Stick-Breaking Policy Learning in Dec-POMDPs

Miao Liu
MIT
Cambridge, MA
miaoliu@mit.edu

Christopher Amato
University of New Hampshire
Durham, NH
camato@cs.unh.edu

Xuejun Liao, Lawrence Carin
Duke University
Durham, NC
{xjliao,lcarin}@duke.edu

Jonathan P. How
MIT
Cambridge, MA
jhow@mit.edu

Abstract

Expectation maximization (EM) has recently been shown to be an efficient algorithm for learning finite-state controllers (FSCs) in large decentralized POMDPs (Dec-POMDPs). However, current methods use fixed-size FSCs and often converge to maxima that are far from the optimal value. This paper represents the local policy of each agent using variable-sized FSCs that are constructed using a stick-breaking prior, leading to a new framework called *decentralized stick-breaking policy representation* (Dec-SBPR). This approach learns the controller parameters with a variational Bayesian algorithm without having to assume that the Dec-POMDP model is available. The performance of Dec-SBPR is demonstrated on several benchmark problems, showing that the algorithm scales to large problems while outperforming other state-of-the-art methods.

1 Introduction

Decentralized partially observable Markov decision processes (Dec-POMDPs) [Amato *et al.*, 2013; Oliehoek, 2012] provide a general framework for solving the cooperative multi-agent sequential decision-making problems that arise in numerous applications, including robotic soccer [Messias *et al.*, 2010], transportation [Amato *et al.*, 2015], extraplanetary exploration [Bernstein *et al.*, 2001], and traffic control [Wu *et al.*, 2013]. Dec-POMDPs can be viewed as a POMDP controlled by multiple distributed agents. These agents make decisions based on their own local streams of information (i.e., observations), and their joint actions control the global state dynamics and the expected reward of the team. Because of the decentralized decision-making, an individual agent generally does not have enough information to compute the global belief state, which is a sufficient statistic for decision making in POMDPs. This makes generating an optimal solution in a Dec-POMDP more difficult than for a POMDP [Bernstein *et al.*, 2002], especially for long planning horizons.

To circumvent the difficulty of solving long-horizon Dec-POMDPs optimally, while still generating a high quality policy, this paper presents scalable learning methods using a finite memory policy representation. For infinite-horizon problems (which continue for an infinite number of steps), significant progress has been made with agent policies represented as finite-state controllers (FSCs) that map observation histories to actions [Bernstein *et al.*, 2009; Amato *et al.*, 2010]. Recent work has shown that expectation-maximization (EM) [Dempster *et al.*, 1977] is a scalable method for generating controllers for large Dec-POMDPs [Kumar and Zilberstein, 2010; Pajarinen and Peltonen, 2011]. In addition, EM has also been shown to be an efficient algorithm for policy-based reinforcement learning (RL) in Dec-POMDPs, where agents learn FSCs based on trajectories, without knowing or learning the Dec-POMDP model [Wu *et al.*, 2013].

An important and yet unanswered question is how to define an appropriate number of nodes in each FSC. Previous work assumes a fixed FSC size for each agent, but the number of nodes affects both the quality of the policies and the convergence rate. When the number of nodes is too small, the FSC is unable to represent the optimal policy and therefore will quickly converge to a sub-optimal result. By contrast, when the number is too large, the FSC overfits data, often yielding slow convergence and, again, a sub-optimal policy.

This paper uses a Bayesian nonparametric approach to determine the appropriate controller size in a variable-size FSC. Following previous methods [Wu *et al.*, 2013; Oliehoek, 2012], learning is assumed to be centralized, and execution is decentralized. That is, learning is accomplished offline based on all available information, but the optimization is only over decentralized solutions. Such a controller is constructed using the stick-breaking (SB) prior [Ishwaran and James, 2001]. The SB prior allows the number of nodes to be variable, but the set of nodes that is actively used by the controller is encouraged to be compact. The nodes that are actually used are determined by the posterior, combining the SB prior and the information from trajectory data. The framework is called the *decentralized stick-breaking policy representation* (Dec-SBPR) to recognize the role of the SB prior.

In addition to the use of variable-size FSCs, the paper also makes several other contributions. Specifically, our algorithm directly operates on the (shifted) empirical value function of Dec-POMDPs, which is simpler than the likelihood functions (a mixture of dynamic Bayes nets (DBNs)) in existing planning-as-inference frameworks [Kumar and Zilberstein, 2010; Wu *et al.*, 2013]. Moreover, we derive a variational Bayesian (VB) algorithm for learning the Dec-SBPR based only on the agents' trajectories (or episodes) of actions, observations, and rewards. The VB algorithm is linear in the number of agents and at most quadratic in the problem size, and is therefore scalable to large application domains. In practice, these trajectories can be generated by a simulator or a set of real-world experiences that are provided, and this batch data scenario is general and realistic, as it is widely adopted in learning from demonstration [Martins and Demiris, 2010], and reinforcement learning. To the best of our knowledge, this is the first application of Bayesian nonparametric methods to the difficult and little-studied problem of policy-based RL in Dec-POMDPs.

2 Background and Related Work

Before introducing the proposed method, we first describe the Dec-POMDP model and some related work.

2.1 Decentralized POMDPs

A Dec-POMDP can be represented as a tuple $\mathcal{M} = \langle \mathcal{N}, \mathcal{A}, \mathcal{S}, \mathcal{O}, \mathcal{T}, \Omega, \mathcal{R}, \gamma \rangle$, where $\mathcal{N} = \{1, \cdots, N\}$ is a finite set of agent indices; $\mathcal{A} = \otimes_n \mathcal{A}_n$ and $\mathcal{O} = \otimes_n \mathcal{O}_n$ respectively are sets of joint actions and observations, with $\mathcal{A}_n$ and $\mathcal{O}_n$ available to agent n. At each step, a joint action $\vec{a} = (a_1, \cdots, a_N) \in \mathcal{A}$ is selected and a joint observation $\vec{o} = (o_1, \cdots, o_N)$ is received; $\mathcal{S}$ is a set of finite world states; $\mathcal{T} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state transition function with $\mathcal{T}(s'|s, \vec{a})$ denoting the probability of transitioning to s' after taking joint action $\vec{a}$ in s; $\Omega : \mathcal{S} \times \mathcal{A} \times \mathcal{O} \to [0, 1]$ is the observation function with $\Omega(\vec{o}|s', \vec{a})$ the probability of observing $\vec{o}$ after taking joint action $\vec{a}$ and arriving in state s'; $\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function with $r(s, \vec{a})$ the immediate reward received after taking joint action $\vec{a}$ in s; $\gamma \in [0, 1)$ is a discount factor. A global reward signal is generated for the team of agents after joint actions are taken, but each agent only observes its local observation. Because each agent lacks access to other agents' observations, each agent maintains a local policy π_n, defined as a mapping from local observation histories to actions. A joint policy consists of the local policies of all agents. For an infinite-horizon Dec-POMDP with initial belief state b_0, the objective is to find a joint policy $\Psi = \otimes_n \Psi_n$, such that the value of Ψ starting from b_0, $V^{\Psi}(b(s_0)) = \mathbb{E}\big[\sum_{t=0}^{\infty} \gamma^t r(s_t, \vec{a}_t) | b_0, \Psi\big]$, is maximized.

An FSC is a compact way to represent a policy as a mapping from histories to actions. Formally, a stochastic FSC for agent n is defined as a tuple $\Theta_n = \langle \mathcal{A}_n, \mathcal{O}_n, \mathcal{Z}_n, \mu_n, W_n, \pi_n \rangle$, where, $\mathcal{A}_n$ and $\mathcal{O}_n$ are the same as defined in the Dec-POMDP; $\mathcal{Z}_n$ is a finite set of controller nodes for agent n; μ_n is the initial node distribution with μ_n^z the probability of agent n initially being in z; W_n is a set of Markov transition matrices with $W_{n,a,o}^{z,z'}$ denoting the probability of the controller node transiting from z to z' when agent n takes action a in z and sees observation o; π_n is a set of stochastic policies with $\pi_{n,z}^a$ the probability of agent n taking action a in z.

For simplicity, we use the following notational conventions. $\mathcal{Z}_n = \{1, 2, \cdots, C_n\}$, where $C_n \overset{\text{def}}{=} |\mathcal{Z}_n|$ is the cardinality of $\mathcal{Z}_n$, and $\mathcal{A}_n$ and $\mathcal{O}_n$ follow similarly. $\Theta = \{\Theta_1, \cdots, \Theta_N\}$ is the joint FSC of all agents. A consecutively-indexed variable is abbreviated as the variable with the index range shown in the subscript or superscript; when the index range is obvious from the context, a simple ":" is used instead. Thus, $a_{n,0:T} = (a_{n,0}, a_{n,1}\ , \ldots, a_{n,T})$ represents the actions of agent n from step 0 to T and $W_{n,a,o}^{z,:} = (W_{n,a,o}^{z,1}, W_{n,a,o}^{z,2}, \cdots, W_{n,a,o}^{z,|\mathcal{Z}_n|})$ represents the node transition probabilities for agent n when starting in node z, taking action a and seeing observation o. Given $h_{n,t} = \{a_{n,0:t-1}, o_{n,1:t}\}$, a local history of actions and observations up to step t, as well as an agent controller, Θ_n, we can calculate a local policy $p(a_{n,t}|h_{n,t},\Theta_n)$, the probability that agent n chooses its action $a_{n,t}$.

2.2 Planning as Inference in Dec-POMDPs

A Dec-POMDP planning problem can be transformed into an inference problem and then efficiently solved by EM algorithms. The validity of this method is based on the fact that by introducing binary rewards R such that $P(R = 1|a, s) \propto r(a, s), \forall a \in \mathcal{A}, s \in \mathcal{S}$ and choosing the geometric time prior $p(T) = \gamma^T(1 - \gamma)$, maximizing the likelihood $L(\Theta) = P(R = 1; \Theta) = \sum_{T=0}^{\infty} P(T) P(R = 1|T; \Theta)$ of a mixture of dynamic Bayes nets is equivalent to optimizing the associated Dec-POMDP policy, as the joint-policy value $V(\Theta)$ and $L(\Theta)$ are related through an affine transform [Kumar and Zilberstein, 2010]

$$L(\Theta) = \frac{(1-\gamma)(V(\Theta) - \sum_T \gamma^T R_{\min})}{R_{\max} - R_{\min}} = \frac{(1-\gamma)\hat{V}(\Theta)}{R_{\max} - R_{\min}}, \quad (1)$$

where $R_{\max}$ and $R_{\min}$ are maximum and minimum reward values and $\hat{V}(\Theta) \overset{\text{def}}{=} V(\Theta) - \sum_T \gamma^T R_{\min}$ is a shifted value.

Previous EM methods [Kumar and Zilberstein, 2010; Pajarinen and Peltonen, 2011] have achieved success in scaling to larger problems by factoring the distribution over states and histories for inference, but these methods require using a POMDP model to construct a Bayes net for policy evaluation. When the exact model parameters $\mathcal{T}$, Ω and $\mathcal{R}$ are unknown, one needs to solve a reinforcement learning (RL) problem. To address this important yet less addressed problem, a global empirical value function extended from the single-agent case [Li *et al.*, 2009], is constructed based on all the action, observation and reward trajectories, and the product of local policies of all agents. This serves as the basis for learning (fixed-size) FSCs in RL settings.

Definition 1. *(Global empirical value function) Let* $\mathcal{D}^{(K)} = \{(\vec{o}_0^k \vec{a}_0^k r_0^k \vec{o}_1^k \vec{a}_1^k r_1^k \cdots \vec{o}_{T_k}^k \vec{a}_{T_k}^k\ r_{T_k}^k)\}_{k=1,\cdots,K}$ *be a set of episodes resulting from* N *agents who choose actions according to* $\Psi = \otimes_n \Psi_n$*, a set of stochastic behavior policies with* $p^{\Psi_n}(a|h) > 0$, $\forall$ *action* a, $\forall$ *history* h*. The global empirical value function is defined as* $\hat{V}\big(\mathcal{D}^{(K)}; \Theta\big) \overset{\text{def}}{=} \sum_{k=1}^{K} \sum_{t=0}^{T_k} \frac{\gamma^t (r_t^k - R_{\min})}{K} \frac{\prod_{\tau=0}^{t} \prod_{n=1}^{N} p(a_{n,\tau}^k | h_{n,\tau}^k, \Theta_n)}{\prod_{\tau=0}^{t} \prod_{n=1}^{N} p^{\Psi_n}(a_{n,\tau}^k | h_{n,\tau}^k)}$ *where* $h_{n,t}^k = (a_{n,0:t-1}^k, o_{n,1:t}^k)$, $0 \leq \gamma < 1$ *is the discount.*

According to the strong law of large numbers [Robert and Casella, 2004], $\hat{V}(\Theta) = \lim_{K \to \infty} \hat{V}\big(\mathcal{D}^{(K)}; \Theta\big)$, i.e., with a large number of trajectories, the empirical value function $\hat{V}\big(\mathcal{D}^{(K)}; \Theta\big)$ approximates $\hat{V}(\Theta)$ accurately. Hence, applying (1), $\hat{V}\big(\mathcal{D}^{(K)}; \Theta\big)$ approximates $L(\Theta)$, and offers an objective for learning the decentralized policies and can be directly maximized by the EM algorithms in [Li *et al.*, 2009].

3 Bayesian Learning of Policies

EM algorithms infer policies based on fixed-size representation and observed data only, it is difficult to explicitly handle model uncertainty and encode prior (or expert) knowledge. To address these issues, a Bayesian learning method is proposed in this section. This is accomplished by measuring the likelihood of Θ using $L\big(\mathcal{D}^{(K)};\Theta\big)$, which is combined with the prior $p(\Theta)$ in Bayes' rule to yield the posterior

$$p(\Theta|\mathcal{D}^{(K)}) = L(\mathcal{D}^{(K)};\Theta)p(\Theta)\left[p\big(\mathcal{D}^{(K)}\big)\right]^{-1}, \qquad (2)$$

where $p\big(\mathcal{D}^{(K)}\big)$ is the marginal likelihood of the joint FSC and, up to additive constant, proportional to the marginal value function,

$$\begin{aligned}\hat{V}\big(\mathcal{D}^{(K)}\big) &\overset{\text{def}}{=} \int \hat{V}\big(\mathcal{D}^{(K)};\Theta\big)p(\Theta)d\Theta \\ &\propto \textstyle\int L\big(\mathcal{D}^{(K)};\Theta\big)p(\Theta)d\Theta = p\big(\mathcal{D}^{(K)}\big). \qquad (3)\end{aligned}$$

To compute the posterior, $p(\Theta|\mathcal{D}^{(K)})$, Markov chain Monte Carlo (MCMC) simulation [Robert and Casella, 2004] is the most straight forward method. However, MCMC is costly in terms of computation and storage, and lacks a strong convergence guarantee. An alternative is a variational Bayes (VB) method [Beal, 2003], which performs approximate posterior inference by minimizing the Kullback-Leibler (KL) divergence between the true and approximate posterior distributions. Because the VB method has a (local) convergence guarantee and is able to trade-off scalability and accuracy, we focus on the derivation of VB method here. Denoting $q(\Theta)$ as the variational approximation to $p(\Theta|\mathcal{D}^{(K)})$, and $q_t^k(\vec{z}_{0:t}^k)$ as the approximation to $p(\vec{z}_{0:t}^k|\vec{o}_{1:t}^k,\Theta)$, a VB objective function [1] is

$$\begin{aligned}\mathrm{KL}\big(\{q_t^k(\vec{z}_{0:t}^k)q(\Theta)\}_{k=1:K}||\{\nu_t^k p(\vec{z}_{0:t}^k,\Theta)\}_{k=1:K}\big) = \\ \ln\hat{V}\big(\mathcal{D}^{(K)}\big) - \mathrm{LB}\big(\{q_t^k(\vec{z}_{0:t}^k)\},q(\Theta)\big), \qquad (4)\end{aligned}$$

where

$$\begin{aligned}&\mathrm{LB}\big(\{q_t^k(\vec{z}_{0:t}^k)\},q(\Theta)\big) \overset{\text{def}}{=} \\ &\sum_{k,t,z_{1:N,0:t}^k}\int \frac{q_t^k(\vec{z}_{0:t}^k)}{K/q(\Theta)}\ln\frac{\nu_t^k p(\vec{z}_{0:t}^k,\Theta|h_{0:t}^k)}{q_t^k(\vec{z}_{0:t}^k)q(\Theta)}d\Theta \qquad (5)\end{aligned}$$

is the lower bound of $\ln\hat{V}\big(\mathcal{D}^{(K)}\big)$ and

$$\nu_t^k \overset{\text{def}}{=} \frac{\gamma^t r_t^k\prod_{n=1}^N p(a_{n,0:t}^k|o_{n,1:t}^k,\Theta)}{\prod_{n=1}^N\prod_{\tau=0}^t p^{\Psi_n}(a_{n,\tau}^k|h_{n,\tau}^k)\widehat{V}(\mathcal{D}^{(K)},\Theta)}, \forall t,k, \qquad (6)$$

is the re-weighted reward. Since $\ln\hat{V}\big(\mathcal{D}^{(K)}\big)$ in equation (4) is independent of Θ and $\{q_t^k(\vec{z}_{0:t}^k)\}$, minimizing the KL divergence is equivalent to maximizing the lower bound, leading to the following constrained optimization problem,

$$\begin{aligned}&\max_{\{q_t^k(\vec{z}_{0:t}^k)\}q(\Theta)} \ \mathrm{LB}\big(\{q_t^k(\vec{z}_{0:t}^k)\},q(\Theta)\big) \\ &\text{subject to: } q_t^k(\vec{z}_{0:t}^k,\Theta) = \textstyle\prod_{n=1}^N q_t^k(z_{n,0:t}^k)q(\Theta_n), \\ &\sum_{k=1}^K\sum_{t=0}^{T_k}\sum_{z_{1:N,0:t}^k=1}^{|\mathcal{Z}|} q_t^k(\vec{z}_{0:t}^k) = K, \quad q_t^k(\vec{z}_{0:t}^k) \geq 0, \forall\vec{z}_t^k,t,k, \\ &\textstyle\int p(\Theta)d\Theta = 1 \text{ and } p(\Theta)\geq 0,\forall\Theta \qquad (7)\end{aligned}$$

where the constraint in the second line arises both from the mean-field approximation and from the decentralized policy representation, and the last two lines summarize the normalization constraints. It is worth emphasizing that we developed this variational mean-field approximation to optimize a decentralized policy representation, showing that the VB learning problem formulation (7) is both a general and accurate method for the multiagent problem considered in this paper.

3.1 Stick-breaking Policy Priors

To solve the Bayesian learning problem described above and obtain the variable-size FSCs, the stick-breaking prior is used to specify the policy's structure. As such, Dec-SBPR is formally given in definition 2.

Definition 2. *The decentralized stick breaking policy representation (Dec-SBPR) is a tuple* $(\mathcal{N},\mathcal{A},\mathcal{O},\mathcal{Z},\mu,\eta,\rho)$*, where* $\mathcal{N},\mathcal{A}$ *and* $\mathcal{O}$ *are as in the definition of Dec-POMDP;* $\mathcal{Z}$ *is an* unbounded *set of nodes indexed by positive integers; for notational simplicity*[2]*,*μ *are assumed to be deterministic with* $\mu_n^1 = 1,\mu_n^{2:\infty} = 0,\forall n$*;* (η,ρ) *determine* (W,π)*, the FSC parameters defined in section 2.1, as follows*

$$W_{n,a,o}^{i,1:\infty}\sim\mathrm{SB}(\sigma_{n,a,o}^{i,1:\infty},\eta_{n,a,o}^{i,1:\infty}),\quad \pi_{n,i}^{1:|\mathcal{A}_n|}\sim\mathrm{Dir}(\rho_{n,i}^{1:|\mathcal{A}_n|}) \qquad (8)$$

where Dir *represents Dirichlet distribution and* SB *represents the stick-breaking process with* $W_{n,a,o}^{i,j} = V_{n,a,o}^{i,j}\prod_{m=1}^{j-1}(1-V_{n,a,o}^{i,m})$ *and* $V_{n,a,o}^{i,j}\sim\mathrm{Beta}(\sigma_{n,a,o}^{i,j},\eta_{n,a,o}^{i,j})$*,* $\eta_{n,a,o}^{i,j}\sim\mathrm{Gamma}(c,d), n=1,\cdots,N$ *and* $i,j=1,\cdots,\infty$*.*

DECSBPR differs from previous nonparametric Bayesian RL methods [Liu *et al.*, 2011; Doshi-Velez *et al.*, 2010]. Specifically, Dec-SBPR performs policy-based RL and generalizes the nonparametric Bayesian policy representation of POMDPs [Liu *et al.*, 2011] to the decentralized domain. Whereas [Doshi-Velez *et al.*, 2010] is a model-based RL method that doesn't assume knowledge about the world's model, but explicitly learns it and then performs planning. Moreover, Dec-SBPR further distinguishes from previous methods [Doshi-Velez *et al.*, 2010; Liu *et al.*, 2011] by the prior distributions and inference methods employed. These previous methods employed hierarchical Dirichlet processes hidden Markov models (HDP-HMM) to infer the number of controller nodes. However, due to the lack of conjugacy between two levels of DPs in the HDP-HMM, a *fully conjugate Bayesian* variational inference does not exist[3]. Therefore, these methods used MCMC which requires high computational and storage costs, making them not ideal for solving large problems. In contrast, Dec-SBPR employs single layer SB priors over FSC transition matrices W and sparse Gamma priors over SB weight hyperparameters η to bias transition among nodes with smaller indices. Although a similar framework has been explored to infer hidden Markov models(HMMs) [Paisley and Carin, 2009] , it has never been used to model the uncertainty of policy representations in multiagent planning and RL problems.

[1] Refer to the supplemental material [URL, b] for more details.

[2] Nonparametric priors over μ can also be used.

[3] The VB method in [Bryant and Sudderth, 2012] imposes point-mass proposals over top level DPs, lacking a uncertainty measure.

Algorithm 1 Batch VB Inference for Dec-SBPR

```
Input: Episodes $\mathcal{D}^{(K)}$, the number of agents $N$, initial policies $\Theta$, VB lower bound $\mathrm{LB}_0 = -\mathrm{Inf}$, $\Delta\mathrm{LB} = 1$, $\mathrm{Iter} = 0$;
while $\Delta\mathrm{LB} > 10^{-3}$ do
    for $k = 1$ to $K$, $n = 1$ to $N$ do
        Update the global rewards $\{\hat{\nu}_t^k\}$ using (9).
        Compute $\{\alpha_\tau^{n,k}\}$ and $\{\beta_{t,\tau}^{n,k}\}$.
    end for
    $\mathrm{Iter} = \mathrm{Iter} + 1$.
    Compute $\mathrm{LB}_{\mathrm{Iter}}$ using (5)
    $\Delta\mathrm{LB} = (\mathrm{LB}_{\mathrm{Iter}} - \mathrm{LB}_{\mathrm{Iter}-1})/|\mathrm{LB}_{\mathrm{Iter}-1}|$
    for $n = 1$ to $N$ do
        Compute $\{\xi_{t,\tau}^{n,k}(i,j)\}$ and $\{\phi_{t,\tau}^{n,k}(i)\}$ using (11).
        Update the hyper-parameters of $\Theta_n$ using (10) .
        Compute $|\mathcal{Z}_n|$ using (13).
    end for
end while
Return: Policies $\{\Theta_n\}_{n=1}^N$, and controller sizes $\{|\mathcal{Z}_n|\}_{n=1}^N$.
```

Table 1: Computational Complexity of Algorithm 1.

VAR	BEST CASE	WORST CASE
α	$\Omega(N\|\mathcal{Z}\|^2KT)$	$O(N\|\mathcal{Z}\|^2KT)$
β	$\Omega(N\|\mathcal{Z}\|^2KT)$	$O(N\|\mathcal{Z}\|^2KT^2)$
ν_t^k	$\Omega(K)$	$O(KT)$
Θ	$\Omega(N\|\mathcal{Z}\|^2KT)$	$O(N\|\mathcal{Z}\|^2KT^2)$
LB	$\Omega(\|\mathcal{Z}\|^2\sum_{n=1}^N\|\mathcal{A}_n\|\|\mathcal{O}_n\|)$	$O(\|\mathcal{Z}\|^2\sum_{n=1}^N\|\mathcal{A}_n\|\|\mathcal{O}_n\|)$

It is also worth noting that SB processes subsume Dirichlet Processes (DPs) [Ferguson, 1973] as a special case, when $\sigma_{n,a,o}^{i,j}{=}1, \forall i,j,n,a,o$ (in Dec-SBPR). The purpose of using SB priors is to encourage a small number of FSC nodes. Compared to a DP, the SB priors can represent richer patterns of sparse transition between the nodes of an FSC, because it allows arbitrary correlation between the stick-breaking weights (the weights are always negatively correlated in a DP).

3.2 Variational Stick-breaking Policy Inference

It is shown in [Ishwaran and James, 2001] that the random weights constructed by the SB prior are equivalently governed by a generalized Dirichlet distribution (GDD) and are therefore conjugate to the multinomial distribution; hence an efficient variational Bayesian algorithm for learning the decentralized policies can be derived. To accommodate an unbounded number of nodes, we apply the retrospective representation of SB priors [Papaspiliopoulos and Roberts, 2008] to the Dec-SBPR. For agent n, the SB prior is set with a truncation level $|\mathcal{Z}_n|$, taking into account the current occupancy as well as additional nodes reserved for future new occupancies. The solution to (7) under the stick-breaking priors is given in Theorem 3, which is proved in the supplementals [URL, b].

Theorem 3. *Let $p(\Theta)$ be constructed by the SB priors defined in* (8) *with hyper-parameters $(\hat{\sigma}, \hat{\eta}, \hat{\rho})$, then iterative application of the following updates leads to monotonic increase of (5), until convergence to a maxima. The updates of $\{q_t^k\}$ are*

$$q_t^k(z_{n,0:t}^k) = \hat{\nu}_t^k p(z_{n,0:t}^k|o_{n,1:t}^k, a_{n,0:t}^k, \widetilde{\Theta}_n), \forall\, n,t,k, \tag{9}$$

where $\hat{\nu}_t^k$ is computed using (6) *with Θ replaced by $\widetilde{\Theta} = \{\widetilde{\pi}, \widetilde{\mu}, \widetilde{W}\}$, a set of under-normalized probability (mass) functions , with $\widetilde{\pi}_{n,i}^a = e^{\langle \ln \pi_{n,i}^a\rangle_{p(\pi|\hat{\boldsymbol{\rho}})}}$, and $\widetilde{W}_{n,a,o}^{i,j} = e^{\langle \ln W_{n,a,o}^{i,j}\rangle_{p(W|\hat{\boldsymbol{\sigma}},\hat{\boldsymbol{\eta}})}}$, and $\langle\cdot\rangle_p$ denotes expectations of $\cdot$ with respect to distributions p. The hyper-parameters of the posterior distribution are updated as*

$$\hat{\sigma}_{n,a,o}^{i,j} = \sigma_{n,a,o}^{i,j} + \zeta_{n,a,o}^{i,j}, \;\; \hat{\eta}_{n,a,o}^{i,j} = \eta_{n,a,o}^{i,j} + \sum_{l=j+1}^{|\mathcal{Z}_n|} \zeta_{n,a,o}^{i,l},$$

$$\hat{\rho}_{n,i}^a = \rho_{n,i}^a + \sum_{k=1}^{K}\sum_{t=0}^{T_k}\sum_{\tau=1}^{t} \frac{\hat{\nu}_t^k}{K} \phi_{t,\tau-1}^{n,k}(i)\mathbb{I}_a(a_\tau^k) \tag{10}$$

with $\zeta_{n,a,o}^{i,j}{=}\sum_{k=1}^K\sum_{t=0}^{T_k}\sum_{\tau=1}^t \frac{\hat{\nu}_t^k}{K}\xi_{t,\tau-1}^{n,k}(i,j)\mathbb{I}_{a,o}(a_{\tau-1}^k, o_\tau^k)$, where $\mathbb{I}(\cdot)$ is the indicator function, and both $\xi_{t,\tau}^{n,k}$ and $\phi_{t,\tau}^{n,k}$ are marginals of $q_t^k(z_{n,0:t}^k)$, i.e.

$$\xi_{t,\tau}^{n,k}(i,j){=}p(z_{n,\tau}^k = i, z_{n,\tau+1}^k = j|a_{n,0:t}^k, o_{n,1:t}^k, \widetilde{\Theta}_n) \tag{11}$$

$$\phi_{t,\tau}^{n,k}(i){=}p(z_{n,\tau}^k = i|a_{n,0:t}^k, o_{n,1:t}^k, \widetilde{\Theta}_n) \tag{12}$$

The update equations in Theorem 3 constitute the VB algorithm for learning a variable-size joint FSCs under SB priors with batch data. In particular, (9) is a policy-evaluation step where the rewards are reweighted to reflect the improved marginal value of the new policy posterior updated in the previous iteration, and (10) is a policy-improvement step where the reweighted rewards are used to further improve the policy posterior. Both steps require (11), which are computed based on $\alpha_\tau^{n,k}(i){=}p\big(z_{n,\tau}^k{=}i|a_{n,0:\tau}^k, o_{n,1:\tau}^k, \widetilde{\Theta}_n\big)$ and $\beta_{t,\tau}^{n,k}(i){=}\frac{p(a_{n,\tau+1:t}^k|z_{n,\tau}^k{=}i, o_{n,\tau+1:t}^k, \widetilde{\Theta}_n)}{\prod_{\tau'=\tau}^t p(a_\tau^k|h_{n,\tau'}^k, \widetilde{\Theta}_n)}, \forall\, n,k,t,\tau$. The (α, β) are forward-backward messages. Their updating equations are derived in the supplemental material [URL, b].

To determine the number of controller nodes $\{|\mathcal{Z}_n|\}_{n=1}^N$, the occupancy of a node is computed by checking if there is a positive reward assigned to it. For example, for action a and node i, $\hat{\rho}_{n,i}^a - \rho_{n,i}^a$ is the reward being assigned. If this quantity is greater than zero, then node i is visited. Summing over all actions gives the value of node i. Hence $|\mathcal{Z}_n|$ can be computed based on the following formula

$$|\mathcal{Z}_n| = \textstyle\sum_{i=1}^{\infty}\mathbb{I}\big(\sum_{a=1}^{|\mathcal{A}_n|}(\hat{\rho}_{n,i}^a - \rho_{n,i}^a) > 0\big). \tag{13}$$

Algorithm 1 describes the complete approach, and after it has converged, point estimates of the decentralized policies may be obtained by calculating $\mathbb{E}[\hat{\mu}_n^i]$, $\mathbb{E}[\hat{\pi}_{n,i}^a]$, and $\mathbb{E}[\hat{W}_{n,a,o}^{i,j}]$.

3.3 Computational complexity

The time complexity of Algorithm 1 for each iteration is summarized in Table 1, assuming the length of an episode is on the order of magnitude of T, and the number of nodes per controller is on the order of magnitude of $|\mathcal{Z}|$. In Table 1, the worst case refers to when there is a nonzero reward at every time step of an episode (dense rewards), while the best case is when nonzero reward is received only at the terminal step. Hence in general, the algorithm scales linearly with the number of episodes and the number of agents. The time dependency on T is between linear and quadratic. In any case, the computational complexity of Algorithm 1 is independent of the number of states, making it is scalable to large problems.

3.4 Exploration and Exploitation Tradeoff

Algorithm 1 assumes off-policy batch learning where trajectories are collected using a separate behavior policy. This is appropriate when data has been generated from real-world or simulated experiences without any input from the learning algorithm (e.g., learning from demonstration). Off-policy learning is efficient if the behavior policy is close to optimal, as in the case when expert information is available to guide the agents. With a random behavior policy, it may take a long time for the policy to converge to optimality; in this case, the agents may want to exploit the policies learned so far to speed up the learning process.

An important issue concerns keeping a proper balance between exploration and exploitation to prevent premature convergence to a suboptimal policy, but allow the algorithm to learn quickly. Since the execution of Dec-POMDP policies is decentralized, it is difficult to design an efficient exploration strategy that guarantees optimality. [Wu *et al.*, 2013] count the visiting frequency of FSC nodes and apply upper-confidence-bound style heuristic to select next controller nodes, and use ϵ-greedy strategy to select actions. However ϵ-greedy might be sample inefficient. [Banerjee *et al.*, 2012] proposed a distributed learning approach where agents take turns to learn the best response to each other's policies. This framework applies an R-max type of heuristic, using the counts of trajectories to distinguish known and unknown histories, to tradeoff exploration and exploitation. However, this method is confined to tree-based policies in finite-horizon problems, and requires synchronized multi-agent learning.

To better accommodate our Bayesian policy learning framework for RL in infinite-horizon Dec-POMDPs, we define an auxiliary FSC, $\Omega_n = \langle \mathcal{Y}, \mathcal{O}_n, \mathcal{Z}_n, W_n, \mu_n, \varphi_n \rangle$, to represent the policy of each agent in balancing exploration and exploitation. To avoid confusion, we refer to Θ_n as a primary FSC. The only two components distinguishing Ψ_n from Θ_n are $\mathcal{Y}$ and φ_n, where $\mathcal{Y} = \{0, 1\}$ encodes exploration ($y = 1$) or exploitation ($y = 0$), and $\varphi_n = \{\varphi_y^{n,z}\}$ with $\varphi_y^{n,z}$ denoting the probability of agent n choosing y in z. One can express $p(y_{n,t}|h_{n,t}, \Psi_n)$ in the same way as one expresses $p(a_{n,t}|h_{n,t}, \Theta_n)$ (which is described in section 2.1). The behavior policy Π_n of agent n is given as

$$p^{\Pi_n}(a|h, \Theta_n, \Omega_n) = \sum_{y=0,1} p(a|y, h) p(y|h, \Omega_n), \qquad (14)$$

where $p(a|y = 0, h) \equiv p(a|h, \Theta_n)$ is the primary FSC policy, and $p(a|y = 1, h)$ is the exploration policy of agent n, which is usually a uniform distribution.

The behavior policy in (14) has achieved significant success in the single-agent case [Cai *et al.*, 2009; Liu *et al.*, 2011; 2013]. Here we extend it to the multi-agent case (centralized learning and decentralized exploration/execution) and provide empirical evaluation in the next section.

4 Experiments

The performance of the proposed algorithms are evaluated on five benchmark problems[4] and a large-scale problem (traffic control) [Wu *et al.*, 2013]. The experimental procedure in [Wu *et al.*, 2013] was used for all the results reported here. For Dec-SBPR, the hyperparameters in (8) are set to $c = 0.1$ and $d = 10^{-6}$ to promote sparse usage of FSC nodes.[5] The policies are initialized as FSCs converted from the episodes with the highest rewards using a method similar to [Amato and Zilberstein, 2009].

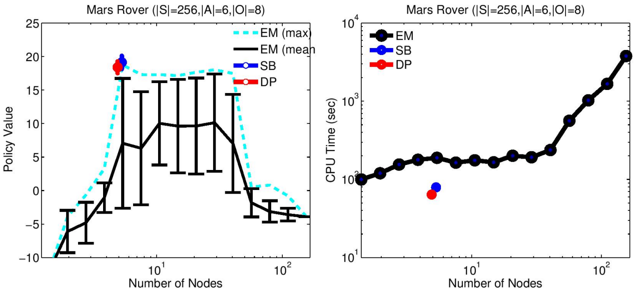

Figure 1: Compares the variable-size controller learned by Dec-SBPR and fixed-size controllers learned by EM. Left: testing value; Right: averaged computation time. Although EM has lower computational complexity per iteration than our VB algorithm, empirically, the VB algorithm takes less time to converge. Moreover, the value of using the SB prior (also its special Dirichlet instance (DP)) is close to the best result of EM (dotted sky-blue line), and is much better than the average results of EM (solid back line). The added flexibility of the SB prior leads to slightly better performance than the DP, as discussed at the end of section 3.1.

Learning variable-size FSC vs learning fixed-size FSC To demonstrate the advantage of learning variable-size FSCs, Dec-SBPR is compared with an implementation of the previous EM algorithm [Wu *et al.*, 2013]. The comparison is for the Mars Rover problem using $K = 300$ episodes [6] to learn the FSCs and evaluating the policy by the discounted accumulated reward averaged over 100 test episodes of 1000 steps. Here, we consider off-policy learning and apply a semi-random policy to collect samples. Specifically, the learning agent is allowed access to episodes collected by taking actions according to a POMDP algorithm (point-based value iteration (PBVI) [Pineau *et al.*, 2003]). Let ϵ be the probability that the agents follow the PBVI policy and $1 - \epsilon$ be the probability that the agents take random actions. This procedure mimics the approach used in previous work [Wu *et al.*, 2013]. The results with $\eta = 0.3$ are reported in Figure 1, which shows the exact value and computation time as a function of the number of controller nodes $|\mathcal{Z}|$. As expected, for the EM algorithm, when $|\mathcal{Z}|$ is too small, the FSCs cannot represent the optimal policy (under-fitting), and when the number of nodes is too large, FSCs overfits a limited amount of data and perform poorly. Even if $|\mathcal{Z}|$ is set to the number inferred by Dec-SBPR, EM can still suffer severely from initialization and local maxima issues, as can be seen from a large error-bar. By setting a high truncation level ($|\mathcal{Z}| = 50$), Dec-SBPR employs Algorithm 1 to integrate out the uncertainty of the policy representation (under the SB prior). As a result, Dec-SBPR can infer both

[4] Downloaded from `http://rbr.cs.umass.edu/camato/decpomdp/download.html`

[5] These values were chosen for testing, but our approach is robust to other values of c and d.

[6] Using smaller training sample size K, our method can still perform robustly, as it is shown in the supplementals [URL, b].

Table 2: Performance of Dec-SBPR on benchmark problems compared to other state-of-art algorithms. Shows policy values (higher value indicates better performance) and CPU times of all algorithms, and the average controller size $|Z|$ inferred by Dec-SBPR.

	POLICY LEARNING (UNKNOWN MODEL)									PLANNING (KNOWN MODEL)					
PROBLEMS ($|\mathcal{S}|, |\mathcal{A}|, |\mathcal{O}|$)	DEC-SBPR(FIXED ITERATION)			DEC-SBPR(FIXED TIME)			MCEM			PERIEM			FB-HSVI		
	VALUE	$|Z|$	TIME	VALUE	$|Z|$	TIME	VALUE	$|Z|$	TIME	VALUE	$|Z|$	TIME	VALUE	$|Z|$	TIME
DEC-TIGER (2, 3, 3)	-18.63	6	96S	-19.42	8	20S	-32.31	3	20S	9.42	7×10	6540S	13.45	52	6.0S
BROADCAST (4, 2, 5)	9.20	2	7S	9.27	2	24S	9.15	3	24S		–		9.27	102	19.8S
RECYCLING ROBOTS (3, 3, 2)	31.26	3	147S	25.16	2	19S	30.78	3	19S	31.80	6×10	272S	31.93	108	0S
BOX PUSHING (100, 4, 5)	77.65	14	290S	58.27	9	32S	59.95	3	32S	106.68	4×10	7164S	224.43	331	1715.1S
MARS ROVERS (256, 6, 8)	20.62	5	1286S	15.2	6	160S	8.16	3	160S	18.13	3×10	7132S	26.94	136	74.31S

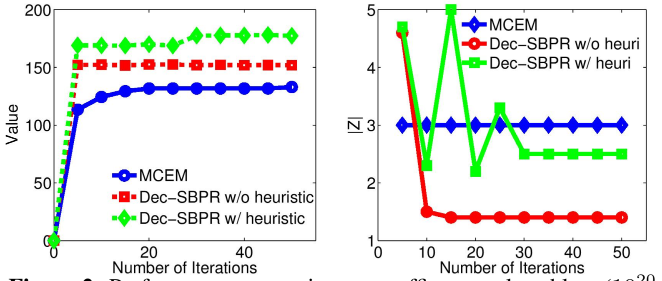

Figure 2: Performance comparison on traffic control problem (10^{20} states and 100 agents). Test reward (left) and inferred controller size (right) of Dec-SBPR, as a function of algorithmic iteration.

the number of nodes that is needed (≈ 5) and optimal controller parameters simultaneously. Furthermore, this inference is done with less computation time and with a higher value and improved robustness (low variance of test value) than EM.

Comparison with other methods The performance of Dec-SBPR is also compared to several state-of-art methods, including: Monte Carlo EM (MCEM) [Wu *et al.*, 2013]. Similar to Dec-SBPR, MCEM is a policy-based RL approach. We apply the exploration-exploitation strategy described in section 3.4 and follow the same experimental procedure in [Wu *et al.*, 2013] to report the results. The rewards after running a fixed number of iterations and a fixed amount of time are summarized (respectively) in Table 2 (the first column under policy-learning category). Dec-SBPR is shown to achieve better policy values than MCEM on all problems [7]. These results can be explained by the fact that EM is (more) sensitive to initialization and (more) prone to local optima. Moreover, by fixing the size of the controllers, the optimal policy from EM algorithms might be over/under fitted . By using a Bayesian nonparametric prior, Dec-SBPR learns the policy with variable-size controllers, allowing more flexibility for representing the optimal policy. We also show the result of Dec-SBPR running the same amount of clock time as MCEM (Dec-SBPR (fixed time)), which indicates Dec-SBPR can achieve a better trade-off between policy value and learning time than MCEM.

Finally, Dec-SBPR is compared to Periodic EM (PeriEM) [Pajarinen and Peltonen, 2011] and feature based heuristic search value iteration (FB-HSVI) [Dibangoye *et al.*, 2014], two state-of-art planing methods (with known models) for generating controllers. Because having a Dec-POMDP model allows more accurate value function calculations than a finite number of trajectories, the value of PeriEM and FB-HSVI are treated as upper-bounds for the policy-based methods. Our Dec-SBPR approach can sometimes outperform PeriEM, but produces lower value than FB-HSVI. FB-HSVI is a boundedly-optimal method, showing that Dec-SBPR can produce near optimal solutions in some of these problems and produces solutions that are much closer to the optimal than previous RL methods. It is also worth noting that neither PeriEM nor FB-HSVI can scale to large problems (such as the one discussed below), while by using a policy-based RL approach, Dec-SBPR can scale well.

Scaling up to larger domains To demonstrate scalability to both large problem sizes and large numbers of agents, we test our algorithm on a traffic problem [Wu *et al.*, 2013], with 10^{20} states. Here, there are 100 agents controlling the traffic flow at 10×10 intersections with one agent located at each intersection. Except for MCEM, no previous Dec-POMDPs algorithms are able to solve such large problems.

Since Wu *et al.* use a hand-coded policy (comparing the traffic flow between two directions) as a heuristic for generating training trajectories, we also use such a heuristic for a fair comparison. In addition, to examine the effectiveness of the exploration-exploitation strategy described in Section 3.4, we also consider the case where the initial behavior policy is random and then it is optimized as discussed. From Figure 2, we can see that, with the help of the heuristic, Dec-SBPR can achieve the best performance. Without using the heuristic (by just using our exploration-exploitation strategy), in a few iterations, Dec-SBPR is able to produce a higher quality policy than MCEM. Moreover, the inferred number of FSC nodes (averaged over all agents) is smaller than the number preselected by MCEM. This shows that not only can Dec-SBPR scale to large problems, but it can also produce higher-quality solutions than other methods for those large problems.

5 Conclusions

The paper presented a scalable Bayesian nonparametric policy representation and an associated learning framework (Dec-SBPR) for generating decentralized policies in Dec-POMDPs. An new exploration-exploitation method, which extends the popular ϵ-greedy method, was also provided for reinforcement learning in Dec-POMDPs. Experimental results show Dec-SBPR produces higher-quality solutions than the state-of-art policy-based method, and has the additional benefit of inferring the number of nodes needed to represent the optimal policy. The resulting method is also scalable to large domains (in terms of both the number of agents and the problem size), allowing high-quality policies for large Dec-POMDPs to be learned efficiently from data.

[7] The results are provided by personal communication with its authors and run on the same benchmarks that are available online.

Acknowledgments

Research supported by US Office of Naval Research under MURI program award #N000141110688 and NSF award #1463945.

References

[Amato and Zilberstein, 2009] C. Amato and S. Zilberstein. Achieving goals in decentralized POMDPs. In *Proc. of the 8th Int'l Conf. on Autonomous Agents and Multiagent Systems*, 2009.

[Amato *et al.*, 2010] C. Amato, D. S. Bernstein, and S. Zilberstein. Optimizing fixed-size stochastic controllers for POMDPs and decentralized POMDPs. *J. Autonomous Agents and Multi-Agent Systems*, 2(3):293–320, 2010.

[Amato *et al.*, 2013] C. Amato, G. Chowdhary, A. Geramifard, N. K. Ure, and M. J. Kochenderfer. Decentralized control of partially observable Markov decision processes. In *Proc. of the Conf. on Decision and Control*, 2013.

[Amato *et al.*, 2015] Christopher Amato, George D. Konidaris, Gabriel Cruz, Christopher A. Maynor, Jonathan P. How, and Leslie P. Kaelbling. Planning for decentralized control of multiple robots under uncertainty. In *ICRA*, 2015.

[Banerjee *et al.*, 2012] B. Banerjee, J. Lyle, L. Kraemer, and R. Yellamraju. Sample bounded distributed reinforcement learning for decentralized POMDPs. In *Proc. of the 26th AAAI Conf. on Artificial Intelligence*, 2012.

[Beal, 2003] Matthew James Beal. *Variational algorithms for approximate Bayesian inference*. PhD thesis, University of London, 2003.

[Bernstein *et al.*, 2001] D. Bernstein, S. Zilberstein, R. Washington, and J. Bresina. Planetary rover control as a Markov decision process. In *6th Int'l Symposium on Artificial Intelligence, Robotics, and Automation in Space*, 2001.

[Bernstein *et al.*, 2002] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein. The complexity of decentralized control of Markov decision processes. *Mathematics of Operations Research*, 27(4):819–840, 2002.

[Bernstein *et al.*, 2009] D. S. Bernstein, C. Amato, E. A. Hansen, and S. Zilberstein. Policy iteration for decentralized control of Markov decision processes. *J. Artificial Intelligence Research*, 34:89–132, 2009.

[Bryant and Sudderth, 2012] Michael Bryant and Erik B Sudderth. Truly nonparametric online variational inference for hierarchical dirichlet processes. In *Advances in Neural Information Processing Systems*, 2012.

[Cai *et al.*, 2009] C. Cai, X. Liao, and L. Carin. Learning to explore and exploit in POMDPs. In *Proc. of the 23th Annual Conf. on Neural Information Processing Systems*, 2009.

[Dempster *et al.*, 1977] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Statistical Society B*, 39:1–38, 1977.

[Dibangoye *et al.*, 2014] J. S. Dibangoye, O. Buffet, and F. Charpillet. Error-bounded approximations for infinite-horizon discounted decentralized POMDPs. In *Machine Learning and Knowledge Discovery in Databases*. Springer, 2014.

[Doshi-Velez *et al.*, 2010] F. Doshi-Velez, D. Wingate, N. Roy, and J. B. Tenenbaum. Nonparametric Bayesian policy priors for reinforcement learning. In *Advances in Neural Information Processing Systems*, 2010.

[Ferguson, 1973] T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *Annals of Statistics*, 1:209–230, 1973.

[Ishwaran and James, 2001] H. Ishwaran and L. F. James. Gibbs sampling methods for stick-breaking priors. *J. Am. Statist. Assoc.*, 96(453), 2001.

[Kumar and Zilberstein, 2010] A. Kumar and S. Zilberstein. Anytime planning for decentralized POMDPs using expectation. In *Proc. of the 26th Conf. on Uncertainty in Artificial Intelligence*, 2010.

[Li *et al.*, 2009] H. Li, X. Liao, and L. Carin. Multi-task reinforcement learning in partially observable stochastic environment. *J. Machine Learning Research*, 10(1):1131–1186, 2009.

[Liu *et al.*, 2011] M. Liu, X. Liao, and L. Carin and. Infinite regionalized policy representation. In *Proc. of the 28th Int'l Conf. on Machine Learning*, 2011.

[Liu *et al.*, 2013] Miao Liu, Xuejun Liao, and Lawrence Carin. Online expectation maximization for reinforcement learning in POMDPs. In *Proc. of 23rd Int Joint Conf. on Artificial Intelligence*, 2013.

[Liu *et al.*, 2015] Miao Liu, Christopher Amato, Xuejun Liao, Lawrence Carin, and Jonathan P. How. Stick-Breaking Policy Learning in Dec-POMDPs. *arXiv:submit/1244756 [cs.AI]*, 2015.

[Martins and Demiris, 2010] Murilo Fernandes Martins and Yiannis Demiris. Learning multirobot joint action plans from simultaneous task execution demonstrations. In *Proc. of the 9th Int'l Conf. on Autonomous Agents and Multiagent Systems*, 2010.

[Messias *et al.*, 2010] J. Messias, M. Spaan, and P. Lima. Multirobot planning under uncertainty with communication: a case study. In *Fifth Workshop on Multi-agent Sequential Decision Making in Uncertain Domains (MSDM)*, 2010.

[Oliehoek, 2012] F. A. Oliehoek. Decentralized POMDPs. In *Reinforcement Learning: State of the Art, Adaptation, Learning, and Optimization*, pages 471–503. Springer Berlin Heidelberg, Berlin, Germany, 2012.

[Paisley and Carin, 2009] J. Paisley and L. Carin. Hidden Markov models with stick-breaking priors. *Signal Processing, IEEE Trans. on*, 57(10):3905–3917, 2009.

[Pajarinen and Peltonen, 2011] J. Pajarinen and J. Peltonen. Periodic finite state controllers for efficient POMDP and DEC-POMDP planning. In *Proc. of the 25th Annual Conf. on Neural Information Processing Systems*, 2011.

[Papaspiliopoulos and Roberts, 2008] O. Papaspiliopoulos and G. O. Roberts. Retrospective Markov chain Monte Carlo methods for Dirichlet process hierarchical models. *Biometrika*, 95(1):169–186, 2008.

[Pineau *et al.*, 2003] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: an anytime algorithm for POMDPs. In *Proc. of the 18th Int Joint Conf. on Artificial Intelligence*, 2003.

[Robert and Casella, 2004] Christian P Robert and George Casella. *Monte Carlo statistical methods*, volume 319. Citeseer, 2004.

[Wu *et al.*, 2013] F. Wu, S. Zilberstein, and N. R. Jennings. Monte-carlo expectation maximization for decentralized POMDPs. In *Proc. 23rd Int'l. Joint Conf. on Artificial Intelligence*, 2013.