Bayesian Reinforcement Learning in Factored POMDPs

Sammie Katt
Northeastern University
Boston, MA, USA
katt.s@husky.neu.edu

Frans A. Oliehoek
Delft University of Technology
Delft, Netherlands
f.a.oliehoek@tudelft.nl

Christopher Amato
Northeastern University
Boston, MA, USA
c.amato@northeastern.edu

ABSTRACT

Model-based Bayesian Reinforcement Learning (BRL) provides a principled solution to dealing with the exploration-exploitation trade-off, but such methods typically assume a fully observable environments. The few Bayesian RL methods that are applicable in partially observable domains, such as the Bayes-Adaptive POMDP (BA-POMDP), scale poorly. To address this issue, we introduce the Factored BA-POMDP model (FBA-POMDP), a framework that is able to learn a compact model of the dynamics by exploiting the underlying structure of a POMDP. The FBA-POMDP framework casts the problem as a planning task, for which we adapt the Monte-Carlo Tree Search planning algorithm and develop a belief tracking method to approximate the joint posterior over the state and model variables. Our empirical results show that this method outperforms a number of BRL baselines and is able to learn efficiently when the factorization is known, as well as learn both the factorization and the model parameters simultaneously.

KEYWORDS

Bayesian reinforcement learning; POMDPs; Monte-Chain Monte-Carlo; Monte-Carlo Tree Search; Bayes Networks

ACM Reference Format:
Sammie Katt, Frans A. Oliehoek, and Christopher Amato. 2019. Bayesian Reinforcement Learning in Factored POMDPs. In *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019,* IFAAMAS, 9 pages.

1 INTRODUCTION

Robust decision-making agents in any non-trivial system must reason over uncertainty in various dimensions such as action outcomes, the agent's current state and the dynamics of the environment. The outcome and state uncertainty are elegantly captured by *Partially Observable Markov Decision Processes* (POMDPs) [21], which enable reasoning in stochastic, partially observable environments. However, POMDP solution methods typically assume complete access to the system dynamics, which unfortunately are often not readily available. When such a model is not available, the problem turns into a partially observable RL (PORL) task, where one must trade off exploration and exploitation of current knowledge While recent model-free, deep RL approaches [30, 41, 46] have shown impressive results on complex tasks, this progress has been driven by improvements to function approximation. These methods often require millions of samples and combining them with effective exploration, although a topic of some studies [3, 5, 33], generally is difficult.

Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), N. Agmon, M. E. Taylor, E. Elkind, M. Veloso (eds.), May 13–17, 2019, Montreal, Canada.

A rather different approach is taken by model-based Bayesian reinforcement learning (BRL) methods [11, 39]. These methods explicitly maintain distributions over the possible models of the environment, and use this knowledge to select actions that, theoretically, can optimally trade off exploration and exploitation. As a result, BRL methods can be very sample efficient.

However, the number of BRL methods that are applicable to partial observable settings are few, and those that do exist are limited in their scalability. For instance, the Bayes-Adaptive POMDP [39] (BA-POMDP), for which we developed an efficient Monte-Carlo Tree Search (MCTS) planner, BA-POMCP [22], models the environment in a tabular fashion. The fact that this approach is based on flat state representations, however, is a bottleneck for scalability. Here, we propose a method to overcome this bottleneck by exploiting structure in the dynamics of *factored* POMDPs [6, 15].

Specifically, we formalize the *Factored* Bayes-Adaptive POMDP (FBA-POMDP), which models the dynamics of partially observable environments through graphical models that exhibit structure, as opposed to tables. The FBA-POMDP framework casts the PORL problem as a planning task, for which we develop FBA-POMCP, a MCTS algorithm that is able to tackle problems of non-trivial length and sizes. Lastly, maintaining a distribution over a potentially large space of factored POMDP models is a challenge. To combat this issue efficiently, we propose a sample-based mechanism to reinvigorate the distribution over graphical models. We show the favorable theoretical guarantees of this approach and demonstrate empirically that we outperform current state-of-the-art methods on three domains, one of which causes previous methods based on the tabular BA-POMDP to fail to learn at all.

2 BACKGROUND

We first provide a summary of the background literature. This section is divided into an introduction to POMDPs and BA-POMDPs, typical solution methods, and factored models.

2.1 The POMDP and BA-POMDP

The POMDP [21] is a general model for decision-making in stochastic and partially observable domains, with execution unfolding over (discrete) time steps. At each step the agent selects an action that triggers a state transition in the system, which generates a reward and observation. The observation is perceived by the agent and the next time step commences. Formally, a POMDP is described by the tuple $\langle S, A, \Omega, D, R, \gamma, h\rangle$, where S is the set of states of the environment; A is the set of actions; Ω is the set of observations; D is the 'dynamics function' that describes the behavior of the system in the form of transition and observation probabilities $D(s', o|s, a)$; R is the immediate reward function $R(s, a)$ that describes the reward of

selecting a in s; $\gamma \in (0,1)$ is the discount factor; and h is the horizon of an episode in the system.

This formulation of the dynamics generalizes the usual formulation with separate transition T and observation functions O: $D(s', o|s, a) = T(s'|a, s)O(o|s, a, s')$. We employ this notation for brevity reasons but used the separation in our implementation.

The agent has no direct access to the system's state, so it can only rely on the *action-observation history* up to the current step t: $h_t = \langle \vec{a}_0^t, \vec{o}_0^t \rangle$, where $\vec{a}_0^t$ and $\vec{0}_0^t$ respectively are the vector of actions and observations from time step 0 to t. When there is no confusion possible, we will also omit the super and subscripts. The agent can use this history to maintain a probability distribution over the state, also called a belief, $b(s)$. The belief is updated at every step through the *belief update* $\tau : (b, a, o) \rightarrow b'$. When the dynamics D are given, the probability of a new state s' after action a and observation o can be computed with the Bayes' Rule:

$$b'(s') = \tau(b, a, o)(s') \propto \sum_s D(s', o|s, a)b(s) \tag{1}$$

The goal of the agent in a POMDP is to find a policy π — a mapping from any belief b to an action a — that maximizes the expectation over the cumulative (discounted) reward, also called the return. Such a policy is called an optimal policy π^*.

In the *Partially Observable Reinforcement Learning* setting (PORL) the dynamics are not known and the belief over states cannot be maintained. The typical Bayesian approach to solving such an RL problem is to maintain a probability distribution over the unknown model, $p(D)$, and select actions with respect to the uncertainty over D. A distribution over D can be represented by a Dirichlet distribution for each $\langle s, a \rangle$ pair. More specifically, each transition $\langle s', o, s, a \rangle$ is associated with a count $\chi_{sa}^{s'o}$, and the collection of all counts, χ, describes a probability distribution over the dynamic function D of the POMDP: $p(D_{sa}) = \chi_{sa}$.

Conceptually, if both the visited states and observations were known, then the agent could 'count' the number of occurrences of each transition by incrementing χ. Over time the counts χ would grow and the belief over the dynamics would converge to the true dynamics. However, the states are hidden to the agent, and thus there is uncertainty over the true counts. Fortunately, this uncertainty can be captured using regular POMDP formalisms.

The Bayes-Adaptive POMDP (BA-POMDP) [39] is a POMDP in which the counts χ are part of the hidden state space. More formally, if X denotes the space of count collections χ, then the BA-POMDP is defined as the tuple $\langle \bar{S}, A, \Omega, \bar{D}, \bar{R}, \gamma, h \rangle$ with (hyper-) state space $\bar{S} = S \times X$. While the observation and action space remain unchanged, a hyper state in the BA-POMDP consists of a domain state and a count collection that represents the belief over the dynamics of the POMDP $p(D)$, $\bar{s} = \langle s, \chi \rangle$. The reward function depends only on the underlying POMDP state: $\bar{R}(\bar{s}, a) = R(s, a)$ and is typically considered known (although could be learned using similar methods). The dynamics function of the BA-POMDP describes the probability $\bar{D}(s', \chi', o|s, \chi, a)$. This factorizes in the probability of the new domain state and observation $p(s', o|s, \chi, a)$ and the update of the counts $p(\chi'|s, \chi, a, s', o)$. The former probability is defined by the ratio of the counts, which also corresponds to the *expected* categorical according to χ_{sa}:

$$p(s', o|s, \chi, a) = p_\chi(s', o|s, a) = \frac{\chi_{sa}^{s'o}}{\sum_{s'o} \chi_{sa}^{s'o}} \tag{2}$$

Rewarding $p(\chi'|s, \chi, a, s', o)$, there is only one new possible set of counts χ', given the previous counts χ and transition $\langle s', o, s, a \rangle$: the one that has $\chi_{sa}^{s'o}$ incremented by 1. More formally, we let $\delta_{sa}^{s'o}$ denote a vector of the length of χ containing all zeros except for the position corresponding to $\langle s, a, s', o \rangle$, where it is 1, and we let $\mathbb{I}_a(b)$ denote the Kronecker Delta function function that indicates (is 1 iff) $a = b$. Then we denote the count update function $\mathcal{U}(\chi, s, a, s', o) = \chi + \delta_{sa}^{s'o}$ and can rewrite $p(\chi|s, \chi, a, s', o) = \mathbb{I}_{\chi'}(\mathcal{U}(\chi, s, a, s', o))$. As a result the dynamics of the BA-POMDP resolves to:

$$\bar{D}(s', \chi', o|s, \chi, a) = \frac{\chi_{sa}^{s'o}}{\sum_{s'o} \chi_{sa}^{s'o}} \mathbb{I}_{\chi'}(\mathcal{U}(\chi, s, a, s', o)) \tag{3}$$

Lastly, the BA-POMDP requires a prior $\bar{b}_0(s, \chi)$, the initial joint belief over the domain state and dynamics. Typically the prior over D can be described with a single set of counts χ_0, and the prior reduces to $\bar{b}_0(s, \chi) = \mathbb{I}_\chi(\chi_0)b_0(s)$, where $b_0(s)$ is the distribution over the initial state of the underlying POMDP.

2.2 Learning by Planning in BA-POMDPs

The BA-POMDP casts the PORL problem as a planning task in a large POMDP where the unknown dynamics are part of the hidden state space. An optimal solution to the BA-POMDP solves the exploration-exploitation trade-off of the underlying RL problem in a principled way (analogous to the observable case [50]).

Unfortunately, the countably infinite state space poses a challenge to offline solution methods due to the curse of dimensionality. As such, previous work has resorted to online solutions. We extended Partially Observable Monte-Carlo Planning (POMCP) [8, 42], a Monte-Carlo Tree Search (MCTS) based algorithm, to the BA-POMDP [22], and will build on this to solve FBA-POMDPs.

At each time step, POMCP incrementally constructs a look-ahead action-observation tree using Monte-Carlo simulations of the POMDP. Each simulation starts by sampling a state from the belief, and traverses the tree by picking actions according to the *Upper Confidence Bound* (UCB [2]), and simulating interactions according to the POMDP model. Upon reaching a leaf-node, the tree is extended with a node for that particular history and the algorithm then propagates the accumulated reward back up into the tree, updating the statistics in each visited node. The action selection terminates by picking the action at the root of the tree that has the highest average return.

The modifications to BA-POMCP, the application POMCP to BA-POMDPs, are two-fold: (1) a simulation starts by sampling a hyper-state $\langle s, \chi \rangle$ at the start and (2) the simulated experiences follows the dynamics of the BA-POMDP: the domain state-observation pair is generated according to χ, which in turn are then used to update χ. Given enough simulations, BA-POMCP converges to the optimal solution with respect to the belief it is sampling states from [22].

2.3 Belief tracking

While the state space is countably infinite, the number of reachable states at any given time t is limited by the prior $b_0(s, \chi)$ and history

$\langle \vec{a}_0^t, \vec{o}_0^t \rangle$. As a result, one can update this belief in closed form by iterating over all possible next states using the dynamics of the BA-POMDP [39]. This quickly becomes infeasible and is only practical for small environments and horizons. More common approaches approximate the belief with *particle filters* [44]. There are several methods to update the particle filter, of which *rejection sampling* has traditionally been used for (BA-)POMCP. Here we use *Importance sampling* [14], however, as it has been shown to be superior in terms of the chi-squared distance [9].

In importance sampling the belief is a weighted particle filter, where each particle x is associated with a weight w_x that represents its probability $p(x)=\frac{w_x}{\sum_{i=1}^{K} w_i}$. Importance sampling computes the new belief given an action a and observation o with respect to the model's dynamics, $b' = \tau(\bar{b}, a, o)$, in three steps. First, each particle is updated using the transition dynamics $\bar{s}' \sim \bar{D}(\cdot|\bar{s}, a)$, and then weighted according to the observation dynamics $w' = w \cdot \bar{D}(o|\bar{s}, a, \bar{s}')$[1]. Note that the sum of weights of the belief after this step $\mathcal{L}^t = \sum w_i^t$ represents the likelihood of the belief update at time t. The likelihood of the entire belief given the observed history h^t can be seen as the product of the likelihood of each update step $\mathcal{L}_{h^t} = \mathcal{L}^t \mathcal{L}_{h^{t-1}}$. Third and last, the belief is resampled, as is the norm in sequential importance sampling.

In between episodes, assuming termination is observable, the agent's belief over the domain state s is reset. However note that, in BA-POMDPs, the belief over the model (counts χ) is retained. In practice, using particle filters as a belief, this results in resetting the domain state in each particle with a sample from $b_0(s)$.

2.4 Factored Models

The dynamics of the POMDP can represented more compactly by exploiting conditional independence between variables. If we factorize the state space into n features $S = \{S^1, \ldots, S^n\}$, and the observation space into m features $\Omega = \{\Omega^1, \ldots, \Omega^m\}$ then the Factored POMDP (F-POMDP) [7] represents the dynamics D as a collection of *Bayes-Nets* (BN) G, one for each action G^a. A BN consists of topology over a set of nodes, which describes the directed edges between the nodes, and a set of *Conditional Probability Tables* (CPTs). The CPTs describe the probability distribution over the values of the nodes given their parent values where we denote θ as the parameters of the CPTs of graphs in G, one for each action. This is illustrated in Figure 1, which shows the topology of the dynamics of a single action of a POMDP with three state features and 2 observation features.

We adopt the notation that given some state s, the probability of the value $v(x)$ of some feature x is given by $\theta(v(x)|PV^{xa}(s))$, where $PV^{xa}(s)$ returns the parent values of feature x given action a. The dynamics of the F-POMDP, is then the joint of all features:

$$D(s', o|s, a) = \prod_{x \in S \cup \Omega} \theta(v(x)|PV^{xa}(s)) \tag{4}$$

The literature contains methods that attempt to exploit the factorization in F-POMDPs [7, 13, 15, 16, 28, 31, 43, 47, 49]. These methods, however, operate under the assumption that the dynamics are known a priori and hence cannot be applied to applications where this is not the case.

[1]Note that this utilizes the assumed factorization of the dynamics into a transition and observation function

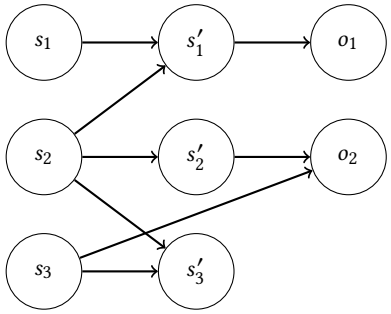

Figure 1: A graph that represents the dynamics associated with a particular action

3 BAYESIAN RL IN FACTORED POMDPS

The BA-POMDP provides a Bayesian framework for RL in POMDPs, but is unable to describe (or exploit) structure that many real world applications exhibit. The representation scales poorly and learns slowly, as the number of parameters grows quadratically in the state space ($O(|S|^2|A||\Omega|)$) and only one (count) is updated after each observation. Here we introduce the Factored BA-POMDP (FBA-POMDP), the Bayes-Adaptive framework for the factored POMDP, that is able to learn and exploit structure in the environment.

3.1 The Factored BA-POMDP

If the structure G of a F-POMDP is known a priori, but its parameters θ are not, then one could consider a Bayes-Adaptive model with counts $\dot{\chi}$ to describe Dirichlet distributions *over the CPTs*.

Known structure: We refer the count associated with value v of a feature x given action a and input state s as $\dot{\chi}(v|PV^{xa}(s))$. The dynamics of this framework is a function of the state, action and counts collection $p(s', \dot{\chi}', o|s, \dot{\chi}, a)$, which factorizes into the probability of a state-observation pair $p_{\dot{\chi}}(s', o|s, a)$, and the counts update $p(\dot{\chi}'|\dot{\chi}, s', o, s, a)$. The probability of the new state and observation corresponds to the joint expectation of all features:

$$p_{\dot{\chi}}(s', o|s, a) = \prod_{x \in S \cup \Omega} p_{\dot{\chi}}(v(x)|PV^{xa}(s)) \tag{5}$$

$$p_{\dot{\chi}}(v|PV^{xa}(s)) = \frac{\dot{\chi}(v|PV^{xa}(s))}{\sum_v \dot{\chi}(v|PV^{xa}(s))} \tag{6}$$

$p(\dot{\chi}'|\dot{\chi}, s', o, s, a)$ corresponds to updating the counts as is the case in the BA-POMDP, denoted ($\mathcal{U}(\dot{\chi}, s, a, s', o)$). Here, as opposed to affecting just a single parameter, it increments a count *per node*.

Unknown structure: It is unrealistic to assume that the topology G of the dynamics is known. Instead, the *Factored BA-POMDP* (FBA-POMDP) also considers G as part of the hidden state. First we define $\mathcal{G} = \{\mathcal{G}^1 \ldots \mathcal{G}^{|A|}\}$ as the set of possible graph topologies for all actions. Then the FBA-POMDP is a POMDP with the state space $\bar{S} = S \times \mathcal{G} \times \dot{X}$, with S as the domain state space of the underlying POMDP and $\dot{X}$ as the space of all possible count collections $\dot{\chi}$.

A (hyper-) state in the FBA-POMDP thus contains a domain state, $|A|$ graph topologies and counts to describe a Dirichlet distribution over all CPTs, $\bar{s} = \langle s, G, \dot{\chi} \rangle$. The dynamics function must then have the form of $\bar{D}(\bar{s}', o|\bar{s}, a) = p(\langle s', G', \dot{\chi}' \rangle, o|\langle s, G, \dot{\chi} \rangle, a)$. This joint distribution can be factored into the state-observation pair transition $p(s', o|\langle s, G, \dot{\chi} \rangle, a)$, the counts update $p(\dot{\chi}'|\langle s, G, \dot{\chi} \rangle, a, s', G', o)$ and the topologies update term $p(G'|\langle s, G, \dot{\chi} \rangle, a, s', o)$. The first two terms have already been discussed above (eq. (5) and $\mathcal{U}$). The latter term, under the common assumption that the (structure of the) underlying POMDP dynamics does not change over time, reduces to the Kronecker Delta function $\mathbb{I}_{G'}(G)$. This results in the following formal definition of the FBA-POMDP as tuple $\langle \bar{S}, A, \Omega, \bar{D}, \bar{R}, \gamma, h \rangle$:

- A, γ, h: Identical to the underlying POMDP.
- $\bar{R}(\bar{s}, a) = R(s, a)$ ignores the counts and reduces to the reward function of the POMDP similar to the BA-POMDP.
- $\bar{\Omega}$: $\{\Omega^0 \times \cdots \times \Omega^m\}$; the set of possible observations defined by their features.
- $\bar{S}$: $\{S^0 \times \cdots \times S^n\} \times \mathcal{G} \times \dot{X}$; the cross product of the domain's factored state space and the set of possible topologies, one for each action a, and their respective Dirichlet distribution counts.
- $\bar{D}$; The dynamics function over that describes the probabilities of transitioning from one hyper state $\bar{s} = \langle s, G, \dot{\chi} \rangle$ to another while generating observation o

$$D(\bar{s}', o|\bar{s}, a) = p_{\dot{\chi}}(s', o|s, a)\mathbb{I}_{\dot{\chi}'}(\mathcal{U}(\dot{\chi}, s, a, s', o))\mathbb{I}_{G'}(G) \qquad (7)$$

  as described above

Lastly we require a prior, a joint distribution, over the FBA-POMDP state space $\bar{b}_0(\langle s, G, \dot{\chi} \rangle)$. In many applications the dependence relationships between features is known a priori for large parts of the domain. For the unknown parts, one could consider a uniform distribution, or distributions that favor few edges.

3.2 Solving FBA-POMDPs

The FBA-POMDP itself is a large POMDP. A solution to this task consists of a method for maintaining the belief $\bar{b}$ and a policy that picks actions with respect to this belief. An optimal solution to the FBA-POMDP is guaranteed to be as sample efficient as possible, maximizing the expected return $V^*(\bar{b})$ *with respect to the uncertainty over the dynamics of the F-POMDP*. For now we assume the belief is given, and focus on developing a planner to generate the policy.

While the representation of the dynamics has changed from tables to graphs, solution methods for the FBA-POMDP, with its large state space, face similar challenges as those for BA-POMDPs: Therefore we draw inspiration from the successful BA-POMDP planning algorithm, BA-POMCP. Recall that the extension of POMCP to BA-POMCP (section 2.2) was summarized by two key parts: sampling both counts χ and a domain state from the belief at the start of each iteration, and simulating interactions with the environment according to the sampled χ. We propose a similar extension for the factored case, and call it Factored BA-POMCP (FBA-POMCP).

A simulation in the FBA-POMCP begins with sampling a FBA-POMDP hyper-state $\bar{s} = \langle s, G, \dot{\chi} \rangle$. The algorithm then traverses through the tree picking actions according to UCB, and simulating interactions according to $\dot{\chi}$ (illustrated in algorithm 1). A simulated step first samples a state-observation pair given the current state and action according to $p_{\dot{\chi}}(\cdot|s, a)$ (line 1), then updates the counts (line 3). Modifications developed specifically for the BA-POMCP, such as root-sampling and expected-transitions, can be applied to FBA-POMCP too. We refer to the original paper for details [22].

Algorithm 1 FBA-POMCP-STEP

Input $\bar{s} = \langle s, G, \dot{\chi} \rangle$: hyper-state, a: simulated action
Output $\bar{s}'$: new FBA-POMDP state, o: simulated observation
1: $s', o' \sim p_{\dot{\chi}}(\cdot|s, a)$
2: // increment the associated CPT counts, skip if root-sampling
3: $\dot{\chi}' \leftarrow \mathcal{U}(\dot{\chi}, s, a, s', o)$
4: $G' \leftarrow G$
5: **return** $\langle s', G', \dot{\chi}' \rangle, o$

3.3 Belief tracking & Particle Reinvigoration

The previous two sections introduced the FBA-POMDP, a large POMDP with Bayes-Nets as part of the state space, and the planning method FBA-POMCP to solve it. Here, we discuss how to maintain the belief. Recall that this belief $\bar{b}(\bar{s})$ is a probability distribution over the FBA-POMDP state, which contains the underlying POMDP state $s \in S$, a set of graph topologies to describe its structure $G \in \mathcal{G}$, and a collection of counts to describe the Dirichlets over the CPTs $\dot{\chi} \in \dot{X}$. It is not practical to maintain a distribution over all possible topologies G, so closed-form approaches are infeasible. Instead, we adopt the particle filter approach that is successful for BA-POMDPs, where now each particle contains $\langle s, G, \dot{\chi} \rangle$. Given an action a and observation o, the belief update, $\bar{b}' = \tau_{\bar{D}}(\bar{b}, a, o)$, is fully specified by the FBA-POMDP dynamics $\bar{D}$. However, $\bar{D}$ assumes that the topology of the underlying POMDP does not change ($p(G'|\bar{s}, a, s', o) = \mathbb{I}_{G'}(G)$ from eq. (7)) and, as a result, it would never modify the topologies in the particles.

Because of this and particle degeneracy, traditional particle filter belief update schemes tend to converge to a single structure, leading to poor performance. To tackle this issue, we propose a *Markov-Chain Monte-Carlo* (MCMC) [17] based sampling scheme to occasionally reinvigorate the belief with new particles according to the (observed) history $\bar{s} \sim p(\langle s, G, \dot{\chi} \rangle|\langle \vec{a}, \vec{o} \rangle, \bar{b}_0)$.

First we re-introduce the notation $\vec{x}_r^t$ which describes the sequence of values of x (a state, action or observation) from time step r to t, with the special case x_t, which corresponds to the value at time step t. For brevity we also use 'model' and the tuple $\langle G, \dot{\chi} \rangle$ interchangeably here, as they both describe the dynamics of a POMDP. Lastly, we refer to T as the last time step in our history, and add that $\vec{x}$ without subscripts is short for the complete sequence $\vec{x}_0^T$.

The distribution $p(\langle s, G, \dot{\chi} \rangle|\langle \vec{a}, \vec{o} \rangle, \bar{b}_0)$ is complex for multiple reasons. First, computing it typically involves integrating out the hidden state sequence. Second, it contains graphs, over which distributions are hard to represent. We propose to sample from this distribution through Gibbs sampling [32, 40], which approximates a joint distribution by sampling variables from their conditional distributions with the remaining variables fixed: we can sample from $p(x, y)$ by picking some initial x, and then continuously sample $y \sim p(y|x)$ and $x \sim p(x|y)$. Here x=$\vec{s}$ and y=$\langle G, \dot{\chi} \rangle$, and we sample:

i. $\vec{s} \sim p(\cdot|G, \dot{\chi}, \langle \vec{a}, \vec{o} \rangle, \bar{b}_0)$, a state sequence
ii. $G, \dot{\chi} \sim p(\cdot|\langle \vec{s}, \vec{a}, \vec{o} \rangle, \bar{b}_0)$, a model

State sequence sampling i.: We approach this task as sampling from a Hidden Markov Model, where the dynamics are determined by the model $\langle G, \dot{\chi} \rangle$ and action history $\vec{a}$. Due to the Markov property, $p(\vec{s}|G, \dot{\chi}, \langle \vec{a}, \vec{o} \rangle, \bar{b}_0)$ decomposes into

$$p(s_0|\bar{b}_0, \langle \vec{a}_0^T, \vec{o}_0^T \rangle, G, \dot{\chi}) \prod_{t=0\ldots T} p(s_t|s_{t-1}, \langle \vec{a}_t^T, \vec{o}_t^T \rangle, G, \dot{\chi}) \quad (8)$$

from which we aim to sample $s_0 \ldots s_T$ hierarchically. For this we require to know $p(s_t|s_{t-1}, \langle \vec{a}_t^T, \vec{o}_t^T \rangle, G, \dot{\chi})$, which we compute through message passing [35]. The forward message pass $\alpha_t(s_t) = p(s_t|s_{t-1}, a_t, o_t, G, \dot{\chi})$ for $t > 0$ can directly be inferred using the Bayes' Rule (with $\alpha_0 = \bar{b}_0(s)$). The backward-message $\beta_t(s_t)$ is computed recursively from $t = T - 1 \ldots 0$:

$$\beta_{t-1}(s) = \sum_{s'} p_{\dot{\chi}}(s', o_t|s, a_t) \cdot \beta_t(s') \quad (9)$$

where β_T is initiated with ones.

Model sampling ii.: Sampling a model from the conditional distribution $p(G, \dot{\chi}|\langle \vec{s}, \vec{a}, \vec{o} \rangle, \bar{b}_0)$ is split into two steps. We first **(a)** sample topologies $G \sim p(\cdot|\langle \vec{s}, \vec{a}, \vec{o} \rangle, \bar{b}_0)$ using Metropolis-Hastings [32]. The second step **(b)** computes the collection of counts given the topologies, prior and history: $\dot{\chi} \sim p(\cdot|\langle \vec{s}, \vec{a}, \vec{o} \rangle, G, \bar{b}_0)$. This is a deterministic function that takes the prior $\dot{\chi}_0$ of G and counts the transitions in the history $\langle \vec{s}, \vec{a}, \vec{o} \rangle$. For the former sample step, **(ii. a)** $G \sim p(\cdot|\langle \vec{s}, \vec{a}, \vec{o} \rangle, \bar{b}_0)$, we adopt a Metropolis-Hastings scheme.

Metropolis-Hastings samples some distribution $p(x)$ using a proposal distribution $q(\tilde{x}|x)$ and a acceptance test operation. The acceptance probability of $\tilde{x}$ is defined as $\frac{p(\tilde{x})q(\tilde{x}|x)}{p(x)q(x|\tilde{x})}$. More specifically, given some initial value x, Metropolis-Hastings consists of:

(1) sample $\tilde{x} \sim q(\tilde{x}|x)$
(2) with probability $MH\text{-}Accept = \frac{p(\tilde{x})q(\tilde{x}|x)}{p(x)q(x|\tilde{x})}$ we set $x \leftarrow \tilde{x}$
(3) store x and go to (1)

Let us take $p(x) = p(G|\langle \vec{s}, \vec{a}, \vec{o} \rangle, \bar{b}_0)$ and q to be domain specific but symmetrical[2], then we derive the following Metropolis-Hastings step for **(ii. a)**:

$$MH\text{-}Accept = \frac{p(\tilde{G}|\langle \vec{s}, \vec{a}, \vec{o} \rangle, \bar{b}_0) \cancel{q(\tilde{x}|x)}}{p(G|\langle \vec{s}, \vec{a}, \vec{o} \rangle, \bar{b}_0) \cancel{q(x|\tilde{x})}} = \frac{\frac{p(\langle \vec{s}, \vec{a}, \vec{o} \rangle, \tilde{G}|\bar{b}_0)}{\cancel{p(\langle \vec{s}, \vec{a}, \vec{o} \rangle|\bar{b}_0)}}}{\frac{p(\langle \vec{s}, \vec{a}, \vec{o} \rangle, G|\bar{b}_0)}{\cancel{p(\langle \vec{s}, \vec{a}, \vec{o} \rangle|\bar{b}_0)}}} = \frac{p(\langle \vec{s}, \vec{a}, \vec{o} \rangle, \tilde{G}|\bar{b}_0)}{p(\langle \vec{s}, \vec{a}, \vec{o} \rangle, G|\bar{b}_0)} \quad (10)$$

Where q cancel out due to symmetry assumptions and the first step applies the Bayes-rule: $p(G|\langle \vec{s}, \vec{a}, \vec{o} \rangle, \bar{b}_0) = \frac{p(\langle \vec{s}, \vec{a}, \vec{o} \rangle, G|\bar{b}_0)}{p(\langle \vec{s}, \vec{a}, \vec{o} \rangle|\bar{b}_0)}$.

Equation (10) is the likelihood ratio between the two graph structures. It has been shown that the likelihood $p(\langle \vec{s}, \vec{a}, \vec{o} \rangle, G|\bar{b}_0)$, given some mild assumptions (such as that the prior is a Dirichlet), is given by the Bayesian-Dirichlet (BD) score metric [19]. Given some initial set of prior counts for G, $\dot{\chi}_0$, and a $\langle \vec{s}, \vec{a}, \vec{o} \rangle$ dataset, we denote N^{nev} as the number of occurrences of v of node n given parent values e and compute the score as follows: $p(\langle \vec{s}, \vec{a}, \vec{o} \rangle, G|\bar{b}_0) =$

$$\prod_n \prod_e \frac{\Gamma(\dot{\chi}_0^{ne})}{\Gamma(\dot{\chi}_0^{ne} + N^{ne})} \prod_v \frac{\Gamma(\dot{\chi}_0^{nev} + N^{nev})}{\Gamma(\dot{\chi}_0^{nev})} \quad (11)$$

Where we abuse notation and denote the total number of counts, $\sum_v \dot{\chi}^{nev}$, as $\dot{\chi}^{ne}$ (and similarly $N^{ne} = \sum_v N^{nev}$). This formula is also used to compute $p(\langle \vec{s}, \vec{a}, \vec{o} \rangle, \tilde{G}|\bar{b}_0)$.

Given this acceptance probability, Metropolis-Hastings can sample a new set of graph structures G with corresponding counts for the CPTs $\dot{\chi}$. This particular combination of MCMC methods — Metropolis-Hastings in one of Gibbs's conditional sampling steps — is also referred to as *MH-within-Gibbs* and, has been known to converge to the true distribution even if the Metropolis-Hastings part only consist of one sample per step [23, 24, 27, 36, 45].

Particle Reinvigoration procedure: The overall particle reinvigoration procedure, assuming some initial $\langle G, \dot{\chi} \rangle$, is summarized as follows:

1. sample from HMM: $\vec{s} \sim p(\cdot|\langle \vec{a}, \vec{o} \rangle, G, \dot{\chi}, \bar{b}_0)$ (i.)
2. sample from MH: $G \sim p(\cdot|\langle \vec{s}, \vec{a}, \vec{o} \rangle, \bar{b}_0)$ (using BD-score) (ii.**a**)
3. compute counts: $\dot{\chi} \sim p(\cdot|\langle \vec{s}, \vec{a}, \vec{o} \rangle, G, \bar{b}_0)$ (ii.**b**)
4. add $\langle s, G, \dot{\chi} \rangle$ to belief and go to 1

It is not necessary to do this operation at every time step. Instead, the log-likelihood $\mathcal{L}$ of the current belief is a useful metric to determine when to resample, which fortunately is a by-product of importance sampling during the belief update. The total accumulated weight, denoted as $\eta_t = \sum w_t^i$ (the normalization constant) is the likelihood of the belief update at time step t. Starting with $\mathcal{L}$=0 at t=0, we maintain the likelihood over time $\mathcal{L}_t = \mathcal{L}_{t-1} + \log \eta_t$ and reinvigorate the posterior $b(\langle s, G, \dot{\chi} \rangle|\langle \vec{a}, \vec{o} \rangle, \bar{b}_0)$ whenever the $\mathcal{L}$ drops below some threshold.

3.4 Theoretical guarantees

Here we consider two theoretical aspects of our proposed solution method. The first part shows guarantees on the planning method given a particular belief, whereas the second part is concerned with guarantees on the belief itself.

We first note that FBA-POMCP converges to the optimal solution with respect to the belief:

Theorem 1. *Given a belief $b(s, G, \dot{\chi})$, FBA-POMCP converges to an ϵ-optimal value function of a FBA-POMDP: $V(b, a) \xrightarrow{p} V^*(b, a)$, where $\epsilon = \frac{precision}{1-\gamma}$.*

Proof (sketch). Analysis from [42] prove that the value function constructed by POMCP, given some suitable exploration constant, converges to the optimal value function with respect to the initial belief. Work on BA-POMCP [22] extends the proofs to the BA-POMDP. Their proof relies on the fact that the BA-POMDP is a POMDP (that ultimately can be seen as a belief MDP), and that BA-POMCP simulates experiences with respect to the dynamics $\bar{D}$. These notions are analogue to our construction of the FBA-POMDP and we can directly apply the proofs to our solution method. □

In the second result we make a claim about the quality of the belief. Previous work on importance sampling and particle filters have shown the consistency of sequential importance sampling

[2]We followed the common approach where proposal method $q(G)$ either adds or removes an edge in G. The prior over the domain specifies the set of edges that are considered by q.

that is used as the belief update [40]. Here we show that the novel particle reinvigoration method is consistent too.

Theorem 2. *Given the observed history $h_t = \langle \vec{a}_0^t, \vec{o}_0^t \rangle$ and the prior belief $b_0(s, G, \dot{\chi})$ in the FBA-POMDP constructed from a POMDP, the samples taken from the MH-within-Gibbs-reinvigoration method converge to the true distribution $p(s, G, \dot{\chi}|h_t, b_0)$ in the limit.*

Proof. This follows directly from the convergence properties of the MCMC sampling methods. The method is an instance of MH-within-Gibbs, where Gibbs is applied on the level to repeatedly sample a model and state history conditioned on the other. The state sequence is sampled directly from the conditional distribution (given the model), and the model is sampled using Metropolis-Hastings. As MH-within-Gibbs is shown to be consistent [1, 36], our reinvigoration scheme converges to the true posterior distribution. □

First note that the consistency claims on Metropolis-Hastings only hold if the proposal distribution gives a non-zero probability of moving to instances (here graph topologies) that have non-zero probability in the target [32]. By proposing to either add or remove any edge of interest, this condition is easily satisfied. Second, Metropolis-Hastings notoriously comes with an initial *burn-in phase* where one should ignore samples that were collected before the stationary distribution is reached. In practice, we avoid this phase but minimize the loss of accuracy by exploiting the fact that our initially sampled topology is taken from the current belief, assuming it is close to a local mode [32]. Lastly, these results hold only in the limit of infinite samples and therefore, under finite samples, the results may still be far for optimal. In the next section we provide an empirical evaluation and show that even with relatively few samples this approach significantly outperforms other methods.

4 EXPERIMENTS

Here we provide an empirical evaluation of our approach on three domains. Factored Tiger, an extension of the well-known Tiger problem [21], demonstrates the need to identify and exploit irrelevant features. Second a Gridworlddomain, inspired by navigational tasks, which has an additional planning challenge of long trajectories without feedback. Lastly, arguably the hardest out of three learning problems is Collision Avoidance taken from [26], where the agent must infer the dynamics of an object of which the location is never observed with high confidence.

4.1 Domains

The Tiger domain describes a scenario where the agent is faced with the task of opening one out of two doors. Behind one door lurks a tiger, a danger and reward of −100 that must be avoided, while the other door opens up to a bag of gold for a reward of 10. The agent can choose to open either doors (which ends the episode) or to listen for a signal: a noisy observation for a reward of −1. This observation informs the agent of the location of the tiger with 85% accuracy. In the **Factored Tiger** domain we increase the state space artificially by adding seven uninformative and stationary binary state features. The challenge for a learning agent is to infer the underlying dynamics in the significantly large domain.

In this particular case, the agent is unsure about the observation function. In particular, the prior belief of the agent assigns 60% expected probability to hearing the tiger correctly, as opposed to the true 85% probability. The prior belief over the structure of the observation model is uniform: each edge from any of the eight state features to the observation feature has a 50% chance of being present in a particle in the initial belief.

Gridworld is a two-dimensional grid in which the agent starts in the bottom left corner and must navigate to a goal cell. The goal cell is chosen from a set of candidates at the start of an episode, and can be fully observed by the agent. The agent additionally observes its own location with a noisy sensor. The agent can move in all four directions, which are generally successful 95% of the attempts. There are, however, specific cells that significantly decrease the chance of success to 15%, essentially trapping the agent. The target of the agent is to reach the goal as fast as possible.

In this domain we assume no prior knowledge of the location or the number of 'trap' cells and the prior assigns 95% probability of transition success on all cells. The observation model in this domain is considered known. Here we factor the state space into the index of the goal state and the (x, y) position of the agent ($s = \langle x, y, \textit{goal-index} \rangle$) and assume the agent knows that its next location is dependent on the previous. However, half of the graph structures in the prior *also* include the value of the goal cell as feature to model the agent's transition function.

In **Collision Avoidance** the agent pilots a plane that flies from right to left (one cell at a time) in a 5 by 5 grid. The agent can choose to stay level for no cost, or move either one cell diagonally with a reward of −1. The episode ends when the plane reaches the last column, where it must avoid collision with a vertically moving obstacle (or face a reward of −1000). The obstacle movement is stochastic, and the agent observes its coordinate with some noise.

While we assume the agent knows the observation and transition model of the plane, the agent initially underestimates the movement strategy of the obstacle: it believes it will stay put 90% of the time and move either direction with 5% probability each, while the actual probabilities are respectively 50% and 25%. The agent knows that the location of the obstacle in the next state depends on its previous location, but otherwise assigns a uniform prior distribution over the topology of the obstacle feature.

4.2 Experimental Setup

The analysis provides an ablation study that includes a comparison with BA-POMCP, a current state-of-the-art method. We study the choice of *model*, type of *belief update* and *planner* (table 1). We consider the BA-POMDP and FBA-POMDP models, importance sampling with and without reinvigoration belief tracking methods, and POMCP variants plus a baseline planners.

While a simple look-ahead method is the most common solution for these frameworks, it performs poorly on the relatively lengthy problems in our experiments. For an interesting comparison, we propose a more sophisticated Thompson-Sampling-inspired planner (TSI) instead. TSI runs POMCP on a single hyper-state from the belief, assuming the sampled domain state is the true current state and that the sampled model defines the true POMDP.

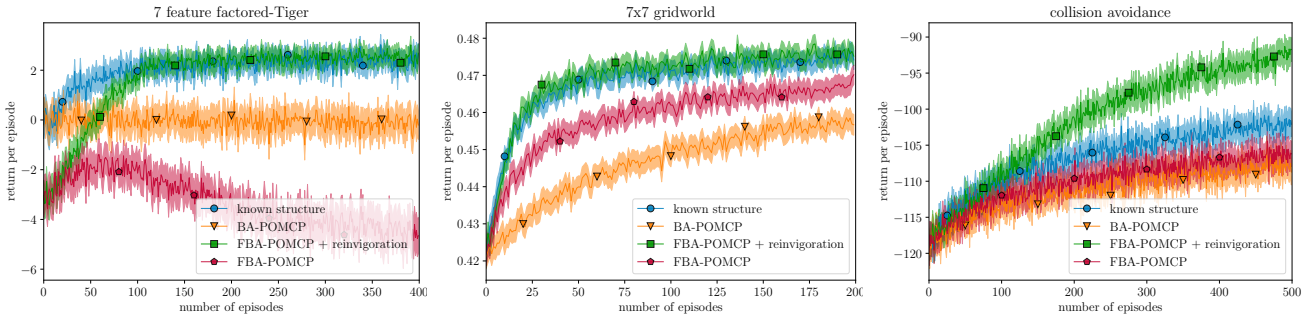

Figure 2: Return of flat vs factored models on Factored Tiger (left), Gridworld (middle) and Collision Avoidance (right)

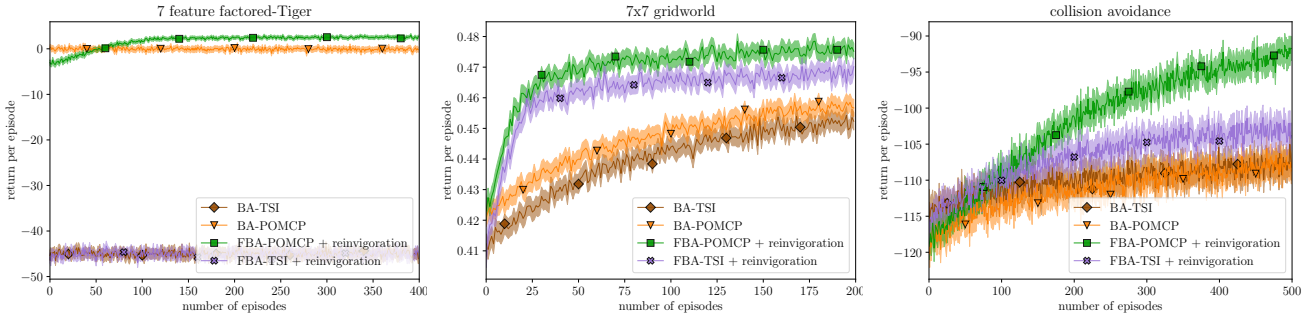

Figure 3: Return of POMCP versus TSI on Factored Tiger (left), Gridworld (middle) and Collision Avoidance (right)

Table 1: Design choices for the solution method

model:	BA-POMDP	FBA-POMDP
belief update:	importance sampling	i.s. + reinvigoration
planner:	TSI	POMCP

Table 2: Parameters per domain (sim refers to simulations)

domain	# sims	# particles	$\mathcal{L}$ threshold	UCB const
f-tiger	4096	1024	−50	100
gridworld	2048	512	−500	1
collision	256	128	−500	1000

Our method uses POMCP with importance sampling and reinvigoration applied to the Factored BA-POMDP. Methods that we compare against are FBA-POMCP, which excludes reinvigoration, and the agent known-structure with the same configurations as FBA-POMCP, but with complete knowledge on the structure of the dynamics a priori. We also consider the tabular BA-POMCP and the methods BA-TSI and FBA-TSI which apply the TSI planner on respectively the BA-POMDP and FBA-POMDP models. FBA-TSI includes reinvigoration to ensure fair comparison.

Due to the wide range of the reward functions, we ran the experiments up to 100000 times in order to produce statistically significant results. The shades in the figures indicate the 95% confidence bound on the reported returns. In these experiments, the parameters of the planning and belief update methods per domain are consistent across methods and described in table 2. The parameters were chosen to keep run time acceptable, and a complete real-time step (both planning and updating the belief) takes less than 2 seconds on average in all our experiments. All methods employ the (F)BA-POMCP modifications root-sampling and expected-transitions [22].

4.3 Results

We present the results in two sets of figures, one with the focus on model comparison and the effect of reinvigoration (Figure 2), and the other with a focus on the planning method (Figure 3).

Model comparison: The Factored representation is able to capture the dynamics with fewer parameters and, as a result, our method and the known-structure agent consistently outperforms the tabular BA-POMCP method (Figure 2). While none of the methods have converged on the Gridworld problem (center image) yet, BA-POMCP is clearly the slowest learner. This is also shown in the Collision Avoidance domain (right), where the learning rate of our method is the highest, and BA-POMCP's is the lowest. Specifically chosen to represent a problem with relatively compact underlying dynamics, the BA-POMCP is unable to learn in the Factored Tiger problem (left), whereas the known-structure agent and our method are able to distinguish the important features and the belief approaches the real model within 100 episodes.

Reinvigoration: A practical issue of particle filters is quality degeneracy. This is particularly obvious in the Factored Tiger problem,

as FBA-POMCP (red line) plummets after 50 episodes. Qualitative analysis shows that in the most likely scenarios FBA-POMCP performs on par with the other two factored approaches. However, occasionally due to poor luck, the belief converges to a posterior that concentrates on a topology where there is no edge between tiger-location-feature and the observation feature, and as a result the agent is unable to represent the true model. In those runs, the agent can only open a door randomly, leading to an average return of −45, which causes the decline in performance. While this phenomenon also happens in the other domains, the result is less dramatic, and thus less obvious. One interesting observation is that reinvigoration not only outperforms no-reinvigoration, but can also be superior to an agent that knows the correct structure a priori (the blue line known-structure for the Collision Avoidance domain). Closer inspection revealed that while reinvigoration is meant to tackle structure degeneracy, it also produces a good approximation of the counts. Given the small number of particles (128), the distribution after reinvigoration represents the belief more closely than regular importance sampling does in this domain.

Planner comparison: Figure 3 compares the performance of the POMCP planner (our method and BA-POMCP) with the baseline TSI planner (FBA-TSI and BA-TSI). The gap in performance (in favor of POMCP) indicates the importance of considering the joint uncertainty over the state and model parameters during planning, as opposed to picking an action optimal with respect to a sample of the belief. In Gridworld (center image) this uncertainty is arguably the least important, as similar states (agent coordinates) and models lead to similar policies, and thus the difference (although significant) are less pronounced. The results on Collision Avoidance, however, demonstrates the need to consider the full posterior more clearly: FBA-TSI performs as poorly as BA-POMCP, while we know the quality of its *belief* is on par with our method. Lastly, the Factored Tiger problem reveals the true nature of the TSI, as both approaches fail horribly. Since the TSI samples a single hyper-state and completely ignores the uncertainty over the current state, the optimal policy is to simply open a specific door, leading to an expected return of −45.

5 RELATED WORK

Much of the recent work in Reinforcement Learning in partially observable environments has been in applications of Deep Reinforcement Learning to POMDPs. To tackle the issue of remembering past observations, researchers have employed Recurrent networks [18, 51]. Others have introduced inductive biases into the network in order to learn a generative model to imitate belief updates [20]. While Deep Reinforcement Learning approaches are able to tackle large-scale problems, these approaches often require millions of interactions with the real world. Another of their main drawbacks is that they do not address the fundamental challenge of the exploration-exploitation trade-off in POMDPs.

More traditional approaches including the U-Tree algorithm [29] (and its modifications), EM-based algorithms such as [25] and policy gradient methods [4], typically do not suffer from the same lack of sample efficiency. They too, however, have similar issues solving the exploration-exploitation trade-off.

Bayesian methods are a good fit for domain where solutions must be learned quickly, as they both address exploration-exploitation in a principled fashion, and allow the user to utilize domain knowledge in the form of a prior distribution. The Infinite-POMDP [12] (iPOMDP), for example, models the probability distribution over the dynamics as a posterior over the space of HMMs. In doing so, the iPOMDP additionally relaxes the assumption that the state space is known, tackling an even more general setting. This complicates the specification of a prior, making it more difficult to encode knowledge. Other BRL methods solve the case of continuous state space, taking on Gaussian assumptions over the dynamics [10, 37].

Work on generalization in model-based BRL methods include [34], which introduces 'tied' parameters, hard-coded sets of states to share transition probabilities. This idea is extended [48] to maintaining a weighted mixture of increasingly 'tied' models. The FBA-MDP [38] learns the transition model as a set of BNs and has been the inspiration of the MH part of our reinvigoration method.

6 CONCLUSION

This paper pushes the state of the art in model-based Bayesian reinforcement learning for partially observable settings. As we demonstrated, such methods can exploit prior information to allow for learning in hundreds rather than millions of episodes. Despite their advantage, previous model-based BRL methods for partially observable settings, such as the BA-POMDP, faced a scalability bottleneck due to their tabular nature.

To overcome this bottleneck we introduced the FBA-POMDP framework, which exploits factored representations to compactly describe the belief over the dynamics of the underlying POMDP. And in order to effectively solve the FBA-POMDP, we introduced a novel particle reinvigorating algorithm to track the complicated belief and paired it with FBA-POMCP, a new Monte-Carlo Tree Search-based planning algorithm. We proved that this method, in the limit of infinite samples, is guaranteed to converge to the optimal policy with respect to the initial belief. In an empirical evaluation we demonstrated that our structure-learning approach is roughly as effective as learning with given structure in two domain, and, surprisingly, even *more effective* on the collision avoidance domain. The results also show the significance of representing and recognizing independent features, as our method either outperforms BA-POMDP based agents or is able to learn in scenarios where tabular methods are not feasible at all.

In order to further scale these methods up future work can take several interesting directions. For domains too large to represent with Bayes Networks one could investigate other models to capture the dynamics. For domains that require learning over long sequences, reinvigoration methods that scale more gracefully with history length would be desirable

ACKNOWLEDGEMENTS

Christopher Amato and Sammie Katt are funded by NSF Grant #1734497, F.A.O. is funded by EPSRC First Grant EP/R001227/1. This project received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 758824 —INFLUENCE).

REFERENCES

[1] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. 2003. An introduction to MCMC for machine learning. *Machine learning* 50, 1-2 (2003), 5–43.

[2] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. 2002. Finite-time analysis of the multiarmed bandit problem. In *Machine Learning*, Vol. 47. 235–256.

[3] Kamyar Azizzadenesheli, Emma Brunskill, and Animashree Anandkumar. 2018. Efficient exploration through Bayesian deep Q-networks. In *Information Theory and Applications Workshop*. 1–9.

[4] Jonathan Baxter and Peter L Bartlett. 2000. Direct gradient-based Reinforcement Learning. In *IEEE International Symposium on Circuits and Systems*, Vol. 3. 271–274.

[5] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. 2016. Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems*. 1471–1479.

[6] Craig Boutilier, Thomas Dean, and Steve Hanks. 1999. Decision-theoretic planning: Structural assumptions and computational leverage. In *Journal of Artificial Intelligence Research*, Vol. 11. 1–94.

[7] Craig Boutilier and David Poole. 1996. Computing optimal policies for partially observable decision processes using compact representations. In *AAAI Conference on Artificial Intelligence*. 1168–1175.

[8] Cameron B Browne, Edward Powley, Daniel Whitehouse, Simon M Lucas, Peter I Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis, and Simon Colton. 2012. A survey of monte carlo tree search methods. In *IEEE Transactions on Computational Intelligence and AI in games*, Vol. 4. 1–43.

[9] Yuguo Chen. 2005. Another look at rejection sampling through importance sampling. In *Statistics & probability letters*, Vol. 72. 277–283.

[10] Patrick Dallaire, Camille Besse, Stephane Ross, and Brahim Chaib-draa. 2009. Bayesian reinforcement learning in continuous POMDPs with Gaussian processes. In *International Conference on Intelligent Robots and Systems*. 2604–2609.

[11] Finale Doshi-Velez. 2009. The infinite partially observable Markov Decision Process. In *Advances in Neural Information Processing Systems*. 477–485.

[12] Finale Doshi-Velez, David Pfau, Frank Wood, and Nicholas Roy. 2015. Bayesian nonparametric methods for partially-observable Reinforcement Learning. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 37. 394–407.

[13] Zhengzhu Feng and Eric A Hansen. 2014. Approximate planning for factored POMDPs. In *European Conference on Planning*.

[14] Neil J Gordon, David J Salmond, and Adrian FM Smith. 1993. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In *IEE Proceedings F*, Vol. 140. 107–113.

[15] Carlos Guestrin, Daphne Koller, and Ronald Parr. 2001. Solving factored POMDPs with linear value functions. In *Workshop on Planning under Uncertainty and Incomplete Information*.

[16] Eric A Hansen and Zhengzhu Feng. 2000. Dynamic Programming for POMDPs Using a Factored State Representation. In *Artificial Intelligence Planning Systems*. 130–139.

[17] W Keith Hastings. 1970. Monte Carlo sampling methods using Markov chains and their applications. In *Biometrika*, Vol. 57. 97–109.

[18] Matthew Hausknecht and Peter Stone. 2015. Deep recurrent q-learning for partially observable MDPS. In *AAAI Conference on Artificial Intelligence Fall Symposium Series*.

[19] David Heckerman, Dan Geiger, and David M Chickering. 1995. Learning Bayesian networks: The combination of knowledge and statistical data. In *Machine Learning*, Vol. 20. 197–243.

[20] Maximilian Igl, Luisa Zintgraf, Tuan Anh Le, Frank Wood, and Shimon Whiteson. 2018. Deep Variational Reinforcement Learning for POMDPs. In *International Conference on Machine Learning*. 2117–2126.

[21] Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. 1998. Planning and acting in partially observable stochastic domains. In *Artificial intelligence*, Vol. 101. 99–134.

[22] Sammie Katt, Frans A Oliehoek, and Christopher Amato. 2017. Learning in POMDPs with Monte Carlo Tree Search. In *International Conference on Machine Learning*. 1819–1827.

[23] Dominic S Lee and Nicholas KK Chia. 2002. A particle algorithm for sequential Bayesian parameter estimation and model selection. In *IEEE Transactions on Signal Processing*, Vol. 50. 326–336.

[24] Faming Liang, Chuanhai Liu, and Raymond Carroll. 2011. *Advanced Markov chain Monte Carlo methods: learning from past samples*. John Wiley & Sons.

[25] Miao Liu, Xuejun Liao, and Lawrence Carin. 2013. Online Expectation Maximization for Reinforcement Learning in POMDPs. In *International Joint Conference on Artificial Intelligence*. 1501–1507.

[26] Yuanfu Luo, Haoyu Bai, David Hsu, and Wee Sun Lee. 2018. Importance sampling for online planning under uncertainty. *The International Journal of Robotics Research* 38, 2-3 (2018), 162–81.

[27] Luca Martino, Jesse Read, and David Luengo. 2015. Independent Doubly Adaptive Rejection Metropolis Sampling Within Gibbs Sampling. In *IEEE Transactions on Signal Processing*, Vol. 63. 3123–3138.

[28] David A McAllester and Satinder Singh. 1999. Approximate planning for factored POMDPs using belief state simplification. In *Uncertainty in Artificial Intelligence*. 409–416.

[29] Andrew Kachites McCallum and Dana Ballard. 1996. *Reinforcement Learning with selective perception and hidden state*. Ph.D. Dissertation. University of Rochester. Dept. of Computer Science.

[30] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *Deep Learning Workshop* (2013).

[31] Felix Müller, Christian Späth, Thomas Geier, and Susanne Biundo. 2012. Exploiting expert knowledge in factored POMDPs. In *European Conference on Artificial Intelligence*. 606–611.

[32] Kevin P Murphy. 2012. *Machine learning: a probabilistic perspective*. MIT press.

[33] Ian Osband, Daniel Russo, Zheng Wen, and Benjamin Van Roy. 2017. Deep exploration via randomized value functions. *arXiv preprint arXiv:1703.07608* (2017).

[34] Pascal Poupart, Nikos Vlassis, Jesse Hoey, and Kevin Regan. 2006. An analytic solution to discrete Bayesian reinforcement learning. In *International Conference on Machine Learning*. 697–704.

[35] Lawrence R Rabiner and Biing-Hwang Juang. 1986. An introduction to hidden Markov models. In *IEEE ASSP magazine*, Vol. 3. 4–16.

[36] Christian Robert and George Casella. 2013. *Monte Carlo statistical methods*. Springer Science & Business Media.

[37] Stephane Ross, Brahim Chaib-draa, and Joelle Pineau. 2008. Bayesian reinforcement learning in continuous POMDPs with application to robot navigation. In *IEEE International Conference on Robotics and Automation*. 2845–2851.

[38] Stéphane Ross and Joelle Pineau. 2008. Model-based Bayesian reinforcement learning in large structured domains. In *Uncertainty in Artificial Intelligence*. 476.

[39] Stéphane Ross, Joelle Pineau, Brahim Chaib-draa, and Pierre Kreitmann. 2011. A Bayesian approach for Learning and planning in partially observable Markov Decision Processes. In *The Journal of Machine Learning Research*, Vol. 12. 1729–1770.

[40] Stuart J Russell and Peter Norvig. 2016. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,.

[41] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. 2016. Mastering the game of Go with deep neural networks and tree search. In *Nature*, Vol. 529. 484.

[42] David Silver and Joel Veness. 2010. Monte-Carlo planning in large POMDPs. In *Advances in Neural Information Processing Systems*. 2164–2172.

[43] Hyeong Seop Sim, Kee-Eung Kim, JinHyung Kim, D-S Chang, and M-W Koo. 2008. Symbolic heuristic search value iteration for factored POMDPs. In *AAAI Conference on Artificial Intelligence*. 1088–1093.

[44] Sebastian Thrun. 1999. Monte Carlo POMDPs. In *Advances in Neural Information Processing Systems*. 1064–1070.

[45] Luke Tierney. 1994. Markov chains for exploring posterior distributions. In *The Annals of Statistics*, Vol. 22. 1701–1728.

[46] Hado Van Hasselt, Arthur Guez, and David Silver. 2016. Deep Reinforcement Learning with Double Q-Learning. In *AAAI Conference on Artificial Intelligence*. 2094–2100.

[47] Tiago Veiga, Matthijs Spaan, and Pedro Lima. 2014. Point-based POMDP solving with factored value function approximation. In *AAAI Conference on Artificial Intelligence*. 2513–2519.

[48] Ngo Anh Vien, Wolfgang Ertel, Viet-Hung Dang, and TaeChoong Chung. 2013. Monte-Carlo tree search for Bayesian reinforcement learning. *Applied intelligence* 39, 2 (2013), 345–353.

[49] Jason D Williams, Pascal Poupart, and Steve Young. 2005. Factored partially observable Markov Decision Processes for dialogue management. In *Workshop on Knowledge and Reasoning in Practical Dialogue Systems*. 76–82.

[50] Jeremy L Wyatt. 2001. Exploration control in Reinforcement Learning using optimistic model selection. In *International Conference on Machine Learning*. 593–600.

[51] Pengfei Zhu, Xin Li, Pascal Poupart, and Guanghui Miao. 2018. On improving deep reinforcement learning for pomdps. *arXiv preprint arXiv:1804.06309* (2018).