Factored Online Planning in Many-Agent POMDPs

Maris F.L. Galesloot[1], **Thiago D. Simão**[2], **Sebastian Junges**[1], **Nils Jansen**[1,3]

[1]Radboud University Nijmegen, The Netherlands
[2]Eindhoven University of Technology, The Netherlands
[3]Ruhr-University Bochum, Germany
maris.galesloot@ru.nl, t.simao@tue.nl, sebastian.junges@ru.nl, n.jansen@rub.de

Abstract

In centralized multi-agent systems, often modeled as multi-agent partially observable Markov decision processes (MPOMDPs), the action and observation spaces grow exponentially with the number of agents, making the value and belief estimation of single-agent online planning ineffective. Prior work partially tackles *value estimation* by exploiting the inherent structure of multi-agent settings via so-called coordination graphs. Additionally, *belief estimation* methods have been improved by incorporating the likelihood of observations into the approximation. However, the challenges of value estimation and belief estimation have only been tackled individually, which prevents existing methods from scaling to settings with many agents. Therefore, we address these challenges simultaneously. First, we introduce *weighted particle filtering* to a sample-based online planner for MPOMDPs. Second, we present a scalable approximation of the belief. Third, we bring an approach that exploits the typical locality of agent interactions to novel online planning algorithms for MPOMDPs operating on a so-called sparse particle filter tree. Our experimental evaluation against several state-of-the-art baselines shows that our methods (1) are competitive in settings with only a few agents and (2) improve over the baselines in the presence of many agents.

1 Introduction

Planning problems with multiple agents, such as teams of mobile robots (Ahmadi et al. 2019) or autonomous surveillance systems (Witwicki et al. 2017), can be modeled by multi-agent partially observable Markov decision processes (MPOMDPs, Messias, Spaan, and Lima 2011). These formal models exhibit sets of (local) observations and actions for each agent that can be shared with a central controller. This controller then makes decisions among the joint actions of all agents. Computationally, the main challenge is that the spaces of joint action and observations grow exponentially with the number of agents (Pynadath and Tambe 2002). Moreover, as the controller only partially observes the system state, it must base its decisions on the history of previous joint actions and observations.

Online algorithms, such as those based on *Monte Carlo tree search* (MCTS, Browne et al. 2012), are a common way to tackle large planning problems. These algorithms search for local solutions in the most promising regions of the search space (Kocsis and Szepesvári 2006). In particular, *partially observable Monte Carlo planning* (POMCP, Silver and Veness 2010) derives Monte Carlo estimates in the form of (1) an approximation of the value function and (2) distributions (beliefs) over states by generating sample trajectories from the simulation of single state particles. However, a naive application of (single-agent) online planning is ill-equipped to handle the high-dimensional MPOMDP setting. First, due to the many actions that must be explored during simulations, the **value estimation** may suffer from high variance. Second, the chance of mismatch between simulated and actual observations is high for large observation spaces, lowering the quality of the approximation in the **belief estimates** (Sunberg and Kochenderfer 2018).

To address the challenge of value estimation, one can exploit the typical locality of interactions between the agents, captured by so-called **coordination graphs** (Guestrin, Lagoudakis, and Parr 2002). In particular, Amato and Oliehoek (2015) estimate the action value for subsets of agents instead of all agents based on such graphs. The main concepts are to (1) factorize the value estimates over the action space of subsets of agents in the **factored statistics** variant and to (2) factorize both the action and the observation space in the **factored trees** variant. These factorizations are key to achieving good performance in settings with many agents. The challenge of belief estimation is also a prevalent issue in single-agent continuous settings. From the likelihood of sampled observations, importance sampling weights are added to the Monte Carlo estimates of the beliefs (Thrun 1999). Such weighted beliefs are also used in single-agent online planners that simulate **weighted belief estimates** instead of single states (Fischer and Tas 2020; Lim, Tomlin, and Sunberg 2020). By simulating belief estimates, these algorithms operate on the set of possible beliefs of the agents, which makes the branching factor insensitive to the number of observations. A particularly effective algorithm is the so-called *sparse particle filter tree* (Sparse-PFT, Lim et al. 2023), which only searches for local solutions in the set of reachable belief estimates.

To the best of our knowledge, these solutions to value and belief estimation have only been studied independently, and weighted belief estimates have not yet been explored in the

Belief Estimation	Simulation	POMDP Value Estimation	MPOMDP Value Estimation	
		Single-Agent	Factored Statistics	Factored Trees
Unweighted	Single States	POMCP (Silver and Veness 2010)	FS-POMCP (Amato and Oliehoek 2015)	FT-POMCP (Amato and Oliehoek 2015)
Weighted	Single States	W-POMCP / POMCPOW (Sunberg and Kochenderfer 2018)	FS-W-POMCP Sect. 4 (*new*)	FT-W-POMCP Sect. 4 (*new*)
	Belief Estimates	Sparse-PFT (Lim et al. 2023)	FS-PFT Sect. 5 (*new*)	FT-PFT Sect. 5 (*new*)

Table 1: Our algorithms and state-of-the-art MCTS methods in the (continuous) POMDP and MPOMDP literature.

online MPOMDP planning setting. Therefore, no method can scale to problems with many agents, and we must tackle both challenges simultaneously. We present multi-agent online planning algorithms that exploit adequate approximations of both the value and the belief and thereby can scale to many agents. In particular, we integrate factored value estimation and weighted belief estimation. Table 1 positions the new algorithms with respect to the existing solutions in the MCTS literature. First, *we add weighted belief estimation to POMCP variants*, namely W-POMCP, and combine this algorithm with *factored statistics* in FS-W-POMCP (Sect. 4.1). Furthermore, we design a weighted belief approximation that is compatible with *factored trees* in FT-W-POMCP (Sect. 4.2). Then, we *introduce two novel variants* of the Sparse-PFT algorithm that exploit coordination graphs similarly to Amato and Oliehoek (2015), namely FS-PFT and FT-PFT (Sect. 5.2). The empirical evaluation (1) shows the improvement of integrating weighted particle filtering in POMCP, and (2) demonstrates the effect of value factorization via coordination graphs in Sparse-PFT. It also shows that exploiting local interactions between agents is beneficial in environments where the way the agents interact may change over time, i. e., there is no naturally sparse graph. The extended version of this article (Galesloot et al. 2023) contains an Appendix with additional details.

Contributions. We present four novel algorithms for many-agent online planning that can scale both value and belief estimations to problems with many agents. Our empirical evaluation, using the nine algorithms in Table 1, shows that our variants improve over the state-of-the-art. For example, in the environments FIREFIGHTINGGRAPH and Multi-Agent ROCKSAMPLE, we scale up to instances with 64 and 6 agents instead of 10 and 2 agents, respectively.

2 Online Planning in MPOMDPs

The set of all distributions over the finite set X is $\Delta(X)$.

MPOMDPs. We study online planning in centralized multi-agent systems that are modeled as MPOMDPs. Intuitively, agents encounter individual observations but can share those via immediate and noiseless broadcast communication, which allows a centralized control paradigm.

Definition 1 (MPOMDP). *An* MPOMDP *is a tuple* $\mathcal{M} = \langle \mathcal{I}, \mathcal{S}, b_0, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma \rangle$*, with the finite set* $\mathcal{I}$ *of* n agents, *the finite set* $\mathcal{S}$ *of* states*, an* initial state distribution $b_0 \in \Delta(\mathcal{S})$*, the set* $\mathcal{A} = \times_{i \in \mathcal{I}} \mathcal{A}_i$ *of* joint actions*, composed of the finite sets* $\mathcal{A}_i$ *of* actions *for each agent* $i \in \mathcal{I}$*, the* transition function $\mathcal{T} \colon \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ *such that* $\mathcal{T}(s' \mid s, \vec{a}) = \Pr(s' \mid s, \vec{a})$ *is the probability of a new state* s' *given the previous state* s *and joint action* $\vec{a}$*, the* reward function $\mathcal{R} \colon \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ *such that* $\mathcal{R}(s, \vec{a})$ *is the* reward *given state* s *and joint action* $\vec{a}$*, the set* $\Omega = \times_{i \in \mathcal{I}} \Omega_i$ *of* joint observations *composed by the finite sets* Ω_i *of* observations *for each agent* $i \in \mathcal{I}$*, the* observation function $\mathcal{O} \colon \mathcal{S} \times \mathcal{A} \to \Delta(\Omega)$ *such that* $\mathcal{O}(\vec{o} \mid s', \vec{a}) = \Pr(\vec{o} \mid s', \vec{a})$ *specifies the probability of observing joint observation* $\vec{o}$ *in the state* s' *given joint action* $\vec{a}$*, and the* discount factor $\gamma \in [0, 1)$.

MPOMDPs generalize POMDPs (Kaelbling, Littman, and Cassandra 1998), which are MPOMDPs with a single agent. An MPOMDP can be treated as a POMDP by ignoring the agent-wise factorization in the action and observation space.

Objective. The *return* $R_t = \sum_{t'=t}^{\infty} \gamma^{t'-t} \mathcal{R}(s_{t'}, a_{t'})$ is the infinite-horizon discounted sum of reward from time $t \in \mathbb{N}$. An observable *history* $\vec{h}_t = (\vec{o}_1, \vec{a}_1, \vec{o}_2, \dots, \vec{a}_{t-1}, \vec{o}_t)$ is a sequence of joint observations and joint actions. Policies determine the action choices. Optimal policies $\pi \colon \Delta(S) \to \mathcal{A}$ for MPOMDPs map the belief $b_t \in \Delta(\mathcal{S})$ to joint actions. The belief b_t is a *sufficient statistic* (Kaelbling, Littman, and Cassandra 1998) for the history h_t, and resembles the state distribution $b_t(s) = \Pr(s_t \mid h_t, b_0)$ at time t, with b_0 from $\mathcal{M}$. Beliefs can be updated $b_{t+1} = \upsilon(b_t, \vec{o}_{t+1}, \vec{a}_t)$ by υ from $\mathcal{T}$ and $\mathcal{O}$, using Bayes' theorem (Spaan 2012). Our aim is to maximize the *joint Q-value* of a belief b, which is the *expected return* under a policy π given action $\vec{a}$ and belief b at time t, and $a_{t'} = \pi(b_t)$ for subsequent $t' > t$:

$$Q^{\pi}(b, \vec{a}) = \mathbb{E}_{\pi} \left[R_t \mid b_t{=}b, \vec{a}_t{=}\vec{a}, a_{t'>t} = \pi(b_{t'}) \right]. \quad (1)$$

Online planning. Online search-based planners interleave planning and execution. They perform a forward search in the set of beliefs reachable from the current belief, incrementally building a look-ahead tree known as a search tree. Monte Carlo planners typically do so with a generative interface $\mathcal{G} \colon \mathcal{S} \times \mathcal{A} \to \mathcal{S} \times \Omega \times \mathbb{R}$ of the model $\mathcal{M}$, i. e., a *simulator* (Kearns, Mansour, and Ng 2002), with $s, \vec{a} \mapsto s', \vec{o}, r$. After searching from b, the planner executes a selected action $\vec{a}$, receives an observation $\vec{o}$. Then, it updates its belief $b' = \upsilon(b, \vec{a}, \vec{o})$, before it starts searching from b'.

MCTS. Contemporary MCTS methods are based on the *upper confidence trees* (UCT, Kocsis and Szepesvári 2006) algorithm. In particular, partially observable UCT (PO-UCT) is the search algorithm that underlies POMCP. It plans with a look-ahead search tree comprised of paths of action and observation nodes. It samples states from the current belief b, and for each state, it expands a trajectory of actions and observations using $\mathcal{G}$ until it reaches a new node in the tree. Then, a (random) *rollout* estimates the value (Kocsis and Szepesvári 2006). The trajectory is used to update a set of statistics for history $\vec{h}$ that includes visit counts $N(\vec{h})$ and $n(\vec{h},\vec{a})$ in the observation and action nodes, respectively. Additionally, the trajectory updates estimates of the Q-values $Q(\vec{h},\vec{a})$ of the action nodes by a running average of the return. The upper confidence bound (UCB1, Auer, Cesa-Bianchi, and Fischer 2002) algorithm decides the most promising actions during the search, balancing exploration and exploitation. It is computed from (an estimate of) the number N^* of visits to the observation node, as well as the number of visits n and the value Q^* of the action node by:

$$\mathcal{UCB}(Q^*, N^*, n^*) = Q^* + c \cdot \sqrt{\tfrac{\log(N^*+1)}{(n^*+1)}}, \qquad (2)$$

where c is an exploration constant. During search, in some history $\vec{h}$, PO-UCT chooses actions via: $\arg\max_{\vec{a}} \mathcal{UCB}(Q(\vec{h},\vec{a}), N(\vec{h}), n(\vec{h},\vec{a}))$.

A separation of beliefs. POMCP is the extension of PO-UCT that gradually builds up beliefs $B(\vec{h})$ consisting of simulated states $s \in \mathcal{S}$, i. e., *particles*, in the observation nodes for history $\vec{h}$, representing a Monte Carlo estimate of the belief b. However, the number of particles in each observation node depends on how often a history has been recorded during forward simulation and might diminish over time due to a lack of diversity in the set of particles. POMCP requires domain-specific particle reinvigorating techniques to mitigate this. Instead, we separate the concerns of an *online* belief *inside the search tree* that is used to estimate Q, and the *offline* belief $\bar{b}$ that represents the current belief over states. Following related works, we write $B(\vec{h})$ for POMCP's online belief and $\tilde{b}$ for Sparse-PFT's weighted online belief.

Problem statement: Given a an MPOMDP $\mathcal{M}$, how do we, at each time step $t \in \mathbb{N}$, both (1) efficiently search for a joint action $\vec{a}_t$ given the current belief estimate $\bar{b}_t$, and (2) effectively find a good belief estimate $\bar{b}_{t+1}$ from $\bar{b}_t, \vec{a}_t$ and the received observation $\vec{o}_{t+1}$.

3 Using Structure in Multi-Agent POMDPs

In this section, we introduce *coordination graphs* to decompose the objective into local sub-problems. In particular, we recap prior work by Amato and Oliehoek (2015).

3.1 Coordination Graphs

A *coordination graph* (CG, Guestrin, Venkataraman, and Koller 2002; Oliehoek and Amato 2016) is an undirected graph $(\mathcal{V},\mathcal{E})$ that represents the local interactions between agents. Each vertex $v \in \mathcal{V}$ corresponds to an agent ($\mathcal{V} \equiv \mathcal{I}$), and each edge $(i,j) \in \mathcal{E}$ indicates that agents $i \in \mathcal{V}$ and $j \in \mathcal{V}$ interact locally. For an edge $e \in \mathcal{E}$, we define the *local action* $\vec{a}_e$ and *local observations* $\vec{o}_e$, which range over the product of the individual agent action $\mathcal{A}_e = \mathcal{A}_i \times \mathcal{A}_j$ and observation spaces $\Omega_e = \Omega_i \times \Omega_j$, with $e = (i,j)$. For three agents $\mathcal{V} = \{1,2,3\}$ connected by a line $\mathcal{E} = \{e_1, e_2\}$, with $e_1 = (1,2)$ and $e_2 = (2,3)$, we have $\vec{a} = \{a_1,a_2,a_3\}$, thus $\vec{a}_{e_1} = \{a_1,a_2\}$ and $\vec{a}_{e_2} = \{a_2,a_3\}$, respectively. To find the Q-value for some history $\vec{h}$ (or equivalent belief b) based on the local actions, we define a local payoff function $Q_e(\vec{h},\vec{a}_e)$ for each edge $e \in \mathcal{E}$, where $\vec{a}_e \in \mathcal{A}_e$ is the projection of $\vec{a}$ to the agents in the edge. Then, $Q(\vec{h},\vec{a}) \approx \sum_e Q_e(\vec{h},\vec{a}_e)$. Instead of finding $Q_e(\vec{h},\vec{a}_e)$, we maintain local predictions $\hat{Q}_e(\vec{h},\vec{a}_e) = \mathbb{E}\left[Q(\vec{h},\vec{a}) \mid \vec{a}_e\right]$ of the joint Q-value. A *mixture of experts* (MoE) combines these local estimates of Q:

$$Q(\vec{h},\vec{a}) \approx \hat{Q}(\vec{h},\vec{a}) = \textstyle\sum_e \omega_e \hat{Q}_e(\vec{h},\vec{a}_e), \qquad (3)$$

where $\omega_e \geq 0$ is the weight for edge e, s.t. $\sum_{e\in\mathcal{E}} \omega_e = 1$. We assume uniform mixture weights $\omega_e = {}^1\!/_{|\mathcal{E}|}$ throughout the remainder of the paper. We pick the estimated maximizing joint action $\vec{a}^{\#} \approx \vec{a}^*$ over the sum of local estimates of Q:

$$\vec{a}^{\#} = \arg\max_{\vec{a}} \textstyle\sum_e \omega_e \hat{Q}_e(\vec{h},\vec{a}_e). \qquad (4)$$

We thus aim to find local actions (for the edges of the graph) that maximize the estimated joint value function. Notice that when finding $\vec{a}^{\#}$, any agent i might belong to multiple edges, and therefore agent i must be assigned the same action $a_i^{\#} \in \mathcal{A}_i$ in all edges $e \in \mathcal{E}$ where $i \in e$. We can compute the maximum with graphical inference algorithms, such as Variable Elimination (VE) and Max-Plus (MP) (Vlassis, Elhorst, and Kok 2004). Appendix D provides an overview.

3.2 Factored-Value POMCP

Factored-value POMCP (Amato and Oliehoek 2015) consists of two techniques that exploit the structure of a CG to scale POMCP to problems with large action and observation spaces. Next, we outline how these techniques factor the action space to introduce statistics for each edge $e \in \mathcal{E}$ for computing the UCB1 value of the local joint action space $\vec{a}_e$. Both these algorithms require an inference algorithm to compute Eq. (4) during and after the simulations.

Factored statistics (FS-POMCP). FS-POMCP uses the structure of MPOMDPs and stores the statistics Q, N, n in a factorized manner. This adaption is more space-efficient and also allows for improved action selection in large action spaces by maximizing over the factored Q-functions. More precisely, the tree structure in FS-POMCP remains the same as in POMCP, representing the history $\vec{h}$ with associated visit counts $N(\vec{h})$ and particles $B(\vec{h})$ in the observation nodes. The action nodes maintain a set of statistics $Q_e(\vec{h},\vec{a}_e)$, $N(\vec{h},\vec{a}_e)$ for each edge $e \in \mathcal{E}$, independently. Thus, the MoE optimization from Eq. (3) is applied directly in each action node of the search tree. This improves over POMCP as the combination of local action spaces $\vec{a}_e$ of each edge $e \in \mathcal{E}$ is smaller than the joint action space

$\vec{a}$. During search, the action $\vec{a}^{\#}$ is selected by maximizing over the UCB1 values (Eq. (2)) of the local Q-functions: $\arg\max_{\vec{a}^{\#}} \sum_{e \in \mathcal{E}} \mathcal{UCB}(Q_e(\vec{h}, \vec{a}_e), N(\vec{h}), N(\vec{h}, \vec{a}_e))$.

Factored trees (FT-POMCP). FT-POMCP constructs a tree for every edge e. This tree represents the factored histories $\vec{h}_e$, which consists of a sequence of factored actions and observations $\vec{h}_{e,t} = (\vec{a}_{e,0}, \vec{o}_{e,1}, \dots, \vec{a}_{e,t-1}, \vec{o}_{e,t})$. This further reduces the scope of $Q_e(\vec{h}, \vec{a}_e)$ to $Q_e(\vec{h}_e, \vec{a}_e)$ by introducing an expert for every $\vec{h}_e, \vec{a}_e$ pair. In each tree, the action nodes maintain statistics $Q_e(\vec{h}_e, \vec{a}_e)$, and $N(\vec{h}_e, \vec{a}_e)$, and the observation nodes maintain $N(\vec{h}_e)$ and $B(\vec{h}_e)$ according to the factored history $\vec{h}_e$. During search, we again maximize with respect to the UCB1 value (using Eq. (2)) of local Q-functions: $\arg\max_{\vec{a}^{\#}} \sum_{e \in \mathcal{E}} \mathcal{UCB}(Q_e(\vec{h}_e, \vec{a}_e), N(\vec{h}_e), N(\vec{h}_e, \vec{a}_e))$.

4 Scalable Particle Filtering

In MPOMDPs with large state spaces, repeated execution of the Bayesian belief update is intractable as each update requires $\mathcal{O}(|\mathcal{S}|^2)$ computations. We represent this belief with *weighted particle filters* to ensure scalability. The following subsection introduces how we incorporate these filters in W-POMCP and FS-W-POMCP. The second subsection introduces our method that uses the structure of a coordination graph to condition the belief on a local part of the observation space, which we apply to FT-W-POMCP and FT-PFT.

Particle filtering. Particle filtering (Thrun, Burgard, and Fox 2005) represents the belief by sequential Monte Carlo approximations, alleviating the bottleneck of the belief update. In an *unweighted* filter, the belief approximation is a set $\bar{b} = \{(s^{(k)})\}_{k=1}^{K}$, with $s^{(k)} \in \mathcal{S}$, and is updated using rejection sampling on the real observation $\vec{o}$; $\bar{b}' = \{s'^{(k)} \colon \vec{o} = \vec{o}^{\,(k)}\}$, where $s'^{(k)}$, $\vec{o}^{\,(k)}$ are generated from $s^{(k)}, \vec{a}$ by $\mathcal{G}$ (Kochenderfer et al. 2015). POMCP implicitly uses an unweighted particle filter by using the online particle belief $B(\vec{h})$ in the observation nodes to represent $\bar{b}$.

4.1 Weighted Particle Filtering

Weighted particle filters approximate the belief by a weighted set of K particles $\bar{b} = \{(s^{(k)}, w^{(k)})\}_{k=1}^{K}$, where $s^{(k)} \in \mathcal{S}$ is a state in the filter and $w^{(k)} \in \mathbb{R}^{+}$ the associated weight. We update beliefs in weighted filters with *importance sampling* as in the *bootstrapped particle filter* (Gordon, Salmond, and Smith 1993). In the bootstrapped particle filter, the proposal distribution is the transition function $s'^{(k)} \sim \mathcal{T}(\cdot \mid s^{(k)}, \vec{a})$, and the importance weights are computed from the observation function $w'^{(k)} \propto w^{(k)} \mathcal{O}(\vec{o} \mid s'^{(k)}, \vec{a})$. Additionally, the posterior belief is re-sampled at every time step to alleviate sample degeneracy, after which the weights are set to $1/K$, which is known as *sequential importance re-sampling* (SIR). We decide whether to re-sample in our SIR filter by comparing the estimated *effective sample size* (ESS) of the particle filters with respect to the number of particles (Septier and Peters 2016). The $\text{ESS}(\bar{b}) \approx (\sum_{k=1}^{K}(w^{(k)})^2)^{-1}$ quantifies weight disparity, which is an indicator for sample degeneracy. The *likelihood* $\mathcal{L}$ of a belief update in a SIR filter represents the probability of the new belief given the observation, action, and previous belief. It is a statistic on the quality of the approximate belief update (Katt, Oliehoek, and Amato 2019). It is computed from the sum of all updated weights multiplied by the previous likelihood $\mathcal{L}(\bar{b}') = \sum_k w'^{(k)} \mathcal{L}(\bar{b})$ where $\mathcal{L}(\bar{b}) = 1$ when $\bar{b}$ was initialized from b_0.

W-POMCP and FS-W-POMCP. In both W-POMCP and FS-W-POMCP, we represent the current root-node belief estimate with an offline weighted filter $\bar{b}$ that we update independently of the search tree instead of using the unweighted online particles B stored inside the tree. We provide the pseudo-code for the SIR filter that updates $\bar{b}$ in Appendix F.2. Additionally, FS-W-POMCP maintains statistics in each action node for the actions of pairs of agents instead of all joint actions, as explained previously for FS-POMCP.

Particle filtering in MPOMDPs. In MPOMDPs, the observation signal becomes increasingly sparse as the number of agents increases, as it commonly depends on the probability of all individual observations. This can result in an impoverishment of the particles. Comparably, the likelihood of matching the received joint observation in the rejection update is small for unweighted filters in larger observation spaces. If the particle filter reaches a deprived state where no particles remain, the planner defaults to a baseline policy.

4.2 Particle Filtering in a Coordination Graph

To increase the scalability and decrease the chance of deprivation of the particle filter in large observation spaces, we introduce a general filtering approach for $\bar{b}$ based on the structure of a coordination graph $(\mathcal{V}, \mathcal{E})$, independent of the online planning algorithm. This method applies to both FT-POMCP (Sect. 3.2) as well as FT-PFT (introduced in Sect. 5). We exploit the structure in the following way. For every edge $e \in \mathcal{E}$, we introduce a separate particle filter $\bar{b}_e$ with K_e particles. We choose K_e such that $K = \sum_e K_e$. This method makes the following assumption.

Assumption 1. *Individual observations probabilities, as given by the individual observation model $\mathcal{O}_i \colon \mathcal{S} \times \mathcal{A} \to \Delta(\Omega_i)$, are conditionally independent given the successor state and the previous action. Therefore, we write the observation model as the product of individual observation probabilities: $\mathcal{O}(\vec{o} \mid s', \vec{a}) = \prod_{i \in \mathcal{I}} \mathcal{O}_i(o_i \mid s', \vec{a})$, with $o_i \in \Omega_i$.*

Note that we condition the individual observations on the joint state and action instead of the assumption of *observational independence* of ND-POMDPs (Nair et al. 2005) and, distinctly from *factored beliefs* (Messias, Spaan, and Lima 2011) and *factored particle filtering* (Ng, Peshkin, and Pfeffer 2002), we do not assume any state space factorization.

Local updates. Using Assumption 1, we update the particle filters for the edges by the local part of the observation space $\vec{o}_e \in \Omega_e$. For an edge $e = (i, j)$, we retrieve the local observation $\vec{o}_e$ by taking the individual observations o_i, o_j from the joint observation $\vec{o}$. Then, we change the importance weights of the filtering procedure to be based on the *local observation probability* $\mathcal{O}_e(\vec{o}_e \mid s', \vec{a})$, instead of the joint observation probability $\mathcal{O}(\vec{o} \mid s', \vec{a})$. For each edge e, the local observation probability $\mathcal{O}_e(\vec{o}_e \mid s', \vec{a}) = \prod_{i \in e} \mathcal{O}_i(o_i \mid s', \vec{a})$ is the probability of observing $o_i \in \Omega_i$ for each agent i in e. Consider an offline *weighted* filter $\bar{b}_e = \{(s^{(k)}, w_e^{(k)})\}_{k=1}^{K_e}$ and joint observation $\vec{o}$. We first propagate the particle ${s'}^{(k)} \sim \mathcal{T}(\cdot \mid s^{(k)}, \vec{a})$ and then compute the new importance sampling weight ${w'_e}^{(k)} \propto {w_e}^{(k)} \mathcal{O}_e(\vec{o}_e \mid s', \vec{a})$, where $\vec{o}_e$ is the local part of the observation $\vec{o}$. Every $\bar{b}_e$ approximates the belief for each $s \in \mathcal{S}$ as: $\bar{b}_e(s) = \sum_{k=1}^{K_e} w_e^{(k)} \delta(s, s^{(k)}) / \sum_{l=1}^{K_e} w_e^{(l)}$, where δ is the Kronecker delta function. Now, $\bar{b}_e$ represents the particle-belief representing the history $\vec{h}^e$ of joint actions $\vec{a}$ and local observations $\vec{o}_e$ for edge e, and $\mathcal{L}(\bar{b}_e) = \sum_{k=1}^{K_e} w_e^{(k)}$ its likelihood. We provide more details in Appendix F.2.

Ensembling. Since we run multiple particle filters in parallel, we must decide how to fuse these beliefs together for sampling. We propose to treat this set of beliefs as an ensemble. We determine a procedure for sampling the ensemble for every simulation iteration. We use the likelihood of the weighted particle filter update as a statistic for the quality of the belief approximation (Katt, Oliehoek, and Amato 2019). Intuitively, we sample more often from higher-quality filters. More precisely, we give filters that contain particles with a higher probability of generating the true observation a higher chance of getting sampled. We sample from the set of filters with probabilities proportional to the likelihoods of the particle filters $\mathcal{L}(\bar{b}_e)$ of the edges: $s \sim \bar{b}_e$ w.p. $\mathcal{L}(\bar{b}_e) / \sum_{e' \in \mathcal{E}} \mathcal{L}(\bar{b}_{e'})$. Altogether, this ensemble particle-filter results in the following approximation $\bar{b}$ of the belief b for each $s \in \mathcal{S}$: $b(s) \approx \bar{b}(s) = \sum_{e \in \mathcal{E}} \mathcal{L}(b_e) / \sum_{e'} \mathcal{L}(b_{e'}) \bar{b}_e(s)$. FT-W-POMCP and FT-PFT maintain an offline belief estimate $\bar{b}$ consisting of the offline local beliefs $\bar{b}_e$ for each e.

Limitation. Our proposal distribution remains $\mathcal{T}$ across the ensemble, but our observation distributions are the local function $\mathcal{O}_e$ for every filter $\bar{b}_e$. Therefore, these local filters will be biased towards the posterior $p(s \mid \vec{h}^e, b_0)$ instead of $p(s \mid \vec{h}, b_0)$, where, as before, $\vec{h}^e$ is the history of factored observations $\vec{o}_e$ and joint actions $\vec{a}$. Since each filter considers only local observations, the local filters cannot recover a joint belief that depends on all agents (Capitán et al. 2011).

5 Sparse-PFT with Value Factorization

In this section, we lift Sparse-PFT to MPOMDPs. We propose extensions that exploit the factorization of the action space as in Sect. 3.2. Firstly, we introduce a particle belief approximation and Sparse-PFT for MPOMDPs. Then, we introduce variants with *factored statistics* (**FS-PFT**) and *factored trees* (**FT-PFT**) to combat large action spaces.

Particle approximation. For POMDPs, it is natural to consider a fully observable belief-MDP, whose state space are the beliefs and the action space is unchanged (Cassandra, Kaelbling, and Littman 1994). The same construction for MPOMDP yields a belief-MMDP. Particle-belief-MDPs approximate belief-MDPs (Lim, Tomlin, and Sunberg 2020; Lim et al. 2023). Similarly, we introduce the particle-belief-MMDP as an approximation of an MPOMDP:

Definition 2 (PB-MMDP). *The* Particle-Belief-MMDP *for an MPOMDP* $\mathcal{M} = \langle \mathcal{I}, \mathcal{S}, \mathcal{A}, \mathcal{T}, r, \Omega, \mathcal{O}, \gamma \rangle$ *is a tuple* $\mathcal{M}' = \langle \mathcal{I}, \Sigma, \mathcal{A}, \tau, \rho, \gamma \rangle$ *with* states $\Sigma = (\mathcal{S} \times \mathbb{R}^+)^C$ *consisting of online weighted particle beliefs* $\tilde{b} = \{(s^{(k)}, w^{(k)})\}_{k=1}^{C}$ *encoded by C particles, the* transition density function $\tau \colon \Sigma \times \mathcal{A} \to \Delta(\Sigma)$ *defined by* $\tau(\tilde{b}' \mid \tilde{b}, \vec{a}) = \sum_{\vec{o} \in \Omega} \Pr(\tilde{b}' \mid \tilde{b}, \vec{a}, \vec{o}) \Pr(\vec{o} \mid \tilde{b}, \vec{a})$, *and the* reward function $\rho \colon \Sigma \times \mathcal{A} \to \mathbb{R}$ *defined by* $\rho(\tilde{b}, \vec{a}) = \sum_k w^{(k)} \mathcal{R}(s^{(k)}, \vec{a}) / \sum_l w^{(l)}$.

Simulating the PB-MMDP requires us to update the associated generative model. We simulate particle beliefs $\tilde{b}$ of size C instead of individual states to estimate Q. Consequentially, the generative model $\mathcal{G}_{PF} \colon \Sigma \times \mathcal{A} \to \Sigma \times \mathbb{R}$ updates the state based on the action and returns the particle-based reward ρ as specified above. This extension increases the complexity of the generative model by a factor $\mathcal{O}(C)$.

5.1 Sparse Particle Filter Tree

Sparse-PFT is an application of UCT to the PB-MMDP. While it was designed for continuous state spaces, the fact that the tree branches on a fixed number of belief nodes instead of the number of joint observations is beneficial in our setting. Sparse-PFT constructs a sparse particle-belief tree incrementally during a forward search by allowing each action node to expand up to C particle-belief nodes. The particle-belief nodes correspond to the states of the particle-belief MMDP (Def. 2). The root particle-belief $\tilde{b} \leftarrow \{(s^{(k)}, 1/C)\}_{k=1}^{C} \sim \bar{b}$ is sampled at every simulation iteration from the current offline belief $\bar{b}$. Following our separation of online and offline beliefs, the number of particles in the offline belief $|\bar{b}| \gg C$ can be much greater than the simulated belief inside the tree $\tilde{b}$. If the number of children $|\mathrm{Ch}(\tilde{b}, \vec{a})|$ of the action node is less than C, then we simulate the particle-belief through $\mathcal{G}_{PF}$ to obtain the next particle-belief $\tilde{b}'$ and particle-based reward ρ. Otherwise, $\tilde{b}'$ and ρ are sampled uniformly from $\mathrm{Ch}(\tilde{b}, \vec{a})$. We continue the simulation and traverse the particle-belief tree until we reach a leaf node or a predetermined maximum depth. If we reach a leaf node, a rollout is performed. Scalability is partially addressed because the branching factor of the belief nodes is independent of the observation size. However, a full enumeration of the action space is still required for selecting actions according to UCB1, which is impractical in MPOMDPs.

5.2 Sparse-PFT for MPOMDPs

We introduce two extensions to improve upon the weakness of Sparse-PFT when operating with large action spaces.

Factored statistics (FS-PFT). We propose to keep factored action statistics in the nodes of the particle filter tree, similar to FS-POMCP. In addition to the node visit count $N(\tilde{b})$, we maintain sets of statistics $Q_e(\tilde{b}, \vec{a}_e)$, $N(\tilde{b}, \vec{a}_e)$ in every particle filter belief node that predicts the Q-function for every edge $e \in \mathcal{E}$, applying MoE optimization from Eq. (3) directly in the nodes. The offline belief $\bar{b}$ is represented by an external weighted particle filter, as in FS-W-POMCP. Finally, similarly to the previous factored statistics algorithms, a graphical inference algorithm selects the maximal joint action $\vec{a}^{\#}$ during the search by maximizing over the UCB1 values: $\arg\max_{\vec{a}^{\#}} \sum_{e\in\mathcal{E}} \mathcal{UCB}(Q_e(\tilde{b}, \vec{a}_e), N(\tilde{b}), N(\tilde{b}, \vec{a}_e))$.

Factored trees (FT-PFT). Additionally, we introduce the construction of multiple particle-belief trees in parallel, one for each $e \in \mathcal{E}$. These trees have a local action space $\vec{a}_e$ and maintain statistics $Q_e(\tilde{b}, \vec{a}_e)$, $N_e(\tilde{b})$, $N_e(\tilde{b}, \vec{a}_e)$ for the agents associated with the edge. Since particle filter trees do not explicitly branch on observations, only the action space is factored inside the trees. We use a single joint particle belief step in each layer to reduce overhead. Thus, every tree is constructed from the same simulated particle filter beliefs. Although the belief nodes might have the same particles, we maintain independent visit count statistics N_e for each belief node and associated local joint actions $\vec{a}_e \in \mathcal{A}_e$, respectively. The inference equation for picking the maximal UCB1 action (using Eq. (2)) is given by: $\arg\max_{\vec{a}^{\#}} \sum_{e\in\mathcal{E}} \mathcal{UCB}(Q_e(\tilde{b}, \vec{a}_e), N_e(\tilde{b}), N_e(\tilde{b}, \vec{a}_e))$. The offline belief $\bar{b}$ is maintain identically to FT-W-POMCP (Sect. 4.2), by the ensemble of offline beliefs $\bar{b}_e$. In addition to the above, we suspect the improvement of FT-PFT over Sparse-PFT is an increase in node re-use and search depth due to the smaller factored action space in the trees.

6 Experimental Evaluation

We evaluate the effectiveness of our methods on MPOMDPs with many agents. Abbreviations follow those in Table 1. The key question is **Q1**: *Does the use of coordination graphs (**CGs**) accelerate online planners for MPOMDPs in general?* We evaluate this question on three benchmarks, one with a given coordination graph and two with an artificially chosen graph. Regarding our novel algorithms introduced in Sect. 4 and 5, we evaluate **Q2**: *Do (FS/FT)-W-POMCP variants improve over (unweighted) (FS/FT)-POMCP variants*, and, **Q3**: *Do FS/FT-PFT improve over Sparse-PFT*?

Benchmarks. FIREFIGHTINGGRAPH (FFG, Oliehoek et al. 2008) has been used to evaluate factored POMCP (Amato and Oliehoek 2015). Agents stand in a line, and houses are located to the left and right of each agent. Agents have two actions: fight fires to their left or right. Multi-agent ROCKSAMPLE (MARS, Cai et al. 2021) extends single-agent RockSample (Smith and Simmons 2004). MARS environments are defined by their size m, the number of agents n, and the number of rocks k, with $k = m = 15$. In CAPTURETARGET (CT), agents are tasked with capturing a moving target. We depict results for CT in Appendix A.2. Detailed benchmark descriptions are in Appendix G.

Experimental set-up. All algorithm variants are implemented in the same Python prototype, published online[1].

[1]https://zenodo.org/records/10409525.

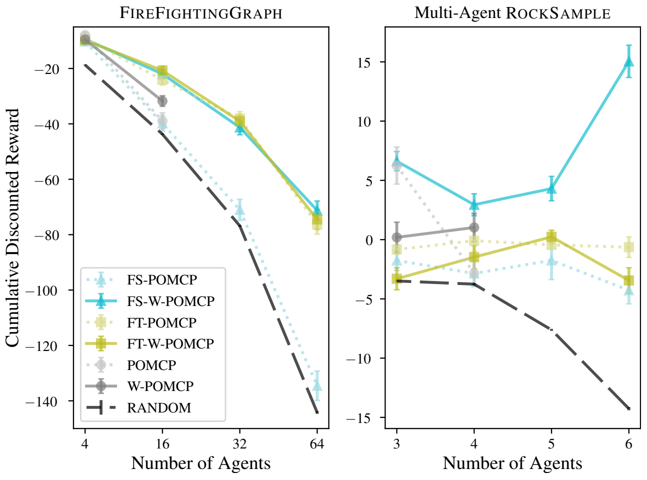

Figure 1: Performance comparison for POMCP variants with (solid) and without (dotted) weighted particle filtering.

All code ran on a machine with an Intel(R) Core(TM) i9-10980XE CPU @ 3.00GHz and 256 GB RAM (8 x 32GB DDR4-3200). Our Python wrapper executed episodes in parallel on 34 threads such that each episode had access to $256/34 \approx 7.5$GB of RAM. All reported results are the achieved returns averaged over 100 episodes with error bars representing 95% confidence intervals. We did not run an extensive hyperparameter optimization for any algorithm, and we list the most important parameters in Tab. 2 of Appendix A.1. All algorithms ran with a maximum of 5s and 15s per step on FFG/CT and MARS, respectively. If the particle filter belief is deprived at any point in time during the episode, the policy defaults to a random policy. We set the number K of particles in the joint filters such that $K = \sum_e K_e$ in the factored filters, e.g., if we have three edges with $K_e = 100$, then the joint counterpart has $K = 300$. For MARS and CT, we chose the CG as a *line* ($n-1$ factors) for odd numbers of agents and a *team* formation ($n/2$ factors) where pairs of agents cooperate for even numbers. The single-agent algorithms could not run with more than 20 and 5 agents on FFG and MARS, respectively.

Discussion. Below, we analyze the results as answers to the three questions. **Q1.** We study Q1 across our different set-ups. The single-agent algorithms (Sparse-PFT, POMCP, and W-POMCP) are out-scaled by their competitors with value factorization (Fig. 1 and 2) in FFG and MARS. However, planning on the joint value performs better in settings with fewer agents. In MARS and CT, the agents move and thus may coordinate dynamically. Therefore, the desired agent coordination does not induce a *sparse* coordination graph, meaning the CG acts as a heuristic. The results show that assuming some arbitrary, sparse static graph is helpful, even if this assumes no coordination between agents that, in principle, should coordinate. We find that the static heuristic performs well when many agents are involved. Thus, **CGs (as a heuristic) accelerate planning**. **Q2.** FS-W-POMCP outperforms FS-POMCP across all three benchmarks, show-

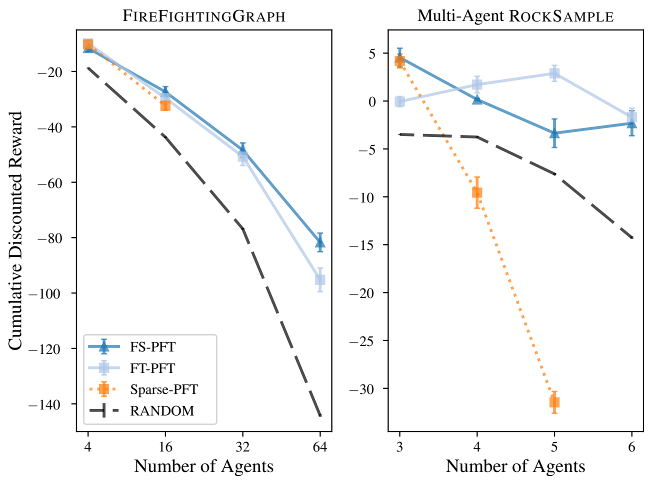

Figure 2: Comparison between Sparse-PFT (dotted) and our PFT variants with value factorization (solid).

ing that **POMCP benefits from offline weighted particle filtering**. The difference between FT-POMCP and FT-W-POMCP is smaller, as FT-POMCP also benefits from the fact that the belief representation, which is offline, consists of local beliefs $\tilde{b}_e$ for each e, albeit unweighted (see Appendix E). **Q3.** Sparse-PFT performs well in MPOMDPs with fewer agents, but with more agents, it runs out of memory or fails to find a good estimate of Q due to its naive enumeration of the action space (Fig. 2). FS-PFT and FT-PFT do scale to settings with many agents. Thus, **CGs alleviate Sparse-PFTs scaling issues in many-agent POMDPs.** However, they achieve comparable (FFG) or lower (MARS) returns in the settings with few agents. **Combining CGs with weighted filtering performs well across.** We cross-evaluate our contributions in (Fig. 3, Appendix A.2). We find that both factored W-POMCP and PFT algorithm variants are suited for many-agent POMDPs and perform well across FFG and MARS, but POMCP variants generally perform better. The improvement of FS-W-POMCP over FS-POMCP is consistent. FT-W-POMCP is slightly better on FFG and significantly improves over FT-POMCP in CT, but performs equal or worse on MARS. **The action selection method has a noticeable influence on performance.** In MARS and CT, VE is the best-performing algorithm. However, MP achieves much higher returns in FFG (Fig. 4, Appendix A.2). The factored algorithms are sensitive to the method that maximizes over the local predictions, as recently also demonstrated in the fully observable setting (Choudhury et al. 2022).

7 Related Work

Multi-agent Markov models. MPOMDPs reside in a realm of models for cooperative multi-agent systems with partial observability. Distributed cooperative systems (Dec-POMDPs, Oliehoek and Amato 2016) remove the communication assumptions of MPOMDPs. However, they are much more computationally complex (doubly exponential), as agents need to reason over each other's policies. Messias, Spaan, and Lima (2011) considered factored MPOMDPs by assuming shared communication in Dec-POMDPs, computing policies over *factored beliefs*. Additionally, they studied lifting the instantaneous communication assumptions by asynchronous execution (Messias, Spaan, and Lima 2013). Our algorithms build on prior work by Amato and Oliehoek (2015). Therefore, their work is summarized in Sect. 3. Zhou et al. (2019) introduce a further decentralized MCTS algorithm for transition-independent MPOMDPs. Choudhury et al. (2022) consider a fully observable MMDP setting and study action selection under state-dependent coordination graphs. Recently, MPOMDPs were also studied with barrier functions over the joint belief (Ahmadi et al. 2019) and to support multi-object tracking (Nguyen et al. 2020).

Single-agent online planning. POMCPOW (Sunberg and Kochenderfer 2018) and Sparse-PFT (Lim et al. 2023), also in Table 1, are algorithms that improved UCT-based planners for POMDPs with continuous spaces. They, i.a., replaced unweighted belief estimates with importance sampling estimates using weighted particle filters (Sect. 4). We summarize Sparse-PFT in Sect. 5. POMCPOW shares characteristics with W-POMCP and Sparse-PFT, simulating single states but maintaining weighted particles in the tree. DESPOT (Ye et al. 2017) and AdaOPS (Wu et al. 2021) are alternative, orthogonal online planners that are distinguishable from MCTS methods (e.g., POMCP). DESPOT utilizes a set of deterministic scenarios and heuristic tree searches to reduce variance in the value estimates instead of the independent simulations via MCTS. Its extensions employ alpha-vectors to fuse similar paths in the tree (Garg, Hsu, and Lee 2019) or (GPU) parallelization in factored simulators (Cai et al. 2021). AdaOPS also employs offline and weighted particle filtering. Distinctively, it uses adaptive particle filtering (Fox 2001), which requires a partitioning of the state space into grids. It relies on a full-width search instead of simulations, during which it fuses similar observation branches. Both algorithms work well with small-sized discrete action spaces. However, it is unclear how value factorization from coordination graphs can be incorporated, as both algorithms expand the full action space at each new node instead of picking the most promising action to simulate via the UCB1 policy. In MPOMDPs, expanding the combinatorial joint action space is impractical.

8 Conclusion

In this paper, we studied how to simultaneously tackle the belief and value estimation challenges in online planning for MPOMDPs. We presented extensions of factored POMCP and novel variants of the Sparse-PFT algorithms tailored specifically for many-agent online planning with partial observability. The empirical evaluation showed the effectiveness of combining weighted particle filtering and value factorization in settings with many agents. However, it is also clear that planning on the joint value suffices when few agents are involved. Future work consists of alleviating the communication assumptions (Spaan, Oliehoek, and Vlassis 2008; Oliehoek and Spaan 2012; Messias, Spaan, and Lima 2013), exploring extensions for continuous MPOMDPs, or learning the coordination graph (Kok et al. 2005).

Acknowledgments

We would like to thank the anonymous reviewers for their valuable feedback. This research has been partially funded by the NWO grant NWA.1160.18.238 (PrimaVera) and the ERC Starting Grant 101077178 (DEUCE). Additionally, we would like to thank the ELLIS Unit Nijmegen and Radboud AI for their support.

References

Ahmadi, M.; Singletary, A.; Burdick, J. W.; and Ames, A. D. 2019. Safe Policy Synthesis in Multi-Agent POMDPs via Discrete-Time Barrier Functions. In *CDC*, 4797–4803. IEEE.

Amato, C.; and Oliehoek, F. A. 2015. Scalable Planning and Learning for Multiagent POMDPs. In *AAAI*, 1995–2002. AAAI Press.

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-time Analysis of the Multiarmed Bandit Problem. *Mach. Learn.*, 47(2-3): 235–256.

Browne, C.; Powley, E. J.; Whitehouse, D.; Lucas, S. M.; Cowling, P. I.; Rohlfshagen, P.; Tavener, S.; Liebana, D. P.; Samothrakis, S.; and Colton, S. 2012. A Survey of Monte Carlo Tree Search Methods. *IEEE Trans. Comput. Intell. AI Games*, 4(1): 1–43.

Cai, P.; Luo, Y.; Hsu, D.; and Lee, W. S. 2021. HyP-DESPOT: A hybrid parallel algorithm for online planning under uncertainty. *Int. J. Robotics Res.*, 40(2-3).

Capitán, J.; Merino, L.; Caballero, F.; and Ollero, A. 2011. Decentralized Delayed-State Information Filter (DDSIF): A new approach for cooperative decentralized tracking. *Robotics Auton. Syst.*, 59(6): 376–388.

Cassandra, A. R.; Kaelbling, L. P.; and Littman, M. L. 1994. Acting Optimally in Partially Observable Stochastic Domains. In *AAAI*, 1023–1028. AAAI Press / The MIT Press.

Choudhury, S.; Gupta, J. K.; Morales, P.; and Kochenderfer, M. J. 2022. Scalable Online Planning for Multi-Agent MDPs. *J. Artif. Intell. Res.*, 73: 821–846.

Fischer, J.; and Tas, Ö. S. 2020. Information Particle Filter Tree: An Online Algorithm for POMDPs with Belief-Based Rewards on Continuous Domains. In *ICML*, volume 119 of *Proceedings of Machine Learning Research*, 3177–3187. PMLR.

Fox, D. 2001. KLD-Sampling: Adaptive Particle Filters. In *NIPS*, 713–720. MIT Press.

Galesloot, M. F. L.; Simao, T. D.; Junges, S.; and Jansen, N. 2023. Factored Online Planning in Many-Agent POMDPs. arXiv:2312.11434.

Garg, N. P.; Hsu, D.; and Lee, W. S. 2019. DESPOT-Alpha: Online POMDP Planning with Large State and Observation Spaces. In *Robotics: Science and Systems*.

Gordon, N. J.; Salmond, D. J.; and Smith, A. F. M. 1993. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings F (Radar and Signal Processing)*, 140(2): 107–113(6).

Guestrin, C.; Lagoudakis, M. G.; and Parr, R. 2002. Coordinated Reinforcement Learning. In *ICML*, 227–234. Morgan Kaufmann.

Guestrin, C.; Venkataraman, S.; and Koller, D. 2002. Context-Specific Multiagent Coordination and Planning with Factored MDPs. In *AAAI/IAAI*, 253–259. AAAI Press / The MIT Press.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and Acting in Partially Observable Stochastic Domains. *Artif. Intell.*, 101(1-2): 99–134.

Katt, S.; Oliehoek, F. A.; and Amato, C. 2019. Bayesian Reinforcement Learning in Factored POMDPs. In *AAMAS*, 7–15. International Foundation for Autonomous Agents and Multiagent Systems.

Kearns, M. J.; Mansour, Y.; and Ng, A. Y. 2002. A Sparse Sampling Algorithm for Near-Optimal Planning in Large Markov Decision Processes. *Mach. Learn.*, 49(2-3): 193–208.

Kochenderfer, M.; Amato, C.; Chowdhary, G.; How, J.; and Reynolds, H. 2015. *Decision Making Under Uncertainty: Theory and Application*. MIT Lincoln Laboratory Series. MIT Press. ISBN 978-0-262-02925-4.

Kocsis, L.; and Szepesvári, C. 2006. Bandit Based Monte-Carlo Planning. In *ECML*, volume 4212 of *Lecture Notes in Computer Science*, 282–293. Springer.

Kok, J. R.; Hoen, P. J.; Bakker, B.; and Vlassis, N. 2005. Utile Coordination: Learning Interdependencies Among Cooperative Agents. In *CIG*. IEEE.

Lim, M. H.; Becker, T. J.; Kochenderfer, M. J.; Tomlin, C. J.; and Sunberg, Z. N. 2023. Optimality Guarantees for Particle Belief Approximation of POMDPs. *J. Artif. Intell. Res.*, 77: 1591–1636.

Lim, M. H.; Tomlin, C. J.; and Sunberg, Z. N. 2020. Sparse Tree Search Optimality Guarantees in POMDPs with Continuous Observation Spaces. In *IJCAI*, 4135–4142. ijcai.org.

Messias, J. V.; Spaan, M. T. J.; and Lima, P. U. 2011. Efficient Offline Communication Policies for Factored Multiagent POMDPs. In *NIPS*, 1917–1925.

Messias, J. V.; Spaan, M. T. J.; and Lima, P. U. 2013. Multiagent POMDPs with asynchronous execution. In *AAMAS*, 1273–1274. IFAAMAS.

Nair, R.; Varakantham, P.; Tambe, M.; and Yokoo, M. 2005. Networked Distributed POMDPs: A Synergy of Distributed Constraint Optimization and POMDPs. In *IJCAI*, 1758–1760. Professional Book Center.

Ng, B.; Peshkin, L.; and Pfeffer, A. 2002. Factored Particles for Scalable Monitoring. In *UAI*, 370–377. Morgan Kaufmann.

Nguyen, H. V.; Rezatofighi, H.; Vo, B.; and Ranasinghe, D. C. 2020. Multi-Objective Multi-Agent Planning for Jointly Discovering and Tracking Mobile Objects. In *AAAI*, 7227–7235. AAAI Press.

Oliehoek, F. A.; and Amato, C. 2016. *A Concise Introduction to Decentralized POMDPs*. Springer Briefs in Intelligent Systems. Springer.

Oliehoek, F. A.; and Spaan, M. T. J. 2012. Tree-Based Solution Methods for Multiagent POMDPs with Delayed Communication. In *AAAI*, 1415–1421. AAAI Press.

Oliehoek, F. A.; Spaan, M. T. J.; Whiteson, S.; and Vlassis, N. 2008. Exploiting locality of interaction in factored Dec-POMDPs. In *AAMAS (1)*, 517–524. IFAAMAS.

Pynadath, D. V.; and Tambe, M. 2002. The Communicative Multiagent Team Decision Problem: Analyzing Teamwork Theories and Models. *J. Artif. Intell. Res.*, 16: 389–423.

Septier, F.; and Peters, G. W. 2016. Langevin and Hamiltonian Based Sequential MCMC for Efficient Bayesian Filtering in High-Dimensional Spaces. *IEEE J. Sel. Top. Signal Process.*, 10(2): 312–327.

Silver, D.; and Veness, J. 2010. Monte-Carlo Planning in Large POMDPs. In *NIPS*, 2164–2172. Curran Associates, Inc.

Smith, T.; and Simmons, R. G. 2004. Heuristic Search Value Iteration for POMDPs. In *UAI*, 520–527. AUAI Press.

Spaan, M. T. J. 2012. Partially Observable Markov Decision Processes. In *Reinforcement Learning*, volume 12 of *Adaptation, Learning, and Optimization*, 387–414. Springer.

Spaan, M. T. J.; Oliehoek, F. A.; and Vlassis, N. 2008. Multiagent Planning Under Uncertainty with Stochastic Communication Delays. In *ICAPS*, 338–345. AAAI.

Sunberg, Z. N.; and Kochenderfer, M. J. 2018. Online Algorithms for POMDPs with Continuous State, Action, and Observation Spaces. In *ICAPS*, 259–263. AAAI Press.

Thrun, S. 1999. Monte Carlo POMDPs. In *NIPS*, 1064–1070. The MIT Press.

Thrun, S.; Burgard, W.; and Fox, D. 2005. *Probabilistic robotics*. Intelligent robotics and autonomous agents. MIT Press.

Vlassis, N.; Elhorst, R.; and Kok, J. R. 2004. Anytime algorithms for multiagent decision making using coordination graphs. In *SMC (1)*, 953–957. IEEE.

Witwicki, S. J.; Castillo, J. C.; Messias, J. V.; Capitán, J.; Melo, F. S.; Lima, P. U.; and Veloso, M. M. 2017. Autonomous Surveillance Robots: A Decision-Making Framework for Networked Muiltiagent Systems. *IEEE Robotics Autom. Mag.*, 24(3): 52–64.

Wu, C.; Yang, G.; Zhang, Z.; Yu, Y.; Li, D.; Liu, W.; and Hao, J. 2021. Adaptive Online Packing-guided Search for POMDPs. In *NeurIPS*, 28419–28430.

Ye, N.; Somani, A.; Hsu, D.; and Lee, W. S. 2017. DESPOT: Online POMDP Planning with Regularization. *J. Artif. Intell. Res.*, 58: 231–266.

Zhou, X.; Wang, W.; Zhu, Y.; Wang, T.; and Zhang, B. 2019. Centralized Patrolling With Weakly-Coupled Agents Using Monte Carlo Tree Search. *IEEE Access*, 7: 157293–157302.