A Semi-Decentralized Approach to Multiagent Control

Anonymous Author(s)
Submission Id: 929

ABSTRACT

We introduce an expressive framework and algorithms for the semi-decentralized control of cooperative agents in environments with communication uncertainty. Whereas semi-Markov control admits a distribution over time for agent actions, semi-Markov communication, or what we refer to as semi-decentralization, gives a distribution over time for what actions and observations agents can store in their histories. We extend semi-decentralization to the partially observable Markov decision process (POMDP). The resulting SDec-POMDP unifies decentralized and multiagent POMDPs and several existing explicit communication mechanisms. We present recursive small-step semi-decentralized A* (RS-SDA*), an exact algorithm for generating optimal SDec-POMDP policies. RS-SDA* is evaluated on semi-decentralized versions of several standard benchmarks and a maritime medical evacuation scenario. This paper provides a well-defined theoretical foundation for exploring many classes of multiagent communication problems through the lens of semi-decentralization.

KEYWORDS

Semi-decentralization, Semi-Markov process, Decentralized partially observable Markov decision process, Multiagent communication, Admissible heuristic search

ACM Reference Format:
Anonymous Author(s). 2026. A Semi-Decentralized Approach to Multiagent Control. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 14 pages.

1 INTRODUCTION

Many complex real-world problems require the coordination of multiple cooperative agents to solve, but feature limited opportunities for information exchange. The decentralized partially observable Markov decision process (Dec-POMDP) formalizes multi-agent planning and control in settings where explicit communication is impossible [1, 4, 5]. Several model variants take advantage of existing, if limited information structures and extend the Dec-POMDP to explicitly incorporate costly [10], delayed [20, 21], lossy [33], or intermittent [34] communication.

In this paper we pursue a general framework that unifies several multiagent communication mechanisms. We are especially interested in problems whose communication dynamics are conditioned with some probability on the underlying state or taken actions. An example domain is area-wide Global Positioning System (GPS) denial via jamming [11]. As seen in the maritime medical evacuation scenario depicted in Figure 1, agents need to coordinate joint tasks in environments with degraded, denied, or disrupted communication. Agents must reason about which actions to take in light of available communication, the influence actions taken have on future communication, and future communication's influence on future actions. In decentralized systems, information may be action-gated in costly communication, constrained by channel capacity in lossy communication, temporally offset in intermittent or delayed communication, and either deterministic or stochastic in nature, while semi-decentralized systems generalize all of these through probabilistic information flow.

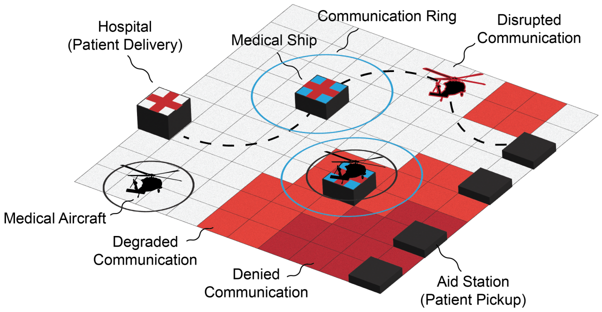

Figure 1: A semi-decentralized multiagent medical evacuation scenario with probabilistic restrictions on communication. Aircraft and watercraft must coordinate under communication constraints to move patients from aid stations to hospitals.

We begin by formally defining the *semi-decentralization property*. The key insight is that **semi-Markov systems for resolving agent control may be co-opted in semi-decentralized systems for agent communication.** Semi-decentralized systems may simultaneously contain a subset of agents that are necessarily decentralized and a subset of agents that are permissibly centralized. We introduce the semi-decentralized POMDP (SDec-POMDP), which unifies several existing multiagent models and explicit communication mechanisms. The SDec-POMDP reveals underlying mechanisms for memory propagation, selector functions, which we toggle here but are otherwise inherently present and constant in existing models. We then detail an exact optimal planning algorithm for solving SDec-POMDPs and apply to a set of new semi-decentralized benchmarks and a maritime medical evacuation application.

Contributions

- We formulate the semi-decentralization property by extending semi-Markov control concepts to communication. Semi-decentralization is then applied to multiagent POMDPs, forming the SDec-POMDP.

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus.

- We prove that the SDec-POMDP unifies the Dec-POMDP, the MPOMDP, k-steps delayed communication, and the Dec-POMDP-COM.
- We introduce Recursive Small-Step Semi-Decentralized A* (RS-SDA*), an exact algorithm for solving semi-decentralized problems. We evaluate performance in semi-decentralized variants of four Dec-POMDP benchmarks, then apply to a complex medical evacuation scenario with joint tasks.

Ultimately, we describe and validate the mechanisms of a novel yet foundational property and model for multiagent decision making in probabilistic communication environments.

2 RELATED WORK

Multiagent Models Centralized planning can occur when agents freely communicate in reliable networks with no latency or when individual agents possess full system observability. This allows a single planner to select joint actions over the global state [22]. A partially observable centralized multiagent system is represented by a multiagent partially observable Markov decision process (MPOMDP). MPOMDP planners benefit from conditioning joint actions on joint-observations but scale poorly and are susceptible to communication failures. MPOMDP therefore have limited application to many challenging real-world problems. By contrast, cooperative agents in decentralized systems can neither communicate explicitly nor observe the entire system state, and therefore must select own actions in accordance with local observations [15]. The decentralized POMDP (Dec-POMDP) and MPOMDP share an underlying model but possess distinct policies and histories. Decentralized models have been used to support unmanned aerial vehicle formation flight [3], maritime traffic management [30], and wildfire surveillance [14]. Decentralized systems permit implicit communication, in which agents transmit information by means of taken actions and received observations [12]. Explicit communication, by contrast, endows agents with formalized communications actions.

The state of the art exact optimal algorithm for solving Dec-POMDPs, recursive small-step multi-agent A* (RS-MAA*) [16], relies on a combination of incremental expansion, clustering, variable-depth recursive heuristics, and heuristic memoization. We extend RS-MAA* to semi-decentralized systems.

Communication Schemes The literature features numerous explicitly modeled communication mechanisms and frameworks, owing to the sheer diversity of information structures in real-world problems. Goldman and Zilberstein formalize the decentralized POMDP with communication (Dec-POMDP-Com), which incorporates an alphabet of possible messages and a communication cost function [9]. The Dec-POMDP-Com provides a basis for costly communication and generally assumes noise-free instantaneous broadcasting. Delayed communication occurs when agents learn the local observation received by others after one or more time steps [22]. In one-step delayed communication, agents determine the latest joint action from the joint policy, but are unaware of the latest joint-observation [20]. In k-steps delayed communication, a sufficient statistic may be used in place of the past joint policy [21]. Other multiagent models consider noisy [33] or intermittent [34] communication channels.

In the costly, delayed, noisy, and intermittent communication cases, the domain environment is orthogonal to the communication channel, and the actions taken by agents and resulting state do not affect their future ability to coordinate. Therefore, while communication directly influences control, control does not in turn directly influence communication. This distinguishes semi-decentralized infrastructure from existing communication schemes.

Communication in Reinforcement Learning Our primary contribution is a foundational model that supports both planning and reinforcement learning in restricted communication environments. We further present an exact algorithm for semi-decentralized planning, in which the communication and model dynamics are known prior to execution [17]. Still, the growing body of multiagent deep reinforcement learning with communication (Comm-MADRL) research is replete with promising techniques for codifying communication and defining communication policy. Zhu et al. categorize Comm-MADRL approaches by communication policy, to include full communication, partial communication, individual control, and global control subcategories [35]. We are chiefly concerned with the individual control literature [13], [29], [28], and seek to learn optimal control policies in light of potential communication links between agents. In the sub-field of *learning tasks with communication*, policies generated using learning algorithms simultaneously maximize environmental rewards and determine effective communication protocols for agents [35]. This parallels recent efforts in planning to design joint communication and control strategies [31], which we also do.

3 PRELIMINARIES

3.1 Dec-POMDPs

The *decentralized partially observable Markov decision process* (Dec-POMDP) is a stochastic, decentralized multiagent model for sequential decision-making under partial observability characterized by tuple $\langle I, S, \bar{A}, \bar{O}, T, O, R, \rangle$, where:

- I is a finite set of k agents,
- S is a finite set of states,
- $\bar{A} = \times_i A_i$ is a finite set of joint actions,
- $\bar{O} = \times_i O_i$ is a finite set of joint-observations, and
- $T : S \times \bar{A} \times S \rightarrow [0, 1]$ is a state transition function where $T(s' \mid s, \bar{a})$ is the probability of being in state s' given joint action $\bar{a}$ being performed in state s,
- $O : \bar{O} \times S \times \bar{A}$ is a joint-observation function where $O(\bar{o}' \mid s', \bar{a})$ specifies the probability of attaining joint-observation $\bar{o}'$ when joint action $\bar{a}$ results in state s', and
- $R : S \times \bar{A} \rightarrow \mathbb{R}$ is a reward function such that $R(s, \bar{a})$ is the immediate reward for performing joint action $\bar{a}$ in state s.

Agents in the *decentralized partially observable Markov decision process* (Dec-POMDP) cannot explicitly share information. Each agent therefore select actions in accordance with a local policy π_i informed by action observation history h_i, where $\bar{h} \in (\prod_{i \in I} A_i O_i)^\star$. We are primarily concerned with deterministic policies, from which agent actions can be inferred using only observation histories $\bar{o}_h$. The collection of individual policies is the decentralized policy set $\bar{\pi} : \langle \pi_1, \pi_2, ... \pi_n \rangle$. Assume an infinite horizon $h = \infty$ and time discount rate γ. The objective is to find a policy set $\bar{\pi}$ maximizing expected reward over states and observation histories:

$$V^{\bar{\pi}}(s, \bar{o}_h) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s^t, \bar{\pi}(\bar{o}_h^t)) \mid b^0, \bar{\pi}\right]$$

$$\bar{\pi}^* \leftarrow \underset{\bar{a} \in \bar{A}}{\operatorname{argmax}} \left(R(s, \bar{a}) + \gamma \sum_{s' \in S} \sum_{\bar{o}' \in O} \Pr(s', \bar{o}' | s, \bar{a}) V^*(s', \bar{o}'_h) \right).$$

3.2 MPOMDPs

The *multiagent partially observable Markov decision process* (MPOMDP) is a stochastic, centralized multiagent model for sequential decision-making under partial observability also characterized by tuple $\langle I, S, \bar{A}, \bar{O}, T, O, R\rangle$. MPOMDP agents also lack a complete picture of the underlying system state, but can share individual actions a_i taken and observations received o_i. This permits a *sufficient statistic* in the form of a probability distribution over states called a *belief* $b \in \Delta^n$, where n is the number of states and Δ^n is the $n-1$ simplex. We notate the multiagent belief using b_α. The belief over successor states s' is updated using the set of agent histories $\bar{h}$ of actions taken and observation received:

$$b'_\alpha(s') = \eta O(\bar{o}' \mid s', \bar{a}) \sum_{s \in S} T(s' \mid s, \bar{a}) b_\alpha(s)$$

where η is a normalizing constant equal to $\Pr(\bar{o}' \mid b_\alpha, \bar{a})^{-1}$, and $\Pr(\bar{o}' \mid b_\alpha, \bar{a}) = \sum_{s' \in S} O(\bar{o}' \mid s', \bar{a}) \sum_{s \in S} T(s' \mid s, \bar{a}) b_\alpha(s)$. Belief $b'_{\alpha\bar{o}\bar{a}}$ is then generated by applying the belief update equation to all $s' \in S$. An initial belief b^0 is assumed to avoid considering the Bellman equation for infinite starting beliefs. From the agent perspective, reward must be calculated as a function of belief, such that $R(b_\alpha, \bar{a}) = \sum_{s \in S} R(s, \bar{a}) b_\alpha(s)$. In an MPOMDP, we seek a single joint policy $\pi_\alpha : \Delta^n \longrightarrow \bar{A}$ that maps beliefs to actions. The expected value of b_α over an infinite horizon may be written as:

$$V^{\pi_\alpha}(b_\alpha) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(b_\alpha^t, \pi_\alpha(b_\alpha^t)) \mid b_\alpha^0 = b_\alpha, \pi_\alpha\right].$$

We seek a π^*_α that maximizes expected reward over time, determined by:

$$\pi^*_\alpha \leftarrow \underset{\bar{a} \in \bar{A}}{\operatorname{argmax}} \left(R(b_\alpha, \bar{a}) + \gamma \sum_{\bar{o}' \in \bar{O}} \Pr(\bar{o}' \mid b_\alpha, \bar{a}) V^*(b'_{\alpha\bar{o}\bar{a}}) \right).$$

3.3 Semi-Markov Processes

Definition 1. The *semi-Markov property for control* admits a distribution over time for state transitions aligned with agent actions.
Definition 2. *Sojourn control time* τ is the assignment of general continuous random variable $\mathcal{T}$.

$$\boxed{Q(\mathcal{T} \le \tau, s' \mid s, a)} \tag{1}$$

The *semi-Markov decision process* (SMDP) [25] is a stochastic single agent model for sequential decision-making with sojourn system control. The SMDP is characterized by state transition distribution Q, which is the probability that the next state transition occurs at or before τ and results in successor state s'. It is mathematically convenient to define Q as the product of $F(\tau \mid s, a) T(s' \mid s, a, \tau)$, or equivalently $F(\tau \mid s', a, s) T(s' \mid s, a)$, where F is the cumulative distribution function of τ. The distribution of τ is conditioned on the current state s and action taken a and is therefore Markov. The "semi" in semi-Markov reflects the arbitrary probability distribution followed by model transitions. When $\tau = 0$, a new action is taken to coincide with a *decision epoch.*

The system natural process time is denoted as $\eta \in \mathbb{R}^+$, such that each decision epoch $t \in \mathbb{N}$ with corresponding sojourn time $\tau^t \in \mathbb{R}^+$ and state $s^t \in S$ occurs at η^t, where η^t is decision epoch start time within the natural process. By consequence, natural process time η is the sum of sojourn times $\sum \tau^t + \epsilon$, where ϵ is the elapsed time since η^t. Similarly, sojourn time τ^t may be expressed as $\eta^{t+1} - \eta^t$, or the difference in natural process times for subsequent decision epochs.

4 SEMI-DECENTRALIZATION

Definition 3. The *semi-Markov property for communication*, or *semi-decentralization*, admits a distribution over time for what information agents can store in memory.
Definition 4. *Sojourn communication time* τ is general continuous random variable representing the time for an agent to return to an information-sharing state.

$$\boxed{Q(\bar{\mathcal{T}} \le \bar{\tau}', s' \mid \bar{\tau}, s, \bar{a}, \bar{a}')} \tag{2}$$

As with SMDPs, we can define Q as the product of $F(\bar{\tau}' \mid s', \bar{a}', \bar{\tau})$ and $T(s' \mid s, \bar{a}, \bar{\tau})$, where $\bar{\tau}'$ may be conditioned on the subsequent joint action set $\bar{a}'$. SMDPs have one agent with an implicit conditioned $\bar{\tau} = 0$. However, SDec-POMDPs have multiple agents with varied $\bar{\tau}$. Thus it is more general with $\bar{\tau}'$ conditioned on $\bar{\tau}$. Semi-decentralized models assume an initial $\bar{\tau}^0$, which can be interpreted as the communicating state of each agent when $\eta = 0$. When $\tau = 0$, information sharing can occur coinciding with a *communication epoch.* We assume noise-free instantaneous broadcast communication resulting in a single communicating agent set as in a blackboard [7, 8]. Semi-decentralization may however incorporate multiple distinct communicating sets. Whereas as semi-Markov control systems toggle model transition dynamics using τ, semi-decentralized systems toggle updating histories using $\bar{\tau}$.

5 THE SDEC-POMDP

The *semi-decentralized partially observable Markov decision process* (SDec-POMDP) is a stochastic, semi-decentralized multiagent model for sequential decision-making under partial observability characterized by tuple $\langle I, S, \bar{A}, \bar{O}, F, T, O, R\rangle$.

Model The **SDec-POMDP model** introduces selector functions f, g, and h to propagate agent memories, actions, and observations, respectively, to M_i and M_c through selector sub-nodes conditioned on $\bar{\tau}$. Figure 2 depicts a SDec-POMDP dynamic decision network where selector sub-nodes are contained within $Z_i^{\text{sel}} = \langle M_{ci}^{\text{sel}}, \bar{A}_i^{\text{sel}}, \bar{O}_i^{\text{sel}}\rangle$ and $Z_c^{\text{sel}} = \langle \bar{M}^{\text{sel}}, \bar{A}^{\text{sel}}, \bar{O}^{\text{sel}}\rangle$. The selector infrastructure defines information-sharing configurations and enables the model to simultaneously maintain sets of centralized and decentralized agents. The set compositions change with changes to $\bar{\tau}$

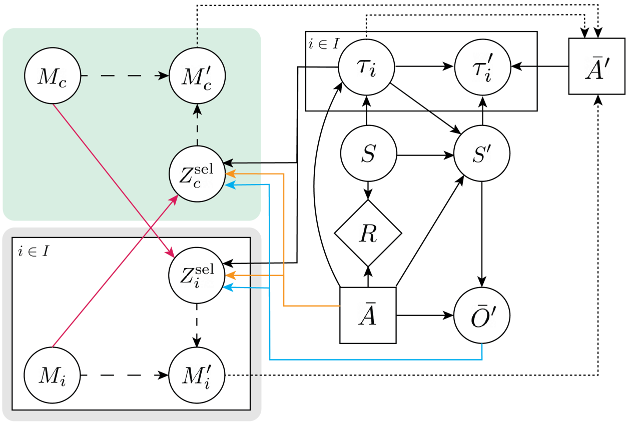

Figure 2: The SDec-POMDP dynamic decision network, with the policy infrastructure on the left and model on the right. The green backdrop contains the blackboard with memory M_c generated from the histories of communicating agents. The gray backdrop with plate notation includes the individual agent memories M_i. Z selector nodes are selectively toggled by $\bar{\tau}$ to facilitate memory propagation η, represented by dashed lines. Policy ψ edges are represented by dotted lines. The SDec-POMDP framework is flexible and can be easily modified to capture the structural and informational characteristics of different problem domains. For example, observations received may be conditioned on $\bar{\tau}$, such that $O(\bar{o}'|s', \bar{a}, \bar{\tau})$.

following each state transition. Define $M_c = (\prod_{i \in I}(\{\varnothing\} \cup A_i O_i))^\star$, and $M_i = (A_i O_i)^\star \times M_c$. The selector functions follow:

$$\forall i, f(m^{\text{sel}}_{ci}) = \begin{cases} m_c & \text{if } \tau_i = 0 \\ \varnothing & \text{if } \tau_i > 0 \end{cases}$$

$$f(m^{\text{sel}}_c) = \begin{cases} \forall i, m_i & \text{if } \tau_i = 0 \\ \varnothing & \text{if } \tau_i > 0 \end{cases}$$

$$\forall i, g(a^{\text{sel}}_i) = \begin{cases} a_j \in \bar{a}, \forall j \mid \tau_j = 0 & \text{if } \tau_i = 0 \\ a_i & \text{if } \tau_i > 0 \end{cases}$$

$$g(a^{\text{sel}}_c) = \{a_i \in \bar{a}, \forall i \mid \tau_i = 0\}$$

$$\forall i, h(o^{\text{sel}}_i) \begin{cases} o_j \in \bar{o}, \forall j \mid \tau_j = 0 & \text{if } \tau_i = 0 \\ o_i & \text{if } \tau_i > 0 \end{cases}$$

$$h(o^{\text{sel}}_c) = \{o_i \in \bar{o}, \forall i \mid \tau_i = 0\}.$$

Policy For generality, assume an **SDec-POMDP policy** with deterministically stored $a_i o_i, \forall i$ and deterministically selected $\bar{a}$, parameterized by both $\pi_i : M_i \rightarrow A$ and $\pi_c : M_c \rightarrow \bar{A}$. Memory propagation η and policy ψ probability functions follow:

$$\psi(\bar{a}) = \begin{cases} 1, \forall i \begin{cases} a_i = (\pi_c(m_c))_i & \text{if } \tau_i = 0 \\ a_i = \pi_i(m_i) & \text{if } \tau_i > 0 \end{cases} \\ 0, \text{otherwise} \end{cases}$$

$$\eta_c(m'_c) = \begin{cases} 1, \text{if } m'_c = m_c \bar{m}^{\text{sel}} \bar{a}^{\text{sel}} \bar{o}^{\text{sel}} \\ 0, \text{otherwise} \end{cases}$$

$$\forall i, \eta(m'_i) = \begin{cases} 1, \text{if } m'_i = m_i m^{\text{sel}}_{ci} a^{\text{sel}}_i o^{\text{sel}}_i \\ 0, \text{otherwise} \end{cases} .$$

Objective Function The **SDec-POMDP objective** is to identify the combination of memory propagation and policy functions that will maximize expected reward. Consider the infinite horizon case:

$$J(\psi, \eta_c, \bar{\eta}) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s^t, \bar{a}^t) \mid b^0\right]$$

$$J^\star = \underset{\psi, \eta_c, \bar{\eta}}{\operatorname{argmax}}\, J(\psi, \eta_c, \bar{\eta}).$$

Rewrite in terms of policies $\bar{\pi}$ and π_c and blackboard and agent memories m_c and $\bar{m}$:

$$\begin{aligned} V^{\pi_c, \bar{\pi}}(m_c, \bar{m}) &= \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s^t, \bar{a}^t) \mid b^0, m_c, \bar{m}, \pi_c, \bar{\pi}\right] \\ &= \sum_{s \in S} V^{\pi_c, \bar{\pi}}(m_c, \bar{m}, s) b^0(s) \end{aligned}$$

$$V^{\pi_c, \bar{\pi}}(m_c, \bar{m}, s) = \sum_{\bar{a} \in \bar{A}} \psi(\bar{a} \mid m_c, \bar{m}) \Bigg[R(s, \bar{a}) + \gamma \sum_{\bar{\tau} \in \bar{T}} F(d\bar{\tau} \mid s, \bar{a})$$
$$\sum_{s' \in S} T(s' \mid s, \bar{a}) \sum_{\bar{o}' \in \bar{O}} O(\bar{o}' \mid s', \bar{a})$$
$$\sum_{m'_c \in M_c} \eta_c(m'_c \mid m_c, f(\bar{m}), g(\bar{a}), h(\bar{o}'))$$
$$\sum_{\bar{m}' \in \bar{M}} \prod_{i \in I} \eta(m'_i \mid m_i, f(m_c), g(\bar{a}), h(\bar{o}')) V^{\pi_c, \bar{\pi}}(m'_c, \bar{m}', s') \Bigg].$$

6 THEORETICAL ANALYSIS

Definition 5. Models X_θ and X_ϕ are equivalent if they reduce to one another via mapping function f, such that $X_\theta \leq X_\phi$ and $X_\phi \leq X_\theta$.

Definition 6. Model-policy structures XY_θ and XY_ϕ are equivalent if they reduce to one another via mapping function g, such that $XY_\theta \leq XY_\phi$ and $XY_\phi \leq XY_\theta$.

Definition 7. Model-policy-objective structures Z_θ and Z_ϕ are equivalent if $\forall \bar{h}\ V_{XY_\theta}(\bar{h}, b^0) = V_{XY_\phi}(\bar{h}, b^0)$.

6.1 MPOMDP

Lemma 1. *SDec-POMDP and MPOMDP models are equivalent.*

Proof. Demonstrate that 1. $X_{\text{MPOMDP}} \leq X_{\text{SDec-POMDP}}$ and 2. $X_{\text{SDec-POMDP}} \leq X_{\text{MPOMDP}}$

1. $X_{\text{MPOMDP}} \leq X_{\text{SDec-POMDP}}$
Let $I' = I$, $S' = S$, $\bar{A}' = \bar{A}$, $R' = R$, and $\bar{O}' = \bar{O}$, where prime notation indicates the SDec-POMDP for purposes of relating models. The state transition and observation functions are defined to reproduce the MPOMDP dynamics independently of $\bar{\tau}$, such that $T'(s' \mid s, \bar{a}, \bar{\tau}) = T(s' \mid s, \bar{a})$ and $O'(\bar{o}' \mid s', \bar{a}, \bar{\tau}) = O(\bar{o}' \mid s', \bar{a})$. Assume a deterministic communication sojourn time function F, where τ' for each agent is fixed to one, resulting in complete centralization

at subsequent decision epochs; $\forall i$, $F'(\tau_i' < 1 \mid s, a_i, \tau_i) = 0$ and $F'(\tau_i' = 1 \mid s, a_i, \tau_i) = 1$. Selector functions therefore return joint actions and observations at each time-step; $\forall i$, $g'(a_i^{\text{sel}}) = g'(a_c^{\text{sel}}) = \bar{a}$ and $\forall i$, $h'(o_i^{\text{sel}}) = h'(o_c^{\text{sel}}) = \bar{o}$.

2. $X_{\text{SDec-POMDP}} \leq X_{\text{MPOMDP}}$
Let $R' = R$ and $\bar{A}' = \bar{A}$. The agent set is extended to include an additional blackboard agent with an independent memory, so that $I' = I \cup I_c$. We assume without loss of generality that $|I_c| = 1$, and that the communication sojourn time for this agent satisfies $\tau_i = 0$ $\forall i \in I_c$. The state space is augmented to include $\bar{\tau}$, such that $S' = S \times (\mathbb{R}^+)^n$. Similarly, the joint observation space is expanded to include the sequence of all action-observation pairs,

$$\bar{O}' = \bar{O} \times \prod_{i \in I} (A_i O_i)^\star.$$

The transition and observation functions adopt the factored state space: $T'(\langle s', \bar{\tau}' \rangle \mid \langle s, \bar{\tau} \rangle, \bar{a}, \bar{a}') = F(\bar{\tau}' \mid s', \bar{a}', \bar{\tau}) T(s' \mid s, \bar{a}, \bar{\tau})$ and $O'(\bar{o}' \mid \langle s', \bar{\tau}' \rangle, \langle s, \bar{\tau} \rangle, \bar{a}) = O(\bar{o}' \mid s', s, \bar{a}, \bar{\tau}', \bar{\tau})$ at its most general.

□

Lemma 2. *SDec-POMDP and MPOMDP model-policy structures are equivalent.*

Proof. Demonstrate that 1. $XY_{\text{MPOMDP}} \leq XY_{\text{SDec-POMDP}}$ and 2. $XY_{\text{SDec-POMDP}} \leq XY_{\text{MPOMDP}}$

1. $XY_{\text{MPOMDP}} \leq XY_{\text{SDec-POMDP}}$
Let each memory selector function return the joint memory, such that $\forall i$, $f'(m_{ci}^{\text{sel}}) = f'(m_c^{\text{sel}}) = \bar{m}$. The update rule for the joint memory is deterministic and concatenates the prior shared memory with the complete set of agent actions and observations from the current time-step, as represented by:

$$\eta'(\bar{m}') = \eta_c'(m_c') = \begin{cases} 1, \text{if } \bar{m}' = \bar{m}\bar{a}^{\text{sel}}\bar{o}^{\text{sel}} \\ 0, \text{otherwise.} \end{cases}$$

Agent actions are selected using a policy over the joint memory:

$$\psi'(\bar{a}) = \begin{cases} 1, \bar{a} = \pi(\bar{m}) \\ 0, \text{otherwise.} \end{cases}$$

2. $XY_{\text{SDec-POMDP}} \leq XY_{\text{MPOMDP}}$
Construct policy $\pi^M(m_i') = \langle a_1, \ldots a_n \rangle$, simulating where the action for agent i is determined to be:

$$a_i = \begin{cases} (\pi_c(\bar{m}_c))_i & \text{if } \tau_i = 0 \\ \pi_i(m_i) & \text{if } \tau_i > 0 \end{cases}$$

as if each agent draws from the blackboard's policy when its communication sojourn time is zero and otherwise following a local policy. The memory update rule is defined to extend the current memory with the observed joint outcome:

$$\eta'(\bar{m}') = \begin{cases} 1, \text{if } \bar{m}' = \bar{m}\bar{o} \\ 0, \text{otherwise.} \end{cases}$$

Joint action selection is consistent with the constructed joint policy:

$$\psi'(\bar{a}) = \begin{cases} 1, \forall i, a_i = \pi^M(m_i') \\ 0, \text{otherwise.} \end{cases}$$

□

Lemma 3. *SDec-POMDP and MPOMDP model-policy-objective structures are equivalent.*

Proof. Show $\forall \bar{h}$ $V_{XY_{\text{MPOMDP}}}(\bar{h}, b_0) = V_{XY_{\text{SDec-POMDP}}}(\bar{h}, b_0)$

Show that the semi-decentralized value function reduces to the standard value function under the original joint policy. Begin with,

$$V^{\pi_c,\bar{\pi}}(m_c, \bar{m}, s) = \underbrace{\sum_{\bar{a} \in \bar{A}} \psi(\bar{a} \mid m_c, \bar{m})}_{1, m_c = \bar{h}} \Bigg[R(s, \bar{a}) + \gamma \underbrace{\sum_{\bar{\tau} \in \bar{T}} F(d\bar{\tau} \mid s, \bar{a})}_{1, \text{ all RV } \perp \text{ of } d\tau}$$

$$\sum_{s' \in S} T(s' \mid s, \bar{a}) \sum_{\bar{o}' \in \bar{O}} O(\bar{o}' \mid s', \bar{a})$$

$$\underbrace{\sum_{m_c' \in M_c} \eta_c(m_c' \mid m_c, f(\bar{m}), g(\bar{a}), h(\bar{o}'))}_{1, m_c' = m_c \bar{a}^{\text{sel}} \bar{o}^{\text{sel}}}$$

$$\underbrace{\sum_{\bar{m}' \in \bar{M}} \prod_{i \in I} \eta(m_i' \mid m_i, f(m_c), g(\bar{a}), h(\bar{o}'))}_{1, \forall i, m_i' = m_i a_i^{\text{sel}} o_i^{\text{sel}}} V^{\pi_c,\bar{\pi}}(m_c', \bar{m}', s') \Bigg].$$

By construction, each under-braced term evaluates deterministically: the blackboard's memory update enforces $m_c' = m_c \bar{a}^{\text{sel}} \bar{o}^{\text{sel}}$, each agent's local memory update yields $m_i' = m_i a_i^{\text{sel}} o_i^{\text{sel}}$, and the distribution over sojourn times collapse to one. Moreover, since $\psi(\bar{a} \mid m_c, \bar{m}, s) = 1$ whenever $\bar{a} = \pi(\bar{h})$, we obtain $V^{\pi_c,\bar{\pi}}(m_c, \bar{m}, s) = V^\pi(s, \bar{h})$, which results in standard value recursion:

$$R(s, \pi(\bar{h})) + \gamma \sum_{s' \in S} \sum_{\bar{o}' \in O} \Pr(s', \bar{o}' \mid s, \pi(\bar{h})) V^\pi(s', \bar{h}').$$

Further observe that $\bar{m}_c = \bar{m}$ and

$$\sum_{s' \in S} T(s' \mid s, \bar{a}) \sum_{\bar{o}' \in \bar{O}} O(\bar{o}' \mid s', \bar{a}) = \sum_{s' \in S} \sum_{\bar{o}' \in O} \Pr(s', \bar{o}' \mid s, \pi(\bar{h})).$$

□

Proposition 1. *The SDec-POMDP and MPOMDP are equivalent.*

6.2 Dec-POMDP

Lemma 4. *SDec-POMDP and Dec-POMDP models are equivalent.*

Proof. 1. Demonstrate that $X_{\text{Dec-POMDP}} \leq X_{\text{SDec-POMDP}}$ and 2. $X_{\text{SDec-POMDP}} \leq X_{\text{Dec-POMDP}}$

1. $X_{\text{Dec-POMDP}} \leq X_{\text{SDec-POMDP}}$
Again set $I' = I$, $S' = S$, $\bar{A}' = \bar{A}$, $R' = R$, and $\bar{O}' = \bar{O}$. Let $T'(s' \mid s, \bar{a}, \bar{\tau}) = T(s' \mid s, \bar{a})$ and $O'(\bar{o}' \mid s', \bar{a}, \bar{\tau}') = O(\bar{o}' \mid s', \bar{a})$. Action and observation selection are specified so that for every agent i, $g'(a_i^{\text{sel}}) = a_i$ and $h'(o_i^{\text{sel}}) = o_i$. Assign to the blackboard memory the null set, such that $g'(a_c^{\text{sel}}) = h'(o_c^{\text{sel}}) = \varnothing$. By the construction of g' and h', τ has no impact and F can be disregarded.
2. $X_{\text{SDec-POMDP}} \leq X_{\text{Dec-POMDP}}$
Reference Lemma 1 proof 2, as $X_{\text{Dec-POMDP}} = X_{\text{MPOMDP}}$.

□

Lemma 5. *SDec-POMDP and Dec-POMDP model-policy structures are equivalent.*

Proof. Demonstrate that 1. $XY_{\text{Dec-POMDP}} \leq XY_{\text{SDec-POMDP}}$ and 2. $XY_{\text{SDec-POMDP}} \leq XY_{\text{Dec-POMDP}}$

1. $XY_{\text{Dec-POMDP}} \leq XY_{\text{SDec-POMDP}}$
Let each memory selector function return the null set, such that $\forall i$, $f'(m_{ci}^{\text{sel}}) = f'(m_c^{\text{sel}}) = \varnothing$. The update rule for each agent's memory is deterministic and concatenates the prior memory with the set of individual agent actions and observations from the current timestep, as represented by:

$$\eta'(m'_i) = \begin{cases} 1, \text{if } m'_i = m_i a_i^{\text{sel}} o_i^{\text{sel}} \\ 0, \text{otherwise} \end{cases}$$

Agent actions are selected using a policy over the agent's memory:

$$\psi'(\bar{a}) = \begin{cases} 1, \forall i, a_i = \pi_i(m_i) \\ 0, \text{otherwise} \end{cases}$$

By the construction of ψ', η'_c can be disregarded.

2. $XY_{\text{SDec-POMDP}} \leq XY_{\text{Dec-POMDP}}$
Construct simulated policy $(\pi^D)_i(m'_i) = a_i$, where:

$$a_i = \begin{cases} (\pi_c(m_c))_i & \text{if } \tau_i = 0 \\ \pi_i(m_i) & \text{if } \tau_i > 0 \end{cases}$$

as if each agent draws from the blackboard's policy when its communication sojourn time is zero and otherwise following a local policy. Each agent's memory update rule appends the taken action and observation to the the current memory:

$$\eta'(m'_i) = \begin{cases} 1, \text{if } m'_i = m_i a_i o_i \\ 0, \text{otherwise} \end{cases}$$

Agent action selection is consistent with the constructed policy:

$$\psi'(\bar{a}) = \begin{cases} 1, \forall i, a_i = (\pi^D)_i(m'_i) \\ 0, \text{otherwise} \end{cases}$$

□

Lemma 6. *SDec-POMDP and Dec-POMDP model-policy-objective structures are equivalent.*

Proof. Show $\forall \bar{h}\ V_{XY_{\text{Dec-POMDP}}}(\bar{h}, b_0) = V_{XY_{\text{SDec-POMDP}}}(\bar{h}, b_0)$
The semi-decentralized value function reduces to the decentralized value function under the original policy set, beginning with:

$$V^{\pi_c,\bar{\pi}}(\underbrace{\cancel{m_c}}_{M_c=\varnothing},\bar{m},s) = \underbrace{\sum_{\bar{a}\in\bar{A}} \psi(\bar{a}\mid \underbrace{\cancel{m_c}}_{M_c=\varnothing},\bar{m})}_{1}\Bigg[R(s,\bar{a})+$$

$$\gamma \underbrace{\sum_{\bar{\tau}\in\bar{T}} F(d\bar{\tau}\mid s,\bar{a})}_{1,\text{ all RV }\perp\text{ of }d\tau} \sum_{s'\in S} T(s'\mid s,\bar{a})\sum_{\bar{o}'\in\bar{O}} O(\bar{o}'\mid s',\bar{a})$$

$$\underbrace{\sum_{\bar{m}'\in\bar{M}}\prod_{i\in I}\eta(m'_i\mid m_i,\underbrace{\cancel{f(m_c)}}_{M_c=\varnothing},g(\bar{a}),g(\bar{a}),h(\bar{o}'))}_{1,\forall i,m'_i=m_i a_i^{\text{sel}} o_i^{\text{sel}}}$$

$$\underbrace{\cancel{\sum_{m'_c\in M_c}\eta_c(m'_c\mid m_c,f(\bar{m}),g(\bar{a}),h(\bar{o}'))}}_{M_c=\varnothing} V^{\pi_c,\bar{\pi}}(\underbrace{\cancel{m'_c}}_{M_c=\varnothing},\bar{m}',s')\Bigg].$$

By construction, each under-braced term evaluates deterministically: the blackboard's memory remains $m'_c = \varnothing$, each agent's local memory update yields $m'_i = m_i a_i^{\text{sel}} o_i^{\text{sel}}$, and the distribution over sojourn times collapses to one. As before, since $\psi(\bar{a} \mid m_c, \bar{m}, s) = 1$ whenever $\bar{a} = \bar{\pi}(\bar{h})$, we obtain $V^{\pi_c,\bar{\pi}}(m_c, \bar{m}, s) = V^{\bar{\pi}}(s, \bar{h})$, which results in standard value recursion:

$$R(s, \bar{\pi}(\bar{h})) + \gamma \sum_{s'\in S}\sum_{\bar{o}'\in O} \Pr(s', \bar{o}' \mid s, \bar{\pi}(\bar{h})) V^{\bar{\pi}}(s', \bar{h}')$$

Again observe that,

$$\sum_{s'\in S} T(s' \mid s, \bar{a}) \sum_{\bar{o}'\in\bar{O}} O(\bar{o}' \mid s', \bar{a}) = \sum_{s'\in S}\sum_{\bar{o}'\in O} \Pr(s', \bar{o}' \mid s, \bar{\pi}(\bar{h})).$$

□

Proposition 2. *The SDec-POMDP and Dec-POMDP are equivalent*

Corollary 1. The SDec-POMDP is the same complexity class as a Dec-POMDP (NEXP-complete)

Proposition 3. *The SDec-POMDP and k-steps-delayed communication are equivalent*

Proposition 4. *The SDec-POMDP and Dec-POMDP-Com are equivalent*

We provide proofs for Propositions 3 and 4 in the technical appendix.

7 RECURSIVE SMALL-STEP SEMI-DECENTRALIZED A*

Recursive small-step semi-decentralized A* (RS-SDA*) is an exact planning algorithm for optimizing expected reward in SDec-POMDP problems. RS-SDA* modifies RS-MAA* [16] by maintaining a stage-specific partition of decentralized and centralized joint-observation histories per probabilistic communication dynamics. Like RS-MAA*, RS-SDA* relies on incremental expansion of a small-step search tree, clustering, recursive heuristics, memoization, and last stage modifications. RS-SDA* generates a fully-specified policy set $\bar{\pi} \in \Pi$ using offline planning. A fully-specified policy contains both fully-specified local policies and, if appropriate for the problem, a blackboard policy $\bar{\pi} = \langle \pi_1, ...\pi_n, \pi_c \rangle$ where $\pi_i : O^h \to A_i$ and $\pi_c : \bar{O}^h \to \bar{A}$. Similarly, $\bar{\varphi} = \langle \varphi_1, ...\varphi_n, \varphi_c \rangle$, $\varphi_i : O^{\leq h-1} \to A_i$, and $\bar{\varphi} : \bar{O}^{\leq h-1} \to \bar{A}$. RS-SDA* is outlined in Algorithm 1.

Small-Step Search Tree We adopt the small-step approach first introduced by Cazenave for A* and used by Koops et al. in RS-MAA* to limit search tree outdegree. Small-step search can be used with both centralized and decentralized components of each policy node, depicted in Figure 4. As shown in Table 1, small-step search provides RS-SDA* with mixed component policies, pre-clustering, a lower bound (complete decentralization) and an upper bound (complete centralization) on the number of considered nodes per stage t. When both $F(\tau \mid s, \bar{a}) = F(\tau \mid \bar{a})$ and given a deterministic policy, we can consolidate the centralized and decentralized components of policies and significantly reduce the search tree size. This results in an RS-SDA* lower bound that is equivalent to RS-MAA* and an RS-SDA* upper bound that has $|O_\star|^{tn}$ levels per stage and considers $|A_\star|^n$ joint actions per level, where $|O_\star|$ is the size of the largest observation set and $|A_\star|$ is the size of the largest action set.

	RS-MAA*		**RS-SDA***		**Classical MAA***	
	(RS-SDA lower bound)*		*upper bound*			
	levels	nodes/stage	levels	nodes/stage	levels	nodes/stage
t	$n\|O_\star\|^t$	$n\|O_\star\|^t\|A_\star\|$	$\|O_\star\|^{tn}$	$\|O_\star\|^{tn}\|A_\star\|^n$	1	$\|A_\star\|^{n\|O_\star\|^t}$
0	2	6	1	9	1	9
1	4	12	4	36	1	81
2	8	24	16	144	1	6,561
3	16	48	64	576	1	>1E6
4	32	96	256	2,304	1	>1E6
5	64	192	1,024	9,216	1	>1E6
6	128	384	4,096	36,864	1	>1E6
7	256	768	16,384	147,456	1	>1E6
8	512	1536	65,536	589,824	1	>1E6

Table 1: Pre-clustered SDec-Tiger levels and nodes per stage by stage for RS-MAA*, RS-SDA*, and classical MAA*. $|A_\star| = 3$, $|O_\star| = 2$, **and** $n = 2$.

Dynamic Programming We apply backward induction over beliefs to rapidly determine the value of centralized policy components. This bypasses expensive recursive heuristic calculations for large portions of the search. For each remaining horizon r and belief b, we compute $V_r(b)$ and $Q_r(b, \bar{a})$ and memoize both V and Q under keys (r, b) and $(r, b, \bar{a})$ to enable extensive reuse during A* expansion. Similarly, observation likelihoods $P(\bar{o} \mid b, \bar{a})$ and posteriors $(b'_{\bar{o},\bar{a}})$ are obtained via a belief update over the model's transition and observation tensors, then cached for subsequent calls.

Ordering Observation Histories Any $\bar{\varphi}$ may contain both decentralized and centralized mappings conditioned on the underlying state and actions taken. We therefore explore joint-observation histories (JOH) and local observation histories (LOH) in a predetermined sequence: stage, JOH then LOH, by agent (for LOHs), then lexicographically. Observe that, for length t, all $\bar{o}^{0:t} \preceq o^{0:t}$ and all $o^{0:t} \preceq \bar{o}^{0:t'}$. Additionally, $\bar{o}^{0:t} \preceq_{\text{lex}} \bar{o}^{0:t}$, and $o_i^{0:t} \preceq o_j^{0:t}$ if $(i < j)$ or $(i = j \wedge o_i^{0:t} \preceq_{\text{lex}} o_j^{0:t})$.

Clustering We implement lossless incremental clustering in decentralized policy components based on a probabilistic equivalence criterion [24]. We similarly cluster centralized policy components based on the resulting joint belief, or $\forall_s$, $\text{P}(s \mid \bar{o}_1^{0:t}) = \text{P}(s \mid \bar{o}_2^{0:t})$.

Admissible Heuristic As with multiagent A* (MAA*) [32], an *admissible heuristic* Q guides a path through a search tree with partial policies as nodes $\bar{\varphi}$. A heuristic is admissible if it equals or over-approximates the true value of the policy node. An *open list* is maintained with nodes under consideration. The node in the open list with the highest heuristic value is expanded and replaced in the open list by its children. The tree search terminates once a fully-specified policy with the highest heuristic value is identified. For each parent node and candidate action, we split each posterior belief by S_τ and A_τ (more generally expressed using $F(s, a)$) and take a probability-weighted sum of the exact centralized value on the communication-dependent posterior and the exact decentralized value on its complement. Because every constituent (centralized, decentralized, and their mixture conditioned on communication) is an exact optimum of a relaxation of the remaining subproblem, the resulting heuristic **never underestimates** the achievable return and is therefore admissible.

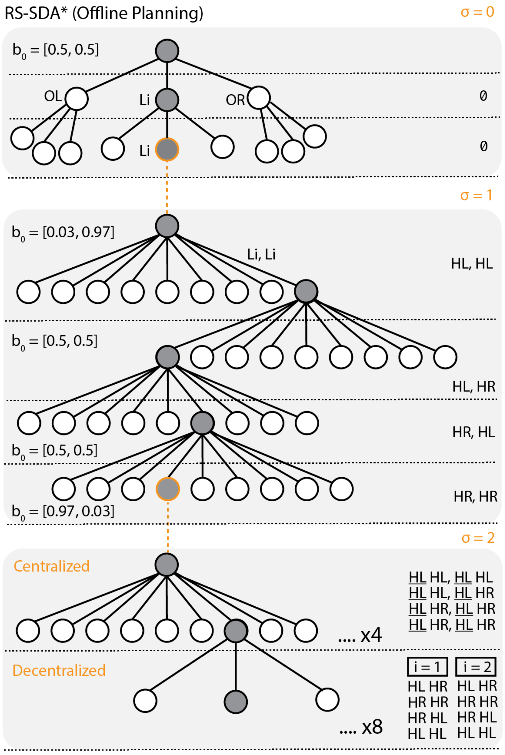

Figure 3: Illustrating RS-SDA* applied to SDec-Tiger using mixed component policies through stage $\sigma = 2$.

8 EXPERIMENTS

Semi-Decentralized Benchmarks We evaluate RS-SDA* in semi-decentralized versions of four common Dec-POMDP benchmarks: Dec-Tiger [18], FireFighting [23], BoxPushing [26], and Mars [2], and in a new MaritimeMEDEVAC benchmark. Problem descriptions with illustrations are disclosed in the technical appendix. All experiments were conducted using an 11th Gen Intel 2.50GHz

Algorithm 1: Recursive Small-Step Semi-Decentralized A*

```
Input  : $S_\tau \subseteq S$ synchronization states; $A_\tau \subseteq A$ synchronization actions; $h$
         horizon; $b$ initial belief; $\varphi$ initial (partial) policy; $d$ heuristic depth;
         $M$ iterations; $\alpha$ threshold; $u$ upper bound
Output : optimal policy $\varphi*$, optimal value $v*$

function RS-SDA*$(h, b, \varphi, d, M, \alpha, u, S_\tau, A_\tau)$:
    $Q \leftarrow$ PriorityQueue$(\uparrow)$
    $Q$.push(min($\varphi$.heuristics), $\varphi$)
    $i \leftarrow 0$
    while true do
        $(v, \varphi) \leftarrow Q$.pop()
        if $\varphi$ fully specified then return $(v, \varphi)$
        if $v < \infty$ then $i \leftarrow i + 1$
        if $i \geq M$ or $v \leq u$ then return (min$(v, u)$, None)
        $k \leftarrow$ current stage of $\varphi$
        if $\sigma_k$ complete then
            $(D_{\text{dec}}, D_{\text{cen}}, P_{\text{dec}}, P_{\text{cen}}, m_{\text{dec}}) \leftarrow$ Terminal$(\varphi, S_\tau, A_\tau)$
            if $m_{\text{dec}} > 0$ then
              $\varphi \leftarrow$ ClusterPolicyDec$(\varphi, D_{\text{dec}}, P_{\text{dec}})$
            if $m_{\text{dec}} < 1$ then
              $\varphi \leftarrow$ ClusterPolicyCEN$(\varphi, D_{\text{cen}}, P_{\text{cen}})$
            $w \leftarrow$ EvaluatePolicy$(\varphi, k)$
            $\varphi$.heuristics.append($w$)
            $\varphi$.depth $\leftarrow$ min$(k, d)$
            $k \leftarrow k + 1$
        if $\sigma_k$ is final and some centralized child is unfilled then
            determine $\bar{a}^*$ for all centralized component LOHs
            $\varphi' \leftarrow \varphi$ with $\bar{a}^*$ set
            $w \leftarrow$ EvaluatePolicy$(\varphi', k)$
            $Q$.push$(w, \varphi')$
        else if $\sigma_k$ is final and one agent's decentralized children is unfilled
          then
            determine $a_i^*$ for all agent decentralized component LOHs
            $\varphi' \leftarrow \varphi$ with $a_i^*$ set
            $w \leftarrow$ EvaluatePolicy$(\varphi', k)$
            if $w = v$ then return $(w, \varphi')$
            $Q$.push$(w, \varphi')$
        else if some centralized child at $\sigma_k$ unfilled then
            foreach joint action $\bar{a} \in \bar{A}$ do
                $\varphi' \leftarrow \varphi$ with $\bar{a}$ set
                $w \leftarrow$ EvaluatePolicy$(\varphi', k)$
                $Q$.push$(w, \varphi')$
            end
        else if some agent $i$ has an unset decentralized entry at $\sigma_k$ then
            foreach $a_i \in A$ do
                $\varphi' \leftarrow \varphi$ with $a_i$ set
                $w \leftarrow$ EvaluatePolicy$(\varphi', k)$
                $Q$.push$(w, \varphi')$
            end
    end
```

i7 CPU, with timeout occurring at 20 minutes and memory out at 16 GB. We adopt hyper-parameters $M = 200$, $d = 3$, and $\alpha = 0.2$ for all experiments. A link to our code repository is provided in the technical appendix to support reproducibility.

Results As shown in Table 2 and Figure 4, RS-SDA* is competitive with the centralized upper bound across most semi-decentralized benchmarks and MaritimeMEDEVAC. The modified benchmarks demonstrate how model dynamics influence the value of information in multi-agent systems. SDec-FireFighting exemplifies problems where centralization benefits are negligible, and the optimal RS-SDA* solution equals the optimal RS-MAA* solution for all considered h. By contrast, SDec-Box exemplifies problems where partial centralization results in complete information sharing, and the optimal RS-SDA* solution equals the fully centralized optimum for all considered h. For SDec-Tiger, the semi-decentralization substantially improves over the lower bound but remains below the centralized upper bound. In MaritimeMEDEVAC, the three regimes are nearly indistinguishable at moderate horizons ($H = 5, 6$), but at $H = 7$ centralized reaches 6.62 while semi-decentralized attains 4.54, clearly outperforming full decentralization (3.27). At $H = 7$, the semi-decentralized policy recovers about 69% of the centralized value. These results indicate that semi-decentralization and RS-SDA* preserve much of the benefit of centralized coordination while staying tractable, with occasional slowdowns or timeouts on problem instances where lossless clustering is largely ineffective.

	lower bound		**our approach**		upper bound	
	decentralized RS-MAA*		*semi-decentralized* **RS-SDA***		*centralized* VI	
h	value	time	value	time	value	time
SDec-Tiger						
8	12.21726	2	27.21518	<1	47.71696	<1
9	15.57244	19	30.90457	<1	53.47353	<1
10	15.18438	TO	34.72370	<1	60.50990	<1
SDec-FireFighting ($n_h = 3, n_f = 3$)						
3	-5.73697	<1	-5.72415	<1	-5.72285	<1
4	-6.57883	8	-6.56419	7	-6.51859	7
5	-7.06987	89	-6.98102	94	-6.98069	94
SDec-BoxPushing						
3	66.08100	<1	66.81000	<1	66.81000	<1
4	98.59613	24	99.55630	<1	99.55630	<1
5	107.72985	MO	109.09251	1	109.09251	1
SDec-Mars (Right Band Rendezvous)						
4	10.18080	3	10.18080	<1	10.87020	<1
5	13.26654	9	14.29038	<1	14.38556	1
6	18.62317	19	20.06430	2	20.06706	3
SDec-Mars (Survey Site Beacons)						
4	10.18080	3	10.54620	<1	10.87020	<1
5	13.26654	9	13.26654	<1	14.38556	1
6	18.62317	19	18.62317	143	20.06706	3
SDec-Mars (Drill Site Beacons)						
4	10.18080	3	10.87020	<1	10.87020	<1
5	13.26654	9	14.38556	<1	14.38556	1
6	18.62317	19	20.06168	2	20.06706	3
MaritimeMEDEVAC						
5	3.46017	<1	3.48345	1	3.49629	<1
6	3.18348	<1	3.19807	28	3.19945	<1
7	3.26710	2	6.36301	37	6.61819	1
8	8.03228	260	10.61275	660	10.88244	1

Table 2: RS-MAA* offline planning and RS-SDA* offline planning/online search performance on semi-decentralized benchmarks and the Maritime MEDEVAC problem. TO and MO denote timeout (>1200s) and memout (>16GB).

9 CONCLUSION

We present a foundational framework for multiagent decision making under probabilistic communication. We formalize the semi- decentralization property and introduce the SDec-POMDP, which unifies the Dec-POMDP, MPOMDP, and several communication mechanisms with delay, loss, or cost. A secondary contribution is RS-SDA*, an admissible heuristic search algorithm for semi-decentralized systems with performance comparable to the state-of-the-art, and semi-decentralized versions of four standard benchmarks and a

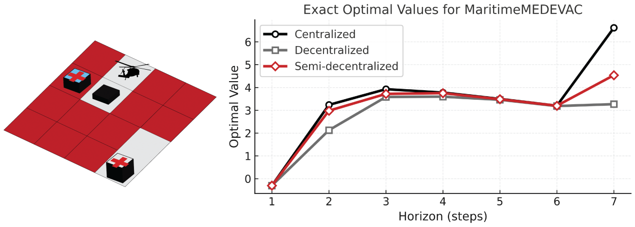

Figure 4: MaritimeMEDEVAC environment representation and centralized/decentralized/semi-decentralized optimal policy values for horizons one through seven.

new medical evacuation scenario. Taken together, SDec-POMDP and RS-SDA* provide a principled basis for studying and exploiting probabilistic communication in cooperative teams. Future work includes exploiting interleaving offline planning and online search to improve approximate RS-SDA* performance and investigating systems with non-stationary sojourn time distributions.

REFERENCES

[1] Christopher Amato, Girish Chowdhary, Alborz Geramifard, N Kemal Üre, and Mykel J Kochenderfer. 2013. Decentralized control of partially observable Markov decision processes. In *52nd IEEE Conference on Decision and Control*. IEEE, 2398–2405.

[2] Christopher Amato and Shlomo Zilberstein. 2009. Achieving goals in decentralized POMDPs. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*. 593–600.

[3] Md Ali Azam, Hans D Mittelmann, and Shankarachary Ragi. 2021. Uav formation shape control via decentralized markov decision processes. *Algorithms* 14, 3 (2021), 91.

[4] Raphen Becker, Shlomo Zilberstein, Victor Lesser, and Claudia V Goldman. 2004. Solving transition independent decentralized Markov decision processes. *Journal of Artificial Intelligence Research* 22 (2004), 423–455.

[5] Daniel S Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. 2002. The complexity of decentralized control of Markov decision processes. *Mathematics of operations research* 27, 4 (2002), 819–840.

[6] Tristan Cazenave. 2010. Partial Move A. In *2010 22nd IEEE International Conference on Tools with Artificial Intelligence*, Vol. 2. IEEE, 25–31.

[7] Iain D Craig. 1988. Blackboard systems. *Artificial Intelligence Review* 2, 2 (1988), 103–118.

[8] Lee D Erman, Frederick Hayes-Roth, Victor R Lesser, and D Raj Reddy. 1980. The Hearsay-II speech-understanding system: Integrating knowledge to resolve uncertainty. *ACM Computing Surveys (CSUR)* 12, 2 (1980), 213–253.

[9] Claudia V Goldman and Shlomo Zilberstein. 2003. Optimizing information exchange in cooperative multi-agent systems. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*. 137–144.

[10] Claudia V Goldman and Shlomo Zilberstein. 2008. Communication-based decomposition mechanisms for decentralized MDPs. *Journal of Artificial Intelligence Research* 32 (2008), 169–202.

[11] Alan Grant, Paul Williams, Nick Ward, and Sally Basker. 2009. GPS jamming and the impact on maritime navigation. *The Journal of Navigation* 62, 2 (2009), 173–187.

[12] Pulkit Grover and Anant Sahai. 2010. Implicit and explicit communication in decentralized control. In *2010 48th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 278–285.

[13] Jiechuan Jiang and Zongqing Lu. 2018. Learning attentional communication for multi-agent cooperation. *Advances in neural information processing systems* 31 (2018).

[14] Kyle D Julian and Mykel J Kochenderfer. 2019. Distributed wildfire surveillance with autonomous aircraft using deep reinforcement learning. *Journal of Guidance, Control, and Dynamics* 42, 8 (2019), 1768–1778.

[15] Mykel J Kochenderfer. 2015. *Decision making under uncertainty: theory and application*. MIT press.

[16] Wietze Koops, Nils Jansen, Sebastian Junges, and Thiago D Simao. 2023. Recursive small-step multi-agent A* for dec-POMDPs. *International Joint Conference on Artificial Intelligence* (2023).

[17] Thomas M Moerland, Joost Broekens, and Catholijn M Jonker. 2020. A framework for reinforcement learning and planning. *arXiv preprint arXiv:2006.15009* 127 (2020).

[18] Ranjit Nair, Milind Tambe, Makoto Yokoo, David Pynadath, and Stacy Marsella. 2003. Taming decentralized POMDPs: Towards efficient policy computation for multiagent settings. In *IJCAI*, Vol. 3. 705–711.

[19] Ashutosh Nayyar, Aditya Mahajan, and Demosthenis Teneketzis. 2010. Optimal control strategies in delayed sharing information structures. *IEEE Trans. Automat. Control* 56, 7 (2010), 1606–1620.

[20] Frans Oliehoek and Matthijs Spaan. 2012. Tree-based solution methods for multiagent POMDPs with delayed communication. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 26. 1415–1421.

[21] Frans Adriaan Oliehoek. 2013. Sufficient Plan-Time Statistics for Decentralized POMDPs.. In *IJCAI*. 302–308.

[22] Frans A Oliehoek, Christopher Amato, et al. 2016. *A concise introduction to decentralized POMDPs*. Vol. 1. Springer.

[23] Frans A Oliehoek, Matthijs TJ Spaan, and Nikos Vlassis. 2008. Optimal and approximate Q-value functions for decentralized POMDPs. *Journal of Artificial Intelligence Research* 32 (2008), 289–353.

[24] Frans A Oliehoek, Shimon Whiteson, Matthijs TJ Spaan, et al. 2009. Lossless clustering of histories in decentralized POMDPs.. In *AAMAS (1)*. 577–584.

[25] Martin L Puterman. 1994. Markov decision processes: Discrete stochastic dynamic programming. Jhon Wiley & Sons. *Inc., New York, NY* (1994).

[26] Sven Seuken and Shlomo Zilberstein. 2007. Improved memory-bounded dynamic programming for decentralized POMDPs. In *Proceedings of the Twenty-Third Conference on Uncertainty in Artificial Intelligence*. 344–351.

[27] Sven Seuken and Shlomo Zilberstein. 2008. Formal models and algorithms for decentralized decision making under uncertainty. *Autonomous Agents and Multi-Agent Systems* 17, 2 (2008), 190–250.

[28] Junjie Sheng, Xiangfeng Wang, Bo Jin, Junchi Yan, Wenhao Li, Tsung-Hui Chang, Jun Wang, and Hongyuan Zha. 2022. Learning structured communication for multi-agent reinforcement learning. *Autonomous Agents and Multi-Agent Systems* 36, 2 (2022), 50.

[29] Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. 2018. Learning when to communicate at scale in multiagent cooperative and competitive tasks. *arXiv preprint arXiv:1812.09755* (2018).

[30] Arambam James Singh, Duc Thien Nguyen, Akshat Kumar, and Hoong Chuin Lau. 2019. Multiagent decision making for maritime traffic management. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 6171–6178.

[31] Sagar Sudhakara, Dhruva Kartik, Rahul Jain, and Ashutosh Nayyar. 2024. Optimal communication and control strategies in a cooperative multi-agent MDP problem. *IEEE Trans. Automat. Control* (2024).

[32] Daniel Szer, François Charpillet, and Shlomo Zilberstein. 2005. MAA*: A Heuristic Search Algorithm for Solving Decentralized POMDPs. In *21st Conference on Uncertainty in Artificial Intelligence-UAI'2005*.

[33] Tze-Yang Tung, Szymon Kobus, Joan Pujol Roig, and Deniz Gündüz. 2021. Effective communications: A joint learning and communication framework for multi-agent reinforcement learning over noisy channels. *IEEE Journal on Selected Areas in Communications* 39, 8 (2021), 2590–2603.

[34] Ruixue Zhang, Jiao Wang, Jun Ge, and Qiyuan Huang. 2024. Multiagent cooperative search learning with intermittent communication. *IEEE Intelligent Systems* 39, 2 (2024), 11–20.

[35] Changxi Zhu, Mehdi Dastani, and Shihan Wang. 2024. A survey of multi-agent deep reinforcement learning with communication. *Autonomous Agents and Multi-Agent Systems* 38, 1 (2024), 4.

A APPENDIX

A.1 k-Steps Delayed Communication

k-steps delayed communication [19], [21], [23] is a model for delayed broadcast communication where each agent receives the complete joint history from $t-k$ at t. This enables k-steps delayed communication to generate a joint belief b^{t-k} on which to conduct subsequent decentralized planning. We formally define the agent histories under k-steps delayed communication, below:

$$H_i^t = (A_i O_i)^t$$

$$\bar{h}_i \in \bar{H}_{i,k} = \bigcup_{t=1}^{\infty} \prod_{j \in I} \begin{cases} H_i^t & \text{if } j = i \\ H_j^{\max\{0,\, t-k\}} & \text{otherwise} \end{cases}$$

$$\bar{\bar{h}} = \prod_{i \in I} \bar{h}_i$$

Lemma 7. *SDec-POMDP and k-steps delayed communication models are equivalent.*

Proof. Demonstrate that 1. $X_{k\text{-steps Delayed}} \leq X_{\text{SDec-POMDP}}$ and 2. $X_{\text{SDec-POMDP}} \leq X_{k\text{-steps Delayed}}$

1. $X_{k\text{-steps Delayed}} \leq X_{\text{SDec-POMDP}}$
Let $I' = I$, $S' = S$, $\bar{A}' = \bar{A}$, $R' = R$, and $\bar{O}' = \bar{O}$. The state transition and communication sojourn time functions are independent of $\bar{\tau}$ such that $T'(s' \mid s, \bar{a}, \bar{\tau}) = T(s' \mid s, \bar{a})$ and $O'(\bar{o}' \mid s', \bar{a}, \bar{\tau}') = O(\bar{o}' \mid s', \bar{a})$. Again assume a deterministic communication sojourn time function F, where τ' for each agent is fixed to one, resulting in complete centralization at subsequent decision epochs; $\forall i$, $F'(\tau_i' < 1 \mid s, a_i, \tau_i) = 0$ and $F'(\tau_i' = 1 \mid s, a_i, \tau_i) = 1$. Blackboard selector functions return joint actions and observations at each time-step; $g'(a_c^{\text{sel}}) = \bar{a}$ and $h'(o_c^{\text{sel}}) = \bar{o}$. Agent selector functions return agent actions and observations at each time-step; $\forall i$, $g'(a_i^{\text{sel}}) = a_i$ and $\forall i$, $h'(o_i^{\text{sel}}) = o_i$.

2. $X_{\text{SDec-POMDP}} \leq X_{k\text{-steps Delayed}}$
Let $k = 0$. See proof of Lemma 4 for construction of a SDec-POMDP model within a Dec-POMDP. □

Lemma 8. *SDec-POMDP and k-steps delayed model-policy structures are equivalent.*

Proof. Demonstrate that $XY_{k\text{-steps Delayed}} \leq XY_{\text{SDec-POMDP}}$ and 2. $XY_{\text{SDec-POMDP}} \leq XY_{k\text{-steps Delayed}}$

1. $XY_{k\text{-steps Delayed}} \leq XY_{\text{SDec-POMDP}}$
For any object X carrying a time index, $X\big|_{0:r}$ denotes the same object with indices $> r$ disabled/ignored. Let each agent memory selector function return the blackboard memory at $t-k$, such that $\forall i$, $f'(m_{ci}^{\text{sel}}) = m_c^t\big|_{0:\,t-k}$. The blackboard memory selector function returns the latest joint agent memory set. The update rule for each agent's memory is deterministic and concatenates the prior memory with the set of individual agent actions and observations from the current time-step:

$$\eta'(m_i') = \begin{cases} 1, \text{if } m_i' = m_i m_{ci}^{\text{sel}} a_i^{\text{sel}} o_i^{\text{sel}} \\ 0, \text{otherwise.} \end{cases}$$

Agent actions are selected using a policy over the agent's memory:

$$\psi'(\bar{a}) = \begin{cases} 1, \forall i, a_i = \pi_i(m_i) \\ 0, \text{otherwise.} \end{cases}$$

Finally, the update rule for the blackboard memory is deterministic and concatenates the prior shared memory with the complete set of agent actions and observations from the current time-step:

$$\eta_c(m_c') = \begin{cases} 1, \text{if } m_c' = m_c \bar{a}^{\text{sel}} \bar{o}^{\text{sel}} \\ 0, \text{otherwise.} \end{cases}$$

2. $XY_{\text{SDec-POMDP}} \leq XY_{k\text{-steps Delayed}}$
Let $k = 0$. See proof of Lemma 5 for construction of a SDec-POMDP model-policy structure within a Dec-POMDP.

□

Lemma 9. *SDec-POMDP and k-steps delayed model-policy-objective structures are equivalent.*

The semi-decentralized value function reduces to the k-steps delayed value function under the original policy set, beginning with:

Proof. Show $\forall \bar{h}$ $V_{XY_{k\text{-steps Delayed}}}(\bar{h}, b_0) = V_{XY_{\text{SDec-POMDP}}}(\bar{h}, b_0)$

$$V^{\pi_c,\bar{\pi}}(m_c, \bar{m}, s) = \underbrace{\sum_{\bar{a} \in \bar{A}} \psi(\bar{a} \mid m_c, \bar{m})}_{1, m_c = \bar{h}} \Bigg[R(s, \bar{a}) + \gamma \underbrace{\sum_{\bar{\tau} \in \bar{T}} F(d\bar{\tau} \mid s, \bar{a})}_{1, \text{ all RV } \perp \text{ of } d\tau}$$

$$\sum_{s' \in S} T(s' \mid s, \bar{a}) \sum_{\bar{o}' \in \bar{O}} O(\bar{o}' \mid s', \bar{a})$$

$$\underbrace{\sum_{m_c' \in M_c} \eta_c(m_c' \mid m_c, f(\bar{m}), g(\bar{a}), h(\bar{o}'))}_{1, m_c' = m_c \bar{a}^{\text{sel}} \bar{o}^{\text{sel}}}$$

$$\underbrace{\sum_{\bar{m}' \in \bar{M}} \prod_{i \in I} \eta(m_i' \mid m_i, f(m_c), g(\bar{a}), h(\bar{o}'))}_{1, \forall i, m_i' = m_i a_i^{\text{sel}} o_i^{\text{sel}}} V^{\pi_c,\bar{\pi}}(m_c', \bar{m}', s') \Bigg]$$

By construction, each under-braced term evaluates deterministically: the blackboard's memory update enforces $m_c' = m_c \bar{a}^{\text{sel}} \bar{o}^{\text{sel}}$, each agent's local memory update yields $m_i' = m_i a_i^{\text{sel}} o_i^{\text{sel}}$, and the distribution over communication sojourn times collapse to one. Moreover, since $\psi(\bar{a} \mid m_c, \bar{m}, s) = 1$ whenever $\bar{a} = \bar{\pi}(\bar{\bar{h}})$, we obtain $V^{\pi_c,\bar{\pi}}(m_c, \bar{m}, s) = V^{\bar{\pi},k}(s, \bar{\bar{h}})$, which results in standard value recursion:

$$R(s, \bar{\pi}(\bar{\bar{h}}) + \gamma \sum_{s' \in S} \sum_{\bar{o}' \in O} \Pr(s', \bar{o}' \mid s, \bar{\pi}(\bar{\bar{h}}) V^{\bar{\pi}}(s', \bar{\bar{h}})$$

Observe that:

$$\sum_{s' \in S} T(s' \mid s, \bar{a}) \sum_{\bar{o}' \in \bar{O}} O(\bar{o}' \mid s', \bar{a}) = \sum_{s' \in S} \sum_{\bar{o}' \in O} \Pr(s', \bar{o}' \mid s, \bar{\pi}(\bar{\bar{h}}))$$

□

Proposition 5. *The SDec-POMDP and k-steps delayed are equivalent.*

A.2 Dec-POMDP-Com

The *Dec-POMDP-Com* [9] extends explicit communication to the Dec-POMDP by including an alphabet of possible messages Σ and communication cost function C_Σ. For a specified cost, each agent takes a communication action after their control action, which under the instantaneous broadcast communication assumption results in all other agents receiving an additional observation. Unlike the SDec-POMDP, agents in a Dec-POMDP-Com are never entirely restricted from communication. The algorithm designer may centralize the agents in a Dec-POMDP-Com (for cost) at will.

Lemma 10. *SDec-POMDP and Dec-POMDP-Com models are equivalent.*

Proof. Demonstrate that $X_{\text{Dec-POMDP-Com}} \leq X_{\text{SDec-POMDP}}$ and 2. $X_{\text{SDec-POMDP}} \leq X_{\text{Dec-POMDP-Com}}$

1. $X_{\text{Dec-POMDP-Com}} \leq X_{\text{SDec-POMDP}}$
Reference $X_{\text{Dec-POMDP-Com}} \leq_p X_{\text{Dec-POMDP}}$ [27] and $X_{\text{Dec-POMDP}} \leq X_{\text{SDec-POMDP}}$ in proof of Lemma 4.

2. $X_{\text{SDec-POMDP}} \leq X_{\text{Dec-POMDP-Com}}$
Reference $X_{\text{SDec-POMDP}} \leq X_{\text{Dec-POMDP}}$ in proof of Lemma 4 and $X_{\text{Dec-POMDP}} \leq_p X_{\text{Dec-POMDP-Com}}$ [27].

□

Lemma 11. *SDec-POMDP and Dec-POMDP-Com model-policy structures are equivalent.*

Proof. Demonstrate that $XY_{\text{Dec-POMDP-Com}} \leq XY_{\text{SDec-POMDP}}$ and 2. $XY_{\text{SDec-POMDP}} \leq XY_{\text{Dec-POMDP-Com}}$

1. $XY_{\text{Dec-POMDP-Com}} \leq XY_{\text{SDec-POMDP}}$
Reference $XY_{\text{Dec-POMDP-Com}} \leq_p XY_{\text{Dec-POMDP}}$ [27] and $XY_{\text{Dec-POMDP}} \leq XY_{\text{SDec-POMDP}}$ in proof of Lemma 5.

2. $XY_{\text{SDec-POMDP}} \leq XY_{\text{Dec-POMDP-Com}}$
Reference $XY_{\text{SDec-POMDP}} \leq XY_{\text{Dec-POMDP}}$ in proof of Lemma 5 and $XY_{\text{Dec-POMDP}} \leq_p XY_{\text{Dec-POMDP-Com}}$ [27].

□

Lemma 12. *SDec-POMDP and Dec-POMDP-Com model-policy-objective structures are equivalent.*

Proof. Demonstrate that $\forall \bar{h}\ V_{XY_{\text{Dec-POMDP-Com}}}(\bar{h}, b_0) = V_{XY_{\text{SDec-POMDP}}}(\bar{h}, b_0)$

Reference $\forall \bar{h}\ V_{XY_{\text{Dec-POMDP}}}(\bar{h}, b_0) = V_{XY_{\text{SDec-POMDP}}}(\bar{h}, b_0)$ in proof of Lemma 6.

□

Proposition 6. *The SDec-POMDP and Dec-POMDP-Com are equivalent.*

A.3 Semi-Decentralized Benchmarks

A.3.1 SDec-Tiger. Consider the following semi-decentralized variation on the Dec-Tiger benchmark [18], depicted in Figure 5. SDec-Tiger has 2 states, 3 actions, and 2 observations. Two cooperative agents stand behind two doors. One door leads to a room containing a tiger while the other leads to a room containing treasure. Each agent has three actions: opening the left door OL, opening the right door OR, and listening L. The problem reward function is fully described in table 3. The problem resets when any door is opened; the probability that the tiger is behind the left door TL and that the tiger is behind the right door TR both become 0.5. Joint actions that do not open doors do not affect the underlying state. Agents have an 75% chance of accurately communicating their action observation histories if they both listen. After taking an action, each agent receives one of two observations, hear tiger on left HL or hear tiger on right HR. Listening to either door gives an 0.75 probability of returning the correct observation. Opening a door and resetting the problem results in both agents receiving either observation with a 0.5 probability. If one agent opens a door while the other listens, the listening agent will not know the problem has been reset.

Table 3: SDec-Tiger Rewards (TL, TR)

$\bar{a}$	OL	OR	L
OL	(-50, 20)	(-100, -100)	(-101, 9)
OR	(-100, -100)	(20, -50)	(9, -101)
L	(-101, 9)	(9, -101)	(-2, -2)

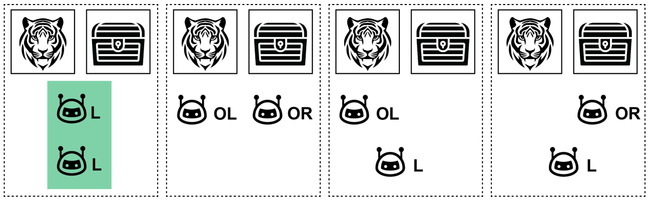

Figure 5: Illustration of four of nine possible joint actions for SDec-Tiger. Agents communicate their observation histories with some probability when they listen to the same door (in green).

A.3.2 SDec-FireFighting. Consider the following semi-decentralized variation on the FireFighting benchmark [23], depicted in Figure 6. SDecBoxPushing ($n_f = 3, n_h = 3$) has 432 states, 3 actions, and 2 observations. 2 agents are tasked with addressing a line of n_h houses, each with fire severity status f in range $[0, n_f]$ initially sampled from a uniform distribution. Each agent selects a house to suppress at each time-step. Single agent suppression decrements f by 1 with probability 1.0 if all adjacent houses have $f = 0$ or with probability 0.6 otherwise. Dual agent suppression resets f to 0. Agents only communicate their observation histories if they suppress the same house. A house without a firefighter present increments its f by 1 with probability 0.8 if an adjacent house has $f > 0$ or with probability 0.4 if all adjacent houses have $f = 0$. A house with $f = 0$ will catch fire (increment f by 1) with probability 0.8 if an adjacent house has $f > 0$. Each agent observes their selected house to be on fire or not with probability 0.2 if $f = 0$, probability 0.5 if $f = 1$, and probability 0.8 if $f \geq 2$. The cooperative agent team is rewarded the summation of $-f$ across all n_h following action selection.

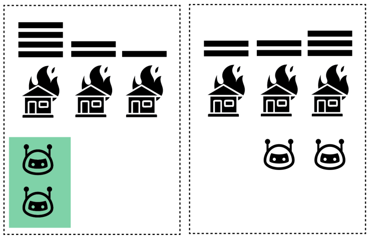

Figure 6: Illustration of two of nine joint actions in SDec-FireFighting ($n_h = 3, n_f = 4$). Agents communicate when they suppress the same house, shown in green.

A.3.3 SDec-BoxPushing. Consider the following semi-decentralized variation on the BoxPushing benchmark [26], depicted in Figure 7. SDecBoxPushing has 100 states, 4 actions, and 5 observations. n agents cooperate to push small and large boxes into an established goal area. Each agent can choose to rotate left, rotate right, move forward, or remain in place. Rotation and movement actions are successful with a 0.9 probability, otherwise the agent remains in place. Forward movement while facing a box will cause the box to translate one unit in the direction of movement, if permissible. A single agent can push a small box but two agents must act in tandem to push a large box. Movement into a wall, or into a large box with one agent, will result in remaining in place. Each agent correctly observes what is in front of them: a wall, a small box, a large box, an empty space, or another agent. Agents share their observation histories when simultaneously occupying one or more established communication grid squares. Agents receive a $-0.1n$ reward after each time-step, a -5 reward for each agent that moves into a wall, a +10 reward for each small box pushed into the goal area, and a +100 reward for each large box pushed into the goal area. The problem resets as soon as any box reaches the goal state. We adopt the environment configuration depicted in Figure 7.

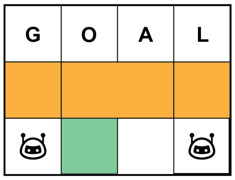

Figure 7: Illustration of the SDec-BoxPushing environment. Agents communicate when they are both in the green square.

A.3.4 SDec-Mars. Consider the following semi-decentralized variation on the Mars benchmark [2], depicted in Figure 8. SDec-Mars has 256 states, 6 actions, and 8 observations. Each of two agents can choose to move north, south, east, and west in a 2x2 grid, or conduct an experiment of choice (drilling or sampling) in their current location. Two grid squares are intended to be sampled by one agent and the other two grid squares require that both agents drill simultaneously. Each agent accurately observes their location in the 2x2 grid and whether an experiment has already been performed there. Agents share their observation histories while simultaneously occupying a designated communication grid square. The problem resets once an experiment is performed in all four grid squares. The cooperative agent team receives a large positive reward for drilling a drill site, a small positive reward for sampling a sample site, a large negative reward for drilling a sample site, and a small positive reward for sampling a drill site. Attempting a second experiment on the same site incurs a small negative reward.

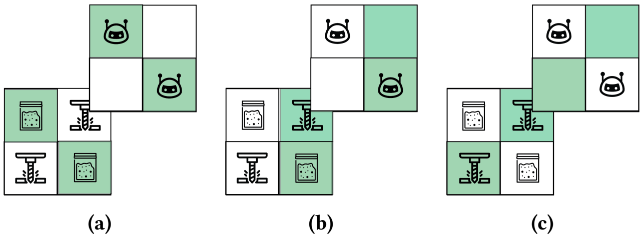

Figure 8: Illustration of the SDec-Mars environment. Agents in (a), the survey site beacon scenario, can communicate when co-located or adjacent to the same not yet surveyed survey site. Agents in (b), the right band rendezvous scenario, communicate alongside the right side of the grid when at least one site remains incomplete. Agents in (c), the drill site beacon scenario, can communicate when co-located or adjacent to the same not yet drilled drill site.

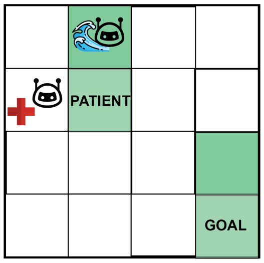

Figure 9: Illustration of the MaritimeMEDEVAC environment. Agents communicate when they are positioned adjacent to patient pickup and drop-off sites.

A.3.5 MaritimeMEDEVAC. We introduce a new semi-decentralized MEDEVAC benchmark involving a 4 × 4 gridworld archipelago, depicted in Figure 9. MaritimeMEDEVAC has 512 states, 3 actions, and 2 observations. Two agents, a medical aircraft and a transport ship, must retrieve a patient at $(1, 1)$ and deliver them to a hospital at $(3, 3)$. At each time-step, agents selects one of Wait, Advance, Exchange. Advance moves one cell toward the current target (patient if $carry = 0$, else hospital), succeeding independently with probability 0.95 for the aircraft and 0.85 for the boat. Wait leaves the agent position unchanged. Exchange attempts a *joint* pickup/drop that succeeds with probability 0.95 when both agents are at the corresponding site (toggling $carry$). Each agent receives a binary observation indicating whether it is *at-target* (patient if $carry = 0$, hospital if $carry = 1$) or *not*. The team incurs -0.3 per step, issuing Exchange away from $\{(1,1),(3,3)\}$ costs -1.0, a solo pickup or solo drop-off incurs -6.0, and joint pickup or drop-off grants +5.0 and +12.0 respectively. Agents share observation histories in a subset of "one-arrived, one-not" states: at the *patient* when the aircraft is at $(1,1)$ and the boat remains at $(1,0)$ with $(carry = 0)$, and at the hospital when one agent is at $(3,3)$ and the other at $(3,2)$ with $(carry = 1)$ (both permutations). Agents cannot communicate in any other states.

A.4 Code

Results for semi-decentralized benchmarks may be reproduced at: https://github.com/csapidus/RSSDA.git