

Deterministic-Probabilistic Models For Partially Observable Reinforcement Learning Problems

M. M. Hassan Mahmud

John W. Lloyd

School of Computer Science

The Australian National University

Canberra, ACT 0200, Australia

MMAHMUD42@GMAIL.COM

JWL@CECS.ANU.EDU.AU

Editor: tba

Abstract

In this paper we consider learning the environment model in reinforcement learning tasks where the environment cannot be fully observed. The most popular frameworks for environment modeling are POMDPs and PSRs but they are considered difficult to learn. We propose to bypass this hard problem by assuming that (a) the sufficient statistic of any history can be represented as one of finitely many states and (b) this state is given by a deterministic map from histories to the finite state space. This finite set of states can be interpreted as the state space of an MDP which can then be used to plan. Now the learning problem is to estimate this deterministic history-state map. One of the earliest approaches in this direction is McCallum's USM algorithm. Our work can roughly be understood as extending this general idea by replacing prediction suffix trees, used in USM, with deterministic-probabilistic finite automata from learning theory. In this paper we describe our model, derive a pseudo-Bayesian inference criterion, and show its consistency. We also describe a heuristic algorithm that uses the criterion to learn the models, along with experiments showing its efficacy.

Keywords: partially observable reinforcement learning, Bayesian reinforcement learning, deterministic probabilistic finite automata, consistency of Bayesian estimators, Bayesian sequence prediction

1. Introduction

In the general reinforcement learning (RL) problem (Bertsekas and Shreve, 1996; Sutton and Barto, 1998) an agent operates in some environment where at each step the agent takes an action and in return receives an observation and a reward. The goal of the agent is to maximize its discounted time averaged future reward. Current RL algorithms can be classified in terms of the type of observation received by the agent at each step. The most common approach is to assume that the environment is a finite Markov decision process where the states are observable (Bellman, 1957; Bertsekas and Shreve, 1996). That is, at each point in time, the observation received by the agent is one of finitely many *states*, each of which is a sufficient statistic of the past and hence can be used to plan for the future. Unfortunately, in many applications, the observation received at each step is (possibly much) less informative than a state. In this case, during planning, the agent also has to determine from the partial information what the underlying state is (if any).

The most prominent approaches to handling this partially observable RL problem include modeling the environment via partially observable Markov decision processes (POMDPs) (Sondik,

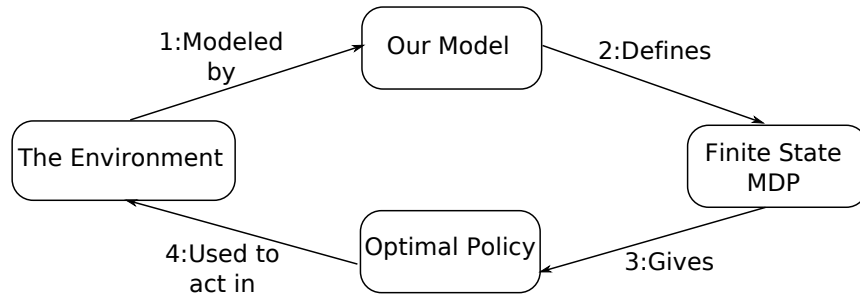


Figure 1: Schematic description of our Approach.

1971; Kaelbling et al., 1998) and predictive state representations (PSRs) (Littman et al., 2002). The parameters describing the relationship between the observations and underlying hidden variable describing the environment dynamics are referred to as the model. Given the true model for an environment, the planning problem is *undecidable* in general (Madani et al., 2003); but in practice this is not a severe issue as effective heuristic planning algorithms have been developed (see the survey Ross et al. (2008)). However, in many problems of interest, the model is not available *a priori* and has to be learned from observations. Unfortunately, despite some promising new results (for instance, Doshi-Velez (2009)), POMDP and PSR models are generally considered quite difficult to learn from data. In this paper¹ we address this issue by taking a step back and looking at a class of models which are simpler (hence less expressive) but easier to learn.

1.1 Our Approach in Brief

The key assumption in our simplified approach to environment modeling is that each history *deterministically* maps to one of finitely many states and this state is a sufficient statistic of the history (McCallum, 1995b; Shalizi and Klinkner, 2004; Hutter, 2009). Given this history-state map, at each point in time we can deterministically compute the state and hence we end up in the finite state MDP planning problem, for which efficient planning algorithms exist. So the learning problem now is to learn this history-state map. Indeed, the well known USM algorithm (McCallum, 1995b) used prediction suffix trees (PST) (Ron et al., 1994) for these maps (each history is mapped to exactly one leaf/state) and was quite successful in benchmark POMDP domains. However, PSTs lack long term memory and have difficulty with noisy environments and so USM was not followed up on for the most part (exceptions being Shani et al. (2004, 2005)). In our work we consider a Bayesian setup and replace PSTs with finite state machines and endow the agent with long term memory. The resulting model is a proper subclass of POMDPs, but (hopefully) maintains the computational simplicity and efficiency that comes with considering deterministic history state maps. Our approach is illustrated in Figure 1.

Interestingly, belief states of POMDPs are also deterministic functions of the history. But this state space is infinite and so POMDP model learning algorithms try to estimate the hidden states (see for instance Doshi-Velez (2009)). As a result, these methods are quite different from algorithms using deterministic history-state maps – we discuss other related methods in Section 1.3.

1. The experimental results presented in this paper previously appeared in Mahmud (2010) and the new material in this paper are a further elaboration of each section and the theoretical development of our method.

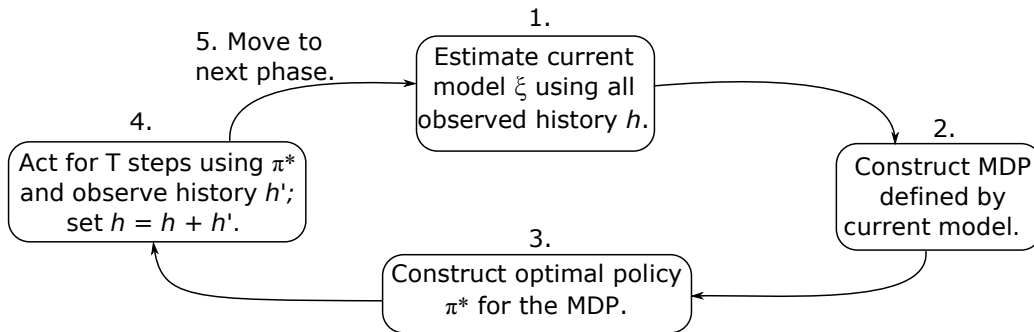


Figure 2: A phase in our learning algorithm.

We now briefly outline our algorithm for learning the environment model. The algorithm runs in phases and in each phase it performs four different tasks (see Figure 2). First, given the history observed so far, it computes an estimate of the correct model. This estimation is done using a MCMC style heuristic search algorithm and is described in Section 4. Given the estimated model it constructs the MDP defined by the model – this step is trivial and described in Section 2.5. It then computes the optimal policy for the constructed MDP using standard policy learning algorithms (Sutton and Barto, 1998). After that, it uses the policy to act in the environment which then generates additional history; this history is appended to the existing history and used in the next phase.

One non-standard aspect of our model is that following Hutter (2009), we do not model the observation distribution explicitly. This is because, for planning, only the reward distribution is relevant and the observation distribution is useful in so far as it helps us predict the next state. In other words, we leave out the irrelevant details of the observation – consider for instance the fact that when crossing a road we are interested only in modeling how the cars move, but not the swaying of the trees lining the road – see Section 2.4 for details. Hence, our model is not a generative model of the environment and so we need to use a non-standard likelihood function to compare goodness of models in the first step in Figure 2. The bulk of the theoretical content of this paper is dedicated toward showing that this likelihood function is in fact *consistent* – that is in the limit of infinite data, the ratio of the likelihood of the correct model and an incorrect model will go to infinity.

1.2 The Exacerbated Exploration-Exploitation Problem

In the next subsection we will discuss related work, most of which use much more expressive and powerful models than us. Interestingly, the use of simpler models can be further motivated by considering what, in our opinion, is *the* problem in model learning in partially observable RL problems. This problem is the exacerbated exploration-exploitation problem (E^3 problem) where the agent has to figure out which regions of the state space to explore before knowing what the relevant states are (!). It is clear that this problem is much more difficult than the traditional exploration-exploitation problem in RL (Sutton and Barto, 1998) where the agent has to choose actions so as to balance exploration of the state space and accumulation of reward (by exploiting its current knowledge). The E^3 problem was first pointed out, to the best of our knowledge, in Chrisman (1992), and has been recognized by later researchers like McCallum (1995a); but it does not seem to have gained the notoriety it deserves amongst researchers working on model learning in partially-observable-RL.

So consideration of simpler models can further be justified because with them we should be able to learn an estimate of the space explored so far more easily (both in terms of computational and sample complexity) and hence explore unseen regions more efficiently. Of course, if the underlying environment can only be modeled by a PSR or a POMDP, then our approach fails (and learning in such environments will be very hard whatever the model used). Hence, the implicit assumption in our work is that deterministic history-state maps are sufficiently expressive for many problems of interest.

1.3 Related Work

As mentioned above, the USM algorithm, using PSTs as the deterministic history-state maps, can be considered to be our earliest precursor. For the most part USM was not followed up on due to the problems mentioned earlier. A couple of exceptions are the papers by Shani et al. (2004, 2005). In the first paper, the authors propose the NUSM algorithm, an extension of USM that is more tolerant of highly noisy observations. In the second paper the authors convert the tree learned by USM to a POMDP and find that in experiments this POMDP model performs much better than pure USM.

Recently, Hutter (2009) proposed a generalization of USM called Φ -MDP to consider arbitrary history-state maps (Φ -functions), along with a MDL style cost criterion to select between different models. The δ functions of our DMMs (see Section 2.4) can be considered to be Φ -functions. However, Φ -MDP does not require that the mapping result in Markovian states (which USM does not either) whereas our states are Markovian by construction. The MDL-style cost criterion for Φ -MDP is also non-standard like our likelihood function and recently the consistency of this function was proven in Sunehag and Hutter (2010). As mentioned earlier, our decision to not model the observation distribution was inspired by the Φ -MDP approach – however our likelihood function is different from that in Hutter (2009). The latter explicitly uses the state transition distribution in the cost function, while we only use the reward distribution. Likewise, our consistency results in Section 3 holds under the identifiability Assumption 12 (see Section 3.2.3) while the Φ -MDP consistency result holds for Φ functions that are bounded-memory finite state machines (this assumption is unrelated to ours).

While not following USM, tree-based representations has also been considered by other researchers for model learning for partially observable environments. Even-Dar et al. (2005) derive a asymptotically convergent algorithm for t -Markov policies (i.e. policies that may be represented by depth t PSTs). The authors also note that the algorithm is computationally very expensive (exponential in relevant parameters) and do not give any experimental results. Another paper that considers tree-based representation of the environment is Farias et al. (2010). In this work, the authors use a context tree (Rissanen, 1983) based representation for the environment and derive an algorithm that, asymptotically, learns the right model and hence the optimal value function (provided that the environment can be represented as a context tree).

Among the more usual approaches, ours is most closely related to the POMDPs. The latter are essentially hidden Markov models with actions, and hence, the most obvious approach to learning POMDP models is to adapt the Baum-Welch algorithm for learning HMMs. This basic idea has been improved using Bayesian approaches (for example Ross et al. (2007) or references in Doshi-Velez (2009)) – these algorithms need the number of states to be specified a priori. Unfortunately, none of these methods is able to go beyond small POMDP benchmark domains ($\leq 5 \times 5$ maze

problems) and USM seems to outperform them all in terms of number of episodes required and computational cost. The main challenge seems to be overcoming the E^3 problem described above.

Finally, finite state controllers for POMDPs (Kaelbling et al., 1998) seem closely related to our work but these are not quite model learning algorithms. They are (powerful) planning algorithms that assume a hidden but known environment model (at the very least, implicitly, an estimate of the number of hidden states).

Aside from POMDPs, the approach that has received the most attention from researchers is representing the environment as a PSR (Littman et al., 2002; Singh et al., 2004; Wolfe et al., 2005). While algorithms have been developed for PSRs, they are considered difficult to learn and do not seem to scale beyond small problems in the benchmark POMDP problem sets (same as POMDP model learning methods) and, depending on the algorithm and setup, require many more examples (McCracken and Bowling, 2005). In particular, determining the *core tests* (Littman et al., 2002) seem to be particularly challenging. While PSRs have been shown to work in higher dimensions under specific domain constraints (for example, Rosencrantz et al. (2004) when the domain can be reset to a start state, Boots et al. (2010) when observations are sufficient to distinguish state-clusters, if not states) the general problem remains unsolved. In particular there does not seem to be any explicit attempt to solve the E^3 problem in PSR approaches.

1.4 Paper Organization

In the following we proceed as follows. In Section 2 we introduce our model, show how it models the environment and how it defines an MDP that can be used to plan in the environment. In Section 3 we derive a novel (pseudo) likelihood function to choose between models and prove its consistency. In Section 4 we describe our learning algorithm for estimating the problem from experience and planning with it. In Section 5 we describe experiments in some suitable problems showing the efficacy of the learning algorithm. Finally, we conclude with a discussion of future development of the method and related work in Section 6. The proofs of the theorems in Sections 2 and 3 appear in the Appendix.

2. Modeling RL Environments

Our approach to learning the hidden environment model is as follows: we represent a general RL environment by our model, the deterministic Markov models (DMMs), and then use the MDP derived from the DMM to plan for the problem. In the following we introduce notation (Section 2.1), define a general RL environment (Section 2.2), define our model, the DMMs (Section 2.3) and show how they can model the environment and define a MDP that can be used for planning (Section 2.5).

2.1 Preliminaries

For a distribution $P(x)$ over some space, $E_{P(x)}[f(x)]$ denotes the expectation of the function f with respect to P . The symbol \triangleq is used for definitions. \mathcal{A} is a finite set of actions, \mathcal{O} a finite set of observations and $\mathcal{R} \subset \mathbb{R}$ a finite set of rewards. We set $\mathcal{H} \triangleq (\mathcal{A} \times \mathcal{O})^*$ to be the set of histories and $\gamma \in [0, 1)$ to be a fixed discount rate. We will need some notation for sequences (over arbitrary finite alphabets). $x_{0:n}$ will denote a string of length $n + 1$ and $x_{<n} \equiv x_{0:n-1}$ will denote a string of length n . $x_{i:j}$ will denote elements i to j inclusive, while x_i will denote the i^{th} element of the sequence. The indices will often be time indices (but not always). If there are two strings $x_{0:n}$ and

$y_{0:n}$, we will use $xy_{0:n}$ to denote the interleaved sequence $x_0y_0x_1y_1 \dots x_ny_n$. We will also use xy_i to denote the i^{th} element of x_iy_i and $xy_{i:j}$ to denote $x_iy_i \dots x_jy_j$. Finally, $x_{0:n}y_{0:m}$ will denote the string $x_{0:n}$ followed by $y_{0:m}$ and λ will denote the empty string.

2.2 General Reinforcement Learning Environment

We define a general RL environment as $\mathcal{G} \triangleq (\mathcal{A}, \mathcal{R}, \mathcal{O}, RO, \gamma)$, where all the quantities except RO were defined above. RO defines the dynamics of the environment: at step t when action a is applied, the next reward-observation pair is selected according to probability $RO(ro|h, a)$ where $h = ao_{0:t-1}$ is the history before step t . We will write the marginals of RO over \mathcal{R} and \mathcal{O} by R (the *reward distribution*) and O (the *observation distribution*) respectively.

The actions at each step are chosen using a policy $\pi : \mathcal{H} \rightarrow \mathcal{A}$; the *value function* V^π of π is defined by:

$$V^\pi(h) = E_{R(r|h,a)}(r) + \gamma E_{O(o|h,a)}[V^\pi(hao)] \quad (1)$$

where $a = \pi(h)$. The goal in RL problems is to learn an optimal policy π^* which satisfies $V^{\pi^*}(h) \geq V^\pi(h)$ for each policy π and history h . In particular the value function of this policy is given by:

$$V^{\pi^*}(h) \triangleq V^*(h) \triangleq \max_a \{E_{R(r|h,a)}(r) + \gamma E_{O(o|h,a)}[V^*(hao)]\} \quad (2)$$

The existence of these functions follows via standard methods (Bertsekas and Shreve, 1996).

2.3 Deterministic Markov Models

Each of our models is defined by a tuple $(\vec{\theta}, \vec{\phi}, \xi)$. We define $\xi \triangleq (q_o, \mathcal{S}, \Sigma, \delta)$ and call it a deterministic Markov model (DMM); it is simply a deterministic finite state automaton (Hopcroft et al., 2006) with q_o as the start state, \mathcal{S} the set of states, $\Sigma = \mathcal{A} \times \mathcal{O}$ the edge-label alphabet and $\delta : \mathcal{S} \times \Sigma \rightarrow \mathcal{S}$ the transition function. If ξ is at state s and $\delta(s, z) = s'$ then ξ transitions to state s' on input $z \in \Sigma$. We overload δ to denote its transitive closure so that $\delta(q, h) = s$ means that ξ transitions to s from q when $h \in \mathcal{H}$ is given as input. In fact, the transitive closure of the δ function is the deterministic history-state map that we mentioned in Section 1. That is, $\kappa(h) \triangleq \delta(q_o, h)$ is the function that maps each history to its sufficient statistic. In the sequel we will use the term ‘model’ to refer to the triples of the form $(\vec{\theta}, \vec{\phi}, \xi)$ and not DMMs.

The first two components of a model are defined thus: $\vec{\theta} \triangleq \{\vec{\theta}_{s,a}\}$ and $\vec{\phi} \triangleq \{\vec{\phi}_{s,a}\}$, indexed by \mathcal{S} and \mathcal{A} . $\vec{\theta}_{s,a}(r)$ is the probability of the next reward being r given that ξ is at state s and action a was chosen. $\vec{\phi}_{s,a}(s')$ gives the probability that ξ transitions to s' given ξ is at state s and action a was chosen. Since both \mathcal{R} and \mathcal{S} are finite, the distributions $\vec{\theta}_{s,a}$ and $\vec{\phi}_{s,a}$ are multinomials satisfying

$$\sum_r \vec{\theta}_{s,a}(r) = 1, \text{ and } \sum_{s'} \vec{\phi}_{s,a}(s') = 1.$$

The dynamics of our model is deterministic or probabilistic depending on the conditioning values. If action a is taken at state s of ξ , then it transitions to state s' with probability $\vec{\phi}_{s,a}(s')$. However, if the next observation o is also given, ξ transitions deterministically to state s' , where $\delta(s, ao) = s'$. This implies that, given a history $ao_{0:n}$, there is exactly one state sequence $s_{0:n}$ that ξ could have transitioned through, where $\delta(q_o, ao_{0:k}) = s_k$. So, DMMs are (roughly speaking) deterministic probabilistic finite automata from learning theory (Vidal et al., 2005) but extended with

actions to a sequential setting. Another way to interpret the DMM state space is as a discretization of POMDP belief state space (Kaelbling et al., 1998)². This rich avenue of research will be pursued in the future.

2.4 Modeling Environments

For the rest of the paper we fix $\mathcal{G} \triangleq (\mathcal{A}, \mathcal{R}, \mathcal{O}, RO, \gamma)$ as the environment. Now our basic assumption is that $\exists \xi^g$ (g stands for ‘generating’) such that $\forall s \in \mathcal{S}^g, a \in \mathcal{A}, h$ such that $\kappa^g(h) = s$ and $r \in \mathcal{R}, \exists \vec{\theta}_{s,a}^g$:

$$\vec{\theta}_{s,a}^g(r) = R(r|h, a) \quad (3)$$

We do not model O because all we actually care about is R (Hutter, 2009). This way, we do not sacrifice reward prediction accuracy or learn an unnecessarily complex model by trying to model task-irrelevant details of O . For example, in a domain with a million states where O is different at each state but R is the same everywhere, it will be a bad idea to try to learn O . In more realistic terms, consider the fact that when crossing a street we care about modeling car movements, but not swaying of the trees lining the road. Note that we do not ignore the observations as the δ s are functions over them. In a sense we only use observations in so far as they help predict rewards. We do need to impose an additional consistency constraint for not modeling O : we further assume that for all $s, s' \in \mathcal{S}^g$ and h such that $\kappa^g(h) = s$,

$$\vec{\phi}_{s,a}^g(s') = \sum_{o|\delta(s,ao)=s'} O(o|h, a) \quad (4)$$

That is, for any s and h with $\kappa^g(h) = s$, the probability of the state s' on action a must be equal to the cumulative probability of the observations o such that $\delta(s, ao) = s'$ ³. Together, this motivates the following definition,

Definition 1 We say a DMM ξ represents \mathcal{G} if it satisfies (3) and (4).

The following is the precise statement of our central modeling assumption.

Assumption 2 The DMM ξ^g represents the target environment \mathcal{G} .

Finally, from a *formal* perspective, DMMs are (trivially) maximally general as all the history-state maps computable using finite time and memory must have a FSM representation. Future research will reveal if this translates to applicability of the model across many practical problems.

2.5 From Models to MDPs and Optimal GRLE Policies

Each of our models $(\vec{\theta}, \vec{\phi}, \xi)$ (with $\xi \triangleq (q_o, \mathcal{S}, \Sigma, \delta)$) defines a MDP $\mathcal{M} \triangleq (\mathcal{A}, \mathcal{R}, \mathcal{S}, R_M, T, \gamma)$ (Bertsekas and Shreve, 1996). The first 3 components are as defined above and

$$R_M(r|s, a) \triangleq \vec{\theta}_{s,a}^g(r), \quad T(s'|s, a) \triangleq \vec{\phi}_{s,a}^g(s') \quad (5)$$

2. The POMDP belief-state space also has deterministic-probabilistic dynamics like our model, but the space is infinite.
 3. Two notes: (a) The fact that $\vec{\phi}_{s,a}^g$ is a distribution follows as $\delta(s, ao)$ maps to only one state. (b) Condition (4) is equivalent to saying that if $\kappa^g(h') = \kappa^g(h) = s$, then $\sum_{o|\delta(s,ao)=s'} O(o|h, a) = \sum_{o|\delta(s,ao)=s'} O(o|h', a)$ for all states $s' \in \mathcal{S}^g$.

where R_M is the MDP reward distribution and T is the state transition distribution. Given a policy $\bar{\pi} : \mathcal{S} \rightarrow \mathcal{A}$ for \mathcal{M} , the value function is defined as follows:

$$V^{\bar{\pi}}(s) \triangleq E_{R_M(r|s,a)}(r) + \gamma E_{T(s'|s,a)}[V^{\bar{\pi}}(s')] \quad (6)$$

where $a = \bar{\pi}(s)$. Using $\bar{\pi}$ we act in the \mathcal{G} as follows – for history $ao_{0:t}$, if $\delta(q_o, ao_{0:t}) = s$ we choose $\bar{\pi}(s)$ as our action. Note that if $h = h'ao$ and $q = \delta(q_o, h')$, then $\delta(q, ao) = s$ and hence we can infer states of our model incrementally. Optimal MDP policies are defined analogously to (2). Denoting the MDP constructed from $(\vec{\theta}^g, \vec{\phi}^g, \xi^g)$ by \mathcal{M}^g , we establish correspondences between policies of \mathcal{G} and \mathcal{M}^g .

Theorem 3 *Let π be a \mathcal{G} policy such that if $\kappa^g(h) = \kappa^g(h') = s$, then $\pi(h) = \pi(h')$. Then the \mathcal{M}^g policy defined by $\bar{\pi}(\kappa^g(h)) = \pi(h)$ is well defined and $V^\pi(h) = V^\pi(h') = V^{\bar{\pi}}(s)$. Conversely, each \mathcal{M}^g policy $\bar{\rho}$ also defines a corresponding \mathcal{G} policy ρ such that if $\kappa^g(h) = \kappa^g(h') = s$, then $\bar{\rho}(s) = \rho(h) = \rho(h')$ and $V^\rho(h) = V^\rho(h') = V^{\bar{\rho}}(s)$.*

The above theorem states something quite simple: any policy that assigns the same action to histories mapping to the same state, gives an \mathcal{M}^g policy, and each \mathcal{M}^g policy corresponds to a \mathcal{G} policy that assigns the same action to histories mapping to the same state.

Theorem 4 *There exists an optimal \mathcal{G} policy μ such that $\kappa^g(h) = \kappa^g(h')$, implies $\mu(h) = \mu(h')$. This further implies that an optimal \mathcal{M}^g policy defines an optimal \mathcal{G} policy.*

This theorem is saying that there is an optimal GRLE policy that is also an MDP policy – and hence the optimal MDP policy is also an optimal GRLE policy. The proofs of both theorems are in Appendix A.

3. Inference of the Correct Model

In this section we describe how to choose between different models given data in a Bayesian framework. First, in Section 3.1 we define our likelihood function and develop a fairly standard Bayesian inference framework around the function. Then in Sections 3.2.2 and 3.2.3 we show that our likelihood function is consistent. Recall that the consistency proof is necessary as our function is not standard (see below for more details). Throughout this section, we assume that all actions are chosen using a fixed \mathcal{G} policy π and that we are given a history + reward sequence (data) $\bar{a}\bar{o}_{0:v}$ and $\bar{r}_{0:v}$ (written in the sequel as $\bar{a}\bar{r}\bar{o}_{0:v}$) generated from \mathcal{G} using π (so in the sequence $\bar{a}\bar{r}\bar{o}_{0:v}$, $\bar{a}_k = \pi(\bar{a}\bar{o}_{<k})$). Since no history is ever repeated, assuming a fixed policy does not result in loss of generality.

3.1 The Bayesian Inference Framework

Our Bayesian inference framework is fairly standard, with the exception of the likelihood function. The main novelty here is that as our models do not model the observation distribution, it is a partial distribution over reward-observation sequences generated by the environment and hence the standard likelihoods used in Bayesian analysis can no longer be defined (see next paragraph). We define the following non-standard likelihood function. Each model $(\vec{\theta}, \vec{\phi}, \xi)$ (with $\xi = (q_o, \delta, \mathcal{S}, \Sigma)$),

defines the following distribution on \mathcal{R}^n conditioned on $\bar{a}\bar{o}_{0:n}$:

$$Pr(r_{0:n}|\bar{a}\bar{o}_{0:n}, \vec{\theta}, \xi) \triangleq \prod_{i=0}^n \vec{\theta}_{\bar{s}_{i-1}, \bar{a}_i}(r_i) \quad (7)$$

where, as usual, $\kappa(\bar{a}\bar{o}_{0:i}) = \bar{s}_i$ and $\bar{s}_{-1} = q_\circ$. This quantity is a distribution over reward sequences that result from the action-state sequence $\bar{a}\bar{s}_{0:n}$. We now use this as the likelihood function of $(\vec{\theta}, \vec{\phi}, \vec{\eta}, \xi)$ – that is the likelihood of ξ given data $\bar{a}\bar{r}\bar{o}_{0:n}$ is taken to be

$$Pr(\bar{r}_{0:n}|\bar{a}\bar{o}_{0:n}, \vec{\theta}, \xi) \quad (8)$$

To see in what sense this likelihood function is non-standard, consider the case where we model the observation distribution by the parameters $\vec{\eta}_{s,a}$ which are elements of the $|\mathcal{O}|$ dimensional simplex (i.e. probability distributions on \mathcal{O}). Then, the (standard) likelihood of a model $(\vec{\theta}, \vec{\phi}, \vec{\eta}, \xi)$ given $\bar{a}\bar{r}\bar{o}_{0:v}$ would simply be:

$$Pr(\bar{r}\bar{o}_{0:v}|\bar{a}_{0:v}, \vec{\theta}, \vec{\eta}, \xi) \triangleq \prod \vec{\theta}_{\bar{s}_{i-1}, \bar{a}_i}(\bar{r}_i) \vec{\eta}_{\bar{s}_{i-1}, \bar{a}_i}(\bar{o}_i)$$

where $\bar{s}_i = \kappa(\bar{a}\bar{o}_{0:i})$ and $\bar{s}_{-1} = q_\circ$. Since, by construction, $\vec{\eta}$ satisfies the relationship,

$$\vec{\phi}_{s,a}(s') = \sum_{o|\delta(s,ao)=s'} \vec{\eta}_{s,a}(o)$$

if we choose between two models using the standard likelihood as the cost function, we will be guaranteed consistency by the usual statistical consistency results. That is, in the limit of the length of the data going to infinity we will choose the model with the better reward and state transition distribution.

However, in (7) the parameters $\vec{\eta}$ and $\vec{\phi}$ are missing and so it is not clear whether the non-standard likelihood (8) is consistent. In particular, we need to show that (8) satisfies two types of consistency – (type 1) consistency that shows that the likelihood chooses models with the correct reward distribution, and (type 2) consistency that shows that the likelihood chooses models with the correct state transition distribution.

We establish these two types of consistency in Sections 3.2.2 and 3.2.3 while in the rest of this subsection we complete development of the Bayesian inference framework, which is fairly standard (see for instance Chipman et al. (1998)). In particular, we discuss how to compute the marginal likelihood of a DMM given the data $\bar{a}\bar{r}\bar{o}_{0:n}$. The marginal likelihood will then be used to infer the correct model in a Bayesian framework. We also discuss how to compute the Bayes estimate of the reward and state transition distributions for the model given the data. The DMM together with the Bayesian estimates give us our estimate of a model.

We start by rewriting (7) as:

$$Pr(r_{0:n}|\bar{a}\bar{o}_{0:n}, \vec{\theta}, \xi) = \prod_{s,a} \prod_r \vec{\theta}_{s,a}(r)^{m_{s,a}(r)}$$

where $m_{s,a}(r)$ is the number of times $r_i = r$ and $\bar{s}_{i-1} = s$ and $\bar{a}_i = a$ in $r_{0:n}$ and $\bar{a}\bar{s}_{0:n}$. Each $\vec{\theta}_{s,a}$ is given an uninformative Dirichlet prior $w(\cdot)$ which is conjugate to the multinomial distribution. The

marginal likelihood of state s and action a is given by integrating out the multinomial parameters given the Dirichlet prior:

$$L(r_{0:n}|\bar{a}\bar{o}_{0:n}, s, a, \xi) \triangleq \int \prod_r \vec{\theta}_{s,a}(r)^{m_{s,a}(r)} w(\vec{\theta}_{s,a}) d\vec{\theta}_{s,a} = \Gamma(1) \frac{\prod_{r'} \Gamma(m_{s,a}(r') + \frac{1}{|\mathcal{R}|})}{\Gamma(\sum_{r'} m_{s,a}(r') + 1)} \quad (9)$$

L here is not a distribution over reward sequences, but over rewards at state s for action a . The notation was chosen because the marginal likelihood of ξ is a distribution over reward sequences (that has $\bar{s}_{0:k}$ as state sequence) and is given by:

$$Pr(r_{0:n}|\bar{a}\bar{o}_{0:n}, \xi) \triangleq \prod_{s,a} L(r_{0:n}|\bar{a}\bar{o}_{0:n}, s, a, \xi) \quad (10)$$

The predictive distribution for the reward at state s and action a is given by:

$$Pr(r|\bar{a}\bar{r}\bar{o}_{0:n}, s, a, \xi) \triangleq \frac{m_{s,a}(r) + \frac{1}{|\mathcal{R}|}}{1 + \sum_{r'} m_{s,a}(r')} \quad (11)$$

Similarly, the state transition distributions, $Pr(s'|s, a, \xi)$, are multinomials and we perform the same trick with Dirichlet priors. This time, we just need the predictive distribution, which is given by,

$$Pr(s'|\bar{a}\bar{r}\bar{o}_{0:n}, s, a, \xi) \triangleq \frac{\hat{m}_{s,a}(s') + \frac{1}{|\mathcal{S}|}}{1 + \sum_q \hat{m}_{s,a}(q)} \quad (12)$$

where $\hat{m}_{s,a}(s')$ is the number of times state $\bar{s}_i = s'$ and $\bar{s}_{i-1} = s$ and $\bar{a}_i = a$ in $\bar{a}\bar{s}_{0:n}$.

To complete the specification of the Bayesian criterion for model selection, we denote the space of DMMs we consider by \mathbf{H} , and we assume we have a prior W over this space. The details of these will be given in Section 4. Given the model space and the prior, the posterior probability of a ξ for the data $\bar{a}\bar{r}\bar{o}_{0:n}$ is now given by:

$$Pr(\xi|\bar{a}\bar{r}\bar{o}_{0:n}) \triangleq \frac{Pr(\bar{r}_{0:n}|\bar{a}\bar{o}_{0:n}, \xi)W(\xi)}{\sum_{\xi' \in \mathbf{H}} Pr(\bar{r}_{0:n}|\bar{a}\bar{o}_{0:n}, \xi')W(\xi')} \quad (13)$$

Our goal will be to learn the maximum a-posteriori (MAP) ξ – i.e. $\arg \max_{\xi'} Pr(\xi'|\bar{a}\bar{r}\bar{o}_{0:n})$ and this learning algorithm is given in Section 4. Hence, (13) is our criterion/cost function for choosing the right model.

3.2 Consistency of the Likelihood

We now prove our consistency theorems. We first introduce some preliminaries that we use throughout and then we state our theorems. In Section 3.2.2 we establish consistency for the reward distribution and then in Section 3.2.3 we establish consistency for the state transition distribution.

3.2.1 PRELIMINARIES

We start by recalling that all actions are chosen using a fixed policy π and in the sequel we will use this fact without further mention. Let Ω be the σ -algebra over $(\mathcal{R} \times \mathcal{O})^\infty$ generated by the set of cylinder sets:

$$\{C(ro_{0:n})|ro_{0:n} \in (\mathcal{R} \times \mathcal{O})^n, n \in \mathbb{N}\}, \quad \text{where } C(ro_{0:n}) \triangleq \{\hat{r}\hat{o}_{0:\infty}|\hat{r}\hat{o}_{0:n} = ro_{0:n}\}$$

We use the following fact often in the sequel. The definition of cylinder sets implies that $C(ro_{0:n}) \subset C(\hat{r}\hat{o}_{0:m})$ if and only if $n \geq m$ and $ro_{0:m} = \hat{r}\hat{o}_{0:m}$.

We need the probability measure, defined by RO and π , over the algebra of cylinder sets:

$$RO_{\pi}[C(ro_{0:n})] \triangleq \prod_{k=0}^n RO(ro_k | ao_{<k}, \pi(ao_{<k}))$$

We extend this measure over Ω in the usual way and denote this extension by RO_{π} as well. We will often write $RO_{\pi}[C(ro_{0:n})]$ as $RO_{\pi}(ro_{0:n})$.

3.2.2 CONSISTENCY FOR REWARD DISTRIBUTION (TYPE 1)

In this section we establish consistency of the reward distribution. We will start by stating precisely what it means for a model to be incorrect. To that end, for a model $(\vec{\theta}, \vec{\phi}, \xi)$ with $\xi = (q_{\circ}, \delta, \mathcal{S}, \Sigma)$ define for $n \in \mathbb{N}$ and $\epsilon > 0$,

$$A_{\pi, \epsilon, n} \triangleq \left\{ ro_{0:\infty} \left| KL \left[\vec{\theta}_{\kappa^g(ao_{0:n}, a_{n+1})}^g \parallel \vec{\theta}_{\kappa(ao_{0:n}, a_{n+1})} \right] \geq \epsilon \right. \right\}$$

where KL is the KL divergence between the distributions. So $A_{\pi, \epsilon, n}$ is the set of reward-observation sequences for which the n length prefixes have ‘ ϵ -incorrect’ reward distribution (i.e. the KL divergence between the true distribution and distribution given by the model is $\geq \epsilon$).

Definition 5 *Define:*

$$A_{\pi, \epsilon} \triangleq \limsup_{n \rightarrow \infty} A_{\pi, \epsilon, n}$$

So $A_{\pi, \epsilon}$ is the formal description of the event that the model $(\vec{\theta}, \vec{\phi}, \xi)$ predicts incorrectly infinitely often. We can now state our first theorem (all the proofs are in Appendix B):

Theorem 6 *If $C(\bar{r}\bar{o}_{0:v}) \subset A_{\pi, \epsilon}$ then for all sequences $ro_{0:\infty} \in C(\bar{r}\bar{o}_{0:v})$ we have*

$$\lim_{n \rightarrow \infty} \frac{Pr(ro_{0:v+n} | ao_{0:v+n}, \vec{\theta}^g, \xi^g)}{Pr(ro_{0:v+n} | ao_{0:v+n}, \vec{\theta}, \xi)} = \infty$$

in RO_{π} probability. Alternatively, if there exists some N such that $n > N$ implies

$$KL[\vec{\theta}_{\kappa^g(ao_{0:n}, a)}^g \parallel \vec{\theta}_{\kappa(ao_{0:n}, a)}] = 0$$

for any sequence $ao_{0:n}$, then the above ratio is bounded in RO_{π} probability.

The theorem says something that is quite intuitive. In the above, $C(\bar{r}\bar{o}_{0:v}) \subset A_{\pi, \epsilon}$ implies that on all extensions of $\bar{r}\bar{o}_{0:v}$ the model predicts incorrectly infinitely often. In this case, as we see more of the output, the ratio of likelihoods of the true model and the incorrect model will go to infinity in probability. Similarly, if in no extension of $\bar{r}\bar{o}_{0:v}$ the model predicts incorrectly infinitely often (and is hence indistinguishable from the true model from a certain point on), then the ratio remains bounded. This establishes type 1 consistency.

We now state the consistency theorem for the marginal likelihood. We start by defining for any $n \in \mathbb{N}$,

$$A_{\pi, \epsilon, n}^M \triangleq \left\{ ro_{0:\infty} \left| KL \left[\vec{\theta}_{\kappa^g(ao_{0:n}, a_{n+1})}^g \parallel Pr(r | ar_{0:n}, \kappa(ao_{0:n}), a_{n+1}, \xi) \right] > \epsilon \right. \right\}$$

Definition 7 *Define:*

$$A_{\pi,\epsilon}^M \triangleq \limsup_{n \rightarrow \infty} A_{\pi,\epsilon,n}^M$$

Our consistency theorem for the marginal likelihood is as follows:

Theorem 8 *If $C(\bar{r}\bar{o}_{0:v}) \subset A_{\pi,\epsilon}^M$ then for all sequences $ro_{0:\infty} \in C(\bar{r}\bar{o}_{0:v})$ we have*

$$\lim_{n \rightarrow \infty} \frac{Pr(r_{0:v+n}|a_{00:v+n}, \xi^g)}{Pr(r_{0:v+n}|a_{00:v+n}, \xi)} = \infty \quad (14)$$

in RO_π probability. Alternatively, if there exists some N such that $n > N$ implies

$$KL[Pr(r|ar_{00:n}, s^g, a, \xi^g) || Pr(r|ar_{00:n}, s, a, \xi)] = 0$$

for any sequence $ar_{00:n}$, then the above ratio is bounded in RO_π probability.

So this theorem is saying that, if a model is such that on the sequence being observed it will continue being incorrect, then its marginal likelihood will be overwhelmed by that of the correct model or any other model that is indistinguishable from the latter. On the other hand, it is trivially true that if the model is correct from a certain point onwards, then the marginal likelihood ratio will remain bounded.

3.2.3 CONSISTENCY FOR STATE TRANSITION DISTRIBUTION (TYPE 2)

We now state type 2 consistency results – that is, we show that if a model has bounded marginal likelihood ratio then it also gives the correct estimate of state transition distribution. To that end, for any $s \in \mathcal{S}$ we define

$$\bar{A}_{\pi,s,n} \triangleq \{ro_{0:\infty} | \kappa(a_{00:n}) = s\}, \text{ and } \bar{A}_{\pi,s} \triangleq \limsup_{n \rightarrow \infty} \bar{A}_{\pi,s,n}$$

We also define $\bar{A}_{\pi,a,n}$, $\bar{A}_{\pi,a}$ for the action a in a similar way. We set $\bar{A}_{\pi,sa,n} = \bar{A}_{\pi,a,n} \cap \bar{A}_{\pi,s,n-1}$ and $\bar{A}_{\pi,sa} = \limsup_{n \rightarrow \infty} \bar{A}_{\pi,sa}$. As in the previous section, these definitions are formal expressions of the events in the conditional occurring infinitely often.

Definition 9 *A state s , action a and pair sa occurs infinitely often in $C(ro_{0:n})$ iff, respectively $C(ro_{0:n}) \subset \bar{A}_{\pi,s}$, $C(ro_{0:n}) \subset \bar{A}_{\pi,a}$ and $C(ro_{0:n}) \subset \bar{A}_{\pi,sa}$. These three notions are denoted, respectively, by $s \in^\infty C(ro_{0:n})$, $a \in^\infty C(ro_{0:n})$ and $sa \in^\infty C(ro_{0:n})$.*

We now define precisely what it means for a state of the DMM to model a state of ξ^g . First define for any state $s \in \mathcal{S}$ and $C(ro_{0:n})$

$$\bar{Z}_{\pi,s}(ro_{0:n}) \triangleq \{\hat{a}\hat{o}_{0:j} | j > n, \exists \hat{r}_{0:j} \text{ with } \hat{r}\hat{o}_{0:n} = ro_{0:n} \text{ and } \kappa(\hat{a}\hat{o}_{0:j}) = s\},$$

So $Z_{\pi,s}(ro_{0:n})$ are the set of histories that are extensions of $ro_{0:n}$ and that also map to state s .

Definition 10 *Let s be a state of the DMM ξ and assume that $s \in^\infty C(ro_{0:n})$. We say that s models a state s' of ξ^g on $C(ro_{0:n})$ if all but a finite number of elements $h \in \bar{Z}_{\pi,s}(ro_{0:n})$ satisfy $\kappa^g(h) = s'$. We denote this by $(s, s') \in^{\equiv} C(ro_{0:n})$.*

So for $s \in \mathcal{S}, s' \in \mathcal{S}^g$, we have $(s, s') \in^{\equiv} C(ro_{0:n})$ if all but a finite number of histories ‘in’ $C(ro_{0:n})$ that map to s , maps only to s' (and hence the use of the word ‘model’). The significance of this definition is that if s models s' , we would expect any statistics collected at s to converge to the true distribution at s' by the law of large numbers.

The following notion is needed for Assumption 12, which is in turn used in the statement of the theorems below.

Definition 11 *The string $h \in \mathcal{H}$ of length k occurs infinitely often after s in $C(ro_{0:n})$ if $C(ro_{0:n}) \subset \limsup_{n \rightarrow \infty} \bar{A}_{\pi, s, h, n}$ where $\bar{A}_{\pi, s, h, n} \triangleq \{ ro_{0:\infty} \mid \kappa(ao_{0:n}) = s, ao_{n+1:n+k} = h \}$. We denote this by $(h, s) \in_{\mathcal{H}}^{\infty} C(ro_{0:n})$.*

Assumption 12 *We assume that the model $(\vec{\theta}^g, \vec{\phi}^g, \xi^g)$ is identifiable given π ; that is for each pair $s, s' \in \mathcal{S}^g$ with $s, s' \in^{\infty} C(\bar{r}\bar{o}_{0:v})$, $\exists h \in \mathcal{H}$ such that*

1. $(h, s), (h, s') \in_{\mathcal{H}}^{\infty} C(\bar{r}\bar{o}_{0:v})$.
2. $\delta^g(s, h) = q$ and $\delta^g(s', h) = q'$.
3. $\vec{\theta}_{q,a}^g \neq \vec{\theta}_{q',a}^g$ where $qa, q'a \in^{\infty} C(\bar{r}\bar{o}_{0:v})$.

This assumption means that for any two states s, s' in the true DMM ξ^g , there must exist a history h such that h leads to states q, q' from s, s' respectively and q and q' have different true reward distributions for an action that is taken infinitely often at q and q' . We view this assumption as weak from a practical perspective because in most domains of interest, for any pair of states of the true model, there exist a history that lead to different regions of the state space and hence different rewards. We now state theorems for type 2 consistency.

Theorem 13 *Assume that for DMM ξ the ratio (14) remains bounded in probability for sequences in $C(\bar{r}\bar{o}_{0:v})$. Then, under Assumption 12, $\Pr(s'|aro_{0:n}, s, a)$ converges almost surely to $\vec{\phi}_{q,a}^g(q')$ where $(s, q), (s', q') \in^{\equiv} C(\bar{r}\bar{o}_{0:v})$ and $sa \in^{\infty} C(\bar{r}\bar{o}_{0:v})$.*

The interpretation of the theorem is that $(s, q), (s', q') \in^{\equiv} C(\bar{r}\bar{o}_{0:v})$ means that s models q and s' models q' . In this case, for a DMM that is correct, i.e. one for which (14) remains bounded, the estimated state transition distribution of each state s of ξ converges to the distribution of the state q of ξ^g it models (but only for actions a that are also taken infinitely often after s).

4. Online Learning and Planning

In this section we describe a heuristic algorithm called DMM-OLP for learning DMMs and planning with them online. DMM-OLP interleaves phases of acting in the environment, and in the process collecting data, and improving its estimate of true DMM ξ^g via learning. During learning, the likelihood function (7) is used (via its marginal in the posterior (13)) to choose between different models. We start by discussing looped-DMMs, the class of DMMs that we use as our hypothesis space. Then we discuss the DMM-OLP algorithm and the stochastic search that is within it used to estimate/learn DMMs.

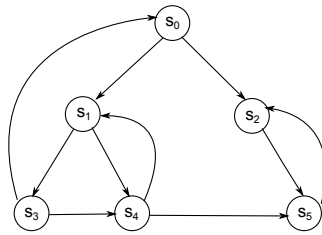


Figure 3: Example of a looped DMM. Edges for internal nodes can only connect to a child or loop back to an ancestor. For leaf nodes, edges can connect to other leaves.

4.1 Looped DMMs

DMMs are sequential versions of DPFA (Vidal et al., 2005) for which learning is NP-hard when states might have equal distributions. Since this is in fact often the case in RL problems, we need to put significant bias in our search algorithm. We introduce bias by restricting ourselves to a particular class of tree-like DMMs, which we refer to as looped DMMs.

In a looped DMM, the initial state q_0 can be considered to be the root node, and then each internal node satisfies the constraint that outgoing edges at each node can go only to a child or an ancestor. Only leaf nodes are allowed to loop to nodes that are not its ancestors (see Figure 3). This is similar to a looping suffix tree (Holmes and Isbell-Jr., 2006), but our model is not a suffix tree structure as each node in the graph is in fact a state. Each node in a looped DMM also has a *context* associated with it given by the string labeling the path from the root to that node. This context is used by the **search** algorithm to perform surgery on looped DMMs while searching.

We now discuss the expressiveness of this subclass. It is easy to see that all DMMs, in fact, have a loopy representation, although with possibly exponentially many states. In particular, a DMM with n states can be represented by a looped DMM of depth n , such that each path from the root to a leaf corresponds to a path of length n starting at q_0 in the original DMM. Hence, the looped representation for a given DMM can be constructed by traveling along a path in the DMM and adding a new node at each step or looping back to an ancestor if the state has been seen before. Hence in any given path from the root in the looped representation, no state appears in more than one node. Finally, when a leaf is reached it will always be possible to loop back to an ancestor for each letter of the edge alphabet Σ because the depth is equal to the number of states in the DMM, and hence any state must have appeared as an ancestor.

Finally, the main reason we choose the looped DMM representation is because it seems particularly suitable for typical POMDP domains. This is because the dynamics in such domains has a neighborhood structure, where we are only able to transition back to states that were visited recently. In the sequel any reference to a DMM is in fact a looped DMM.

4.2 The DMM-OLP Algorithm

This algorithm was outlined in Figure 2 and is now given in full in Algorithm 1. The initializations are self explanatory and so we focus on the main **for** loop. The first step (line 4), learning a better estimate, is the heart of the algorithm and discussed in detail below. The second step (line 5) is

performed as follows. First, a MDP corresponding to ξ_i is constructed using the method in Section 2.5, but using the estimates (11) and (12) for, respectively, the MDP reward and state-transition distributions (5) respectively. Then the optimal policy is learned offline by using Q-learning on the constructed MDP (we do not use value iteration because Q learning was seen to be quicker without any noticeable difference in performance).

Algorithm 1 Algorithm for online learning and planning.

function: DMM-OLP()

Require: T the length of each phase of acting in the world, N the number of phases.

- 1: Generate a random sample $\bar{a}\bar{r}\bar{o}_{0:T-1}$ from the environment by choosing actions at random.
 - 2: Let ξ_0 , the DMM with a single state (i.e. all transitions are to q_o), be the initial estimate of ξ^g .
 - 3: **for** $i = 1$ to N **do**
 - 4: Learn ξ_i from ξ_{i-1} using history so far $\bar{a}\bar{r}\bar{o}_{0:T_{i-1}}$ by setting ξ_i to **search** ($\xi_{i-1}, \bar{a}\bar{r}\bar{o}_{0:T_{i-1}}$).
 - 5: Estimate optimal policy $\hat{\pi}^*$ for MDP corresponding to ξ_i using Q-learning.
 - 6: Generate a sample $\bar{a}\bar{r}\bar{o}_{T_i:T(i+1)-1}$ from the environment by choosing actions according to $\hat{\pi}^*$.
 - 7: **end for**
-

In the third step (line 6) we follow the policy as described at the end of Section 2.5. Additionally, whenever the MDP/DMM transitions from state s to s' on action a and we observe reward r , the algorithm performs Q-learning updates on the Q-value of s at a (the Q-values learned in line 5 are used as initial values for this step). This additional Q-learning is to ensure that we overcome incorrect parameter estimates due to mismatch between δ^i, δ^g and $\mathcal{S}^i, \mathcal{S}^g$ of ξ_i and ξ^g and still properly explore the domain. This is our approach to dealing with the infamous ‘exacerbated exploration problem’ described in Section 1.2. During this Q-learning, we also take a random action with probability 0.1 to further aid in exploration. This approach is entirely heuristic and a formal solution to this problem is a research topic in and of itself. The idea behind our heuristic is that we start by acting according to what our current model tells us is the best course of action. Since our model is likely incomplete/incorrect, we allow the computed Q-values to be changed based in experience. Finally, the $\epsilon = 0.1$ exploration is used to further observe effects of actions on hidden states not modeled in the current estimate ξ_i , and hence discover it during the estimation step of the next phase.

4.3 The Stochastic Search Algorithm

We now discuss how to estimate ξ_i in the first step of the **for** loop of DMM-OLP (line 4) using a search algorithm. The search algorithm (**search**, Algorithm 2) is a stochastic search over this space with the posterior (13) as the selection criterion. At each step, **search** performs one of two possible operations. It chooses a leaf node, and either extends the node by adding another leaf node (**extend**, Algorithm 3); or it chooses an internal or a leaf node and loops an unextended transition of the leaf back to an ancestor (**loop**, Algorithm 4). These operations are chosen with probabilities 0.6 and 0.4 respectively. While other values of these probabilities worked, these values were experimentally found to be the quickest. The modification is then accepted as the estimate of ξ^g in the next iteration if it satisfies a Metropolis-Hastings (MH) style condition (line 6). The condition contains a parameter α which was set to 2^5 in all our experiments (this simulates the proposal

Algorithm 2 Stochastic Search

function: $\text{search}(\xi, \bar{a}\bar{r}\bar{o}_{0:n})$
Require: K , the number of iterations.

- 1: Set ξ_{cur} to ξ .
 - 2: **for** $i = 1$ to K **do**
 - 3: With probability 0.6 and 0.4 respectively, set ξ_{prop} to **extend** $(\xi_{cur}, \bar{a}\bar{r}\bar{o}_{0:n})$ or **loop** $(\xi_{cur}, \bar{a}\bar{r}\bar{o}_{0:n})$.
 - 4: Choose $rnd \in [0, 1]$ uniformly at random.
 - 5: **if** $\frac{Pr(\xi_{prop}|\bar{a}\bar{r}\bar{o}_{0:n})}{\alpha Pr(\xi_{cur}|\bar{a}\bar{r}\bar{o}_{0:n})} > rnd$ **then**
 - 6: Set ξ_{cur} to ξ_{prop} .
 - 7: **end if**
 - 8: **end for**
 - 9: **Return** ξ_{cur} .
-

Algorithm 3 Function for Extending Edges

function: $\text{extend}(\xi, \bar{a}\bar{r}\bar{o}_{0:n})$

- 1: Sample a leaf node/state s according to frequency in $\bar{s}_{0:n}$ ($\bar{s}_k = \delta(q_o, \bar{a}\bar{o}_{0:k})$).
 - 2: Sample an outgoing edge ao at s according to the empirical (w.r.t. $\bar{a}\bar{r}\bar{o}_{0:n}$) next step reward at ao .
 - 3: Create a new state s_{new} with *context* $x_s ao$ where x_s is the context of s (the context of root is empty).
 - 4: Set $\delta(s, ao)$ to s_{new} ; set $\delta(s_{new}, a'o') = \arg \max \{length(x_{s'}) | x_{s'} \text{ is a suffix of } x_{s_{new}}\}$.
 - 5: Add s_{new} to the states of ξ .
 - 6: **Return** ξ .
-

distribution ratio in the MH acceptance probability). This value was chosen because experimentally this value was found to result in the best models.

Algorithm 4 Function for Looping Edges.

function: $\text{loop}(\xi, \bar{a}\bar{r}\bar{o}_{0:n})$

- 1: Sample a leaf node/state s according to frequency in $\bar{s}_{0:n}$ ($\bar{s}_k = \delta(q_o, \bar{a}\bar{o}_{0:k})$).
 - 2: From all the outgoing edges of s that were not constructed by **extend** sample an outgoing edge ao according to the empirical (w.r.t. $\bar{a}\bar{r}\bar{o}_{0:n}$) next step reward at ao .
 - 3: Choose an ancestor s' of s according $2^{-m}/Z$, where m is the no. of ancestors of s between s and s' .
 - 4: Set $\delta(s, ao)$ to s' .
 - 5: **Return** ξ .
-

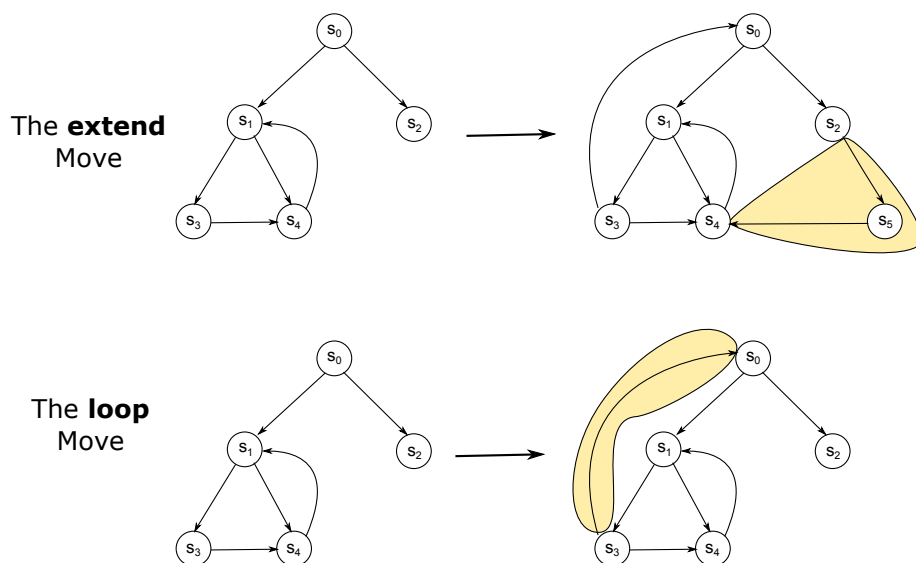


Figure 4: The two operators, **extend** and **loop**, used to search through looped DMM space.

4.4 Operators on Looped DMM

We now discuss the heuristics used in **extend** and **loop** to choose nodes to modify. In line 1 of both procedure, we sample states according to their frequency in the data because we expect the impact to the likelihood to be the greatest if we modify high frequency states. Then in line 2 of both we sample an outgoing edge ao of the state s from line 1 according to the average reward seen at s after taking action a and observing o . The intuition is that if the reward at that edge is high, then it might be a good idea to create a new state there (in **extend**) or move it to a different state (in **loop**) to get a better model. A more sophisticated version of this heuristic would be to use the future discounted reward at the edge. In lines 3 and 4 **extend** constructs the new state for the chosen edge. In line 3 **loop** chooses an ancestor to loop back to, choosing a closer ancestor with higher probability. This is because in typical POMDP domains we expect the state to transition back to states visited very recently. **loop** then loops the edge in line 4.

5. Experiments

In the following we describe experiments we performed to test the efficacy of our algorithm. We first describe the set up for the algorithms, the domains we used and then we present our results and their interpretation.

5.1 Setup

We conducted experiments to illustrate that we do in fact extend McCallum’s Utile Suffix Memory in desirable ways. We first ran experiments on three POMDP maze problems to evaluate our model on standard problems. Then we ran experiments on three novel, non-episodic domains to test the long term memory ability of our method. The domains for the first are given in Fig. 5 (numbered 1,2,

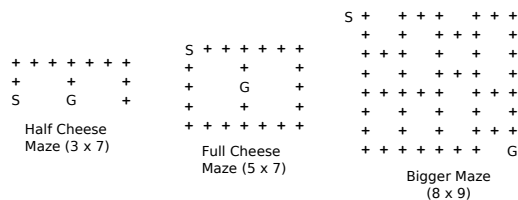


Figure 5: The three POMDP mazes. S is the start state and G is the goal state.

and 3 respectively) and they have the same dynamics as in McCallum (1995b) with state transition and observation noise.

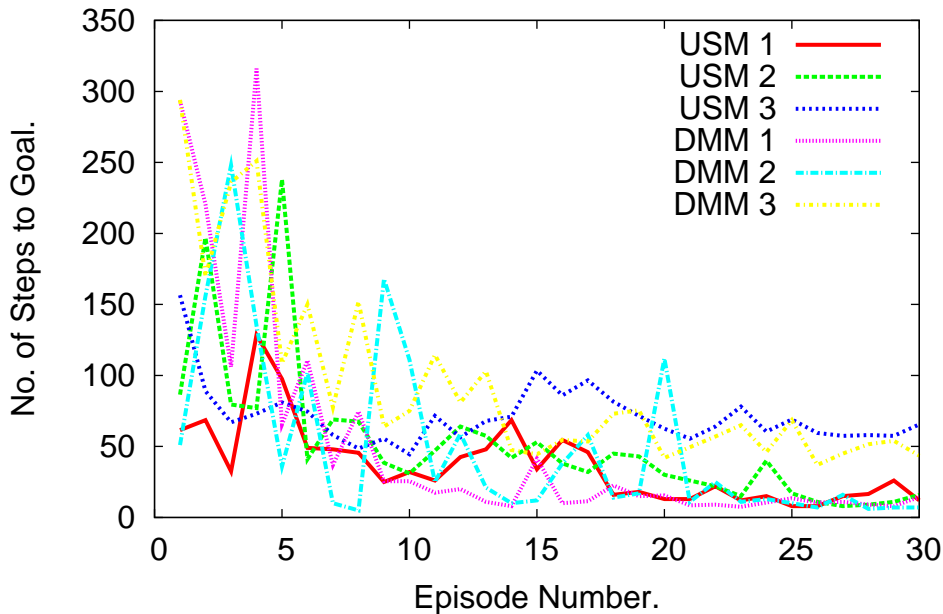


Figure 6: Results for the maze domains; USM i is the number of steps to goal state for maze i at the given trial. Similarly for DMM i etc.

The domains for the second type of experiments are a set of ‘harvesting’ tasks with m crops. A crop i become ready to be harvested after being developed through k_i different phases. It is ready to be harvested for just 1 time step once development is complete and then it spoils immediately, requiring to be grown again. The agent has $m + 1$ different actions. Action a_i develops crop i to its

next phase with probability 0.7 and develops some other crop with probability 0.3. The remaining action allows the agent to harvest any crop ready to be harvested at that step. The observation at each step is the crop that has been developed at that step. So essentially this is a counting task where the agent has to remember which phase each crop is at, and hence a finite history suffix is not sufficient to make a decision. The agent receives -1.0 reward for each action, and 5.0 reward for a successful harvest. There are no episodes and the each problem instance essentially runs forever.

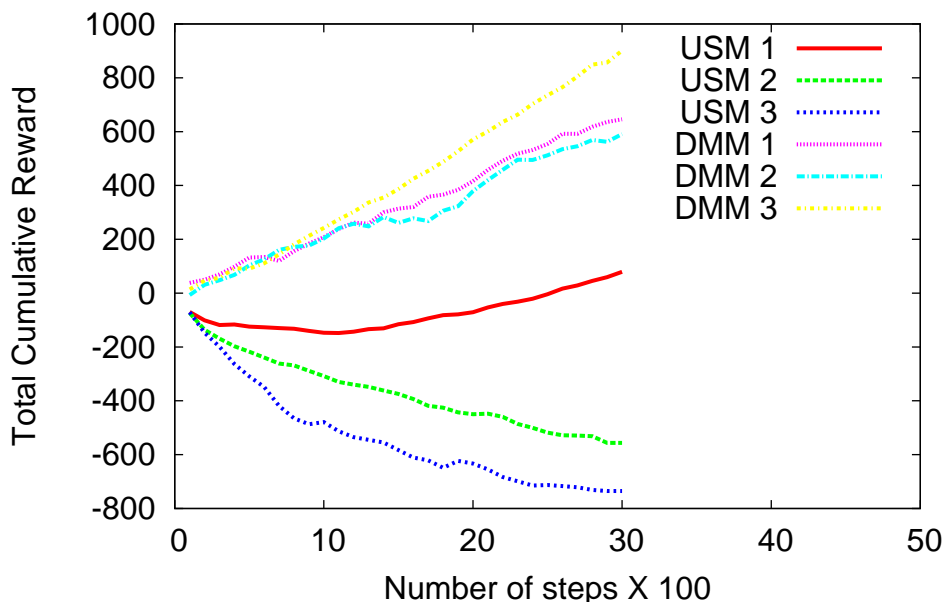


Figure 7: Results for the mining domains. USM i is the total reward accumulated by USM on mining problem i at the given step. Similarly for DMM i etc.

We used several different values of $[m, (k_1, k_2, \dots, k_m)] : [4, (2, 2, 2, 2)]$, $[5, (2, 2, 2, 2, 2)]$ and $[7, (2, 2, 2, 2, 2, 2, 2)]$ (referred to as harvesting problems 1, 2 and 3 respectively from now on). The POMDP representation of these problems have, respectively, $4 \cdot 2^4 = 64$, $5 \cdot 2^5 = 160$, $7 \cdot 2^7 = 896$ states. In each of these cases, we let each USM and DMM run for 3000 steps and report the total accumulated reward for each method averaged over 10 trials.

5.2 Results

The results for the three maze domains are given in Fig. 6 we report the median time per trial to get to the goal state to be comparable to McCallum (1995b) – the performance of our implementation

of USM matches that of McCallum (1995b). In these experiments in Alg. 1, T is variable, equal to the number of steps the agent reaches the goal, while $N \triangleq 30$; and $K \triangleq 5000$.

As can be seen, our algorithm more or less matches USM, except at the beginning where the number of steps is much higher. This is because our model is trying to learn a recursive model and hence gets confused when there is little data. Not surprisingly, the average number of states estimated by our method was 45.3, 67.5 and 90.5 compared to 204.5, 231.5 and 623.5 for USM. Similarly, our stochastic search took significantly longer off-line processing to learn (5 minutes compared to 10s of seconds for USM). So for these types of episodic tasks, it might be better to first learn a model using USM and then compress it to a DMM Shalizi and Klinkner (2004). Regardless, the experiments give evidence that DMMs are viable for modeling hidden RL environments, and that our inference criterion (7) is correct.

The results for the 3 harvesting/counter domains are in Fig. 7. Here the difference between the two methods becomes quite significant, with our method dominating USM completely. The differences in computational time were quite similar to that in the previous set of experiments. The average number of states inferred differed by about 400, with the USM continually creating new states.

6. Conclusion

In this paper we proposed a novel model, the DMM, for modeling the environment in partially observable reinforcement learning problems. We proposed an inference criteria for choosing between the models and then proved consistency of this criterion. We also presented a heuristic algorithm for learning and planning with DMMs along with experiments to show the promise of our approach. Directions for future research are many, the foremost being constructing a principled algorithm for learning these models – this is currently under active development. Another problem that needs to be solved is the exacerbated exploration problem which, as we pointed out, is quite hard. Other important extensions include more experiments on more difficult domains to further establish viability of this approach; dealing with very large, possibly continuous state spaces via state-aggregation schemes; and extension to relational, multi-agent domains.

Acknowledgments.

We would like to thank Marcus Hutter for detailed comments on an early draft of the paper and various other discussions. The research was supported by the Australian Research Council Discovery Project DP0877635 “Foundation and Architectures for Agent Systems”.

Appendix A. Proofs for Section 2.5

Definition 14 Define the following n -step V function for any \mathcal{G} policy π :

$$V_0^\pi(h) := 0, \quad V_n^\pi(h) := E_{R(r|h, \pi(h))}(r) + \gamma E_{O(o|h, \pi(h))}[V_{n-1}^\pi(h\pi(h)o)]$$

Similarly, define the following n -step V function for any MDP policy $\bar{\pi}$ for the MDP $(\mathcal{A}, \mathcal{R}, \mathcal{S}, R_M, T, \gamma)$:

$$V_0^{\bar{\pi}}(s) := 0, \quad V_n^{\bar{\pi}}(s) := E_{R_M(r|s, \bar{\pi}(s))}(r) + \gamma E_{T(s'|s, \bar{\pi}(s))}[V_{n-1}^{\bar{\pi}}(s')]$$

Lemma 15 For each history h , $\lim_{n \rightarrow \infty} V_n^\pi(h) = V^\pi(h)$ (see (1)). Similarly, for each MDP policy $\bar{\pi}$, $\lim_{n \rightarrow \infty} V_n^{\bar{\pi}}(s) = V^{\bar{\pi}}(s)$ (see (6)).

Proof Using standard value iteration convergence proof methods Bertsekas and Shreve (1996). ■

Proof [Proof of Theorem 3] That $\bar{\pi}$ is well defined is obvious. We will show that $V_n^\pi(h) = V_n^{\bar{\pi}}(s)$ for all n , and hence in the limit, whenever $\kappa^g(h) = s$; this will complete the proof via Lemma 15. We show this by induction on n ; for $n = 0$, the theorem holds trivially. Now assume that the statement holds for $n = m > 1$. Then, we can write with $a = \pi(h)$,

$$\begin{aligned}
 & V_{m+1}^\pi(h) \\
 & \stackrel{(1)}{=} E_{R(r|h,a)}(r) + \gamma E_{O(o|h,a)}[V_m^\pi(hao)] \quad \stackrel{(2)}{=} E_{\bar{\theta}_{s,a}^g}(r) + \gamma \sum_o O(o|h,a) V_m^\pi(hao) \\
 & \stackrel{(3)}{=} E_{\bar{\theta}_{s,a}^g}(r) + \gamma \sum_{s'} \sum_{o|\kappa^g(hao)=s'} O(o|h,a) V_m^{\bar{\pi}}(s') \\
 & \stackrel{(4)}{=} E_{\bar{\theta}_{s,a}^g}(r) + \gamma \sum_{s'} \bar{\phi}_{s,a}^g(s') V_m^{\bar{\pi}}(s') \quad \stackrel{(5)}{=} V_{m+1}^{\bar{\pi}}(s)
 \end{aligned}$$

(1) is just definition; (2) follows because $(\bar{\theta}^g, \bar{\phi}^g, \xi^g)$ represents \mathcal{G} ; (3) follows as κ^g partitions the history space and by the inductive hypothesis; (4) is by another application of the fact that $(\bar{\theta}^g, \bar{\phi}^g, \xi^g)$ represents \mathcal{G} . (5) is by definition of $V_{m+1}^{\bar{\pi}}$. This proves the first part of the theorem.

For the second part, the \mathcal{G} policy ρ satisfies the conditions of the first part of the theorem and hence the result follows. ■

Definition 16 Define the following n -step optimal Q function:

$$Q_0^*(h, a) := 0, \quad Q_n^*(h, a) := E_{R(r|h,a)}(r) + \gamma E_{O(o|h,a)}[\max_{a' \in \mathcal{A}} Q_{n-1}^*(hao, a')]$$

Define the optimal Q function to be:

$$Q^*(h, a) := E_{R(r|h,a)}(r) + \gamma E_{O(o|h,a)}[\max_{a' \in \mathcal{A}} Q^*(hao, a')]$$

Lemma 17 For each action a and history h , $\lim_{n \rightarrow \infty} Q_n^*(h, a) = Q^*(h, a)$.

Proof Using standard value iteration convergence proof methods (Bertsekas and Shreve, 1996). ■

We note that by construction, $V^*(h) = \arg \max_a Q^*(h, a)$; indeed,

$$Q^*(h, a) := E_{R(r|h,a)}(r) + \gamma E_{O(o|h,a)}[V^*(hao)]$$

Proof [Proof of Theorem 4]

We will show that if $\kappa^g(h) = \kappa^g(h') = s$ then $Q_n^*(h, a) = Q_n^*(h', a)$ for all actions a . Coupled with the convergence of $Q_n^*(\bar{h}, a) \rightarrow Q^*(\bar{h}, a)$ from Lemma 17, this will show that $Q^*(h, a) = Q^*(h', a)$. This implies that the sets $\arg \max_a Q^*(h, a)$ and $\arg \max_a Q^*(h', a)$ are identical if $\kappa^g(h) = \kappa^g(h')$. Since $\arg \max_a Q^*(h, a)$ contains only actions chosen by an optimal policy, existence of a policy satisfying the conditions of the theorem is immediate, and hence first part of the theorem is proved.

To show that $Q_n^*(h, a) = Q_n^*(h', a)$ we proceed by induction on n . The base case $n = 0$ holds trivially by definition. Now assume that $Q_n^*(h, a) = Q_n^*(h', a)$ for all $n \leq m$. For $n = m + 1$ we write:

$$\begin{aligned}
 Q_{m+1}^*(h, a) &\stackrel{(1)}{=} E_{R(r|h,a)}(r) + \gamma E_{O(o|h,a)}[\max_{a' \in \mathcal{A}} Q_m^*(hao, a')] \\
 &\stackrel{(2)}{=} E_{R(r|h',a)}(r) + \gamma \sum_{s'} \sum_{o|\kappa^g(hao)=s'} O(o|h, a) \max_{a' \in \mathcal{A}} Q_m^*(hao, a') \\
 &\stackrel{(3)}{=} E_{R(r|h',a)}(r) + \gamma \sum_{s'} \max_{a' \in \mathcal{A}} Q_m^*(hao^{s'}, a') \sum_{o|\kappa^g(hao)=s'} O(o|h, a) \\
 &\stackrel{(4)}{=} E_{R(r|h',a)}(r) + \gamma \sum_{s'} \max_{a' \in \mathcal{A}} Q_m^*(h'ao^{s'}, a') \sum_{o|\kappa^g(hao)=s'} O(o|h, a) \\
 &\stackrel{(5)}{=} E_{R(r|h',a)}(r) + \gamma \sum_{s'} \max_{a' \in \mathcal{A}} Q_m^*(h'ao^{s'}, a') \sum_{o|\kappa^g(h'ao)=s'} O(o|h', a) \\
 &\stackrel{(6)}{=} E_{R(r|h',a)}(r) + \gamma \sum_o O(o|h', a) \max_{a' \in \mathcal{A}} Q_m^*(h'ao, a') \stackrel{(7)}{=} Q_{m+1}^*(h', a)
 \end{aligned}$$

(1) follows by definition. The first term in (2) follows as $R(r|h, a) = R(r|h', a)$ whenever $\kappa^g(h) = \kappa^g(h')$; the second term because κ^g partitions the history space. In (3), $o^{s'}$ is an arbitrary element of $B := \{o \mid \kappa^g(hao) = s'\}$. So for each $o \in B$, by the inductive hypothesis, $Q_m^*(hao, a') = Q_m^*(hao^{s'}, a')$ for all $a' \in \mathcal{A}$. So we can replace all the $\arg \max_{a'} Q_m^*(hao, a')$ by $\arg \max_{a'} Q_m^*(hao^{s'}, a')$ and then bring that term out. (4) is yet another application of the inductive hypothesis and the fact that $\kappa^g(hao^{s'}) = \kappa^g(h'ao^{s'})$ due to $\kappa(h) = \kappa(h')$. (5) follows because $\kappa^g(h) = \kappa^g(h') = s$ and hence both h, h' satisfy (4). (6) is the argument for (3) applied in reverse. And finally, (7) is by definition. This completes the proof of the first part.

For the second part of the theorem, let the optimal \mathcal{G} policy from the first part be μ . Then we claim that the \mathcal{M}^g policy ρ , obtained from μ by the first part of Theorem 3, is an optimal \mathcal{M}^g policy. To see this, let $\bar{\rho}$ be an optimal \mathcal{M}^g policy and let $\bar{\mu}$ be the corresponding \mathcal{G} policy from the second part of Theorem 3. We need to show that $V^\rho(s) = V^{\bar{\rho}}(s)$. By Theorem 3, for any h and s such that $\kappa^g(h) = s$, we have $V^\rho(s) \leq V^{\bar{\rho}}(s) = V^{\bar{\mu}}(h) \leq V^\mu(h) = V^\rho(s)$. ■

Appendix B. Proofs for Section 3

B.1 Proof of Type-1 Consistency (Reward Distribution)

The proofs in this subsection are a simple consequence of the definition of lim sup and the conditionally-deterministic nature of the transitions in the model. We assume the context of Section 3, in particular

that all actions are selected using a fixed policy π , and that we are given data $\bar{a}\bar{r}\bar{o}_{0:v}$. We prove each theorem in turn.

B.1.1 PROOF OF THEOREM 6

First some notation:

Definition 18 We say $\hat{r}\hat{o}_{0:m+k}$, $0 \leq k \leq \infty$ is an extension of $ro_{0:m}$ if $\hat{r}\hat{o}_{0:m} = ro_{0:m}$.

Now a definition:

Definition 19 For $n > v$, we define for the observed data $\bar{a}\bar{r}\bar{o}_{0:v}$ the following quantity:

$$EC_n(\vec{\theta}^g, \xi^g | \vec{\theta}, \xi) \triangleq \sum_{ro_{0:n} | ro_{0:v} = \bar{r}\bar{o}_{0:v}} RO_\pi(ro_{0:n}) \log \frac{Pr(r_{0:n} | ao_{0:n}, \vec{\theta}^g, \xi^g)}{Pr(r_{0:n} | ao_{0:n}, \vec{\theta}, \xi)}$$

We need two non-trivial lemmas for our theorem – here is the first one:

Lemma 20 We can write:

$$EC_n(\vec{\theta}^g, \xi^g | \vec{\theta}, \xi) = K_v + \sum_{j=v+1}^n \sum_{ro_{<j} | ro_{0:v} = \bar{r}\bar{o}_{0:v}} RO_\pi(ro_{<j}) KL[\vec{\theta}_{\kappa^g(ao_{<j}), a_j}^g || \vec{\theta}_{\kappa(ao_{<j}), a_j}] \quad (15)$$

where $K_v > -\infty$ is a constant depending only on $\bar{r}\bar{o}_{0:v}$.

Proof Now we have the following equalities giving us the lemma:

$$\begin{aligned} EC_n(\vec{\theta}^g, \xi^g | \vec{\theta}, \xi) &= \sum_{ro_{0:n} | ro_{0:v} = \bar{r}\bar{o}_{0:v}} RO_\pi(ro_{0:n}) \log \frac{Pr(r_{0:n} | ao_{0:n}, \vec{\theta}^g, \xi^g)}{Pr(r_{0:n} | ao_{0:n}, \vec{\theta}, \xi)} \\ &\stackrel{(0)}{=} \sum_{ro_{0:n} | ro_{0:v} = \bar{r}\bar{o}_{0:v}} RO_\pi(ro_{0:n}) \sum_{j=0}^n \log \frac{\vec{\theta}_{\kappa^g(ao_{<j}), a_j}^g(r_j)}{\vec{\theta}_{\kappa(ao_{<j}), a_j}(r_j)} \\ &\stackrel{(1)}{=} \sum_{j=0}^n \sum_{ro_{0:n} | ro_{0:v} = \bar{r}\bar{o}_{0:v}} RO_\pi(ro_{0:n}) \log \frac{\vec{\theta}_{\kappa^g(ao_{<j}), a_j}^g(r_j)}{\vec{\theta}_{\kappa(ao_{<j}), a_j}(r_j)} \\ &\stackrel{(2)}{=} K_v + \sum_{j=v+1}^n \sum_{ro_{0:n} | ro_{0:v} = \bar{r}\bar{o}_{0:v}} RO_\pi(ro_{0:n}) \log \frac{\vec{\theta}_{\kappa^g(ao_{<j}), a_j}^g(r_j)}{\vec{\theta}_{\kappa(ao_{<j}), a_j}(r_j)} \\ &\stackrel{(3)}{=} K_v + \sum_{j=v+1}^n \sum_{ro_{0:j} | ro_{0:v} = \bar{r}\bar{o}_{0:v}} RO_\pi(ro_{0:j}) \log \frac{\vec{\theta}_{\kappa^g(ao_{<j}), a_j}^g(r_j)}{\vec{\theta}_{\kappa(ao_{<j}), a_j}(r_j)} \\ &\stackrel{(4)}{=} K_v + \sum_{j=v+1}^n \sum_{ro_{<j} | ro_{0:v} = \bar{r}\bar{o}_{0:v}} RO_\pi(ro_{<j}) \sum_{ro_j} RO(ro_j | ao_{<j}, a_j) \log \frac{\vec{\theta}_{\kappa^g(ao_{<j}), a_j}^g(r_j)}{\vec{\theta}_{\kappa(ao_{<j}), a_j}(r_j)} \\ &\stackrel{(5)}{=} K_v + \sum_{j=v+1}^n \sum_{ro_{<j} | ro_{0:v} = \bar{r}\bar{o}_{0:v}} RO_\pi(ro_{<j}) \sum_{r_j} R(r_j | ao_{<j}, a_j) \log \frac{\vec{\theta}_{\kappa^g(ao_{<j}), a_j}^g(r_j)}{\vec{\theta}_{\kappa(ao_{<j}), a_j}(r_j)} \end{aligned}$$

$$\stackrel{(6)}{=} K_v + \sum_{j=v+1}^n \sum_{ro_{<j} | ro_{0:v} = \bar{r}\bar{o}_{0:v}} RO_{\pi}(ro_{<j}) KL[\vec{\theta}_{\kappa^g(ao_{<j}), a_j}^g || \vec{\theta}_{\kappa(ao_{<j}), a_j}]$$

(⁰) follows by chain rule of probability and definition of the log function. In (¹) we simply switched the sums and in (²) we introduced the term $K_v > -\infty$ for the terms in the outer sum prior to $v + 1$. In (³) we marginalized out the terms $ro_{j+1:n}$ because the log terms depend only on $ao_{0:j}$ and because, by definition of cylinder sets, $ro_{j+1:n}$ varies over all possible values in $(\mathcal{R} \times \mathcal{O})^{n-j}$. (⁴) is an application of the chain rule of probability. In (⁵) we marginalized out the o_j terms because the log terms do not depend on the o_j and, by definition of cylinder sets, o_j varies over all possible values in \mathcal{O} . Finally, (⁶) follows by definition of KL divergence and because, again by the definition of cylinder sets, r_j varies over all possible values in \mathcal{R} . \blacksquare

Our second non-trivial lemma gives a bound on EC_n defined above:

Lemma 21 *If $C(\bar{r}\bar{o}_{0:v}) \subset A_{\pi, \epsilon}$, for any $L \in \mathbb{N}$ we have that for some finite m_L ,*

$$EC_{m_L}(\vec{\theta}^g, \xi^g | \vec{\theta}, \xi) \geq K_v + \frac{L\epsilon}{2} RO_{\pi}(\bar{r}\bar{o}_{0:v})$$

where K_v was defined in Lemma 20.

Proof For convenience, we set $\alpha \triangleq RO_{\pi}[C(\bar{r}\bar{o}_{0:v})]$. Now, since $C(\bar{r}\bar{o}_{0:v}) \subset A_{\pi, \epsilon}$, for each $ro_{0:\infty} \in C(\bar{r}\bar{o}_{0:v})$ there exists an infinite sequence $1 \leq n_1 < n_2 \dots$ such that $C(ro_{0:v+n_i}) \subset A_{\pi, \epsilon, v+n_i}$. Let $N(ro_{0:v+n})$ be the set of numbers n_i in the prefix $ro_{0:v+n}$ of $ro_{0:\infty}$. Define

$$M_{L,n} := \{ro_{0:\infty} | ro_{0:v} = \bar{r}\bar{o}_{0:v}, |N(ro_{0:v+n})| \geq L\}$$

$M_{L,n} \uparrow C(\bar{r}\bar{o}_{0:v})$ and this implies that $\lim_{n \rightarrow \infty} RO_{\pi}(M_{L,n}) = \alpha$. Now set m_L to be the smallest number such that $RO_{\pi}(M_{L,m_L}) \geq \alpha/2$. We now prove an intermediate proposition:

Proposition 22

$$\sum_{ro_{0:v+m_L} \in Z_L} \sum_{n_i \in N(ro_{0:v+m_L})} RO_{\pi}(ro_{0:v+n_i}) \geq \frac{\alpha}{2} L \quad (16)$$

where Z_L are the set of $v + m_L + 1$ length prefixes of elements of M_{L,m_L} .

Proof First, observe that for any $ro_{0:v+m_L} \in Z_L$, as $|N(ro_{0:v+m_L})| \geq L$, for any $k < L$, there exists a proper suffix $ro_{0:v+n_i(k)}$ such that $|N(ro_{0:v+n_i(k)})| = k$. By definition of cylinder sets $RO_{\pi}(ro_{0:v+m_L}) \leq RO_{\pi}(ro_{0:v+n_i(k)})$, which implies

$$\sum_{ro_{0:v+m_L} \in Z_L} RO_{\pi}(ro_{0:v+n_i(k)}) \geq \sum_{ro_{0:v+m_L} \in Z_L} RO_{\pi}(ro_{0:v+m_L}) = RO_{\pi}(M_{L,m_L}) \geq \frac{\alpha}{2}$$

Observe that $n_i(k) \neq n_i(k')$ for $k \neq k'$ and $n_i(k), n_i(k') \in N(ro_{0:v+m_L})$. Therefore

$$\sum_{k=1}^L \sum_{ro_{0:v+m_L} \in Z_L} RO_{\pi}(ro_{0:v+n_i(k)}) \geq \frac{\alpha}{2} L$$

By construction, the set of strings we sum over in the left hand side of (16) contains the set of strings we sum over in the left hand side of the above inequality; hence this proves the inequality (16). ■

To complete the proof of this lemma, we start by rewriting the statement of Lemma 20 as:

$$\text{EC}_{m_L}(\vec{\theta}^g, \xi^g | \vec{\theta}, \xi) = K_v + \sum_{j=v+1}^{m_L} \sum_{C(ro_{<j}) \subset C(\bar{r}\bar{o}_{0:v})} \text{RO}_\pi(ro_{<j}) \text{KL}[\vec{\theta}_{\kappa^g(ao_{<j}), a_j}^g || \vec{\theta}_{\kappa(ao_{<j}), a_j}]$$

Now setting $D_{j-1} \triangleq C(\bar{r}\bar{o}_{0:v}) \cap A_{\pi, \epsilon, j-1}$ we can write that the above is:

$$\begin{aligned} &\stackrel{(0)}{\geq} K_v + \sum_{j=v+1}^{m_L} \sum_{ro_{<j} | C(ro_{<j}) \subset D_{j-1}} \text{RO}_\pi(ro_{<j}) \epsilon \\ &\stackrel{(1)}{=} K_v + \sum_{ro_{0:v+m_L} | ro_{0:v} = \bar{r}\bar{o}_{0:v}} \sum_{\substack{ro_{<j} | C(ro_{<j}) \subset A_{\pi, \epsilon, j-1} \\ v < j \leq m_L}} \text{RO}_\pi(ro_{<j}) \epsilon \\ &\stackrel{(2)}{\geq} K_v + \frac{\alpha}{2} L \epsilon \end{aligned}$$

In the above, ⁽⁰⁾ follows by definition of $A_{\pi, \epsilon, j-1}$ and because $D_{j-1} \subset C(\bar{r}\bar{o}_{0:v})$. ⁽¹⁾ is a rewrite of ⁽⁰⁾. Finally, ⁽²⁾ follows from (16) as each $C(ro_{0:v+n_i}) \subset A_{\pi, \epsilon, j-1}$ for some $j-1$ by definition of n_i . ■

We can now prove Theorem 6:

Proof [Proof of Theorem 6] We have

$$\lim_{n \rightarrow \infty} \text{EC}_n(\vec{\theta}^g, \xi^g | \vec{\theta}, \xi) = \infty \quad (17)$$

because in Lemma 21, we can take the right hand side to ∞ as a m_L exists for each possible L . So (17) shows that the expectation of the log of the ratio in the theorem statement diverges. The theorem now follows by arguments similar (but not identical) to those used to show that convergence in expectation implies convergence in probability.

First define for any $C(\hat{r}\hat{o}_{0:k}) \subset C(\bar{r}\bar{o}_{0:v})$ and $n > k$,

$$\text{EC}_n^{C(\hat{r}\hat{o}_{0:k})}(\vec{\theta}^g, \xi^g | \vec{\theta}, \xi) \triangleq \sum_{ro_{0:n} | ro_{0:k} = \hat{r}\hat{o}_{0:k}} \text{RO}_\pi(ro_{0:n}) \log \frac{\text{Pr}(r_{0:n} | ao_{0:n}, \vec{\theta}^g, \xi^g)}{\text{Pr}(r_{0:n} | ao_{0:n}, \vec{\theta}, \xi)}.$$

Clearly, if $\text{RO}_\pi(C(\hat{r}\hat{o}_{0:k})) > 0$, then

$$\lim_{n \rightarrow \infty} \text{EC}_n^{C(\hat{r}\hat{o}_{0:k})}(\vec{\theta}^g, \xi^g | \vec{\theta}, \xi) = \infty \quad (18)$$

by the same arguments used to derive (17).

Now, let

$$E_n^M \triangleq \left\{ ro_{0:\infty} \mid ro_{0:v} = \bar{r}\bar{o}_{0:v}, \log \frac{\text{Pr}(r_{0:v+n} | ao_{0:v+n}, \vec{\theta}^g, \xi^g)}{\text{Pr}(r_{0:v+n} | ao_{0:v+n}, \vec{\theta}, \xi)} \leq M \right\}$$

The theorem statement asserts that for each M , $1_{E_n^M} = 0$ in probability where $1_{E_n^M}$ is the indicator function of E_n^M . That is, $\forall M, \epsilon > 0, \exists N^\epsilon$ such that $n > N^\epsilon$ implies $RO_\pi(E_n^M) < \epsilon$. By definition of E_n^M this means that if $n > N^\epsilon$ and $C(ro_{0:v+n}) \subset C(\bar{r}\bar{o}_{0:v})$, then

$$\log \frac{Pr(r_{0:v+n}|ao_{0:v+n}, \vec{\theta}^g, \xi^g)}{Pr(r_{0:v+n}|ao_{0:v+n}, \vec{\theta}, \xi)} \leq M \text{ implies } RO_\pi[C(ro_{0:v+n})] < \epsilon$$

So if the theorem is false then there exists some $M', \epsilon' > 0$ and $\hat{r}\hat{o}_{0:v+N}, C(\hat{r}\hat{o}_{0:v+N}) \subset C(\bar{r}\bar{o}_{0:v})$, such that

$$RO_\pi(C(\hat{r}\hat{o}_{0:v+N})) > \epsilon' \text{ and } \log \frac{Pr(r_{0:v+N+m}|ao_{0:v+N+m}, \vec{\theta}^g, \xi^g)}{Pr(r_{0:v+N+m}|ao_{0:v+N+m}, \vec{\theta}, \xi)} \leq M'$$

for all $m \in \mathbb{N}$ and $C(ro_{0:v+N+m}) \subset C(\hat{r}\hat{o}_{0:v+N})$. This in turn implies that $EC_n^{C(\hat{r}\hat{o}_{0:v+N})}(\vec{\theta}^g, \xi^g | \vec{\theta}, \xi)$ remains bounded as $n \rightarrow \infty$ as it is the expectation of the log ratio. This contradicts (18) and hence the first part of the theorem is proved.

For the second part of the theorem, the rewrite in Lemma 21 implies that $\lim_{n \rightarrow \infty} EC_n(\vec{\theta}^g, \xi^g | \vec{\theta}, \xi)$ is now bounded. Hence, the ratio is also bounded in probability as convergence in expectation implies convergence in probability. \blacksquare

B.1.2 PROOF OF THEOREM 8

Lemma 23 *Under the hypothesis of Theorem 8, for sequences $ro_{0:\infty} \in C(\bar{r}\bar{o}_{0:v})$ we have*

$$\lim_{n \rightarrow \infty} \frac{Pr(r_{0:v+n}|ao_{0:v+n}, \vec{\theta}^g, \xi^g)}{Pr(r_{0:v+n}|ao_{0:v+n}, \xi)} = \infty$$

in RO_π probability.

Proof (Sketch) The proof is analogous to the proof for Theorem 6 and hence not repeated. \blacksquare

Proof [Proof of Theorem 8] By the law of large numbers $Pr(r|aro_{0:n}, s, a, \xi^g)$ converges almost surely to $\vec{\theta}_{s,a}^g$ in total variation whenever $sa \in C(\bar{r}\bar{o}_{0:v})$ (see Definition 9). Furthermore, by definition, for DMM ξ ,

$$\log Pr(r_{0:n}|ao_{0:n}, \xi) = \sum_i \log Pr(r_i|aro_{<i}, s_{i-1}, a_i, \xi)$$

or in other words,

$$\begin{aligned} \log \frac{Pr(r_{0:v+n}|ao_{0:v+n}, \xi^g)}{Pr(r_{0:v+n}|ao_{0:v+n}, \xi)} &= \sum_i \left[\log Pr(r_i|aro_{<i}, s_{i-1}^g, a_i, \xi^g) - \log \vec{\theta}_{s_{i-1}^g, a_i}^g(r_i) \right] \\ &\quad + \sum_i \left[\log \vec{\theta}_{s_{i-1}, a_i}^g(r_i) - \log Pr(r_i|aro_{<i}, s_{i-1}, a_i, \xi) \right] \end{aligned}$$

Since the number of states and actions are finite, for $ro_{0:\infty} \in C(\bar{r}\bar{o}_{0:v})$, as $n \rightarrow \infty$ the terms in the first sum converges (as mentioned above) almost surely to 0 and hence the sum converges to a bounded constant; the second sum goes to infinity in probability by Lemma 23. All together this implies the theorem.

The second part of the theorem follows by arguments analogous to the ones for the second part of Theorem 6. ■

B.2 Proof of State Transition Distribution Consistency (Type 2)

Lemma 24 For any ξ and $s \in \mathcal{S}, s' \in \mathcal{S}^g$, if $(s, s') \in^{\equiv} C(\bar{r}\bar{o}_{0:v})$, then for all other $q' \in \mathcal{S}^g$ $(s, q') \notin^{\equiv} C(\bar{r}\bar{o}_{0:v})$.

Proof By definition of \in^{\equiv} . ■

Lemma 25 If a state s of ξ models $s^g \in \mathcal{S}^g$ in $C(\bar{r}\bar{o}_{0:v})$ (that is, $(s, s^g) \in^{\equiv} C(\bar{r}\bar{o}_{0:v})$), then for each action a such that $sa \in^{\infty} C(\bar{r}\bar{o}_{0:v})$, for sequences $ro_{0:\infty} \in C(\bar{r}\bar{o}_{0:v})$, $Pr(s'|ar_{0:n}, s, a, \xi)$ converges RO_{π} almost surely to $\vec{\phi}_{s^g, a}^g$ in total variation as $n \rightarrow \infty$.

Proof Assume that $a \in \mathcal{A}$ and $s \in \mathcal{S}, sa \in^{\infty} C(\bar{r}\bar{o}_{0:v})$ and that $s \in \mathcal{S}$ models s^g . Then for all but a finite number of histories h with $C(h) \subset C(\bar{r}\bar{o}_{0:v})$, $\kappa(h) = s$ implies $\kappa^g(h) = s^g$. Therefore the required almost sure convergence happens via (12) by the law of large numbers. ■

Proof [Proof of Theorem 13] Assume that the almost sure convergence does not happen for some action a such that $sa \in^{\infty} C(\bar{r}\bar{o}_{0:v})$. Then by lemmas 24 and 25 s corresponds to (without loss of generality) two states $s', s'' \in \mathcal{S}^g$ infinitely often – that is for all but a finite number of elements of $h \in \bar{Z}_{\pi, s}(\bar{r}\bar{o}_{0:v})$ (see Definition 10) satisfy $\kappa^g(h) = s'$ or $\kappa^g(h) = s''$, both infinitely often, and $s', s'' \in^{\infty} C(\bar{r}\bar{o}_{0:v})$. We will refer to this condition as s being *confused* w.r.t. s', s'' .

As $s', s'' \in^{\infty} C(\bar{r}\bar{o}_{0:v})$, let $\hat{h} \in \mathcal{H}$ be the corresponding sequence for s' and s'' from Assumption 12 and let $\delta^g(s', \hat{h}) = q'$ and $\delta^g(s'', \hat{h}) = q''$. So $\vec{\theta}_{q', \hat{a}}^g \neq \vec{\theta}_{q'', \hat{a}}^g$ and $\hat{a}q'', \hat{a}q' \in^{\infty} C(\bar{r}\bar{o}_{0:v})$ for some $\hat{a} \in \mathcal{A}$. By determinism there exists a single state q such that $\delta(s, \hat{h}) = q$ and $q \in^{\infty} C(\bar{r}\bar{o}_{0:v})$ as \hat{h} occurs infinitely often after s', s'' by Assumption 12. Hence q is confused w.r.t. q', q'' .

Since q corresponds to states q' and q'' both infinitely often and since $\vec{\theta}_{q', \hat{a}}^g \neq \vec{\theta}_{q'', \hat{a}}^g$, it means that $C(\bar{r}\bar{o}_{0:v}) \subset A_{\pi, \epsilon}$ for some $\epsilon > 0^4$. By Theorem 8, this means that the assumed boundedness of the ratio (14) is false and so this is a contradiction of hypothesis of this theorem. This implies that the almost sure convergence takes place. ■

References

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

4. In fact, $\epsilon \geq \frac{1}{2} \|\vec{\theta}_{q', \hat{a}}^g - \vec{\theta}_{q'', \hat{a}}^g\|_2^2$.

- Dimitri P. Bertsekas and Steven E. Shreve. *Stochastic Optimal Control: The Discrete-Time Case*. Athena Scientific, paperback edition, 1996.
- Byron Boots, Geoff Gordon, and Sajid Siddiqi. Closing the learning-planning loop with predictive state representations (extended abstract). In *Proceedings of the 9th International Conference Autonomous Agents and Multiagent Systems (AAMAS)*, 2010.
- Hugh A. Chipman, Edward I. George, and Robert E. McCulloh. Bayesian CART model search. *Journal of the American Statistical Association*, 93:935 – 948, 1998.
- Lonnie Chrisman. Reinforcement learning with perceptual aliasing: The perceptual distinctions approach. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 183–188, 1992.
- Finale Doshi-Velez. The infinite partially observable markov decision process. In *Proc. of the 20th Neural Information Processing Systems Conference*, 2009.
- Eyal Even-Dar, Sham M. Kakade, and Yishay Mansour. Reinforcement learning in POMDPs without reset. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, 2005.
- Vivek F. Farias, Ciamac C. Moallemi, Benjamin Van Roy, and Tsachy Weissman. Universal reinforcement learning. *IEEE Transactions on Information Theory*, 56(5):2441–2454, 2010.
- Michael P. Holmes and Charles L. Isbell-Jr. Looping suffix tree-based inference of partially observable hidden state. In *Proceedings of the 23^rd International Conference on Machine Learning*, 2006.
- John E. Hopcroft, Rajiv Motwani, and Jeffrey D. Ullman. *Introduction Automata Theory, Languages and Computation*. Addison Wesley, 3rd edition, 2006.
- Marcus Hutter. Feature markov decision process. In *Proceedings of the 2nd AGI Conference*, 2009.
- Leslie P. Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101:99 – 134, 1998.
- Michael Littman, Richard Sutton, and Satindar S. Singh. Predictive representations of state. In *Proceedings of the 15th Neural Information Processing Systems Conference*, 2002.
- Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artificial Intelligence*, 147(1-2):5–34, 2003.
- M. M. Hassan Mahmud. Constructing states for reinforcement learning. In *Proceedings of the 27th International Conference on Machine Learning*, 2010.
- Andrew K. McCallum. *Reinforcement Learning with Selective Perception and Hidden State*. PhD thesis, University of Rochester, 1995a.
- Andrew K. McCallum. Instance-based utile distinctions for reinforcement learning. In *Proceedings of the 12th International Machine Learning Conference*, 1995b.

- Peter McCracken and Michael Bowling. Online discovery and learning of predictive state representations. In *Proceedings of the 18th Neural Information Processing Systems Conference*, 2005.
- Jorma Rissanen. A universal data compression system. *IEEE Transactions on Information Theory*, 29:656–664, 1983.
- Dana Ron, Yoram Singer, and Naftali Tishby. Learning probabilistic automata with variable memory length. In *Proceedings of the 7th Conference on Computational Learning Theory*, 1994.
- Matthew Rosencrantz, Geoff Gordon, and Sebastian Thrun. Learning low dimensional predictive representations. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.
- Stephane Ross, Brahim Chaib-draa, and Jolle Pineau. Bayes adaptive POMDP. In *Proceedings of the 20th Conference on Neural Information Processing Systems*, 2007.
- Stephane Ross, Joelle Pineau, Sebastian Paquet, and Brahim Chaib-draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32:663–704, 2008.
- C. Shalizi and K. Klinkner. Blind construction of optimal nonlinear recursive predictors for discrete sequences. In *Proc. of the 20th Conference on Uncertainty in Artificial Intelligence*, 2004.
- Guy Shani, Ronen I. Brafman, and Solmon E. Shimony. Resolving perceptual aliasing in the presence of noisy sensors. In *Proceedings of the 17th Neural Information Processing Systems Conference*, 2004.
- Guy Shani, Ronen I. Brafman, and Solmon E. Shimony. Model-based online learning of POMDPs. In *Proceedings of the 15th European Conference on Machine Learning*, 2005.
- Satinder Singh, M. R. James, and M. R. Rudary. Predictive state representations: A new theory for modeling dynamical systems. In *Uncertainty in Artificial Intelligence*, 2004.
- Edward J. Sondik. *The optimal control of partially observable Markov processes*. PhD thesis, Stanford University, 1971.
- Peter Sunehag and Marcus Hutter. Consistency of feature markov processes. In *Proceedings of the 21st International Conference on Algorithmic Learning Theory*, 2010.
- Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.
- Enrique Vidal, Frank Thollard, Colin de la Higuera, Francisco Casacuberta, and Rafael C. Carrasco. Probabilistic finite-state machines part I. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1013–1025, 2005.
- Britton Wolfe, Michael R. James, and Satinder Singh. Learning predictive state representations in dynamical systems without reset. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.