Efficient Multiagent Planning via Shared Action Suggestions

Dylan M. Asmar, Mykel J. Kochenderfer

Stanford Intelligent Systems Laboratory, Stanford University, Stanford, CA
asmar@stanford.edu, mykel@stanford.edu

Abstract

Decentralized partially observable Markov decision processes with communication (Dec-POMDP-Com) provide a framework for multiagent decision making under uncertainty, but the NEXP-complete complexity for finite-horizon problems renders solutions intractable in general. While sharing actions and observations can reduce the complexity to PSPACE-complete, we propose an approach that bridges POMDPs and Dec-POMDPs by communicating only suggested joint actions, eliminating the need to share observations while retaining near-centralized performance. Our algorithm estimates joint beliefs using shared actions to prune infeasible beliefs. Each agent maintains possible belief sets for other agents, pruning them based on suggested actions to form an estimated joint belief usable with any centralized policy. This approach requires solving a POMDP for each agent, reducing computational complexity while preserving performance. We demonstrate its effectiveness on several Dec-POMDP benchmarks, showing performance comparable to centralized methods when shared actions enable effective belief pruning. This action-based communication framework offers a natural avenue for integrating human-agent cooperation, opening new directions for scalable multiagent planning under uncertainty, with applications in both autonomous systems and human-agent teams.

Code — github.com/sisl/MCAS

1 Introduction

From complex engineering projects to emergency response teams, effective coordination between individuals is vital for success. The ability of humans to work together, communicate intuitively, and adapt to changing conditions has inspired researchers to explore cooperation in autonomous systems (Albrecht and Stone 2018). However, achieving such seamless collaboration in autonomous teams remains a significant challenge.

For multiagent decision making under uncertainty, where agents act without complete knowledge and outcomes are stochastic, the decentralized partially observable Markov decision process (Dec-POMDP) (Bernstein et al. 2002) is a widely used model. Agents must reason about both their environment and other agents' possible actions and beliefs without direct communication. While powerful, Dec-POMDPs are notoriously hard to solve, making them impractical for many real-world problems (Oliehoek and Amato 2016).

When agents can communicate, the computational burden can be reduced under certain assumptions (Pynadath and Tambe 2002; Goldman and Zilberstein 2003). However, sharing complete information is often impractical, and when communication is not lossless and free, the complexity of solving a finite-horizon Dec-POMDP with communication (Dec-POMDP-Com) remains NEXP-complete (Goldman and Zilberstein 2004).

As autonomous systems become more capable, human-machine collaboration becomes increasingly relevant (Johnson and Vera 2019; Dafoe et al. 2020). While combining human intuition with machine computation could create superior teams, this requires addressing both multiagent coordination complexities and human-machine communication challenges (Grosz and Kraus 1996; Crandall et al. 2018; Tabrez, Luebbers, and Hayes 2020). Humans naturally communicate through action suggestions like "let's move there" without sharing detailed observations or beliefs. For instance, suggesting "We should eat at restaurant X" implicitly communicates beliefs about the restaurant being open, suitable, and reasonably priced, thus encapsulating complex reasoning in a simple action proposal.

Action-based communication is natural for humans but underexplored in autonomous and human-agent teams. We propose an approach that narrows the focus of communication to suggested joint actions. Instead of sharing observations or beliefs, agents communicate recommended actions, using suggestions to convey information about their understanding of the situation.

Our proposed method estimates joint beliefs by maintaining sets of reachable beliefs and inferring other agents' beliefs. The key insight is that an action suggestion implies the agent's belief is within a particular subspace of the belief space, allowing us to prune infeasible beliefs. The agent can then more accurately infer the other agents' beliefs, enabling the construction of an estimated joint belief that can be used with a policy assuming centralized execution. This method requires solving n multiagent POMDPs (MPOMDPs) for an n agent problem, online computation of belief updates for all of the beliefs in the belief set, and a joint policy using the centralized assumptions (solving another MPOMDP). We evaluate this approach on several standard Dec-POMDP

benchmarks and more complex variations, demonstrating performance similar to fully centralized methods when shared actions enable effective belief pruning.

2 Related Work

This work builds upon key areas in decentralized decision making, including communication in Dec-POMDPs, sufficient statistics for planning, and action-based coordination methods. The introduction of communication to Dec-POMDPs has been explored to reduce computational burden and improve coordination. Pynadath and Tambe (2002) examined how communication strategies improve multi-agent teamwork, while Goldman and Zilberstein (2003) investigated optimizing information exchange in these models. The concept of sufficient statistics has played an important role in simplifying Dec-POMDP planning. Oliehoek (2013) introduced probability distributions over joint action-observation histories as sufficient plan-time statistics. Dibangoye et al. (2016) recast Dec-POMDPs as continuous state MDPs using occupancy states, enabling POMDP techniques.

This work is also related to research on action-based coordination in multi-agent systems. Previous work has explored the use of suggested actions as a means of communication between agents, treating these suggestions as observations of the environment (Asmar and Kochenderfer 2022). However, that work assumed that suggested actions were conditioned on the true state of the environment, which becomes less reliable when agents make suggestions based on their beliefs about the state rather than the state itself. A practical example can be found in aircraft collision avoidance systems like TCAS and ACAS X (Asmar and Kochenderfer 2013), which use action advisories (e.g., "do not descend") to restrict other aircraft's actions, effectively coordinating decisions without direct observation sharing by modifying costs of incompatible actions (Asmar 2013).

This work builds on the idea that communication of suggested joint actions can help reduce computational complexity. By using these suggestions to construct distributions over other agents' beliefs, we provide sufficient statistics for histories while allowing all agents to maintain partial observability. The approach reduces the need for full communication while maintaining coordination efficiency, and provides a natural framework for human-agent teaming where action suggestions are an intuitive mode of communication.

3 Background

A partially observable Markov decision process (POMDP) models sequential decision making under uncertainty (Smallwood and Sondik 1973). In a POMDP $(\mathcal{S}, \mathcal{A}, \mathcal{O}, T, O, R, \gamma)$, an agent in state $s \in \mathcal{S}$ chooses an action $a \in \mathcal{A}$, transitions to s' based on $T(s, a, s') = P(s' \mid s, a)$, and receives an observation $o \in \mathcal{O}$ based on $O(s', a, o) = P(o \mid s', a)$. The agent receives a reward $R(s, a) \in \mathbb{R}$ with discount factor $\gamma \in [0, 1)$. One method to solve a POMDP is to infer a belief distribution $b \in \mathcal{B}$ over $\mathcal{S}$ and then solve for a policy π that maps the belief to an action where $\mathcal{B}$ is the set of beliefs over $\mathcal{S}$ (Kochenderfer, Wheeler, and Wray 2022). Executing with this type of policy requires maintaining b through updates after each time step.

A Decentralized POMDP (Dec-POMDP) extends the POMDP framework to multiple cooperative agents with tuple $(\mathcal{I}, \mathcal{S}, \{\mathcal{A}^i\}, \{\mathcal{O}^i\}, T, O, R, \gamma)$. Each agent $i \in \mathcal{I}$ selects an action $a^i \in \mathcal{A}^i$ and receives an observation $o^i \in \mathcal{O}^i$. We use superscripts to represent the agent index and bold variables to represent the joint collection across all agents, e.g., $\mathbf{a} = (a^1, \ldots, a^{|\mathcal{I}|})$. The true state is shared by all agents, while the reward, transition, and observation functions are defined over joint actions and observations (Oliehoek and Amato 2016; Kochenderfer, Wheeler, and Wray 2022).

In a Dec-POMDP with communication (Dec-POMDP-Com), agents can communicate using messages from alphabets $\{\Sigma^i\}$ with communication cost function C_Σ (Pynadath and Tambe 2002; Oliehoek and Amato 2016). In both models, agents make decisions based on their individual action-observation histories (and messages in Dec-POMDP-Com). Solving a finite horizon Dec-POMDP or Dec-POMDP-Com is NEXP-complete (Bernstein et al. 2002; Oliehoek and Amato 2016). If agents can communicate actions and observations perfectly without cost, the model becomes a multiagent POMDP (MPOMDP), which can be solved using POMDP approaches (Pynadath and Tambe 2002; Oliehoek and Amato 2016).

In our approach, we use MPOMDP policies instead of solving the Dec-POMDP directly. Policies for POMDPs and MPOMDPs can be generated offline or computed online during execution. In this work, we integrate our method with policies generated offline and leave the application to online solvers for future work. In particular, we use SARSOP (Kurniawati, Hsu, and Lee 2008) to generate the policies and represent the policy as a set of alpha vectors, but our approach is not limited to SARSOP or alpha vectors and can be applied to policies generated by other methods.

4 Problem Formulation and Notation

We study Dec-POMDP-Coms with discrete spaces, focusing on infinite-horizon problems, though the methods also apply to finite horizons. We consider a system of n agents, each receiving individual observations and maintaining their own belief over the state space. The individual belief of agent i at time t is designated as $b_t^i \in \mathcal{B}^i$. At each time step, after performing an action and receiving an observation, agents communicate by suggesting an action that is optimal using their policy. Therefore, the message alphabet for each agent equals their action space $\Sigma^i = \mathcal{A}^i$. Messages are assumed to be transmitted and received perfectly, without cost, noise, or loss. We denote the message from agent i to agent j at time t as $\sigma_t^{i,j} \in \Sigma^i$.

Each agent also maintains a set of possible beliefs that the other agents might possess. For estimations maintained by agents, we use a *hat* symbol. A superscript of two indices i, j on an estimation refers to agent i's estimate about agent j. For example, $\hat{b}_{t_k}^{i,j}$ represents the k^{th} estimated belief agent i has for agent j's belief at time t. Using similar notation, $\hat{\mathcal{B}}^{i,j}$ is the set of all estimated beliefs agent i has for agent j's belief. When referring to joint beliefs, we use a tilde, $\tilde{b}$.

We assume each agent has access to surrogate policies for other agents, where the surrogate policy $\hat{\pi}^{i,j}$ is agent i's model of agent j's policy. In our experiments where we compute the policies in a centralized manner offline, $\hat{\pi}^{i,j} = \pi^j$. In a slight abuse of notation, we use subscripts to indicate the time step (b_t), counting of the number of variables of a collection (subscript to the time step, b_{t_k}), and for indexing actions and observations (a_ℓ, o_m). For actions and observations, subscripts reference the index within the space, e.g. $a^i_\ell \in \mathcal{A}^i$ is the ℓ^{th} action in $\mathcal{A}^i$.

5 Using Action Suggestions

There are several ways agents can use suggested actions. The simplest option is to ignore the messages and choose actions as if there was no communication, which is equivalent to a Dec-POMDP. Alternatively, agents could designate a leader at each time step and follow the leader's suggested actions, which is sufficient in some environments where one agent's observations provide enough information, as in the Broadcast Channel problem (section 6.3). Another approach is hierarchical action selection, where agents select actions and communicate following a specific communication order. In this scheme, each agent can select an action with knowledge of the previous messages received for that time step. The order of communication becomes important as agents earlier in the process have to make decisions with less information. This approach is similar to other prioritization schemes (Dibangoye et al. 2009).

In our approach, we use suggested actions to infer beliefs. In a cooperative scenario, we assume agents act optimally to maximize shared rewards. Therefore, we assume the suggested action is the one that maximizes the expected sum of discounted rewards based on the agent's belief of the environment. Referencing back to the restaurant example from the introduction, we can infer aspects of the friend's belief from their action suggestion by assuming they are acting optimally and want to maximize the happiness of the group. For instance, if a friend suggests a restaurant, we can infer they believe it is open and suitable for the group's preferences. Each action suggestion thus contains information related to the suggester's belief of the environment, which we can use to infer their belief.

5.1 Inferring the Belief Subspace

We can use the suggested action and the fact that the suggested action is the optimal action from the suggester's perspective to infer the possible beliefs the agent could have. For example, if agent i receives a suggested action a_s from agent j using policy π^j, then we know $b^j \in \mathcal{B}^j_{a_s}$ where $\mathcal{B}^j_{a_s} = \{b \mid \pi^j(b) = a_s, \forall b \in \mathcal{B}^j\}$.

In an alpha vector policy, this would be the subspace of beliefs that are dominated by alpha vectors associated with the suggested action. With a set of alpha vectors Γ representing the policy and a suggested action a_s

$$\mathcal{B}^j_{a_s} = \{\mathbf{b} \mid (\boldsymbol{\alpha}_i - \boldsymbol{\alpha}_j) \cdot \mathbf{b} \geq 0, \forall \boldsymbol{\alpha}_i \in \Gamma_{a_s}, \forall \boldsymbol{\alpha}_j \in \Gamma\} \quad (1)$$

where $\mathbf{b}$ is a belief vector representing probabilities over states and $\Gamma_{a_s} \subseteq \Gamma$ is the set of alpha vectors corresponding to action a_s.

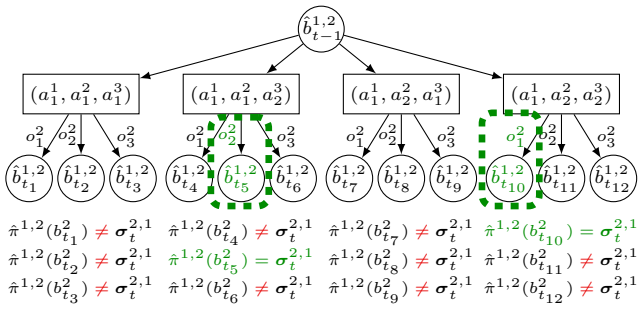

Figure 1: Example of pruning reachable beliefs that do not align with the received message $\boldsymbol{\sigma}^{2,1}_t$. This example has $n = 3$, $|\mathcal{A}^i| = 2$, and $|\mathcal{O}^i| = 3$. The process is from agent 1's perspective, expanding a single belief estimate for agent 2.

5.2 Pruning Beliefs

At each time step, agents update their beliefs based on individual observations and actions performed. From agent i's perspective, there are $|\mathcal{O}^j| \prod_{i \neq j} |\mathcal{A}^k|$ possible beliefs reachable from $\hat{b}^{i,j}_t$ for agent j. The size of this set grows exponentially in time, reaching $\left(|\mathcal{O}^j| \prod_{i \neq j} |\mathcal{A}^k|\right)^\ell$ after ℓ time steps. This exponential growth is one of the primary factors in the NEXP complexity of solving Dec-POMDPs.

To help manage this growth, we can prune infeasible beliefs using the suggested actions. We can rigorously define the belief subspace in which the suggester's belief must lie (eq. (1)) and this subspace is an infinite set of beliefs. While we cannot easily construct the subspace, we can test if a belief is within this subspace by evaluating the policy at that belief.

Without loss of generality, we will discuss this process from the perspective of agent i maintaining a belief estimate for agent j. We start with an initial belief set $\hat{\mathcal{B}}^{i,j}_0 = \{b^j_0\}$, where in our approach, we assume all agents begin with the same initial belief. After performing an action and receiving a local observation, we expand the beliefs considering all possible actions and observations, resulting in $|\hat{\mathcal{B}}^{i,j}_t| = |\hat{\mathcal{B}}^{i,j}_{t-1}||\mathcal{O}^j| \prod_{j \neq i} |\mathcal{A}^j|$ at time t. We then evaluate each belief with the surrogate policy for agent j and prune the beliefs where the optimal action does not match the received message

$$\hat{\mathcal{B}}^{i,j}_t \leftarrow \{b \in \hat{\mathcal{B}}^{i,j}_t \mid \hat{\pi}^{i,j}(b) = \sigma^{j,i}\}. \quad (2)$$

Figure 1 illustrates this pruning process in an example with three agents. If we know the actions performed at the last time step, we only need to consider observations for a single joint action, increasing our estimated belief set by a factor of $|\mathcal{O}^j|$ instead of $|\mathcal{O}^j| \prod_{j \neq i} |\mathcal{A}^j|$. This knowledge significantly reduces the size of the reachable belief set.

After pruning the infeasible beliefs, we can further reduce our set by removing beliefs that are sufficiently close to other beliefs in the set. Zhang, Littman, and Chen (2012) showed that for any two beliefs b and b', if $||b - b'||_1 \leq \delta$, then $|P(o \mid b, a) - P(o \mid b', a)| \leq \delta$. Additionally, Hsu, Lee, and Rong

(2007) proved that the value function of POMDPs satisfies the Lipschitz condition, i.e., $|V(b)-V(b')| \leq \frac{||R||_\infty}{1-\gamma}\delta$ if $||b-b'||_1 \leq \delta$ and Wu et al. (2021) used this bound to combine beliefs in their proposed POMDP algorithm. Building on this previous work, we can reduce the size of our reachable belief set by removing beliefs within the same δ-ball for some parameter δ_ℓ.

5.3 Joint Belief Estimation

Exact Reconstruction: Theoretical Possibility. When inferring other agents' beliefs through our pruning process, we generate both the posterior beliefs and the action-observation histories that led to them. Rather than using the beliefs directly, we can leverage these inferred actions and observations to update an estimated joint belief without approximation error. Using the estimates of the observations and actions, we can update the joint belief using:

$$\hat{\tilde{b}}^i_t(s') \propto O(\hat{\mathbf{o}} \mid \hat{\mathbf{a}}, s') \sum_s T(s' \mid s, \hat{\mathbf{a}}) \hat{\tilde{b}}^i_{t^-}(s) \tag{3}$$

where $\hat{\tilde{b}}^i_{t^-}$ is the joint belief estimation from the previous time step, $\hat{\mathbf{o}} = (\hat{o}^{i,1}, \ldots, o^i, \ldots, \hat{o}^{i,n})$, and $\hat{\mathbf{a}} = (\hat{a}^{i,1}, \ldots, a^i, \ldots, \hat{a}^{i,n})$.

While theoretically achievable, this exact approach presents computational challenges. The memory requirements increase substantially, as we must store $|\hat{\mathcal{B}}^{i,j}|$ beliefs for each agent plus $\prod_{j\neq i} |\hat{\mathcal{B}}^{i,j}|$ joint belief combinations, where the joint combinations can grow exponentially with the number of agents. The computational overhead similarly increases from $\sum_{j\neq i} |\hat{\mathcal{B}}^{i,j}|$ individual belief updates to $\sum_{j\neq i} |\hat{\mathcal{B}}^{i,j}| + \prod_{j\neq i} |\hat{\mathcal{B}}^{i,j}|$ total updates per time step. These computational challenges motivate consideration of approximate methods that capture the essential benefits while maintaining practical feasibility.

Practical Approximation: Conflation. After inferring beliefs of other agents, we can combine the inferred beliefs with the receiving agent's own belief to estimate a joint belief. Various methods exist for combining probability distributions (Genest and Zidek 1986). While mixture distributions are a straightforward approach, they require assigning and justifying potentially unequal weights.

An alternative method is conflation (Hill 2011):

$$\hat{\tilde{b}}^i(s) = \frac{b^i(s) \prod_{j\neq i} \hat{b}^{i,j}(s)}{\sum_{s'\in\mathcal{S}} b^i(s') \prod_{j\neq i} \hat{b}^{i,j}(s')}. \tag{4}$$

Unlike many combination methods, conflation possesses several desirable properties. Notably, conflation is not idempotent (i.e., $T(P, \ldots, P) \neq P$), which is beneficial when consolidating results from independent observations. As noted by Hill (2011), conflation minimizes the loss of Shannon information when consolidating multiple distributions, automatically prioritizes more confident beliefs, and requires no ad hoc weight assignments, making it a principled choice for belief combination. From a practical standpoint, conflation operates directly on available belief estimates without requiring the increased memory or computation of exact reconstruction, producing a single joint belief estimate that can be immediately used for action selection.

5.4 Belief and Action Selection

Using the suggested joint actions to prune the reachable beliefs and removing similar beliefs is effective in reducing the size of our estimated belief set. However, the belief subspace dominated by the suggested action can be composed of disjoint subsets, and pruning does not guarantee the reduction to a single belief

To form our set of estimated joint beliefs, we consider all possible estimated beliefs of other agents, resulting in at most $\prod_{j\neq i} |\hat{\mathcal{B}}^{i,j}|$ combinations. In practice, when the information implied by an action results in a small belief subspace, we often do not have many beliefs to consider. We demonstrate this in our experiments by sharing the alpha vector index instead of the action, thus sharing a single subspace region that is dominated by the optimal action. However, in cases where an action does not imply much information, the pruning is less effective, and we must determine how to use the set of estimated joint beliefs to select an action.

Rather than combining joint belief estimations (e.g., through centroids or averages), we propose a heuristic using counts for each unique belief. Counts increment when a similar belief is pruned, indicating how frequently each belief is reached through different paths. The estimated joint belief with the highest count is selected, breaking ties randomly to avoid bias. We then use this selected estimated joint belief to choose an action using a policy based on the assumption of shared observations and actions (a centralized joint policy).

This approach of maintaining belief counts and selecting based on weights balances computational efficiency and decision quality in our experiments, though effectiveness varies with problem characteristics. Future improvements could include optimal belief and action selection strategies for various problem scenarios like implementing history-based selection for more nuanced belief choice and using regret minimization across all estimated joint beliefs (Auer, Cesa-Bianchi, and Fischer 2002) for an action selection strategy.

5.5 Multiagent Control via Action Suggestions (MCAS) Algorithm

Our approach begins by solving $n+1$ MPOMDPs. For each agent $i \in 1, \ldots, n$, we solve an MPOMDP where agent i receives individual observations (observation space $\mathcal{O}^i$) but has control over all agents (action space $\mathcal{A}^1 \times \mathcal{A}^2 \times \cdots \times \mathcal{A}^n$). This results in policies $\pi^1, \ldots, \pi^n$ which have joint actions as the action space. We also require a policy $\tilde{\pi}$ that assumes joint observations and uses a joint belief, which can be generated by any suitable solver (online or offline).

The MCAS algorithm (algorithm 1) operates from the perspective of agent 1, arbitrarily designated as the coordinating agent. This approach builds upon leader-based coordination but differs by integrating information from all agents. Unlike hierarchical action selection, it does not rely on a fixed communication order; instead, it treats all agents' suggestions

Algorithm 1: Multiagent Control via Action Suggestions

Given: n ▷ Number of agents
$\mathcal{P}^1, \ldots, \mathcal{P}^n$ ▷ Agents' MPOMDPs
$\pi^1, \ldots, \pi^n$ ▷ Agents' policies
$\tilde{\mathcal{P}}, \tilde{\pi}$ ▷ Joint MPOMDP and policy
$\delta_{\text{joint}}, \delta_{\text{single}}$ ▷ Similarity thresholds
$\overline{B}_{\max}$ ▷ Maximum number of estimated beliefs

1 Initialize belief b^1 for agent 1
2 Initialize surrogate belief sets $(\hat{\mathcal{B}}^{1,j}, w^{1,j}) = \{(b_0^j, 1.0)\}$ for $j = 2, \ldots, n$
3 **while** *not done* **do**
4 Receive messages $\sigma^{j,1}$ from agents $j = 2, \ldots, n$
5 **for** $j \leftarrow 2$ ***to*** n **do**
6 $\hat{\mathcal{B}}^{1,j} \leftarrow$ PRUNEBELIEFS$(\pi^j, \hat{\mathcal{B}}^{1,j}, \boldsymbol{\sigma^{j,1}})$
7 $\hat{\mathcal{B}}^{1,j} \leftarrow$ REDUCETOMAXLIMIT$(\hat{\mathcal{B}}^{1,j}, \overline{B}_{\max})$
8 $\hat{\tilde{b}} \leftarrow$ SELECTJOINTBELIEF$(\{(\hat{\mathcal{B}}^{1,j}, w^{1,j})\}_{j=2}^n, b^1, \delta_{\text{joint}})$
9 $\tilde{\mathbf{a}} \leftarrow \tilde{\pi}(\hat{\tilde{b}})$
10 Broadcast $\tilde{\mathbf{a}}$ to all agents
11 Execute $\tilde{\mathbf{a}}[1]$ and observe o^1 ▷ Agent 1's action
12 $b^1 \leftarrow$ UPDATE$(\mathcal{P}^1, b^1, \tilde{\mathbf{a}}, o^1)$
13 **for** $j \leftarrow 2$ ***to*** n **do**
14 $\hat{\mathcal{B}}^{1,j}, w^{1,j} \leftarrow$ UPDATEESTBELIEFS$(j, \mathcal{P}^j, \hat{\mathcal{B}}^{1,j}, w^{1,j}, \tilde{\mathbf{a}}, \delta_{\text{single}})$
15 **Function** PRUNEBELIEFS*($\pi, \hat{\mathcal{B}}, \sigma$)*
16 **return** $\{b \in \hat{\mathcal{B}} \mid \pi(b) = \sigma\}$

equally to infer a comprehensive joint belief. The coordinating agent receives action suggestions from others, estimates a joint belief, and suggests a final joint action based on the centralized policy, which all agents then follow. This coordination role is not strictly necessary, but simplifies the discussion and presentation of our results.

While presented with a designated coordinator for simplicity, MCAS can operate in a fully decentralized manner once action suggestions have been shared. Each agent can independently maintain its estimated joint belief and select actions according to their MPOMDP policy, often achieving performance comparable to the coordinator-based method when effective pruning results in small belief sets. Alternative coordination strategies that leverage the shared action suggestions include random delays where agents execute the most recently received suggestion, belief set considerations where agents prioritize decisions from those with the smallest belief set size, and voting schemes for collective action determination.

The algorithm can be implemented using either actions or alpha vector indices as messages. When using alpha vector indices (which we call MCAS-α), each agent communicates which alpha vector dominates their belief rather than just the resulting action. Since multiple alpha vectors can correspond to the same action but define distinct regions of the belief space, this provides a more precise subspace for pruning, leading to more effective belief state estimation. While

Algorithm 2: Update Estimated Beliefs

1 **Function** UPDATEESTBELIEFS*($j, \mathcal{P}, \hat{\mathcal{B}}, w, \mathbf{a}, \delta_{single}$)*
2 $\hat{\mathcal{B}}_{\text{new}} \leftarrow \emptyset$, $w_{\text{new}} \leftarrow \emptyset$
3 **for** $i \leftarrow 1$ ***to*** $|\hat{\mathcal{B}}|$ **do**
4 **for** $o \in \mathcal{O}^j$ **do**
5 $b' \leftarrow$ UPDATE$(\mathcal{P}, \hat{\mathcal{B}}[i], \mathbf{a}, o)$
6 $w' \leftarrow w[i] + 1$ ▷ Count-based approach
7 **if** $\forall b'' \in \hat{\mathcal{B}}_{new} : \|b' - b''\|_1 \geq \delta_{single}$ **then**
8 $\hat{\mathcal{B}}_{\text{new}} \leftarrow \hat{\mathcal{B}}_{\text{new}} \cup \{b'\}$
9 $w_{\text{new}} \leftarrow w_{\text{new}} \cup \{w'\}$
10 **else**
11 $k \leftarrow \text{argmin}_{b'' \in \hat{\mathcal{B}}_{\text{new}}} \|b' - b''\|_1$
12 $w_{\text{new}}[k] \leftarrow w_{\text{new}}[k] + w'$
13 **return** $\hat{\mathcal{B}}_{\text{new}}, w_{\text{new}}$

sharing alpha vector indices requires agents to have access to identical policies and greater computational coordination, MCAS-α demonstrates the effectiveness of our belief pruning approach when additional information is available.

The ESTIMATEJOINTBELIEF function (line 4 of algorithm 3) can be implemented using various methods such as weighted averaging or conflation (section 5.3). If maintaining estimated joint beliefs from inferred observations, the UPDATEESTBELIEFS function (algorithm 2) would need to return associated observations, and the belief combination process would involve updates for all possible observation combinations, potentially improving the accuracy of the joint belief estimate at the cost of increased computational complexity.

Pruning based on the suggested action is effective in practice; however, the number of reachable beliefs can still grow exponentially in the worst case. The REDUCETOMAXLIMIT function (line 7) limits the size of the belief set to $\overline{B}_{\max}$. Our implementation computes the $\mathcal{L}_1$ norm between all belief pairs, sorts these distances, and iteratively removes the lower-weighted belief of the closest pair, adding its weight to the remaining belief, until reaching $\overline{B}_{\max}$.

Algorithm 3: Select Joint Beliefs

1 **Function** SELECTJOINTBELIEF*($\{(\hat{\mathcal{B}}^j, w^j)\}_{j=2}^n, b^1, \delta_{joint}$)*
2 $\mathcal{B}_{\text{combined}} \leftarrow \emptyset$, $w_{\text{combined}} \leftarrow \emptyset$
3 **for** $(\hat{b}^2, \ldots, \hat{b}^n) \in \hat{\mathcal{B}}^2 \times \cdots \times \hat{\mathcal{B}}^n$ **do**
4 $b^c \leftarrow$ ESTIMATEJOINTBELIEF$(b^1, \hat{b}^2, \ldots, \hat{b}^n)$
5 $w^c \leftarrow \sum_{j=2}^n w^j[\text{index}(\hat{b}^j)]$ ▷ Count-based approach
6 **if** $\forall b' \in \mathcal{B}_{combined} : \|b^c - b'\|_1 \geq \delta_{joint}$ **then**
7 $\mathcal{B}_{\text{combined}} \leftarrow \mathcal{B}_{\text{combined}} \cup \{b^c\}$
8 $w_{\text{combined}} \leftarrow w_{\text{combined}} \cup \{w^c\}$
9 **else**
10 $k \leftarrow \text{argmin}_{b' \in \mathcal{B}_{\text{combined}}} \|b^c - b'\|_1$
11 $w_{\text{combined}}[k] \leftarrow w_{\text{combined}}[k] + w_c$
12 $w_{\text{normalized}} \leftarrow w_{\text{combined}} / ||w_{\text{combined}}||_1$
13 $k \leftarrow \arg\max_i w_{\text{normalized}}[i]$
14 **return** $\mathcal{B}_{\text{combined}}[k]$

6 Experiments

All experiments were implemented and executed using Julia (Bezanson et al. 2017) with the POMDPs.jl framework (Egorov et al. 2017). Problem implementations were based primarily on originating papers, with additional references to the Multiagent Systems Planning Page (Spaan et al. 2014) and the Dec-POMDP page (Amato 2024) to ensure consistency with previous work. For context, we include the best-reported results from Dec-POMDP solvers when available, noting that our approach's use of communication makes these comparisons informative but not equivalent.

6.1 Benchmark Problems

We tested MCAS on several Dec-POMDP benchmarks: Decentralized Tiger (Nair et al. 2003), Broadcast Channel (Hansen, Bernstein, and Zilberstein 2004), Meeting in a 2×2 Grid (Bernstein, Hansen, and Zilberstein 2005), Meeting in a 3×3 Grid (Amato, Dibangoye, and Zilberstein 2009), Cooperative Box Pushing (Seuken and Zilberstein 2007), Wireless Networking (Pajarinen and Peltonen 2011a), and Mars Rover (Amato and Zilberstein 2009). For detailed problem descriptions and implementations, we refer readers to the original papers and our accompanying repository.

The original problems were designed without considering communication. In our experiments, we found that when we allowed one agent, using only its individual observations, to control all agents, it often achieved performance similar to a full MPOMDP (with shared observations and actions). To better demonstrate the value of integrating different beliefs, we introduced modifications to increase difficulty and emphasize the importance of different agent observations.

We use qualifiers to denote problem modifications from the original implementation in our results:

- *UI*: Uniform initial belief distribution.
- *WP*: Added penalties for wall collisions or message sending.
- *DP*: Modified Broadcast buffer fill probabilities for three agents (0.2, 0.4, 0.4).
- *SS*: Meet 2×2, changed starting positions from corners to same row or column.
- *AG*: Meet 3×3, rewarded agents for meeting at any grid location, not just two corners.
- *SO*: Box Push with stochastic observations ($50\,\%$ accuracy).
- 5*G*: Additional Mars Rover sampling site (accessible from the original top-right location).
- *Meet* 27: Expanded version of Meet 2×2 with 27 grid locations. Observation space expanded to also include *no walls* and *both walls*.

6.2 Baseline Methods and Implementation Details

We compared MCAS against the following baselines:

- *MMDP*: Multiagent MDP assuming full observability.
- *MPOMDP*: Multiagent POMDP with centralized control.
- *MPOMDP-C*: MPOMDP policy with joint beliefs generated by conflating the true individual agent beliefs.
- *MCAS*$-\alpha$: MCAS using alpha vector indices. Used conflation with similarity parameters δ_{single} and δ_{joint} set to 10^{-5}.
- *MCAS*: As described in section 5.5, using same parameters as MCAS$-\alpha$ with maximum estimated beliefs $\overline{B}_{\max} = 200$.
- *MPOMDP-I*: Single agent controls all agents, using only its individual observations.
- *Independent*: Agents execute individual policies (assuming control of other agents), ignoring messages.
- *Dec-POMDP*: Best reported results from literature (experiments not conducted by us).

Hyperparameter selection prioritized computational tractability: $\overline{B}_{\max} = 200$ ensures reasonable simulation times while preserving belief diversity, and similarity thresholds $\delta_{\text{single}}, \delta_{\text{joint}} = 10^{-5}$ were set through empirical testing to merge functionally equivalent beliefs without impacting solution quality.

All POMDP policies were computed using SARSOP (Kurniawati, Hsu, and Lee 2008). Experiments for POMDP-based methods were conducted on a MacBook Pro with an Apple M1 Max processor and 32 GB of memory, running each scenario 2000 times. Results for these methods are reported with $95\,\%$ confidence intervals. MMDP results represent the converged policy value and are reported without confidence intervals. Most problems used 50 time steps with a discount factor of 0.9, while the Wireless Network problem used 450 steps and a 0.99 discount factor.

6.3 Results

The results in section 6.3 offer several insights into the performance of MCAS across various Dec-POMDP benchmarks. MPOMDP-C has similar performance to MPOMDP across all problems, suggesting that using conflation to combine beliefs is an effective approach, particularly in these scenarios where observations are independent.

MCAS$-\alpha$ closely matches MPOMDP-C results through effective use of alpha vector indices for belief subspace refinement. While MCAS performs marginally worse using only shared actions for pruning, it still maintains effective joint belief estimates, achieving comparable results with less refined belief subspaces.

MCAS effectively pruned beliefs, with $|\hat{\mathcal{B}}^{1,j}|$ exceeding the maximum set size limit in only two problems: $3.2\,\%$ of Meet 27 and $87.8\,\%$ of Box Push-SO runs. The largest performance decreases for MCAS compared to MCAS$-\alpha$ occurred in Meet 27, Box Push-SO, Mars Rover-UI, and Mars Rover-5G-UI. This difference is due to MCAS$-\alpha$'s more effective pruning. Section 6.3 shows the maximum estimated belief set sizes for problems with a noticeable increase for MCAS. Despite larger set sizes, MCAS still achieved high performance approaching that of MCAS$-\alpha$. We anticipate this gap will decrease with improved belief selection.

Comparing MPOMDP and MPOMDP-I results reveals that in problems like Broadcast, Meeting, and Wireless, sharing observations provides no advantage over using only individual observations. While this suggests opportunities for

Problem	*Qualifiers*	*# Agents*	*Solution Method*							
			MMDP	*MPOMDP*	*MPOMDP-C*	*MCAS−α*	*MCAS*	*MPOMDP-I*	*Dec-POMDP*	*Independent*
Dec-Tiger	—	2	200.0	59.5 ± 0.9	59.5 ± 0.9	58.5 ± 0.9	58.5 ± 0.8	34.3 ± 1.7	13.5[1]	-68.1 ± 3.5
	—	3	300.0	108.5 ± 1.0	108.5 ± 1.0	108.5 ± 1.0	108.5 ± 1.0	82.1 ± 1.5	—	-95.5 ± 4.1
	—	4	400.0	153.0 ± 0.7	153.0 ± 0.7	152.8 ± 0.7	152.8 ± 0.7	121.3 ± 1.5	—	-121.4 ± 4.4
Broadcast	—	2	9.4	9.4 ± 0.0	9.4 ± 0.0	9.4 ± 0.0	9.4 ± 0.0	9.4 ± 0.0	9.3[2]	7.6 ± 0.1
	DP, WP	3	6.7	6.6 ± 0.0	6.6 ± 0.0	6.6 ± 0.0	6.6 ± 0.0	5.5 ± 0.0	—	-0.6 ± 0.1
Meet 2×2	—	2	8.0	6.4 ± 0.1	6.1 ± 0.2	6.1 ± 0.2	6.1 ± 0.2	5.9 ± 0.1	6.1*[3]	1.7 ± 0.1
	SS	2	8.4	6.9 ± 0.1	6.8 ± 0.1	6.8 ± 0.1	6.8 ± 0.1	6.8 ± 0.1	7.0*[2]	2.3 ± 0.1
	UI, WP	2	8.7	5.8 ± 0.2	5.3 ± 0.2	5.3 ± 0.2	5.3 ± 0.2	4.5 ± 0.2	—	3.5 ± 0.2
Meet 3×3	—	2	5.9	5.8 ± 0.1	5.8 ± 0.1	5.8 ± 0.1	5.7 ± 0.1	3.6 ± 0.1	5.8[4]	3.7 ± 0.1
	AG, UI, WP	2	8.1	7.3 ± 0.1	7.3 ± 0.1	7.3 ± 0.1	7.1 ± 0.1	3.5 ± 0.1	—	2.8 ± 0.1
	AG, UI, WP	3	7.2	6.4 ± 0.1	6.4 ± 0.1	6.4 ± 0.1	6.2 ± 0.1	1.0 ± 0.1	—	1.7 ± 0.1
Meet 27	UI, WP	2	6.3	2.2 ± 0.1	2.1 ± 0.1	2.0 ± 0.1	1.6 ± 0.1	0.6 ± 0.1	—	0.6 ± 0.1
Box Push	—	2	240.1	222.9 ± 2.2	223.4 ± 2.1	223.4 ± 2.1	223.0 ± 2.2	199.6 ± 2.6	224.4[4]	163.6 ± 3.4
	SO	2	240.1	204.3 ± 2.5	203.4 ± 2.5	203.2 ± 2.5	199.8 ± 2.5	178.8 ± 2.7	—	138.5 ± 3.8
Wireless	—	2	-143.6	-152.8 ± 2.3	-152.8 ± 2.3	-152.8 ± 2.3	-153.0 ± 2.4	-152.8 ± 2.3	−167.1†[2]	-219.8 ± 3.9
	WP	2	-154.5	-165.8 ± 2.4	-166.5 ± 2.4	-166.5 ± 2.4	-166.5 ± 2.4	-172.4 ± 2.3	—	-240.2 ± 4.1
Mars Rover	—	2	29.2	29.0 ± 0.1	29.0 ± 0.1	29.0 ± 0.1	29.0 ± 0.1	24.4 ± 0.3	26.9[4]	26.0 ± 0.2
	UI	2	24.9	23.9 ± 0.1	23.9 ± 0.1	23.9 ± 0.1	19.8 ± 0.2	16.4 ± 0.2	—	15.3 ± 0.2
	UI	3	26.2	25.2 ± 0.1	25.2 ± 0.1	25.2 ± 0.1	23.8 ± 0.2	19.7 ± 0.1	—	16.6 ± 0.1
	5G, UI	2	21.4	20.7 ± 0.1	20.7 ± 0.1	20.7 ± 0.8	18.0 ± 0.2	14.8 ± 0.1	—	13.1 ± 0.2

[1] Pajarinen and Peltonen (2011b); [2] MacDermed and Isbell (2013); [3] Amato, Bonet, and Zilberstein (2010); [4] Dibangoye, Buffet, and Charpillet (2014)
* The papers reporting the best scores for Meeting 2×2 do not discuss the initial state. We associated the best-reported result with an initial condition based on the MPOMDP solutions (which is an upper bound on Dec-POMDP results). Other reported scores: Pajarinen and Peltonen (2011b): 6.9, Amato and Zilberstein (2009): 5.6.
† Dibangoye, Buffet, and Charpillet (2014) reported a value of -140.4, but we were unable to verify the implementations details. The reported value -140.4 is better than the performance of the MPOMDP on our implementation which implies there is a difference in implementation. Previously highest reported score prior to MacDermed and Isbell (2013) was -175.4 by Pajarinen and Peltonen (2011b).

Table 1: Average cumulative discounted reward (with $95\,\%$ confidence intervals) for various Dec-POMDP problems.

simplification in certain multiagent problems, determining optimal leadership roles requires further study.

A challenge in conducting these experiments was the generation of MPOMDP policies. While this process is more tractable compared to Dec-POMDP solvers, the complexity of solving MPOMDPs grows exponentially with the number of agents. The online execution of MCAS, however, did not pose a major computational burden, with all simulations conducted on a standard laptop. This balance between offline policy generation and lightweight online execution makes MCAS promising for practical multiagent problems.

Problem	*Qualifiers*	*# Agents*	*Solution Method*	
			MCAS−α	*MCAS*
Meet 3×3	—	2	1.0 ± 0.0	2.5 ± 0.0
Meet 27	UI, WP	2	1.5 ± 0.0	16.8 ± 1.6
Box Push	SO	2	4.8 ± 0.1	192.1 ± 1.2
Wireless	—	2	1.0 ± 0.0	18.0 ± 0.9
Mars Rover	UI	2	1.0 ± 0.0	2.0 ± 0.0
	5G, UI	2	1.0 ± 0.0	3.0 ± 0.0

Table 2: Maximum size of $\hat{\mathcal{B}}^{1,j}$ per simulation.

7 Conclusions and Future Work

This paper introduced the Multiagent Control via Action Suggestions (MCAS) algorithm for coordinating agents in partially observable environments. By leveraging suggested actions as a form of communication, MCAS demonstrated performance comparable to centralized methods across Dec-POMDP benchmarks, while maintaining computational efficiency. The algorithm prunes the reachable belief space, enabling inference of other agents' beliefs and estimation of a joint belief.

Several research directions remain open. A key area is theoretical analysis, including convergence of belief estimation, bounds relative to centralized methods, and information-theoretic properties of action-based communication. Another area is relaxing assumptions, including surrogate-policy mismatch and noncompliance with suggestions. Extending MCAS to online solvers (e.g., AdaOPS (Wu et al. 2021), BetaZero (Moss et al. 2024)) could scale to larger problems, but requires real-time belief-subspace estimation and handling stochastic online policies.

Our results indicate that action-based communication can be a powerful tool for multiagent coordination, potentially bridging the gap between decentralized and centralized approaches. This approach lays the groundwork for more intuitive coordination in human-agent teams, opening possibilities for mixed-initiative planning and decision making in real-world applications.

References

Albrecht, S. V.; and Stone, P. 2018. Autonomous Agents Modelling Other Agents: A Comprehensive Survey and Open Problems. *Artificial Intelligence*, 258: 66–95.

Amato, C. 2024. Decentralized POMDPs. http://rbr.cs.umass.edu/camato/decpomdp. Accessed: 2024-07.

Amato, C.; Bonet, B.; and Zilberstein, S. 2010. Finite-State Controllers Based on Mealy Machines for Centralized and Decentralized POMDPs. In *AAAI Conference on Artificial Intelligence*.

Amato, C.; Dibangoye, J. S.; and Zilberstein, S. 2009. Incremental Policy Generation for Finite-Horizon Dec-POMDPs. In *International Conference on Automated Planning and Scheduling (ICAPS)*.

Amato, C.; and Zilberstein, S. 2009. Achieving Goals in Decentralized POMDPs. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

Asmar, D. M. 2013. *Airborne Collision Avoidance in Mixed Equipage Environments*. Master's thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics.

Asmar, D. M.; and Kochenderfer, M. J. 2013. Optimized Airborne Collision Avoidance in Mixed Equipage Environments. Project Report ATC-408, MIT Lincoln Laboratory, Lexington, MA.

Asmar, D. M.; and Kochenderfer, M. J. 2022. Collaborative Decision Making Using Action Suggestions. In *Advances in Neural Information Processing Systems (NeurIPS)*.

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

Bernstein, D. S.; Givan, R.; Immerman, N.; and Zilberstein, S. 2002. The Complexity of Decentralized Control of Markov Decision Processes. *Mathematics of Operations Research*, 27(4): 819–840.

Bernstein, D. S.; Hansen, E. A.; and Zilberstein, S. 2005. Bounded Policy Iteration for Decentralized POMDPs. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Bezanson, J.; Edelman, A.; Karpinski, S.; and Shah, V. B. 2017. Julia: A Fresh Approach to Numerical Computing. *SIAM Review*, 59(1): 65–98.

Crandall, J. W.; Oudah, M.; Tennom; Ishowo-Oloko, F.; Abdallah, S.; Bonnefon, J.-F.; Cebrian, M.; Shariff, A.; Goodrich, M. A.; and Rahwan, I. 2018. Cooperating with machines. *Nature Communications*, 9(1): 233.

Dafoe, A.; Hughes, E.; Bachrach, Y.; Collins, T.; McKee, K. R.; Leibo, J. Z.; Larson, K.; and Graepel, T. 2020. Open Problems in Cooperative AI. arXiv:2012.08630.

Dibangoye, J. S.; Amato, C.; Buffet, O.; and Charpillet, F. 2016. Optimally solving Dec-POMDPs as continuous-state MDPs. *Journal of Artificial Intelligence Research (JAIR)*, 55(1): 443–497.

Dibangoye, J. S.; Buffet, O.; and Charpillet, F. 2014. Error-Bounded Approximations for Infinite-Horizon Discounted Decentralized POMDPs. In *Machine Learning and Knowledge Discovery in Databases*.

Dibangoye, J. S.; Shani, G.; Chaib-draa, B.; and Mouaddib, A. I. 2009. Topological order planner for POMDPs. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Egorov, M.; Sunberg, Z. N.; Balaban, E.; Wheeler, T. A.; Gupta, J. K.; and Kochenderfer, M. J. 2017. POMDPs.jl: A Framework for Sequential Decision Making under Uncertainty. *Journal of Machine Learning Research (JMLR)*, 18(26): 1–5.

Genest, C.; and Zidek, J. V. 1986. Combining Probability Distributions: A Critique and an Annotated Bibliography. *Statistical Science*, 1(1): 114–135.

Goldman, C. V.; and Zilberstein, S. 2003. Optimizing Information Exchange in Cooperative Multi-agent Systems. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.

Goldman, C. V.; and Zilberstein, S. 2004. Decentralized Control of Cooperative Systems: Categorization and Complexity Analysis. *Journal of Artificial Intelligence Research (JAIR)*, 22(1): 143–174.

Grosz, B. J.; and Kraus, S. 1996. Collaborative Plans for Complex Group Action. *Artificial Intelligence*, 86(2): 269–357.

Hansen, E. A.; Bernstein, D. S.; and Zilberstein, S. 2004. Dynamic Programming for Partially Observable Stochastic Games. In *AAAI Conference on Artificial Intelligence*.

Hill, T. 2011. Conflations of probability distributions. *Transactions of the American Mathematical Society*, 363(6): 3351–3372.

Hsu, D.; Lee, W. S.; and Rong, N. 2007. What makes some POMDP problems easy to approximate? In *Advances in Neural Information Processing Systems (NeurIPS)*.

Johnson, M.; and Vera, A. 2019. No AI Is an Island: The Case for Teaming Intelligence. *AI Magazine*, 40(1): 16–28.

Kochenderfer, M. J.; Wheeler, T. A.; and Wray, K. H. 2022. *Algorithms for Decision Making*. MIT Press.

Kurniawati, H.; Hsu, D.; and Lee, W. S. 2008. SARSOP: Efficient Point-Based POMDP Planning by Approximating Optimally Reachable Belief Spaces. In *Robotics: Science and Systems (RSS)*.

MacDermed, L. C.; and Isbell, C. L. 2013. Point Based Value Iteration with Optimal Belief Compression for Dec-POMDPs. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Moss, R. J.; Corso, A.; Caers, J.; and Kochenderfer, M. J. 2024. BetaZero: Belief-State Planning for Long-Horizon POMDPs using Learned Approximations. In *Reinforcement Learning Conference (RLC)*.

Nair, R.; Tambe, M.; Yokoo, M.; Pynadath, D. V.; and Marsella, S. 2003. Taming Decentralized POMDPs: Towards Efficient Policy Computation for Multiagent Settings. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Oliehoek, F. A. 2013. Sufficient Plan-Time Statistics for Decentralized POMDPs. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

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

Pajarinen, J.; and Peltonen, J. 2011a. Efficient Planning for Factored Infinite-Horizon Dec-POMDPs. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Pajarinen, J.; and Peltonen, J. 2011b. Periodic Finite State Controllers for Efficient POMDP and Dec-POMDP Planning. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Pynadath, D. V.; and Tambe, M. 2002. The Communicative Multiagent Team Decision Problem: Analyzing Teamwork Theories and Models. *Journal of Artificial Intelligence Research (JAIR)*, 16: 389–423.

Seuken, S.; and Zilberstein, S. 2007. Improved Memory-Bounded Dynamic Programming for Decentralized POMDPs. In *Conference on Uncertainty in Artificial Intelligence (UAI)*.

Smallwood, R. D.; and Sondik, E. J. 1973. The Optimal Control of Partially Observable Markov Processes over a Finite Horizon. *Operations Research*, 21: 1071–1088.

Spaan, M.; Amato, C.; Oliehoek, F.; and Witwicki, S. 2014. Multi-Agent Systems Planning. http://masplan.org/. Accessed: 2024-07.

Tabrez, A.; Luebbers, M. B.; and Hayes, B. 2020. A Survey of Mental Modeling Techniques in Human–Robot Teaming. *Current Robotics Reports*, 1(4): 259–267.

Wu, C.; Yang, G.; Zhang, Z.; Yu, Y.; Li, D.; Liu, W.; and HAO, J. 2021. Adaptive Online Packing-guided Search for POMDPs. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Zhang, Z.; Littman, M.; and Chen, X. 2012. Covering Number as a Complexity Measure for POMDP planning and learning. In *AAAI Conference on Artificial Intelligence*.