delta_y(y')$ is 1 if $y' = y$ and is equal to 0 otherwise.

To complete our picture of the relationships between POMDPs, MOMDPs, and MEMDPs, we note that each POMDP $(Q, \mathcal{A}, \delta, r, \mathcal{Z}, \mathcal{O}, \lambda)$ can be viewed as a MOMDP with $\mathcal{X} = \{d\}$, where d is a dummy element, $\mathcal{Y} = Q$, $\delta_{\mathcal{X}}(d|d, q, a) = 1$, and $\delta_{\mathcal{Y}}(q'|d, q, a, d) = \delta(q'|q, a)$ for all $(q, a) \in Q \times \mathcal{A}$. Hence, MOMDPs encompass the whole structural complexity of POMDPs. On the other hand, the form of MEMDPs is truly restricted, since there cannot be a transition of positive probability between two states that may yield the same observation.

To summarize, the special structure of MEMDPs rests in the fact that the partially observable features are actually

- *hidden*, i.e. $\mathcal{Z}_{\mathcal{Y}}$ is a singleton, the only observations we receive are the perfect observations of observable features.
- *static*, i.e. once the initial partially observable (PO) feature (the environment) is sampled, it is not changed ever again.

In particular, MEMDPs are indeed not an alternative to (P/M)OMDPs, but a special case, with a special structure exploitable for more efficient analysis.

In the following paragraphs we demonstrate three concrete advantages of this special structure.

Sparse Transitions. As follows from the definition of a MOMDP, to encode its transition function we need at most $|\mathcal{X}|^2 \cdot |\mathcal{Y}| \cdot |\mathcal{A}| + |\mathcal{X}|^2 \cdot |\mathcal{Y}|^2 \cdot |\mathcal{A}|$ numerical entries (the first term corresponds to $\delta_{\mathcal{X}}$, the second to $\delta_{\mathcal{Y}}$). On the other hand, representing a transition function of a MEMDP requires at most $|S|^2 \cdot |\mathcal{I}| \cdot |\mathcal{A}|$, which is asymptotically better than what MOMDP might need, as there is no quadratic dependence on $\mathcal{Y}$ (recall that in the MEMDP–MOMDP correspondence we have $\mathcal{X} = S$ and $\mathcal{Y} = \mathcal{I}$). Having a compact representation of a transition function allows for more efficient utilization of memory, which is important for analysis of large POMDPs.

Faster Belief Updates. The belief update is a very computationally demanding part of algorithmic POMDP analysis. Given a previous belief b, a last played action a and a consequently received observation o, the task is to compute the current belief $U^b_{a,o}$. The full belief update can be computed via a straightforward application of Bayes' formula: the probability of being in a state q under the new belief is

$$U^b_{a,o}(q) = \frac{\mathcal{O}(o|q) \cdot \sum_{q' \in Q} \left(b(q') \cdot \delta(q|q', a)\right)}{\sum_{q' \in Q} b(q') \cdot \sum_{q'' \in Q} \delta(q''|q', a) \cdot \mathcal{O}(o|q'')}. \quad (1)$$

The denominator can be computed using quadratic (in size of the state set) number of arithmetic operations, due to the double summation, but it is independent of q. The nominator requires a linear number of operations, but needs to be computed for each q. Hence, the standard update needs a quadratic number of operations, a rather demanding task if the state space is large. In MOMDPs one only needs to maintain the belief over the partially observable features, so in (1) the summations range over $\mathcal{Y}$. Although this somewhat improves the computation, since the number of partially observable features can be smaller than the number of states, the quadratic dependence remains. In MEMDPs, the only uncertainty is about the environment, i.e. beliefs can be viewed as tuples (s, b), where $s \in S$ and $b \in \mathcal{D}(\mathcal{I})$ is an *environment belief*. Given a previous belief (s, b), an action a, and an observation $t \in S$ (recall that observations in MEMDPs are the same as control states), the new belief is $(t, U^{(s,b)}_{a,t})$, where

$U_{a,t}^{(s,b)}$ is computed via (1) as

$$U_{a,t}^{(s,b)}(i) = \frac{\sum_{i' \in \mathcal{I}} b(i') \cdot \delta((t,i)|(s,i'),a)}{\sum_{i' \in \mathcal{I}} b(i') \cdot \sum_{i'' \in \mathcal{I}} \delta((t,i'')|(s,i'),a)}. \quad (2)$$

But due to the partially observable features being static in MEMDPs, we have $\delta((t,i)|(s,i'),a) = \delta_i(t|s,a)$ if $i = i'$ and $\delta((t,i)|(s,i'),a) = 0$ otherwise. Hence, (2) reduces to

$$U_{a,t}^{(s,b)}(i) = \frac{b(i) \cdot \delta_i(t|s,a)}{\sum_{i' \in \mathcal{I}} b(i') \cdot \delta_{i'}(t|s,a)}. \quad (3)$$

The nominator can be computed, for each i, with a single arithmetic operation, while the denominator can be computed once with linearly many (in $|\mathcal{I}|$) operations and again re-used for all i. Hence, in MEMDPs, the environment belief update requires *linearly* many operations.

Monotonic Average Belief Entropy. In POMDPs and MOMDPs it may inevitably happen that the uncertainty about the current state strictly increases. Indeed, consider a simple POMDP with three states q, u, v, a single action a, two observations o_1, o_2 such that in q we always receive o_1 and in u and v we always receive o_2, and transition function δ s.t. $\delta(u|q,a) = \delta(v|q,a) = \frac{1}{2}$ and $\delta(u|u,a) = \delta(v|v,a) = 1$. Whenever we are in state q, we know it exactly, i.e. the belief in such a situation is $\bar{b} = (1,0,0)$. However, after playing the only possible action a the new belief is $U_{a,o_2}^{\bar{b}} = (0, \frac{1}{2}, \frac{1}{2})$. To formalize the notion of "uncertainty about the current state," we use the standard notion of *Shannon entropy* (Shannon 2001). The Shannon entropy (or simply, entropy) of a discrete distribution $f \in \mathcal{D}(X)$, where X is a finite set, is defined as

$$H(f) := \sum_{x \in X} f(x) \cdot \log_2 \left(\frac{1}{f(x)} \right)$$

(a convention stipulates that $0 \cdot \log_2(1/0) = 0$). The higher the entropy, the more uncertainty there is about the outcome of sampling from f. In particular, the entropy of a belief b represents the uncertainty about the current state. In the example above, $H(\bar{b}) = 0$ and $H(U_{a,o_2}^{\bar{b}} = (0, \frac{1}{2}, \frac{1}{2})) = 1$. MEMDPs differ very much in this respect: no matter the environment belief and which action we take, the entropy of the subsequent belief will not increase on average (where the average is taken over all observations, i.e. states of the MEMDP, that we can receive after taking the action).

Theorem 1 *Let $M = (\mathcal{I}, S, \mathcal{A}, \{\delta_i\}_{i \in \mathcal{I}}, \{r_i\}_{i \in \mathcal{I}}, s_0, \lambda)$ be a MEMDP and let $(s,b) \in S \times \mathcal{D}(\mathcal{I})$ and $a \in \mathcal{A}$ be arbitrary. Denote by $U_{a,-}^{(s,b)}$ a random vector which for each $t \in S$ returns the updated environment belief $U_{a,t}^{(s,b)}$. Then*

$$\mathbb{E}[H(U_{a,-}^{(s,b)})] \leq H(b).$$

We note the existence of POMDP solving methods that use the expected change of entropy as a heuristic to guide the analysis (Cassandra 1998).

Despite MEMDPs having a very specific structure, they can be used to model and solve interesting problems. We cover these in the rest of the paper.

4 Optimized Solvers

In this section we show how the special structure of MEMDPs can be leveraged to optimize the execution of several standard POMDP-solving algorithms.

Sparse Point-Based Value Iteration (SPBVI). Value iteration (VI) is a standard algorithm for solving MDPs (Bellman 1957), which was also extended for a use in POMDPs (Sondik 1971), where it iteratively builds a more and more precise approximation of the optimal value function (which can be then used to identify optimal behaviour). After n iterations the output of VI consist of set V_n of $|Q|$-dimensional α-*vectors*, and the optimal payoff achievable from any belief b can then be approximated as $\max_{\alpha \in V_n} b \cdot \alpha$, where $\cdot$ denotes the vector dot-product. The main drawback of VI is that the set V_n may grow very fast with the number of iterations. The Point-Based Value Iteration (PBVI) algorithm (Pineau et al. 2003) alleviates this scalability issue by approximating an exact value iteration solution on a *restricted* set of beliefs and retaining only some α-vectors in each step.

For MEMDPs, we consider a modified version of PBVI, the *sparse* PBVI (SPBVI). This algorithm utilizes the special structure of MEMDPs in several ways. First, it reduces memory consumption by working with factorized transition function. Next, we modify the algorithm to work with beliefs of the form $(s,b) \in S \times \mathcal{D}(\mathcal{I})$ instead of beliefs over the entire space $S \times \mathcal{I}$ (this is straightforward, as in the POMDP reformulation of a MEMDP M there is no reachable belief assigning a positive probability to two tuples $(s,i),(t,j)$ with $s \neq t$). This modification is similar to the one used for MOMDPs, although (Ong et al. 2010) modify the SARSOP algorithm instead of PBVI. We also integrate the linear-time belief updates mentioned in Section 3 into SPBVI.

Monte Carlo MEMDP Planning. Off-line methods based on value iteration may not scale to very large POMDPs. To overcome this issue, on-line methods for solving POMDPs were developed, which, instead of computing the whole (ϵ)-optimal policy, compute only its local approximations needed to select an (ϵ)-optimal action for the current belief (Ross et al. 2008; Geffner and Bonet 1998). One such well-known algorithm is the *partially observable Monte Carlo planning* (POMCP, (Silver and Veness 2010)). POMCP performs, in each step, when the history is h and current belief is b, a number of finite-horizon simulations starting from b in order to compute an approximation of the optimal value function. After all the simulations proceed, the best action according to the estimated values is played, a new observation is received, and the process continues as above. The crucial aspect of POMCP is that the results of past simulations are stored in a so-called *search tree* $\mathcal{T}$, whose nodes correspond to some of the histories that extend h. The information stored in $\mathcal{T}$ affects the way in which actions are selected in future simulations, in order to balance exploration of yet undiscovered paths with exploitation of information gained in the previous simulations. If a simulation produces a history not yet stored in $\mathcal{T}$, a new node is added with information on the outcome of this simulation. In particular, each node of $\mathcal{T}$ stores the approximation of a belief for the corresponding

history[2], so that beliefs do not have to be updated all the time. To make belief updates even more efficient, the beliefs are approximated using a Monte Carlo particle filter.

We consider several ways in which POMCP can be optimized for use on MEMDPs. First, in all our modifications of POMCP we exploit the factored transition function to allow for more efficient sampling of successor states during simulations. We also work with beliefs over the environment rather than over $S \times \mathcal{I}$. On top of these modifications, we consider *exact belief updates* and *past-aware POMCP*.

Belief updates in POMCP. POMCP performs fast and approximate belief updates using particle filters. We call this standard variant POMCP-pf. While the method is highly efficient, it can lead to accuracy issues due to particle deprivation. As exact belief updates can be done in linear time in MEMDPs, we consider a variant of POMCP with exact belief representation and updates according to Equation (3). We call this variant POMCP-ex. We aim to show that this modification of POMCP, which was hinted as useful for POMDPs with small state space in (Silver and Veness 2010), can be efficiently applied to MEMDPs with a large number of states.

Past-Aware POMCP (PAMCP). When POMCP finishes the round of simulations for history h, selects an action a, and gets observation o, it only keeps the sub-tree $\mathcal{T}'$ rooted in the node corresponding to history hao as the new search tree; the remaining branches are pruned away. However, in many application domains of MEMDPs, such as recommendation systems described below, we need to perform many executions of the planning algorithm in quick succession: e.g. for a recommender system, customers arrive continuously, and serving each customer amounts to a single execution of the online planning algorithm starting with empty history of actions and observations. Hence, it makes sense to pass all the information from the simulation to future executions. We denote by PAMCP ("past-aware") the variant of POMCP which does not prune the search tree during its execution, so that the whole tree can be passed to future executions. We distinguish PAMCP-pf and PAMCP-ex depending on whether PAMCP uses particle filters or an exact representation of beliefs.

5 Applications and Experiments

5.1 Applications

We apply MEMDPs in several domains, in particular for recommendation systems and parametric MDPs.

Recommender Systems. Recommender systems have multiple real-life applications such as e-commerce or media recommendation platforms. Their aim is to predict a user's preference in order to make adequate recommendations. Two main categories of such systems are usually found in the literature: content-based filtering approaches seek to map user profiles to items based on the item's characteristics, while collaborative filtering approaches exploit the knowledge of user's past choices to make recommendations. We focus on the latter setting: (Shani, Heckerman, and Brafman 2005) and (Brafman, Heckerman, and Shani 2003) formulated a sequential-decision making approach to recommendations systems as MDPs, where states encode the user's past behaviour, and actions account for the recommendations[3] made by the system. We extend this formulation to MEMDPs in order to take the variability between users' profiles into account. That is, in the MDP formulation the transition probabilities (e.g. a probability that the user clicks on a link given her previous history of interactions and the recommendation performed) are inferred from aggregated historical data of all user's interaction with the system. In contrast, in our MEMDP formulation we cluster customers according to various attributes (age, sex, etc.), and for each cluster we infer a probabilistic transition function characterizing the behaviour of customers within the cluster. In our experiments, we first consider a synthetic dataset designed to exhibit high difference between environments. Secondly, we experiment with real-life data, by exploiting actual past sequences of users choice. We first cluster the sequences in order to build the environments, and we then infer the system's transition probabilities from these sequences using a classical k-gram model (where k is the history length) with maximum likelihood estimation (Indurkhya and Damerau 2010, Ch. 15).

Parametric MDPs. Parametric MDPs consist of MDPs where certain probabilities are specified in terms of parameter values; e.g., given parameters x and y, outgoing transition probabilities in state s_0 can be x and $1 - x$; outgoing transition probabilities in state s_1 can be $2x$ and $1 - 2x$; and outgoing transition probabilities in state s_2 can be $x + y$ and $1 - x - y$; and so on. For parametric MDPs where the values of the parameters are unknown and each parameter has a discrete range, every possible combination of values for the parameters represent an environment. Thus they are naturally modeled as MEMDPs.

Multi-Environment Robot Navigation. We also evaluate our algorithms on an MEMDP variant of the standard *Hallway* benchmark (Littman, Cassandra, and Kaelbling 1995). We consider a model of a robot navigating towards a goal state with potential errors in movement (i.e. each action has a small probability to fail and not be performed, dependent on the environment) in a labyrinth whose exact structure is hidden, but is known to be drawn out of a pool of mazes.

5.2 Experiments

Recommendation Experiments. We run the MEMDP-optimized algorithms on two instances of the recommender model: First, an instance obtained from an artificially generated dataset, where each customer shows a strong preference towards buying a certain item. Second, we consider an instance obtained from a real-life Microsoft Foodmart retail dataset[4]. While similar datasets exist in the data mining and pattern analysis community, they are often anonymized, which makes it harder to cluster users, or they contain very short sequences, unfit for reliable probability estimates.

To evaluate the algorithms' performance, we use several

[2] In each step, the current belief is uniquely determined by the current history and the initial belief.

[3] These are represented as an ordered lists of recommended items, similar to online search engines.

[4] Microsoft Foodmart retail database, http://www.philippe-fournier-viger.com/spmf/index.php?link=datasets.php

(synth)	MDP	SPBVI	POMCP-pf	POMCP-ex
accuracy	0.10 ± 0.03	0.77 ± 0.09	0.67 ± 0.19	**0.75** ± 0.08
env. pred.	-	**0.96** ± 0.04	0.83 ± 0.23	0.94 ± 0.05
time	0.26s	~ 12h	19.9s	20.5s

(a) Results for $N = 10$ and $k = 2$ ($|\mathcal{I}| = 10$, $|S| \sim 100$)

(synth)	MDP	SPBVI	POMCP-pf	POMCP-ex
accuracy	0.12 ± 0.03	-	0.64 ± 0.27	**0.77** ± 0.07
env. pred.	-	-	0.79 ± 0.33	**0.96** ± 0.04
time	5h30mn	Out of mem.	9mn36s	14s

(b) Results for $N = 8$ and $k = 5$ ($|\mathcal{I}| = 8$, $|S| \sim 38000$)

(synth)	MDP	SPBVI	POMCP-pf	POMCP-ex
accuracy	0.02 ± 0.01	-	0.04 ± 0.09	**0.40** ± 0.15
precision	0.03 ± 0.01	-	0.08 ± 0.09	**0.42** ± 0.15
env. pred.	-	-	0.06 ± 0.10	**0.50** ± 0.15
time	6mn8s	Out of mem.	~1h47mn	7mn10s

(c) Results for $N = 60$ and $k = 2$ ($|\mathcal{I}| = 60$, $|S| \sim 4000$)

(F-mart)	MDP	SPBVI	POMCP-pf	POMCP-ex
accuracy	0.15 ± 0.06	**0.16** ± 0.06	0.11 ± 0.05	0.13 ± 0.06
precision	0.19 ± 0.05	-	0.26 ± 0.05	**0.29** ± 0.06
env. pred.	-	**0.73** ± 0.26	0.23 ± 0.21	0.49 ± 0.20
time	12s	4mn34s	1mn46s	1mn20s

(d) Results for $N = 22$ and $k = 2$ ($|\mathcal{I}| = 5$, $|S| = 507$)

(F-mart)	MDP	SPBVI	POMCP-pf	POMCP-ex
accuracy	0.61 ± 0.14	**0.62** ± 0.14	0.62 ± 0.14	0.62 ± 0.14
precision	0.74 ± 0.09	-	**0.78** ± 0.08	0.78 ± 0.08
env. pred.	-	**0.60** ± 0.31	0.54 ± 0.35	0.53 ± 0.36
time	11mn57s	12mn38s	46s	23s

(e) Results for $N = 3$ and $k = 8$ ($|\mathcal{I}| = 5$, $|S| = 9841$)

(synth)	POMCP-pf	POMCP-ex	PAMCP-pf	PAMCP-ex
accuracy	0.64 ± 0.27	**0.77** ± 0.07	0.68 ± 0.24	0.75 ± 0.08
env. pred.	0.79 ± 0.33	**0.96** ± 0.04	0.85 ± 0.30	0.94 ± 0.06
time	9mn36s	15s	14s	36s

(f) Comparison of the POMCP variants in the setting (b).

Table 1: Recommendation system results for the synthetic and real-life (Foodmart) dataset. N = number of items, k = maximal length of customer's past behaviour we track. For each setting we consider an MDP formulation of the problem (MDP, solved via Value Iteration), and an MEMDP formulation solved by various algorithms: sparse PBVI (SPBVI), and several modifications of POMCP, which differ in whether they use exact (-ex) or Monte Carlo particle filter (-pf) belief updates, and in whether they are past-aware (PAMCP-) or not (POMCP-). We do not report the precision for SPBVI as we could only access the best action recommended by the policy and not the scores for each action in the underlying PBVI implementation.

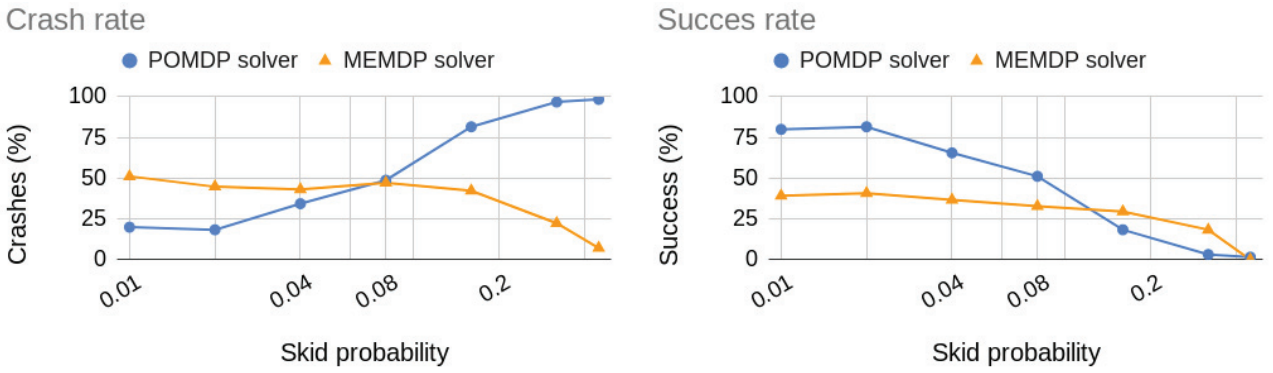

Figure 2: Plots of success and crash rates in the parameterized Hallway benchmark. The x axis is log-scale. Another case is one where the robot neither reaches a target state (success) nor a sink "trap" state (crash), but instead keeps wandering in loops.

information retrieval evaluation measures (Rijsbergen 1979, Chapter 7), such as the *prediction accuracy*, which measures how often the user selects the first item on the recommendation list. When it is possible to recommend many items at once, the accuracy carries limited information on the system's performance as it only accounts for the best prediction. Hence, when possible, we also report *precision* of the recommender, which assesses how well the user's actual preference is ranked in the recommendations list. More formally, at each planning step the POMCP algorithm returns an approximation of optimal value for each action (i.e., each item) and we use these values to rank the recommendations (recommendation with the best value is ranked first, etc). Then we compute the reciprocal rank of the item the user actually selects in this list. We average these reciprocal ranks over all time steps, hence, the precision can be interpreted as a mean reciprocal rank (in each step we assume only one relevant item - the one selected by the user). Since each user's interaction can be in principle of any length, we aggregate the scores of the metrics using a discounted payoff with discount factor $\gamma = 0.95$. We also measure how well each algorithm managed to sort customers it interacts with into correct profiles (*environment accuracy*, env. pred.: since all the POMDP solvers keep (in some form) track of the current belief about the environment, we compute the environment identification success rate according to the average probability it assigns to the correct environment.

The Monte Carlo algorithms were evaluated with 1000 simulation iterations and horizon 2. Using larger horizon only leads to slight improvement in performance. VI's and SPBVI's parameters are chosen so as to give the best results in reasonable solving time. As baselines, we compare with the performance of POMCP-pf (which corresponds to standard POMCP with additional optimization for computational efficiency, hence is comparable in terms of accuracy) to study the benefits of our optimizations, and with that of the MDP

formulation (solved via Value Iteration), to highlight the importance of explicitly considering different environments for this type of applications.

The results of the experiments, including the total running time of each algorithm, are presented in Table 1. We note that the POMDP obtained from MEMDP has $|\mathcal{I}| \cdot |S|$ states, i.e. we solve POMDPs (of a special shape) whose size ranges from 1000 to more than 200,000 states. We observe that when there is a significant difference between the environments, the MEMDP-based recommender systems always outperform the MDP-based one; This is especially true in the synthetic dataset experiments by design (tables (a), (b), (c) and (f)). Secondly, POMCP-ex outperforms POMCP-pf both in recommendation and environment prediction accuracy, especially when the model contains numerous environments, which shows the benefits of leveraging the MEMDP structure. The performance of MEMDP algorithms against the MDP baseline is less competitive in the Foodmart scenario, where there are high correlations between user's preferences. Still, MEMDP models typically yield higher precisions, which is a meaningful measure in a real-life scenario as users are typically suggested several recommendations, not just one. The MDP approach uses value iteration over the whole state space. This method is inefficient for large state spaces, since the complexity of a single iteration is worst-case quadratic.

In Table 1 (f), we also see that POMCP-ex offers, among all 4 POMCP variants, the best trade-off between running time and accuracy (PAMCP-pf is slightly faster, but has worse accuracy). Overall, the experiments suggest that in terms of prediction accuracy, MEMDPs are most valuable where there are many different types of customers with varying preferences. However we still observe some improvements even in the smaller settings: In particular, POMCP-ex, i.e. the variant with exact belief updates, almost always outperforms POMCP-pf. This shows the importance of having exact belief updates, which the MEMDP formulations allow to perform efficiently even for large state spaces. Also, SPBVI is much more computationally efficient than PBVI (which typically runs out of memory or does not finish in a reasonable time on most of these settings), while preserving the accuracy.

Parametrized MDPs We consider a parameterized version of the classical maze-solving Hallway benchmark. The goal is to reach a target cell without being destroyed in a trap. The layout of the maze is known in advance (we use a maze of dimension 5×10), but the probability p of an error in movement ("skidding" off the desired direction of movement) is an unknown parameter from the set $\{0, 0.01, 0.02, 0.04, 0.08, 0.16, 0.32, 0.45\}$. We compared two approaches to the situation: modeling it as a POMDP (akin to (Arming et al. 2018)) solved with the POMCP algorithm, and modeling as a MEMDP solved with PAMCP-ex. We evaluated the "success rate" and "crash" metrics (the ratio of evaluation runs in which we reached a target cell, and in which we fell into a trap, respectively). The results are pictured in Figure 2. We observed that the POMDP solver always takes the "risky" path close to the trap, no matter the environment, due to its inability to detect a safer but longer alternative path. This daredevil strategy performs well for safe environments but fails for the riskier ones. We see that the POMDP version only behaves reasonably for low values of p, effectively failing to take into account the risk-taking aspect. In particular, the crash rate rises very dramatically for the POMDP approach. On the other hand, in the MEMDP formulation we achieve a much more balanced performance throughout the parameter range. For small values of p, the success rate of PAMCP-ex is smaller, since the algorithm hedges for the possibility that the value of p is high. On the other hand, the crash rate is much more even and with increasing p: the MEMDP approach is able to detect danger and act extremely conservatively, significantly decreasing the crash rate. Moreover, the MEMDP was much faster: a single evaluation episode took, on average, 1479.35s for the POMDP approach and only 30.15s for the MEMDP approach.

	length	env. pred.	success	time
(A)	28.0 ± 23.4	0.97 ± 0.32	0.67 ± 0.32	16mn12s
(B)	50.8 ± 10.3	0.38 ± 0.35	0.37 ± 0.34	1h21mn
(C)	14.3 ± 8.3	0.50 ± 0.32	0.92 ± 0.21	2mn22s

Table 2: POMCP-pf for the maze-solving problem.

MEMDP Hallway. Finally we consider a MEMDP version of the Hallway benchmark in which the skid probability is fixed but there are multiple possible mazes in which the agent can be placed. We evaluate the algorithms in term of this expected payoff (`length` is the average length of a found path to the goal), probability of successfully reaching the target (`success` is again the ratio of simulations in which the robot manages to reach one of the goal states), and in terms of environment prediction accuracy (denoted as `env. pred.`). We present results of POMCP-pf on three instances: instance (A), with 25×25 rather densely populated by randomly placed walls ($|\mathcal{I}| = 60$, $|S| \sim 2500$), and instance (B) 50×50 grid with *no walls* ($|\mathcal{I}| = 30$, $|S| \sim 10000$) and (C) 5×5 grid with no walls ($|\mathcal{I}| = 60$, $|S| \sim 100$). Absence of walls in (B) and (C) makes finding the correct environment harder, as without sensing the walls in this large settings the only way to gain information is to enter a position where a goal is in one of the mazes. The results are shown in Table 2.

We see that despite environment classification being indeed harder in (B) and (C), in all scenarios the accuracy is much larger than that of simple guessing, and the solver is able to finish in reasonable times. Both PBVI and SPBVI timed out on the large instances ((A) and (B)), but interestingly, for the smallest instance, (C), where $|\mathcal{I}| \cdot |S| \sim 6000$, our SPBVI terminates in about 75 minutes while PBVI did not terminate even after 11 hours, which shows the computational advantage of exploiting the sparsity structure in MEMDPs.

6 Conclusion and Future Work

In this work we consider MEMDPs with discounted-sum payoff, their practical advantages, and applications. There are several possible directions of future work, including exploration of other heuristics (such as those based on belief entropy) and the applications of MEMDPs in other domains.

Acknowledgements

Krishnendu Chatterjee is supported by the Austrian Science Fund (FWF) NFN Grant No. S11407-N23 (RiSE/SHiNE), and COST Action GAMENET. Petr Novotný is supported by the Czech Science Foundation grant No. GJ19-15134Y.

References

Arming, S.; Bartocci, E.; Chatterjee, K.; Katoen, J.; and Sokolova, A. 2018. Parameter-independent strategies for pmdps via pomdps. In *QEST*, 53–70.

Bellman, R. 1957. A Markovian Decision Process. *Indiana Univ. Math. J.* 6:679–684.

Brafman, R. I.; Heckerman, D.; and Shani, G. 2003. Recommendation as a Stochastic Sequential Decision Problem. In *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS 2003), June 9-13, 2003, Trento, Italy*, 164–173.

Cassandra, A. 1998. *Exact and approximate algorithms for partially observable Markov decision processes*. Brown University.

Ceska, M.; Jansen, N.; Junges, S.; and Katoen, J. 2019. Shepherding hordes of markov chains. In *TACAS (II)*, 172–190.

Delgado, K. V.; Sanner, S.; and de Barros, L. N. 2011. Efficient solutions to factored mdps with imprecise transition probabilities. *Artificial Intelligence* 175(9):1498 – 1527.

Doshi-Velez, F., and Konidaris, G. 2013. Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. *arXiv preprint arXiv:1308.3513*.

Fern, A., and Tadepalli, P. 2010. A computational decision theory for interactive assistants. In *Advances in Neural Information Processing Systems*, 577–585.

Geffner, H., and Bonet, B. 1998. Solving Large POMDPs using Real Time Dynamic Programming. In *In Proc. AAAI Fall Symp. on POMDPs*.

Howard, H. 1960. *Dynamic Programming and Markov Processes*. MIT Press.

Indurkhya, N., and Damerau, F. J. 2010. *Handbook of Natural Language Processing*. Chapman & Hall/CRC, 2nd edition.

Itoh, H., and Nakamura, K. 2007. Partially observable markov decision processes with imprecise parameters. *Artificial Intelligence* 171(8):453 – 490.

Kurniawati, H.; Hsu, D.; and Lee, W. 2008. SARSOP: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable Belief Spaces. In *Robotics: Science and Systems*, 65–72.

Littman, M. L.; Cassandra, A. R.; and Kaelbling, L. P. 1995. Learning Policies for Partially Observable Environments: Scaling Up. In *ICML*, 362–370.

Littman, M. L. 1996. *Algorithms for Sequential Decision Making*. Ph.D. Dissertation, Brown University.

Ong, S. C. W.; Png, S. W.; Hsu, D.; and Lee, W. S. 2010. Planning under Uncertainty for Robotic Tasks with Mixed Observability. *I. J. Robotics Res.* 29(8):1053–1068.

Papadimitriou, C. H., and Tsitsiklis, J. N. 1987. The complexity of Markov Decision Processes. *Mathematics of Operations Research* 12:441–450.

Pineau, J.; Gordon, G.; Thrun, S.; et al. 2003. Point-based value iteration: An anytime algorithm for POMDPs. In *IJCAI*, volume 3, 1025–1032.

Poupart, P.; Vlassis, N.; Hoey, J.; and Regan, K. 2006. An analytic solution to discrete bayesian reinforcement learning. In *Proceedings of the 23rd international conference on Machine learning*, 697–704. ACM.

Puterman, M. L. 1994. *Markov Decision Processes*. John Wiley and Sons.

Raskin, J., and Sankur, O. 2014. Multiple-environment Markov decision processes. *arXiv preprint arXiv:1405.4733*.

Rijsbergen, C. J. V. 1979. *Information Retrieval*. Newton, MA, USA: Butterworth-Heinemann, 2nd edition.

Ross, S.; Pineau, J.; Paquet, S.; and Chaib-draa, B. 2008. Online Planning Algorithms for POMDPs. *J. Artif. Intell. Res. (JAIR)* 32:663–704.

Sen, K.; Viswanathan, M.; and Agha, G. 2006. Model-checking markov chains in the presence of uncertainties. In Hermanns, H., and Palsberg, J., eds., *Tools and Algorithms for the Construction and Analysis of Systems*, 394–410. Berlin, Heidelberg: Springer Berlin Heidelberg.

Shani, G.; Brafman, R. I.; and Shimony, S. E. 2007. Forward search value iteration for pomdps. In Veloso, M. M., ed., *IJCAI*, 2619–2624.

Shani, G.; Heckerman, D.; and Brafman, R. I. 2005. An MDP-Based Recommender System. *Journal of Machine Learning Research* 6:1265–1295.

Shannon, C. E. 2001. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.* 5(1):3–55.

Silver, D., and Veness, J. 2010. Monte-Carlo Planning in Large POMDPs. In Lafferty, J. D.; Williams, C. K. I.; Shawe-Taylor, J.; Zemel, R. S.; and Culotta, A., eds., *Advances in Neural Information Processing Systems 23*. Curran Associates, Inc. 2164–2172.

Smith, T., and Simmons, R. G. 2012. Point-based POMDP algorithms: Improved analysis and implementation. *CoRR* abs/1207.1412.

Sondik, E. J. 1971. *The Optimal Control of Partially Observable Markov Processes.* Stanford University.

Spaan, M. 2004. A point-based POMDP algorithm for robot planning. In *ICRA*, volume 3, 2399–2404. IEEE.

Wang, Y.; Won, K. S.; Hsu, D.; and Lee, W. S. 2012. Monte carlo bayesian reinforcement learning. *arXiv preprint arXiv:1206.6449*.

White III, C. C., and Eldeib, H. K. 1994. Markov decision processes with imprecise transition probabilities. *Operations Research* 42(4):739–749.