Bayesian Nonparametric Methods for Partially-Observable Reinforcement Learning

Finale Doshi-Velez, David Pfau, Frank Wood, and Nicholas Roy, *Member, IEEE*

Abstract—Making intelligent decisions from incomplete information is critical in many applications: for example, robots must choose actions based on imperfect sensors, and speech-based interfaces must infer a user's needs from noisy microphone inputs. What makes these tasks hard is that often we do not have a natural representation with which to model the domain and use for choosing actions; we must learn about the domain's properties while simultaneously performing the task. Learning a representation also involves trade-offs between modeling the data that we have seen previously and being able to make predictions about new data. This article explores learning representations of stochastic systems using Bayesian nonparametric statistics. Bayesian nonparametric methods allow the sophistication of a representation to scale gracefully with the complexity in the data. Our main contribution is a careful empirical evaluation of how representations learned using Bayesian nonparametric methods compare to other standard learning approaches, especially in support of planning and control. We show that the Bayesian aspects of the methods result in achieving state-of-the-art performance in decision making with relatively few samples, while the nonparametric aspects often result in fewer computations. These results hold across a variety of different techniques for choosing actions given a representation.

Index Terms—Artificial intelligence, machine learning, reinforcement learning, partially-observable Markov decision process, hierarchial Dirichlet process hidden Markov model

1 Introduction

Many sequential decision-making problems occur in domains where the agent must choose actions with little or no information about how the world works. A robot without a prior map may be tasked with finding its way to a goal. Similarly, recommender systems must be able to make useful recommendations with no prior knowledge of the user. However, by using data acquired over time and refining the agent's predictions, the performance of the system can quickly improve. While searching for a path to a goal, the robot can incrementally build a map and improve its navigation performance. When making recommendations based on a limited purchasing history, recommender systems can start with items that are generally popular and then refine their knowledge about user's preferences.

Partially-observable reinforcement learning [1] formalizes this sequential decision-making problem as a series of exchanges between an agent and an environment (Fig. 1). At each time step, the agent interacts with the environment through an action a. The environment returns an observation o and a reward r. Given a history $h_t = \{a_0, o_0, r_0, ...a_t, o_t, r_t\}$ of length t, the agent's goal is to choose the next action a_{t+1} such that it will maximize the discounted sum of its expected rewards $E[\sum_t \gamma^t r_t]$, where $\gamma \in [0, 1)$ trades off current and future rewards.

A fundamental question in partially-observable reinforcement learning is how the agent should represent the knowledge that it derives from its interactions from the environment. Of course, the agent could simply just store the history h_t, but making decisions from such a high-dimensional variable is challenging, and most approaches compact histories into a lower-dimensional knowledge representation. These compact "models" of the environment can then be used by decison-making algorithms to choose future actions. However, the "right" knowledge representation is not always clear: for example, can all the information about a patient be summarized by their symptoms? Their genetic sequence?

We argue that the "right" representation is ultimately the one that allows the agent to maximize the discounted sum of its expected rewards $E[\sum_t \gamma^t r_t]$ (combined with an action-selection strategy). Specifically, let K_t be the agent's knowledge representation at any time t and define a function $K_{t+1} = f(K_t, a_t, o_t, r_t)$ that describes how to transform the knowledge representation given new data (a_t, o_t, r_t). Paired with a policy $a_{t+1} = \pi(K_t)$, another function which describes what action to take given the knowledge representation, we can then empirically evaluate our objective $E[\sum_t \gamma^t r_t]$.

We use Bayesian nonparametric methods to address this fundamental question of representation choice. We divide the knowledge representation for the reinforcement-learning problem into two parts $K_t = (\mu, \xi_t)$: a static part μ and a dynamic part $\xi_t \in \Xi$ that changes with each action

- *F. Doshi-Velez and N. Roy are with the Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge MA 02139 USA. E-mail: finale@alum.mit.edu; nickroy@mit.edu.*
- *D. Pfau is with the Center for Theoretical Neuroscience, Columbia University, New York, NY 10032 USA. E-mail: pfau@neurotheory.columbia.edu.*
- *F. Wood is with the Department of Engineering, University of Oxford, Oxford, OX1 3PJ, U.K. E-mail: fwood@robots.ox.ac.uk.*

Manuscript received 14 Sep. 2012; revised 2 May 2013; accepted 10 Sep. 2013. Date of publication 30 Sep. 2013; date of current version 14 Jan. 2015. Recommended for acceptance by Y. W. Teh.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier 10.1109/TPAMI.2013.191

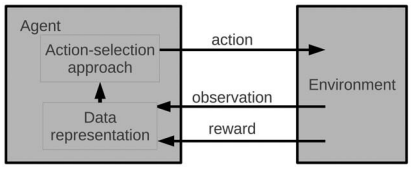

Fig. 1. Reinforcement learning framework. At each time-step, the agent sends an action to the environment and receives an observation and reward.

a_t. We can think of the static part μ as a set of parameters that encode a set of rules of how the environment responds in different situations, and think of the dynamic part ξ_t as our current situation.

While the parameters μ are static, in most real-world systems, they are rarely known exactly and must be inferred from data (such as via system identification). Even worse, it is not always clear what are the "right" representations of μ and Ξ. An overly simple representation may not allow the agent to understand the effect of its actions. An overly complex representation may require a large amount of data to train.

Unlike more traditional approaches to learning in decision-making problems, which often try to recover the parameters of some assumed "actual" environment [2]–[7], our emphasis is on inferring a representation (μ, ξ) from data that simply allows us to make good predictions. Specifically, we posit that an infinite number of parameters μ are required to completely describe the effect of every possible action a in every possible situation ξ. Thus, unlike with parametric approaches, we never have to worry whether our representation is "large enough" to explain the data. However, given a finite set of interactions with the environment, Bayesian nonparametric approaches guarantee that the agent can only gather information about a finite number of these parameters. Thus, the agent never has to perform inference over an infinite number of parameters. Drawing on concepts from [8], [9], and [10], Bayesian nonparametric methods allow us to think of our representation not as an approximation of the physical environment, but as information needed to make the underlying system Markovian. By focusing our efforts on prediction, we ignore aspects of the "actual" environment that are irrelevant for our sequential decision-making problem.

In this article, we present two Bayesian nonparametric representations for partially-observable reinforcement learning. The infinite partially-observable Markov decision process (iPOMDP)[1] is a hidden-state representation in which the observations and rewards are emitted from a hidden state that evolves based on the agent's actions [12]. In contrast, the states in our infinite deterministic Markov model (iDMM) transition deterministically given the most recent action and observation.

Our main contribution is a careful empirical comparison of the iPOMDP and the iDMM to other standard approaches to partially-observable reinforcement learning on a series of benchmark domains. These models had different expected strengths. Robust inference methods exist for Bayesian nonparametric models over hidden state representations. In contrast, history-based representations require no latent variables, potentially providing more efficient learning.

We compare our approaches to Bayesian parametric approaches, non-Bayesian parametric approaches, and non-Bayesian nonparametric approaches. We find that the Bayesian aspect of the iPOMDP results in strong performance with relatively few samples, while the nonparametric aspect often result in fewer computations. These results hold across a variety of different techniques to choose actions given a representation. Finally, although iDMM-like models have outperformed HMM-like models on prediction tasks [13], we find that the iPOMDP is superior to the iDMM in all of our experimental settings, leaving interesting questions for future work on appropriate choices of knowledge representations in reinforcement learning settings.

2 Representations for Partially-Observable Reinforcement Learning

Given a history $h_t = \{a_0, o_0, r_0, ...a_t, o_t, r_t\}$ of length t, the problem we address is choosing the next action a_{t+1} such that it will maximize the discounted sum of its expected rewards $E[\sum_t \gamma^t r_t]$. Our goal is to choose a representation Ξ and μ that allows us to infer the current value of ξ_t, the parameters of μ from the history. By choosing the representation and inferring μ, we can predict the effect of actions given the current state ξ_t and compute a policy that maximizes the objective function.

Following the definition of state described in [9], [14], we define the *information* state $\xi_t = g(h_t)$ as any statistic of the history h_t which is sufficient to predict the distribution of future rewards and observations. More formally, let f_t be the random variable representing the agent's future after time t: $\{a_{t+1}, o_{t+1}, r_{t+1}, ...\}$. If the state $\xi_t = g(h_t)$ is a sufficient statistic for the history h_t, then the conditional probability of the future conditioned on the state and history does not depend on the history at all: $Pr(f_t|\xi_t, h_t) = Pr(f_t|\xi_t)$, and if two histories h_t and h'_t have the same statistic ξ, then $Pr(f_t|h_t) = Pr(f_t|h'_t)$. (Equivalently, the mutual information between the current and future states is equal to the mutual information between past and future: $I(h_t; f_t) = I(g(h_t), f_t)$.) This information state is known to be sufficient for making optimal decisions [15].

Approaches for inferring a representation Ξ of the information state and the corresponding prediction representation μ fall into two broad categories: implicit representations using hidden variables and explicit representations using features of histories. In this section, we provide an overview of methods in each category; Sections 3 and 4 present Bayesian nonparametric versions of each representation.

2.1 States Using Hidden Variables

Hidden-variable approaches to defining state introduce a set of hidden variables s that, if known, would themselves

1. The abbreviation iPOMDP is consistent with abbreviations for other infinite models such as the infinite HMM (iHMM). However, the iPOMDP model should not be confused with the interactive POMDP [11], which has the same abbreviation.

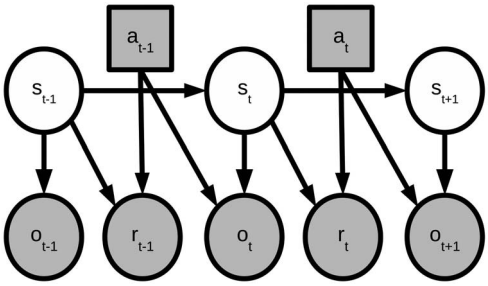

Fig. 2. Graphical model showing a time-slice of a POMDP. The world-state s_t is a hidden variable.

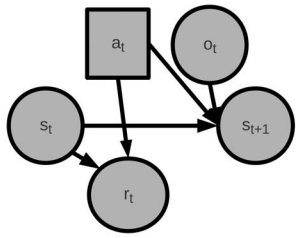

Fig. 3. Graphical model showing a time-slice of the DMM. Note that variables are observed at all times.

be a sufficient statistic for the history[2]. Then, the distribution over possible world-states $\xi_t = b_t(s) = p(s|h_t)$ is a sufficient statistic for the history [16]. Historically, hidden-variable representations are derived from application areas in which it was natural to think of the agent being in some "true" world-state that was made ambiguous by inadequate sensing [16]–[19].

The partially-observable Markov decision process (POMDP) provides a very general formulation of this hidden-variable approach (Fig. 2). A POMDP is a 7-tuple $\{T, \Omega, R, A, S, O, \gamma\}$ where S, A, O are sets of world-states, actions, and observations, respectively; the transition function $T(s'|s, a)$ is the probability of the next world-state s' given the current world-state s and action a; the observation function $\Omega(o|s, a)$ is the probability of receiving observation o after taking action a and arriving at state s; the reward function $r = R(s, a)$ is the immediate reward; and γ is the discount factor. Given the transition, observation, and reward parameters, a variety of sophisticated techniques exist for solving for the optimal policy $\pi^*(a|b_t(s))$ [20]–[22]. A fixed choice of S determines the representation $\xi = p(s|h_t)$; many partially observable techniques assume that S (and hence Ξ) is known ahead of time, focusing on learning the corresponding R, A and O defining μ.

Learning POMDPs is a hard problem [23], but a variety of methods exist for inferring the transition, observation, and reward functions. Parametric methods such as expectation-maximization [24], Bayesian model learning [2]–[4], [25]–[27], and system identification [28] all assume that the number of world-states is known, but the transition, observation, and reward parameters are not. For example, when modeling a pendulum, one might know that the mass and length of the arm are the relevant parameters but not know their values. Parametric techniques allow expert knowledge about the underlying system to be encoded directly into the representation; in some cases parts of the representation may even be trained or calibrated separately.

However, in many applications, the concept of a "true" or "real" world-state S is ambiguous. For example, in a medical setting, the definition of a patient's state could range from a set of symptoms to their their complete molecular make-up. Our Bayesian nonparametric hidden-variable approaches circumvent this issue by first positing that the full state space S is infinite but then trying to explain finite history with as few visited-states as possible. The Bayesian aspect of these approaches also allows us to begin with informed expert biases but allows the agent's interactions with the environment to ultimately dictate what notion of states are useful for future predictions. In uncontrolled settings, spectral techniques have been used to learn hidden-variable models [29]. Bayesian nonparametric models, such as the hierarchical Dirichlet process hidden Markov model (HDP-HMM), and their precursors have been used to find predictive structure in a variety of dynamical systems [8]–[10], [30]–[33].

2.2 States from Features of Histories

Another approach to summarizing the history h_t into a statistic $\xi_t = g(h_t)$ is to directly search for certain patterns in that history: a sub-sequence of elements, a suffix, or some more complex feature. For example, knowing a customer bought the first two books in a trilogy may be sufficient for inferring whether to recommend the third book, regardless of his other purchasing history. A chess player's most recent moves may be sufficient to infer the her current strategy, regardless of her initial moves.

History-based approaches have the advantage of operating directly on the data. There are no hidden variables to to be defined, inferred, or controlled. Instead, states are simply aspects of the history that are useful for predicting future rewards or future observations. However, it is often harder to incorporate direct knowledge, such as a sensor calibration, into the representation. Furthermore, because histories are so general, large amounts of experience may be required to discover the aspects of the histories that are valuable for making decisions.

States as History Sub-sequences: Finite Automata. One approach to grouping sub-sequences of the history together is to use finite automata [34]. Finite automata consist of a set of nodes n; while the transitions between nodes may be deterministic or stochastic, the current node is always observed (see Fig. 3). The deterministic Markov model (DMM) is a specific finite automaton described by the 6-tuple $(S, A, \Omega, \delta, R, s_0)$, where S, A, and Ω are discrete sets of nodes, actions, and observations; the node s_0 is the initial node; δ:S x A x $\Omega \rightarrow S$ describes which node which follows the current state, action, and observation. The reward function $R(r|s, a)$ gives the probability of receiving reward r after doing action a in node s.

Because the nodes are observed, the information state ξ is simply the node-state s. Learning a DMM corresponds to finding reward distributions $R(r|s, a)$ and transition

2. We refer to the hidden "world-state" as s and the sufficient statistic for the history (the "information-state") as ξ.

mappings $s' = \delta(s, a, o)$ so that rewards can reliably be predicted. While also an NP-hard problem, several approximation techniques have been developed using PAC [35], information-theoretic [36], and Bayesian methods [13], [37]. Solving a DMM for an optimal policy is relatively simple since the nodes are observed.

By conditioning transitions on both actions and observations, the DMM decouples learning the transition model from learning the stochasticity of the observations. While finite automata are a strictly smaller class of models than POMDPs, they can be made arbitrary close to—and thus their nodes can be sufficient statistics for—any RL environment (similar to uncontrolled automata of [38]). In uncontrolled settings, simpler models—such as PDFAs rather than HMMs—have had superior performance because the are easier to learn [13], and recent work has suggested that this may also be true for the reinforcement learning setting [13], [37]. In Section 4, we extend the work of [13] to create an infinite deterministic Markov model which posits that there are an infinite number of possible nodes s.

States as History-Suffixes: U-Tree Another related approach to grouping sub-sequences of the history together is the U-Tree algorithm [39], which builds a suffix tree of the agent's histories $h_1, h_2, ...h_t$. Each branch of the tree is trimmed to a particular depth, corresponding to how large a window of recent recent history is relevant. The state $\xi = g(h_t)$ in U-Tree is the leaf-node to which the history h_t corresponds. If the depth of that leaf-node is d, then all histories h sharing the same d-length suffix will be grouped together into a single state.

Given a particular suffix tree, planning is simple: the agent always knows its current state $s = \xi$ (that is, the leaf-node associated with its history h), and that state is always a discrete scalar. By tracking how often it sees various state transitions $T(s'|s, a)$ and rewards $R(s, a)$, the agent can build a Markov decision process (MDP) model of the dynamics of these node states; choosing actions is simply a matter of solving the MDP. However, robustly learning the suffix tree is an area of continuing research [40]–[42]. Related approaches that use windows of history as state have also been applied to learning the dynamics of uncontrolled systems [9], [43].

3 The Infinite Partially-Observable Markov Decision Process

The infinite partially-observable Markov decision process (iPOMDP) is a Bayesian nonparametric hidden-state representation for reinforcement learning. A standard POMDP can be constructed from a hidden Markov model (HMM)—4-tuple $\{T, \Omega, S, O\}$ where S is the set of world-states, O is the set of observations, $T(s'|s)$ is the transition function, and $\Omega(o|s)$ is the observation function—by adding actions and rewards. The infinite POMDP is constructed from the HDP-HMM, a Bayesian nonparametric model that places a prior over HMMs with countably infinite discrete world-states. The time-dependent aspect of the knowledge representation ξ_t is therefore given as the posterior $p(m, s|h_t)$ over infinite POMDP models m and their internal world-states s.

We describe the iPOMDP prior with a generative model over infinite POMDPs. We take a Monte Carlo approach to inference, which means we approximate the posterior distribution by a set of samples. The distribution over states in the ensemble of samples at each time step represents the information state ξ_t. The transition functions, observation and reward functions in the samples give the distribution over μ.

We first describe the process of sampling the prior iPOMDP, and then the process of inferring the posterior given a history, and lastly the process of choosing actions given the inferred state ξ_t.

3.1 Model

The HDP-HMM has the following generative model:

$$\bar{T} \sim \text{GEM}(\lambda) \tag{1}$$

$$T(\cdot|s) \sim \text{DP}(\alpha, \bar{T}) \forall s \tag{2}$$

$$\Omega(\cdot|s) \sim H \; \forall s \tag{3}$$

where the GEM distribution is used to sample a mean transition distribution $\bar{T}$ with concentration parameter λ, the Dirichlet process DP is used to sample transition distributions $T(\cdot|s)$ from each world-state with concentration parameter α, and the base distribution H is a prior over observation distributions $\Omega(\cdot|s)$. For discrete observation spaces, H is often chosen to be Dirichlet distributions; however, the framework is general enough to allow for any choice of prior. The ensemble of states resulting from sampling $\bar{T}$ in equation 1 implicitly defines the space S.

The GEM distribution is a stick-breaking process that divides the unit interval into an infinite number of segments [44]. Under the GEM distribution, $\bar{T}(k)$, decays exponentially with k based on the concentration parameter λ. Small values of λ place a bias toward a very few popular world-states; large λ imply many world-states will be likely destinations of a transition. The concentration parameter α governs how closely $T(\cdot|s)$ matches $\bar{T}$. Large α will result in transition functions $T(\cdot|s, a)$ that look almost identical to $\bar{T}$. Small values will result in near-deterministic transition functions $T(\cdot|s, a)$ with mean $\bar{T}$.

The iPOMDP (Fig. 4) adds actions and rewards to the HDP-HMM. While the reward $r = R(s, a)$ is traditionally a deterministic function, we let $R(r|s, a)$ be the probability of receiving reward r after performing action a in world-state s. The full generative process is given by

$$\begin{aligned} \bar{T} &\sim \text{GEM}(\lambda) \\ T(\cdot|s, a) &\sim \text{DP}(\alpha, \bar{T}) \forall s, a \\ \Omega(\cdot|s, a) &\sim H \; \forall s, a \\ R(\cdot|s, a) &\sim H_R \; \forall s, a \end{aligned}$$

where the base distribution H_R is the prior over rewards $R(\cdot|s, a)$. While any choice of prior could be used for H_R, here we choose to discretize the rewards and use a Dirichlet distribution for the prior H_R. We fix the POMDP discount parameter $\gamma = .95$ as is standard in many RL applications.

3.2 Inference

By construction, our sufficient statistic ξ_t is the posterior distribution $p(s, m|h_t)$ over current world state s and the model

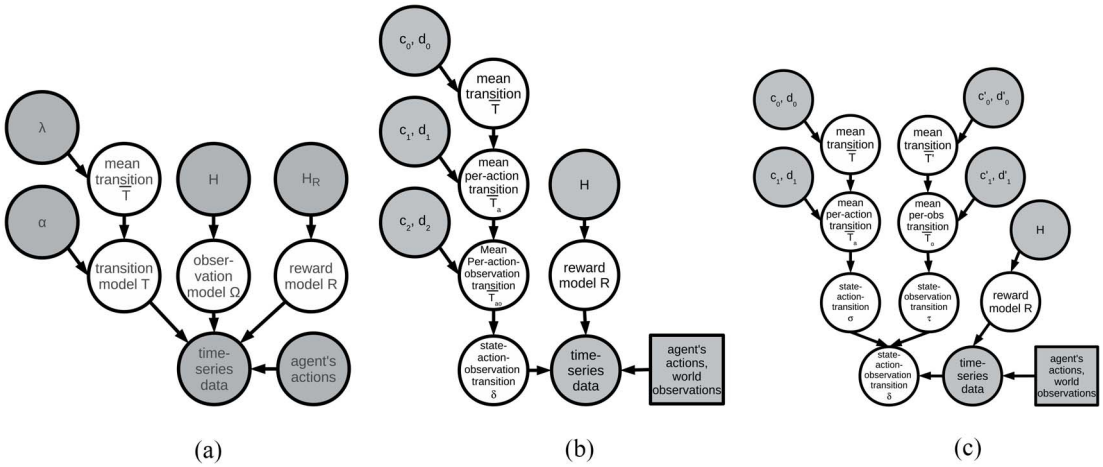

Fig. 4. Graphical models showing the generative process for the iPOMDP and the iDMM: (a) iPOMDP graphical model. (b) Asymmetric iDMM graphical model. (c) Symmetric iDMM graphical model.

m.[3] The joint posterior $\xi_t = p(s, m|h_t)$ cannot be represented in closed form. We factor the posterior into $p(s, m|h)$ into two components $p(s|m, h)p(m|h)$. We approximate the posterior over models $p(m|h)$ with a set of samples $\{m\}$. For each model $m = \{T, \Omega, R\}$, the distribution over world-states $p(s|m, h)$ can be computed exactly for discrete spaces using standard belief update equations [17]. Thus, we focus in this section on describing how to draw samples m from $p(m|h)$.

Updating $\{m\}$ through Importance Sampling. The generative process outlined in Section 3.1 describes how to draw samples from the prior over models m. More generally, suppose we have a set of models m that have been drawn from the correct posterior $p(m|h_t)$ at time t. To represent the posterior $p(m|h_{t+1})$, we can simply assign weights to set of models $\{m\}$ drawn at time t. The importance weight $w(m)$ on model m is:

$$w_{t+1}(m) \propto \Omega(o_t|m, a_t)R(r_t|m, a_t)w_t(m), \tag{4}$$

where $\Omega(o|m, a) = \sum_{s' \in S} \Omega(o|s', m, a)b_m(s')$, $R(r|m, a) = \sum_{s \in S} R(r|s, m, a)b_m(s)$, and, because the true model does not change, $T(m'|m, a) = \delta_m(m')$

The advantage of simply reweighting the samples is that the belief update is computationally fast. It is also an effective way of updating the posterior if we do not expect a large change to have happened: usually, a single round of experience from time t to $t + 1$ does not change the agent's beliefs about the model m significantly. However, over time, new experience may render all of the current model samples unlikely. Very small likelihood weights, which mean that none of the sampled models are likely, indicate that the predictions made by those models are poor and therefore unlikely to perform well in decision-making situations. Very uneven weights—a few likely models and many unlikely models—result in low effective sample sizes and make it difficult to estimate our uncertainty over models. To alleviate this issue, we periodically resample the models m_i from the posterior.

3. We will use the words *posterior* and *belief* interchangeably; both refer to the probability distribution over the hidden state given some initial belief (or *prior*) and the history of actions and observations.

Updating $\{m\}$ through Beam Sampling. We resample a new set of models directly from the current belief using the beam-sampling approach of [45], with minor adaptations to allow for observations with different temporal shifts (since the reward r_t depends on the world-state s_t, whereas the observation o_t is conditioned on the world-state s_{t+1}) and for transitions indexed by both the current world-state and the most recent action. The correctness of our sampler follows directly from the correctness of the beam sampler. The details required to adapt the beam-sampler to the iPOMDP setting, as well as discussion about its performance, are detailed in the supplementary materials, which is available in the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/191.

3.3 Action Selection

Our belief $b(s, m)$ is a joint distribution over the space of world-states and models, which we represent by a set of sampled models m with exact beliefs over world-states $b(s|m)$ in each. Reflecting our representation of the belief, we divide our action-selection strategy into two parts: (1) evaluating the value of an action a with respect to some model m and the conditional belief $b(s|m)$, which we write as $Q_m(a, b_m(s))$, and (2) choosing an action given a set of (possibly weighted) models m.

Solving Individual Models. When sampling the posterior over models, we only instantiate model parameters $T(s_j|s_i, a)$, $\Omega(o|s_i, a)$, and $R(s_i, a)$ for world-states s_i and s_j that the agent believes that it has visited. The sampled transition models encode this fact: for a visited world-state s_i, the sum $\sum_{s_j} T(s_j|s_i, a)$ over all visited world-states s_j will be less than 1, reflecting the fact that it is also possible that the agent will transition to a new completely new world-state after taking action a in world-state s_i. Because the posterior distribution for parameters corresponding to unvisited world-states is equal to the prior distribution, there was no reason to instantiate those parameters. However, those (infinite) unvisited world-states still exist. We cannot ignore them when solving a policy for the model.

The correct Bayesian approach to handling these infinite unvisited, uninstantiated states during planning would

be to marginalize over them, however, integrating over the all possible visits to all possible states is computationally intractable. Instead, we introduce a new world-state s^* that represents all of the unvisited world-states. This approximation is reasonable because we have no data to differentiate these unvisited world-states. We set the transitions into this new world-state s^* as the "leftover probability" $T(s^*|s_i, a) = 1 - \sum_{k=1}^{K} T(s_k|s_i, a)$ for all the visited world-states s_i, $i = 1..K$. We approximate the remaining parameters for s^* by using mean values from the prior. This approximation ignores the variation in these mean parameters but posits that the large number of world-states encompassed by s^* should collectively act close to their mean. Specifically, we set the transitions from s^*, $T(\cdot|s^*, a)$, to the mean transition function $\bar{T}$, the observations $\Omega(\cdot|s^*, a)$ to their mean from H, and the rewards $R(\cdot|s^*, a)$ to their mean from H_R.[4]

Now that the models $\{m\}$ have a finite number of world-states and well-formed transition and observation functions, we can plan with standard approximations (we used point-based value iteration (PBVI) [20]). Many POMDP solution techniques allow us to compute the action-value function $Q_m(a, b_m(s))$, which gives the expected value of performing action a under belief $b_m(s)$ and model m and then following the optimal policy. This action-value function fully takes into account world-state uncertainty given a model; if there was no model uncertainty then the action $a = \arg\max_a Q_m(a, b_m(s))$ would be the optimal action for the agent to take. Thus, as the model uncertainty decreases with more experience, we expect our agent to start behaving near-optimally.

Solving the Model-Uncertainty-POMDP. In the previous section, we summarized how we first create a finite summary of, and then (approximately) solve, an individual model m. Now we describe how we approximate the solution to the full model-uncertainty-POMDP, which consists of a set of models, each individually tracking its own world-state uncertainty. The state space of the model-uncertainty-POMDP is too large to apply global solutions methods (such as PBVI); instead we apply a stochastic forward search in model space which locally approximates the model-uncertainty-POMDP solution.

Forward search in POMDPs[22] uses a forward-looking tree to compute action-values. Starting from the agent's current belief, the tree branches on each action the agent might take and each observation the agent might see. At each action node, the agent computes its expected immediate reward $R(a) = E_m[E_{s|m}[R(\cdot|s, a)]]$.

The value of taking action a in belief $b(s, m)$ is

$$Q(a, b) = R(b, a) + \gamma \sum_o \Omega(o|b, a) \max_{a'} Q(a', b^{ao}) \qquad (5)$$

where b^{ao} is the agent's belief after taking action a and seeing observation o from belief b. The value $R(b, a)$ is

$$R(b, a) = \sum_m b(m) \sum_s b(s|m) R(s, a) \qquad (6)$$

where $b(m)$ is simply the weight $w(m)$ on the model m. The update to the conditional belief $b^{ao}(s|m)$ can be computed in closed form. To update the belief over models $b^{ao}(m)$, we use eqn. 4 to update the belief over models $b(m)$ via the weights $w(m)$. Eqn. 5 is evaluated recursively for each $Q(a', b^{ao})$ up to some depth D.

The number of evaluations $(|A||O|)^D$ grows exponentially with the depth D, so a full expansion is only feasible for tiny problems. We use a stochastic approximation that samples a few observations from the distribution $P(o|a) = \sum_m P(o|a, m)w(m)$. Eqn. 5 reduces to:

$$Q(a, b) = R(b, a) + \gamma \frac{1}{N_O} \sum_i \max_{a'} Q(a', b^{ao_i}) \qquad (7)$$

where N_O is the number of sampled observations and o_i is the i^{th} sampled observation.

Once we reach a prespecified depth in the tree, we must approximate the value of the leaves $Q(a, b^f)$, where f is the future that corresponds the actions and observations along the branches from the root b to the leaf. For each leaf-model m, we can efficiently compute the value $Q_m(a, b_m(s))$ from our approximate solution to the POMDP m. We approximate the value of action a as

$$Q(a, b^f) \approx \sum_m w(m) Q_m(a, b_m^f(s)). \qquad (8)$$

This approximation always overestimates the value, as it assumes that the uncertainty over models—but not the uncertainty over world-states—will be resolved in the following time step (the proof follows directly from the fact that eqn. 8 applies the QMDP approximation [46] in the space of models). As the iPOMDP posterior becomes peaked and the uncertainty over models decreases, the approximation becomes more exact.

The quality of the action selection largely follows from the bounds presented in [47] for planning through forward search. The key difference is that now our belief representation is particle-based; during the forward search we approximate expected rewards over all possible models with rewards from the particles in our set. Because our models are drawn from the true posterior over models, this approach is a standard Monte Carlo approximation of the expectation.

4 Infinite Deterministic Markov Models

The iPOMDP is a nonparametric prior over hidden-variable representations. While all hidden-variable representations are variants of POMDPs, there are many kinds of history-based representations. PDFA-like models have outperformed HMMs in uncontrolled settings [13], and recent work [37] has suggested that DMMs are more succinct sufficient statistics of the history than (i)POMDPs. Motivated by these examples, we now introduce the infinite deterministic Markov model (iDMM), which is a nonparametric prior over DMMs with a countably infinite number of nodes.

DMMs are an extension of probablistic-deterministic finite automata (PDFA) to include rewards. A PDFA is described by the 5-tuple $\{S, A, \Omega, \delta, s_0\}$, where S, A, and Ω are discrete sets of nodes, actions, and observations; the

4. Of course, the correct Bayesian thing to do would be to integrate over all the (infinite) uninstantiated states in the model. Using a catch-all state is a heuristic way of roughly achieving the smoothing effect of the desired integration in a computationally-efficient manner.

node s_0 is the initial node; δ:S x A x $\Omega \rightarrow S$ outputs which node which follows the current state, action, and observation. Just as the DMM $m = \{S, A, R, \Omega, \delta, s_0\}$ is constructed by adding rewards to the PDFA, we extend the probabilistic-deterministic infinite automata (PDIA) [13] to construct the iDMM[5].

4.1 Model

The original PDIA prior, designed for an uncontrolled system, has a generative process based on the hierarchical Pitman-Yor process[6] (HPYP) for the transitions [49]:

$$\begin{aligned} \mathcal{G} &\sim \text{PYP}(c_0, d_0, H) \\ \mathcal{G}_o &\sim \text{PYP}(c_1, d_1, \mathcal{G})\, \forall o \in \Omega \\ \delta(s, o) &\sim \mathcal{G}_o\, \forall o \in \Omega, s \in S \end{aligned}$$

where, as with the HDP-HMM, the base distribution H is the prior over observation distributions $\Omega(\cdot|s)$, and the constants c_0, d_0, c_1, and d_1 are the parameters of the Pitman-Yor process[7]. Here, $\mathcal{G}$ is a general mean transition function and $\mathcal{G}_o$ is the mean transition function for each observation o (encoding the bias that we expect the most recent observation o to be the most informative when predicting the next node). Since transitions in the PDIA are deterministic, we still have to sample a deterministic $s' = \delta(s, o) \sim \mathcal{G}_o$ for each state s.

Adding rewards to extend the PDIA prior to the iDMM is straightforward: as in previous work with DMMs [37], we assumed that each node s had some reward emission probability $R(r|s, a)$. However, deciding how to incorporate actions requires modeling choices of what biases should be encoded. We considered three different ways of incorporating actions:

1) **Consider observations then actions** To encode an explicit bias for the most recent observation to be more informative than the most recent action—such as a robot with good sensors but poor motor control—we sample transitions with:

$$\begin{aligned} \bar{T} &\sim \text{PYP}(c_0, d_0, H \otimes H_R) \\ \bar{T}_o &\sim \text{PYP}(c_1, d_1, \bar{T})\, \forall o \in \Omega \\ \bar{T}_{oa} &\sim \text{PYP}(c_2, d_2, \bar{T}_o)\, \forall o \in \Omega, a \in A \\ \delta(s, a, o) &\sim \bar{T}_{oa}\, \forall o \in \Omega, a \in A, s \in S \end{aligned}$$

   Here, all of the transition distributions $\bar{T}_{oa}$ will be similar to $\bar{T}_o$, and thus the observation will have the stronger generalizing influence (see Fig. 4).

2) **Consider actions then observations** In contrast, if we believe that the most recent action is more informative than the most recent observation—such as if there exist actions that reset the environment to a reliable state, we reverse the ordering of actions and observations in the hierarchy:

$$\begin{aligned} \bar{T} &\sim \text{PYP}(c_0, d_0, H \otimes H_R) \\ \bar{T}_a &\sim \text{PYP}(c_1, d_1, \bar{T})\, \forall a \in A \\ \bar{T}_{ao} &\sim \text{PYP}(c_2, d_2, \bar{T}_a)\, \forall a \in A, o \in \Omega \\ \delta(s, a, o) &\sim \bar{T}_{ao}\, \forall a \in A, o \in \Omega, s \in S \end{aligned}$$

   Here, all of the transition distributions $\bar{T}_{oa}$ will be similar to $\bar{T}_a$, and thus the action will have the stronger generalizing effect.

3) **Consider actions and observations equally** Incorporating the actions by extending the hierarchy forces a bias towards either actions or observations being more informative (depending on their ordering). Our final model breaks the transition function into two stages to remove this bias. In the first stage, the agent takes an action, gets a reward, and updates its node (before receiving the observation). In the second stage, the agent receives an observation and updates its node based only the observation. Thus the full transition function can be expressed as the composition of two partial-transition functions, σ:$S \times A \rightarrow S'$ and τ:$S' \times \Omega \rightarrow S$, where S' is the set of *intermediate* states, following an action but before an observation. We then use the original PDIA prior for both partial-transition functions:

$$\begin{aligned} \bar{T} &\sim \text{PYP}(c_0, d_0, H \otimes H_R) & \\ \bar{T}_a &\sim \text{PYP}(c_1, d_1, \bar{T}) & \forall a \in A \\ \sigma(s, a) &\sim \bar{T}_a & \forall a \in A, s \in S \\ \bar{T}' &\sim \text{PYP}(c_0', d_0', H') & \\ \bar{T}_o' &\sim \text{PYP}(c_1', d_1', \bar{T}') & \forall o \in \Omega \\ \tau(s', o) &\sim \bar{T}_o' & \forall o \in \Omega, s' \in S' \\ \delta(s, a, o) &= \tau(\sigma(s, a), o) & \end{aligned}$$

   Here H' is any distribution such that samples are almost surely unique, and can be thought of as a label for intermediate states. Since nodes in S can only transition to nodes in S', the combined transitions σ and τ form a symmetric graph. We refer to this model as a *symmetric iDMM* (see Fig. 4).

4.2 Inference

The belief-monitoring step, which requires performing the inference over iDMMs, follows directly from the Metropolis-Hastings sampler used in the original PDIA work of [13]. To generate samples from the posterior distribution over iDMMs given experience, we randomly sample the state to which edges transition according to a Metropolis-Hastings scheme. Given an edge $\delta(s, a, o)$, we propose a new state for it to transition to and evaluate the likelihood of the agent's experience given the new model. Since changing one edge affects the whole graph, we may have to introduce other new edges to evaluate the likelihood. These are sampled in a "just-in-time" manner from the prior, and treated as if they were already part of the model. The proposed new $\delta(s, a, o)$ is then accepted with probability given by the ratio of the

5. In contrast, the prior work of [37] used a uniform prior over DMMs, but restricted the search space.

6. The Pitman-Yor process (PYP) [48] is an extension of the Dirichlet process that allows for the generation of heavy-tailed distributions.

7. Note that $\delta(s, o)$ is an element of the base space of H, which is the space of distributions over Ω. Since samples from H are almost surely unique, we can index states by their emission distribution, so $\delta(s, o)$ is both the state that is transitioned to and the emission distribution of that state.

likelihood of experience between the old and new value of $\delta(s, a, o)$. We use an exponential prior for concentrations and a uniform prior for discounts.

4.3 Action Selection

As with the iPOMDP, we maintain a distribution over iDMMs with a set of weighted samples. To choose an action, we first choose an iDMM model based on its importance weight. While the iDMM itself does not explicitly model observation probabilities, we can use the histories h to compute the probability of each transition $T(s'|s, a) = \sum_o I(\delta(s, a, o) = s')p(o|s, a)$ using empirical observation probabilities $p(o|s, a)$. The resulting MDP can be solved via standard techniques.

5 Experiments

We begin with a set of illustrations that demonstrate the properties of the Bayesian nonparametric approach to learning POMDP representations before providing comparisons of the iPOMDP and the iDMM with other learning approaches on several benchmark problems.

All of our experiments used the following settings for any applicable priors and inference:

- **Observation Prior** H We used a uniform Dirichlet prior for H with a concentration $H_o = 1$ for each element, providing a bias toward smoother observation distributions.
- **Reward Prior** H_R We assumed that rewards took on discrete values and used a uniform Dirichlet prior for H_R with a concentration $H_o = .1$ for each element. Using a low concentration encoded our prior belief that $R(r|s, a)$ was highly peaked.
- **Updating Models** Beliefs were approximated with set of 10 models. Models were updated after every 100 interactions of experience after an initial run of 250 interactions. The beam-sampler had an initial burn-in of 50 iterations and then took every 10th sample as an output. Each round of MCMC was "hot-started" with the last model from the previous round. The initial round of MCMC was intialized with a random sequence with 5 possible states.
- **Evaluating Agent Progress** Following each full update over models using MCMC, we ran 50 "catch" test episodes (not included in the agent's experience) with the new models and policies to empirically evaluate the current value of the agents' policy.
- **Solving Models** Models were solved using PBVI [20]. The solver was run with 500 beliefs $b(s|m)$ per model; the number of backups was increased from 10 to 35 linearly with the interactions of experience (so that we would spend more effort trying to solve models in the later stages of the learning process).
- **Selecting Actions** Forward-search of depth 3.
- **Trial Length** A trial consisted of 7500 interactions with the environment. Within each trial, each episode was capped at 75 interactions of experience.
- **Repeated Trials** Each trial was repeated 10 times.

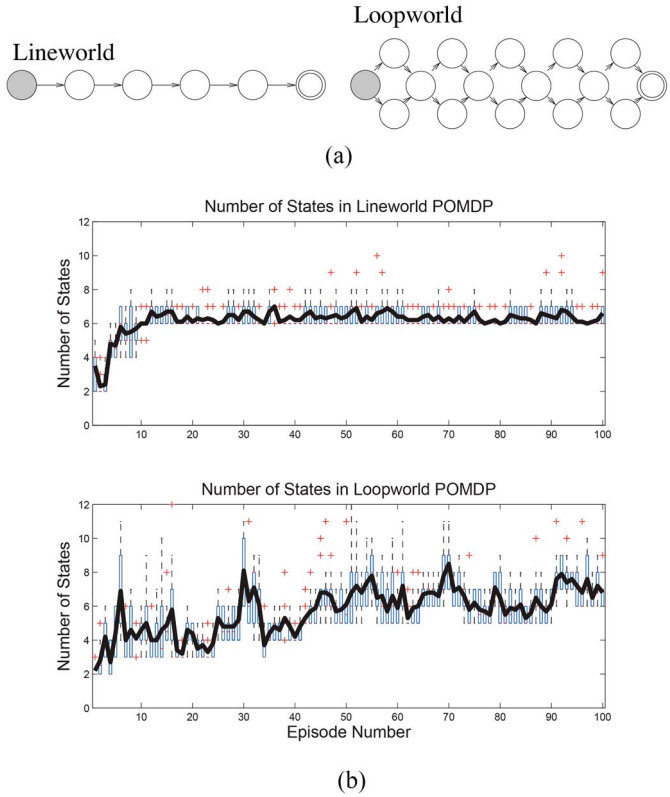

Fig. 5. Both lineworld and loopworld have hallways with a start (shaded), goal (double line), and identical middle states (a). The plots in (b) show the number of states inferred by the iPOMDP against the number of times the agent has traversed the hallways over 50 repeated trials: the black line shows the mean number of inferred iPOMDP states, and boxplots show the medians, quartiles, and outliers at each episode. Of note is that loopworld infers only necessary states, ignoring the more complex (but irrelevant) structure: (a) Cartoon of models. (b) Evolution of number of states.

The hyper-parameters were not sampled. We found that as long as the MCMC was started with a relatively small number of initial states, the choice of hyper-parameter had relatively little effect.

5.1 Illustrations

We begin with a pair of illustrative examples demonstrating the properties of the iPOMDP. The first, lineworld and loopworld, shows how the iPOMDP learns only the structure that is needed to make predictions. The second, tiger-3, shows how the infinite capacity of the iPOMDP allows it to adapt when "additional state" is added to the environment.

Avoiding unnecessary structure: Lineworld and Loopworld. We designed a pair of simple environments to show how the iPOMDP infers states only as it can distinguish them. The first, lineworld, was a length-six corridor in which the agent could either travel left or right. Loopworld consisted of a corridor with a series of loops (see Fig. 5; now the agent could travel though the upper or lower branches. In both environments, only the two ends of the corridors had unique observations. Actions produced the desired effect with probability 0.95, observations were correct with probability 0.85 (that is, 15% of the time the agent saw an incorrect observation). The agent started on the far

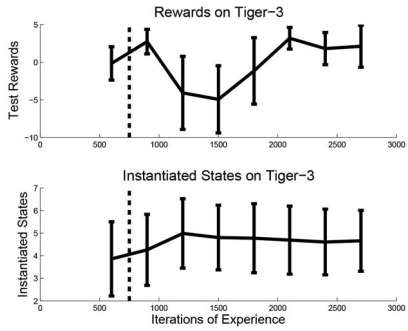

Fig. 6. Evolution of reward from tiger-3. The agent's performance dips slightly after the third door is introduced, but then it adapts its representation to the new environment. The number of instantiated states also grows to accommodate this new possibility.

left and received a reward of -1 until it reached the opposite end (reward 10).

While the agent eventually inferred that the lineworld environment consisted of six states (by the number of steps required to reach the goal), it intially inferred distinct states only for the ends of the corridor and groups the middle region as one state. Trials with the loopworld agent also showed a growth in the number of states over time (Fig. 5), but it never inferred separate states for the identical upper and lower branches. By inferring states as they were needed to explain its observations—instead of relying on a prespecified number of states—the agent avoided reasoning about irrelevant structure in the environment. The agent (unsurprisingly) learned optimal performance in both environments.

Adapting to new situations: Tiger-3. The iPOMDP's flexibility also lets it adapt to new situations. In the tiger-3 domain, a variant of the tiger problem of [46] the agent had to choose one of three doors to open. Two doors had tigers behind them ($r = -100$) and one door had a small reward ($r = 10$). At each time step, the agent could either open a door or listen for the "quiet" door. Each attempt at listening identified the good door correctly with probability 0.85.

During the agent's first 750 interactions with the environment, the reward was equally likely to be behind doors 1 or 2. The improving rewards (averaged over 10 independent runs and blocks of 300 steps) in Fig. 6 show the agent steadily learning the dynamics of its world; it learned never to open door 3. The dip in Fig. 6 following iteration 750 occurred when we next allowed the reward to be behind all three doors (the lag is because the plot averages over time, the length of the dip represents the time required to adjust to the new parameters). The agent adapted to the new possible state of its environment by instantiating new state. Thus, the iPOMDP enabled the agent to first adapt quickly to its simplified environment but add complexity as needed.

5.2 Results on Standard Problems

We next completed a set of experiments on POMDP problems from the literature (listed in Table 1). We compared the accrued rewards for the iPOMDP agent and iDMM variants with five baselines:

TABLE 1
Summary of iPOMDP Benchmarks

Domain	States	Actions	Observations
Tiger [46]	2	3	2
Network [46]	7	4	2
Shuttle [51]	8	3	5
Cheese [39]	23	4	8
5x5 Gridworld [46]	25	4	16
Follow (adapted from [2]	26	5	6
Hallway [46]	60	5	21
Beach [52]	100	5	2
Rocksample [53]	257	9	2
Image [52]	673	5	9
Tag [20]	870	5	30
Harvest [37]	896	8	7

1) **EM:** The agent knew the "true" state count K and used expectation-maximization (EM) [50], a greedy maximum likelihood approach prone to local optima, to train its model.
2) **FFBS:** The agent knew the "true" state count K and that used the forward-filtering backward-sampling (FFBS) algorithm to sample models m from the posterior over finite models with K states. We used the same hyperparameters as the iPOMDP. FFBS was used in the inner loop of the iPOMDP beam-sampler to sample state sequences once a finite model has been sliced; thus the only difference between FFBS and iPOMDP was that FFBS considered a class of finite models.
3) **EM-Big:** The agent used EM with ten times the "true" number of states $10K$. This option represented a "safe" strategy if the number of hidden states was not initially known; by guessing too high one could guarantee that the belief-state $b(s)$ would be a sufficient statistic for the history.
4) **FFBS-Big:** The agent used FFBS with ten times the true number of states $10K$ (i.e., a truncated iPOMDP), and
5) **U-Tree:** The agent used an optimized version of the U-Tree algorithm [39]. Suffixes of the history were used as a sufficient statistic for predicting future rewards. Our version of U-Tree did an all-pairs comparison when considering whether to split the nodes. We optimized over values of the KS hypothesis test threshold, the minimum node-size threshold, and number of MDP backups. The tree-depth was limited to 6 for computational reasons.

Fig. 7 plots the iPOMDP agent's learning curve, averaged over multiple trials, against all of the other baselines. The iPOMDP and the FFBS agents led the other methods with very similar learning curves; however, the iPOMDP agent did so without knowledge of the number of hidden variables needed to encode the environment. While the iDMM agents often outperformed their history-based baseline U-tree, their learning rates were slower than the hidden-variable approaches.[8]

Fig. 8 shows the rewards from "catch tests" on several benchmark problems. The problems are organized

8. While the iDMM results here are presented using the MH inference of Section 4.2, we also experimented with a variety of blocked Gibbs, tempered MH, and simulated-annealing from HMM inference schemes—all of which resulted in similar performance.

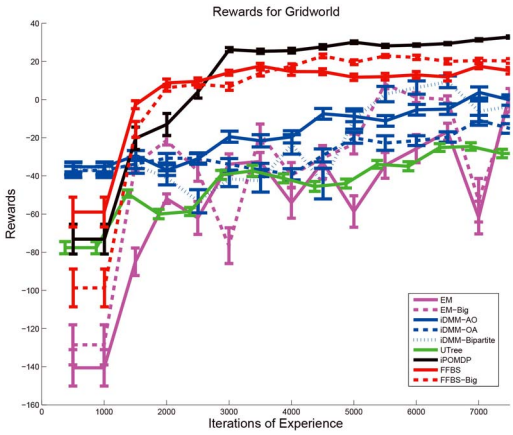

Fig. 7. Learning rates for various algorithms in the gridworld domain. The iDMM models outperform U-Tree, a simpler history-based learning method, but do not outperform the hidden-variable methods. Among the hidden-variable methods, the Bayesian methods (iPOMDP and FFBS) outperform EM by avoiding local optima.

in order of size: tiger has only two underlying states, while tag and harvest have over 800. All algorithms have difficulty with the larger problems; the variations in performance are most clear in the small and mid-sized problems. The first major difference between the algorithms is that the history-based representations, including the iDMM, tend to learn more slowly than the hidden-variable representations. (All results, of course, depend on parameter and inference choices; we optimized several parameters in U-Tree and tried several inference approaches for the iDMM.)

The impact of the inference techniques is clear when one compares the EM-based approaches with the iPOMDP and FFBS-based approaches. FFBS is a clean, robust inference technique, and the beam-sampler inherits many of its good properties. As a result, these samplers are able to find high-probability samples m and not get caught in local optima (unlike EM). This effect is most clear in tiger, where EM quickly finds the locally optimal greedy strategy of never opening a door—and thus has the worst performance. FFBS and iPOMDP often have similar performance—however, iPOMDP achieves this performance without having to prespecify the number of states (which would not be available in a real application). The advantage of using the iPOMDP is clear when one observes the less consistent performance of the "big" versions of the algorithms: inferring the number of states (iPOMDP) does as well or better than knowing the true number of states, whereas guessing conservatively leads to poorer performance.

Inferring the size of the (visited) state space also has computational benefits. In Fig. 8, certain algorithms do not appear in later plots because we were unable to run them on Matlab with 3-GB of RAM. Fig. 9 shows the running times of all the comparison algorithms relative to the iPOMDP. Because the iPOMDP often infers a smaller number of hidden states than the true count, ignoring distinctions not supported by the agent's experience, it has faster running times than similar-performing (and predicting) algorithms such as FFBS.

5.3 Comparison of Action-Selection Approaches

Reinforcement learning consists of (1) choosing (and learning) a representation and (2) selecting actions based on this representation. In Section 5.2, we saw how the learning mechanism—sampling vs. greedy optimization—can have a large impact on an agent's performance even when the choice of representation is the same. We saw

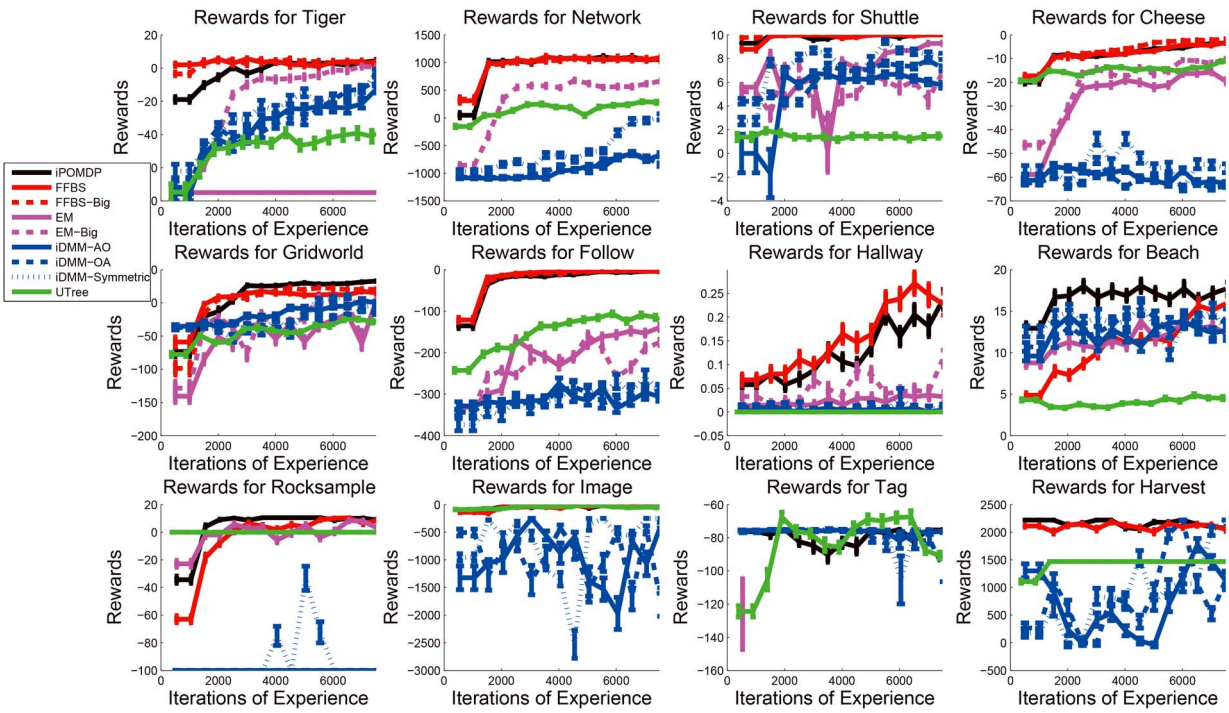

Fig. 8. Performance of various algorithms on Benchmark Problems. The Bayesian hidden-variable approaches perform best overall, and iPOMDP matches or bests the performance of FFBS with less information about the state space.

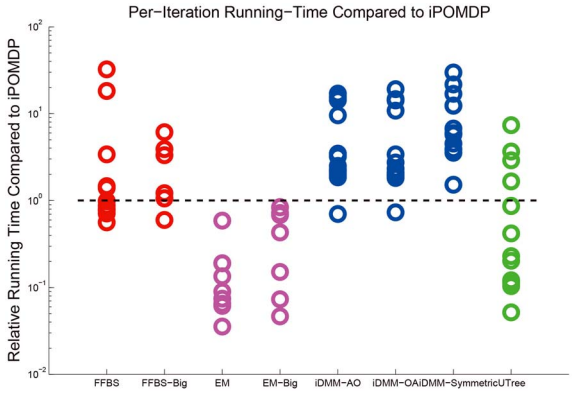

Fig. 9. Wall-clock running time of various algorithms on Benchmark Problems. All approaches were run on a computer with a 2.6GHz CPU. The iPOMDP, FFBS, and EM approaches were all coded in highly-optimized Matlab and shared subcomputations. The U-Tree approach was also coded in highly-optimized Matlab. The iDMM inference was coded in Java. Each circle in the top figure shows the running time of each comparison algorithm compared to the iPOMDP for a particular domain. A value greater than 1 means that the comparison algorithm was slower than the iPOMDP. While simple algorithms, such as EM and U-Tree, generally run faster than iPOMDP, they had significantly worse performance. FFBS-Big appears faster than FFBS because it could only be run on the six smallest domains.

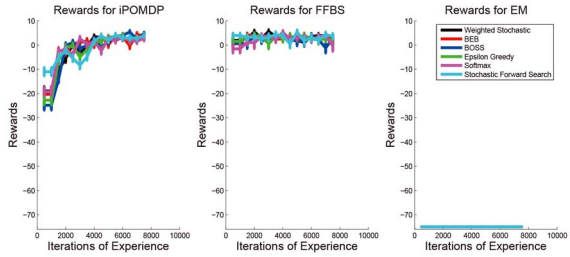

Fig. 10. Action selection comparison on tiger.

sample-based inference with hidden-variable representations outperformed history-based methods on all of our benchmarks. We now examine the impact of the action-selection scheme on the performance of the EM, FFBS, and iPOMDP learning algorithms.

This empirical analysis is motivated in part because stochastic forward search, while guaranteed to be Bayes-optimal in the limit, requires a significant amount of computation to expand the search tree—even to short depths such as five or six. However, while one may be able to plan reasonably well in many *domains* by only considering five actions into the future, it is unclear to what extent the benefits of learning a model can be ascertained by considering such a small amount of future data. In this section, we test several other action-selection heuristics on two domains, tiger and gridworld:

1) **Epsilon-Greedy** The QMDP heuristic of [46] takes the action with the highest expected reward $\sum_m Q_m(a)b(m)$, ignoring the future value of information (such as actions that might help the agent distinguish between models). The epsilon-greedy approach executes the QMDP solution $1-\epsilon$ proportion of the time, and performs a random action otherwise. We set $\epsilon = .1$ in our experiments.
2) **Softmax** Each model maintains an action-value function $Q_m(a)$. Instead of taking the action which maximizes $\sum_m Q_m(a)b(m)$, the softmax algorithms takes action a in proportion to expected future reward: $P(a) \propto \exp(\lambda \sum_m Q_m(a)b(m))$.
3) **Weighted Stochastic (WS)** Weighted stochastic is another very simple but fast heuristic: when selecting actions, we first choose a model m according to its weight $b(m)$. Then we take the action a that m believes is optimal. Choosing a model based on its weight, rather than the most likely model, allows for "potentially useful" actions to be tried some of the time. However, this approach also does not take into account the value of future information.
4) **Bayesian Exploration Bonus (BEB)** The original BEB algorithm [54], designed for MDPs, inflates rewards for state-action pairs that the agent has rarely tried (hence "exploration bonus"). In the POMDP setting, we apply the same bonus to state-action visit counts based that the agent *thinks* it has visited. By encouraging the agent to visit regions that it thinks it knows less well, we hope that BEB would help the agent discover more quickly how world-states should be organized.
5) **Best of Sampled Set (BOSS)** The original BOSS algorithm [55], also designed for MDPs, uses the most optimistic—or the best of the sampled set—transition model in each iteration of the iterative process for computing the value of an action. This optimism results in either get high rewards or a quick discovery that the models are too optimistic. While the original paper describes the work in the context of value iteration, which cannot be applied to POMDPs, we find it straight-forward to implement the concept in the context of a forward-search.

Of the three alternatives to the basic forward search, epsilon-greedy, softmax, and weighted stochastic do not explicitly consider the value of future information. They ensure that the agent explores enough by introducing various kinds of stochasticity into the action-selection process. BOSS and BEB expand use a stochastic forward search with different optimism-under-uncertainty heuristics to encourage exploration; these heuristics help compensate for lower search depths.

Figs. 10 and 11 compare the different action-selection strategies on two problems, tiger and gridworld, and three learning approaches, iPOMDP, FFBS, and EM. In all the plots, the action-selection approach makes little difference in performance; the simple, fast heuristics (epsilon-greedy, softmax, weighted stochastic) often do quite well compared to the more computationally intensive approaches basic forward search, BEB, and BOSS. In the tiger problem, we confirm that the poor performance of EM was not just due to our stochastic forward search; none of the action-selection strategies can prevent EM from getting caught in local optima.

We have several hypotheses for why the action-selection strategy seems to have little impact. First, we note that all of the strategies above are only trying to make decisions in the face of model uncertainty $b(m)$. For each sampled model m,

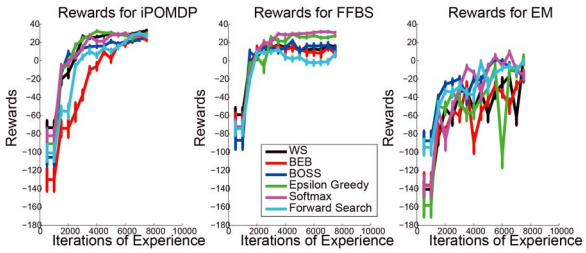

Fig. 11. Action selection comparison on Gridworld. All of the action-selection strategies have similar performance; the main differences in performance come from the inference techniques.

we can and do maintain the belief $b(s|m)$ in closed form and compute the value of each action $Q_m(b(s|m), a)$. If the model uncertainty is small compared to the state uncertainty, it is possible that the models are almost always in agreement; if all models m from $b(m)$ are in agreement, then there is little value in trying to differentiate them. We find that all the action-selection strategies chose the greedy action 70-80% of the time, suggesting that the models were often in agreement. Thus, even solving the model-uncertainty POMDP offline—as done in [25] for a set of four *extremely* different possible models—may not result in better performance.

The model uncertainty may appear to be small either because the posterior $b(m)$ is truly peaked or because of artifacts in the MCMC sampler. We know that Dirichlet priors are fairly smooth, and, in practice, our samplers often give us similar models. We experimented with using multiple restarts to help alleviate potential mixing issues, still ending up with models that make very similar predictions. Thus, it seems plausible that there may not be enough information to be discovered in our crude forward search. Our models took hundreds of interactions with the environment to learn; our forward search had a depth of 4 to 6 actions. A deeper forward search may have resulted in more optimal learning, but the depth required for an effective forward search may be too large to be practical (of course, a deep forward search would also require a discount factor larger than 0.95 to make far-future rewards attractive).

Finally, we might ask if the forward search is not really exploring, how is the learning happening? We hypothesize that all of the domains are relatively "friendly" in the sense that many policies will provide enough information about the domain for the agent to learn a model without explicitly selecting actions to do so. Even action-selection strategies that do not have an explicit stochastic component, such as BOSS or BEB, still rely on sampled models which introduce some randomness into the policy. This randomness, combined with spaces where just acting may provide much information, may be sufficient for simple techniques to perform well.

6 Discussion and Conclusion

Past work in learning POMDP models include [4], which uses a set of Gaussian approximations to allow for analytic value function updates in the POMDP space, and [2], which reasons over the joint space of Dirichlet parameters and world-states when planning in discrete POMDPs. Sampling-based approaches include Medusa [3] and [25], which learn using state and policy queries, respectively. All of these approaches assume a known number of world-states; all but [25] focus on learning only the transition and observation models.

In contrast, the iPOMDP provides a principled framework for an agent to refine more complex models of its world as it gains more experience. By linking the number of instantiated parameters to the agent's experience, the agent is not forced to consider uncertainties in large numbers of parameters at the beginning of the planning process, but it can still infer accurate models of the world as it has data to do so. We show that it outperforms both basic history-based approaches, such as U-Tree, and our own iDMM alternative.

We were surprised to find that the iPOMDP outperformed the iDMM in all environments. Prior work in uncontrolled environments [13] showed that the PDIA model, on which the iDMM is based, outperformed EM-trained HMMs on predictive tasks. In these settings, using a simpler model allowed for more robust inference and less over-fitting. The reinforcement learning setting is more complex than the uncontrolled setting, and dramatic innovations in iDMM inference—akin to the recent development of the beam-sampler—may be needed to improve the iDMM's performance on these more complex tasks. Learning models of controlled environments may also differ from learning models of uncontrolled environments in fundamental ways not addressed by any of the iDMM variants presented here. We do note, however, that while it is true that many of the benchmarks were created with "hidden states" in mind, the Harvest [37] and Cheese tasks [39] were first introduced as demonstrations of history-based methods.

More generally, there appear to be certain problem characteristics for which Bayesian nonparametric representations such as the iPOMDP are well-suited:

- **Representations Must Be Built Online, and Training Time Performance Matter.** The iPOMDP had the fastest per-iteration computational time among all of the well-performing models (Fig. 9). Most of the computational wins were early in the training phase, when instantiating fewer parameters resulted in both faster inference and faster planning. These wins became less significant as the model was refined, but the iPOMDP models often still used fewer instantiated parameters than the "true" model.
- **Data Is Limited or Fundamentally Sparse.** When gathering experience is easy, such as with a game-playing AI, almost any representation will discover reasonable structures: long suffix-trees or many-node DMMs can be trained without overfitting, statistics can be precisely computed for PSRs, and sufficient data exists to train the many parameters in a POMDP. In Fig. 8, all the approaches improved with time; if we were given a batch of trajectory data, we could have just chosen a model via standard model comparison methods such as cross-validation. What distinguished the iPOMDP's performance is that it did well early on by quickly discovering gross structure in the dynamics.

- **The "Right" Representation Is Non-Obvious.** All of the benchmark problems had one thing in common: even when the domain had a very specific structure—such as the gridworld—none of the structure was given to the agent (unlike previous work like [56]). In reality, of course, if the domain structure is well-understood, then it makes sense to use models calibrated to the task. For example, if we know there's only one "listen" parameter to learn in tiger, we found learning occurs much faster.
- **Predictive Accuracy Is the Priority.** Bayesian nonparametric approaches instantiate parameters to make accurate predictions about the future, and in general, hidden variables cannot be thought of as "real" world-states. These approaches cannot overcome fundamental non-identifiability questions—such as the identical parallel tracks in loopworld. Instead, the iPOMDP finds a representation that allow it to make good predictions.

By giving the agent an unbounded state space—but strong locality priors—the iPOMDP provides one principled framework to learning POMDP structure. Natural directions for extensions include expanding the HDP-based construction described in Section 3.1 to include deeper hierarchies, which can be used to encode structure in the state transitions (for example, clusters of states might behave similarly). Infinite dynamic bayes nets [57] could provide a more structured hidden-variable representation. More radical alternatives include infinite probabilistic context-free grammars or probabilistic programs.

However, our negative results with the iDMM and our action-selection results in Section 5.3 emphasize that one must think carefully when choosing knowledge representations. We hypothesize that one reason why the choice of action-selection strategies had little effect was that the posterior $b(m)$ was too smooth, making it difficult to discover the value of differentiating models. Even the HPYP prior in the iDMM did not truly encourage diversity. "Spiky" models with biases toward multimodal posteriors might make it easier to quickly explore the space of possible models. However, as seen with the iPOMDP and iDMM, a model is only as good as its inference: unless these posterior modes can be reliably found, the models will have little value for reinforcement-learning applications.

Acknowledgments

The authors thank David Hsu for insightful discussions on action-selection in Bayesian reinforcement learning.

References

[1] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.
[2] S. Ross, B. Chaib-draa, and J. Pineau, "Bayes-adaptive POMDPs," in *Proc. NIPS*, 2008.
[3] R. Jaulmes, J. Pineau, and D. Precup, "Learning in non-stationary partially observable Markov decision processes," in *Proc. ECML Workshop*, 2005.
[4] P. Poupart and N. Vlassis, "Model-based Bayesian reinforcement learning in partially observable domains," in *Proc. ISAIM*, 2008.
[5] R. Dearden, N. Friedman, and D. Andre, "Model based Bayesian exploration," in *Proc. 15th Conf. UAI*, San Francisco, CA, USA, 1999, pp. 150–159.
[6] M. O. Duff, "Optimal learning: Computational procedures for Bayes-adaptive Markov decision processes," Ph.D. dissertation, Univ. Massachusetts, Amherst, MA, USA, 2002.
[7] D. J. MacKay, "Ensemble learning for hidden Markov models," Cambridge Univ., Cambridge, U.K., Tech. Rep., 1997.
[8] A. Stolcke and S. Omohundro, "Hidden Markov model induction by Bayesian model merging," in *Advances in Neural Information Processing Systems*. Burlington, MA, USA: Morgan Kaufmann, 1993, pp. 11–18.
[9] C. R. Shalizi and K. L. Klinkner, "Blind construction of optimal nonlinear recursive predictors for discrete sequences," in *Uncertainty in Artificial Intelligence: Proceedings of the Twentieth Conference*, M. Chickering and J. Y. Halpern, Eds. Arlington, VA, USA: AUAI Press, 2004, pp. 504–511.
[10] G. L. Drescher, *Made-Up Minds: A Constructivist Approach to Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1991.
[11] P. J. Gmytrasiewicz and P. Doshi, "Interactive POMDPs: Properties and preliminary results," in *Proc. 3rd AAMAS*, New York, NY, USA, 2004, pp. 1374–1375.
[12] F. Doshi-Velez, "The infinite partially observable Markov decision process," in *Advances in Neural Information Processing Systems*, Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, Eds., Vancouver, BC, Canada, 2009, pp. 477–485.
[13] D. Pfau, N. Bartlett, and F. Wood, "Probabilistic deterministic infinite automata," in *Advances in Neural Information Processing Systems 23*, J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., Vancouver, BC, Canada, 2010, pp. 1930–1938.
[14] N. Z. Tishby, F. Pereira, and W. Bialek, "The information bottleneck method," in *Proc. 37th Allerton Conf. Commun., Control, Comput.*, 1999.
[15] D. Blackwell and M. Girshick, *Theory of Games and Statistical Decisions*. New York, NY, USA: Wiley, 1954.
[16] E. J. Sondik, "The optimal control of partially observable Markov processes," Ph.D. dissertation, Stanford Univ., Stanford, CA, USA, 1971.
[17] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artif. Intell.*, vol. 101, no. 1–2, pp. 99–134, 1995.
[18] J. Williams and S. Young, "Scaling up POMDPs for dialogue management: The "summary POMDP" method," in *Proc. IEEE ASRU Workshop*, San Juan, CA, USA, 2005.
[19] J. Pineau, N. Roy, and S. Thrun, "A hierarchical approach to POMDP planning and execution," in *Proc. Workshop Hierarchy Memory Reinforcement Learning (ICML)*, Jun. 2001.
[20] J. Pineau, G. Gordon, and S. Thrun, "Point-based value iteration: An anytime algorithm for POMDPs," in *Proc. 18th IJCAI*, San Francisco, CA, USA, 2003.
[21] H. Kurniawati, D. Hsu, and W. S. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces," in *Proc. RSS*, 2008.
[22] S. Ross, J. Pineau, S. Paquet, and B. Chaib-Draa, "Online planning algorithms for POMDPs," *J. Artif. Intell. Res.*, vol. 32, pp. 663–704, Jul. 2008.
[23] R. Sabbadin, J. Lang, and N. Ravoanjanahry, "Purely epistemic Markov decision processes." in *Proc. AAAI Conf. Artif. Intell.*, 2007, pp. 1057–1062.
[24] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.
[25] F. Doshi, J. Pineau, and N. Roy, "Reinforcement learning with limited reinforcement: Using Bayes risk for active learning in POMDPs," in *Proc. 25th ICML*, New York, NY, USA, 2008.
[26] M. Strens, "A Bayesian framework for reinforcement learning," in *Proc. 17th ICML*, San Francisco, CA, USA, 2000.
[27] J. Veness, K. S. Ng, M. Hutter, and D. Silver, "A Monte Carlo aixi approximation," *CoRR*, vol. abs/0909.0801, 2009.
[28] P. S. Maybeck, *Stochastic Models, Estimation, and Control*. Boston, MA, USA: Academic Press, Inc., 1979.
[29] L. Song, B. Boots, S. M. Siddiqi, G. J. Gordon, and A. J. Smola, "Hilbert space embeddings of hidden Markov models," in *Proc. 27th ICML*, Haifa, Israel, 2010.
[30] E. B. Fox, E. B. Sudderth, M. I. Jordan, and A. S. Willsky, "An HDP-HMM for systems with state persistence," in *Proc. ICML*, Helsinki, Finland, 2008.

[31] T. Stepleton, Z. Ghahramani, G. Gordon, and T. S. Lee, "The block diagonal infinite hidden Markov model," in *Proc. 12th AISTATS*, Clearwater Beach, FL, USA, 2009.
[32] M. Johnson and A. Willsky, "The hierarchical Dirichlet process hidden semi-Markov model," in *Proc. UAI*, 2010.
[33] J. Huggins and F. Wood, "Infinite structured hidden semi-Markov models," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published.
[34] M. Rabin, "Probabilistic automata," *Inform. control*, vol. 6, no. 3, pp. 230–245, 1963.
[35] J. Castro and R. Gavalda, "Towards feasible PAC-learning of probabilistic deterministic finite automata," in *Proc. 9th ICGI*, Saint-Malo, France, 2008, pp. 163–174.
[36] F. Thollard, P. Dupont, and C. De La Higuera, *Probabilistic DFA Inference Using Kullback-Leibler Divergence and Minimality*. San Francisco, CA, USA: Morgan Kaufmann, 2000, pp. 975–982 [Online]. Available: www.sfs.nphil.uni-tuebingen.de/~thollard/Recherches/Icml2k/icml2k.html
[37] M. M. H. Mahmud, "Constructing states for reinforcement learning," in *Proc. 27th ICML*, Haifa, Israel, 2010.
[38] P. Dupont, F. Denis, and Y. Esposito, "Links between probabilistic automata and hidden Markov models: Probability distributions, learning models and induction algorithms," *Pattern Recognit.*, vol. 38, no. 9, pp. 1349–1371, 2005.
[39] A. R. McCallum, "Overcoming incomplete perception with utile distinction memory," in *Proc. 10th ICML*, Amherst, MA, USA, 1993, pp. 190–196.
[40] L. Breslow, "Greedy utile suffix memory for reinforcement learning with perceptually-aliased states," Navy Center Research Lab., Tech. Rep. 890178, 1996.
[41] R. I. Brafman and G. Shani, "Resolving perceptual aliasing in the presence of noisy sensors," in *Proc. NIPS*, 2004.
[42] L. Zheng and S.-Y. Cho, "A modified memory-based reinforcement learning method for solving POMDP problems," *Neural Process. Lett.*, vol. 33, no. 2, pp. 187–200, 2011.
[43] C. Dimitrakakis, "Bayesian variable order Markov models," in *Proc. 13th AISTATS*, vol. 9. Sardinia, Italy, 2010, pp. 161–168.
[44] J. Sethuraman, "A constructive definition of Dirichlet priors," *Statistica Sinica*, vol. 4, no. 1, pp. 639–650, 1994.
[45] J. van Gael, Y. Saatci, Y. W. Teh, and Z. Ghahramani, "Beam sampling for the infinite hidden Markov model," in *Proc. 25th ICML*, vol. 25. New York, NY, USA, 2008.
[46] M. L. Littman, A. R. Cassandra, and L. P. Kaelbling, "Learning policies for partially observable environments: Scaling up," in *Proc. ICML*, San Francisco, CA, USA, 1995.
[47] D. McAllester and S. Singh, "Approximate planning for factored POMDPs using belief state simplification," in *Proc. 15th UAI*, 1999.
[48] J. Pitman and M. Yor, "The two-parameter Poisson-Dirichlet distribution derived from a stable subordinator," *Ann. Probab.*, vol. 25, no. 2, pp. 855–900, 1997.
[49] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei, "Hierarchical Dirichlet processes," *J. Amer. Statist. Assoc.*, vol. 101, no. 476, pp. 1566–1581, 2006.
[50] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc.*, vol. 39, no. 1, pp. 1–38, 1977.
[51] L. Chrisman, "Reinforcement learning with perceptual aliasing: The perceptual distinctions approach," in *Proc. 10th Natl. Conf. Artif. Intell.*, Palo Alto, CA, USA: AAAI Press, 1992.
[52] F. Doshi-Velez, D. Wingate, N. Roy, and J. Tenenbaum, "Nonparametric Bayesian policy priors for reinforcement learning," in *Proc. NIPS*, 2010.
[53] T. Smith and R. Simmons, "Heuristic search value iteration for POMDPs," in *Proc. 20th Conf. UAI*, Banff, AB, Canada, 2004.
[54] J. Z. Kolter and A. Ng, "Near-Bayesian exploration in polynomial time," in *Proc. 26th ICML*, Montreal, QC, Canada, 2009.
[55] J. Asmuth, L. Li, M. Littman, A. Nouri, and D. Wingate, "A Bayesian sampling approach to exploration in reinforcement learning," in *Proc. 25th UAI*, Arlington, VA, USA, 2009.
[56] S. Ross, B. Chaib-draa, and J. Pineau, "Bayesian reinforcement learning in continuous POMDPs with application to robot navigation," in *Proc. ICRA*, Pasadena, CA, USA, 2008.
[57] F. Doshi-Velez, D. Wingate, N. Roy, and J. Tenenbaum, "Infinite dynamic Bayesian networks," in *Proc. 28th ICML*, Washington, DC, USA, 2011.

Finale Doshi-Velez is a research associate with Harvard University, Cambridge, MA, USA. Her Ph.D. dissertation was at the Massachusetts Institute of Technology, Cambridge, MA, USA, in computer science, in 2012. She focused on Bayesian nonparametric approaches for reinforcement learning in partially-observable domains. Her current research interests include Bayesian nonparametric models, sequential decision-making, and health informatics.

David Pfau is a Ph.D. candidate at the Center for Theoretical Neuroscience, Columbia University, New York, NY, USA. He received the B.S. degree in physics from Stanford University, Stanford, CA, USA, in 2007. His current research interests include machine learning for complex, structured time series such as natural language, music or neural recordings, as well as neural decoding and neural rehabilitation.

Frank Wood is a lecturer in information engineering at Oxford University, Oxford, U.K. He received the Ph.D. degree in computer science from Brown University, Providence, RI, USA, in 2007. His current research interests include intersection of neuroscience, computer science, and statistics, particularly models and algorithms that shed light on the path towards artificial intelligence. He is also an accomplished entrepreneur.

Nicholas Roy is an associate professor of Aeronautics and Astronautics at the Massachusetts Institute of Technology, Cambridge, MA, USA. He received the Ph.D. degree from Carnegie Mellon University, Pittsburgh, PA, USA, in 2003. His current research interests include robotics, machine learning, autonomous systems, planning and reasoning, human-computer interaction, and micro-air vehicles.