Approximate Heuristic Search for Semi-Decentralized Systems

Mahdi Al-Husseini[1] , **Kyle H. Wray**[2] , **Isaac R. Ward**[1] and **Mykel J. Kochenderfer**[1]
[1] Stanford University, Stanford, CA
[2] Northeastern University, Boston, MA
mah9@stanford.edu, k.wray@northeastern.edu, irward@stanford.edu, mykel@stanford.edu

Abstract

Achieving optimal coordination in multiagent systems involves a trade-off between intractable centralized planning and suboptimal decentralized execution. We bridge this gap by introducing Approximate Recursive Small Step-Semi-Decentralized A* (RS-SDA*), a tree search algorithm that exploits time-varying centralization during periods of available communication conditioned on the environment or joint actions. By interleaving offline planning with online search and using relaxed heuristics, RS-SDA* achieves high solution quality with reduced computational overhead. SDec-POMDP benchmark experiments show that Approximate RS-SDA* often finds near exact optimal solutions in less than 1% of the time required by exact algorithms. We show scalability in six labyrinth environments both deterministic and stochastic state transitions, and demonstrate real-world feasibility with a multi-drone search-and-rescue simulation in the DARPA Subterranean Challenge cave system.

1 Introduction

Many real-world multiagent systems must operate rapidly in challenging environments characterized by communication uncertainty. For example, drones jointly navigating subterranean cave systems may communicate only upon establishing line-of-sight (Fig. 1), while medical aircraft coordinating patient movement in tactical environments may be subject to adversarial jamming. Effective coordination in these scenarios requires agents to balance autonomous decision-making with the ability to synchronize beliefs when possible. The Decentralized Partially Observable Markov Decision Process (Dec-POMDP) provides a rigorous formalism for distributed multiagent planning [Becker *et al.*, 2004; Bernstein *et al.*, 2002; Amato *et al.*, 2013]. While the standard Dec-POMDP formulation captures implicit communication, it does not model the explicit communication dynamics of many deployed systems, where information exchange is probabilistic and constrained by range, latency, or cost.

To overcome this gap, recent research has extended the Dec-POMDP to incorporate explicit communication costs [Goldman and Zilberstein, 2008], delays [Oliehoek and Spaan, 2012; Oliehoek, 2013], or intermittency [Zhang *et al.*, 2024]. In this work, we focus on semi-decentralized systems characterized by intermittent connectivity (formally modeled as "sojourn" or semi-Markov information sharing). In this paradigm, agents operate independently for variable durations before re-synchronizing their histories based on specific triggers. While this offers a realistic model for systems that drift in and out of connectivity, it presents a formidable computational challenge: the planning algorithm must reason over a joint belief space that stochastically expands and collapses, rendering exact solution methods intractable for non-trivial horizons. We assume the model dynamics, to include communication, are known a priori and therefore primarily concerned with *planning* rather than *learning*.

Existing approaches commonly trade scalability for

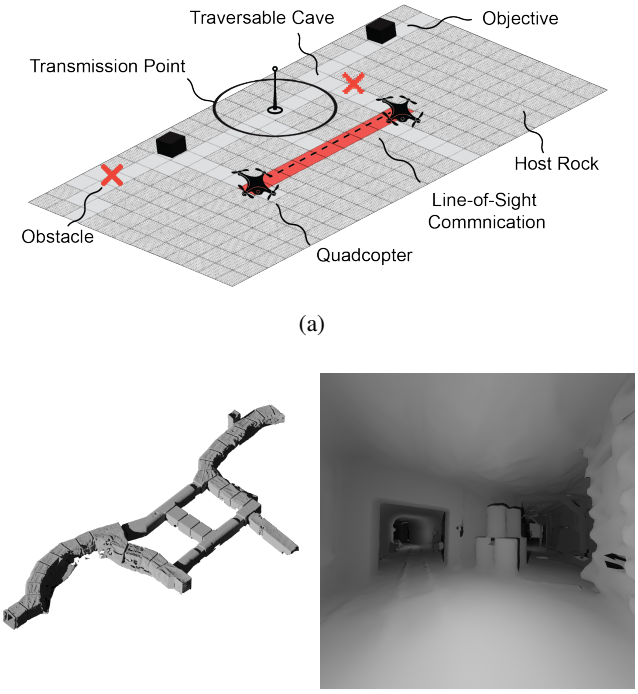


Figure 1: A semi-decentralized multiagent subterranean objective finding scenario with line-of-sight restrictions on communication.

optimality. Exact algorithms like Recursive Small-Step Semi-Decentralized A* (RS-SDA*) provide theoretical guarantees but struggle to scale beyond small benchmarks due to double-exponential complexity. Conversely, scalable approximations like Approximate RS-MAA* [Koops *et al.*, 2024] are limited to fully decentralized execution and cannot efficiently exploit periods of connectivity. We therefore introduce Approximate RS-SDA*, an approximate heuristic search algorithm for semi-decentralized systems. Our approach interleaves offline planning with online search, effectively "collapsing" the joint belief state during communication windows. We also apply a progress-based pruning metric that accounts for probability mass across both centralized and decentralized policy components and various heuristic relaxation techniques.

Contributions

- We introduce Approximate RS-SDA*, a heuristic search algorithm that scales to large semi-decentralized problems by interleaving offline planning with online search, pruning the priority queue, and relaxing heuristics.
- We validate that Approximate RS-SDA* regularly finds near optimal solutions in less than 1% of the time required by exact solvers on standard benchmarks and both deterministic and stochastic labyrinth problems, and conduct ablation studies.
- We demonstrate real-world feasibility by executing unmanned multiagent path planning under line-of-sight communication constraints in two high-fidelity, three-dimensional DARPA Subterranean Challenge simulation environments.

Ultimately, we present a scalable algorithmic framework for multiagent control in probabilistic communication environments, bridging the gap between theoretical semi-decentralized models and deployable autonomous systems.

2 Preliminaries

We formulate the problem as a semi-decentralized partially observable Markov decision process (SDec-POMDP) [Al-Husseini *et al.*, 2026]. The SDec-POMDP is a model for cooperative multiagent systems subject to probabilistic communication constraints, unifying the standard Dec-POMDP and MPOMDP frameworks into a single formalism.

2.1 SDec-POMDPs

The *semi-decentralized partially observable Markov decision process* (SDec-POMDP) is a stochastic, semi-decentralized multiagent model for sequential decision-making under partial observability and probabilistic communication characterized by tuple $\langle I, S, \bar{A}, \bar{\mathcal{O}}, T, O, R, F\rangle$, 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,
- $\bar{\mathcal{O}} = \times_i \mathcal{O}_i$ is a finite set of joint observations, and
- $T : S \times \bar{A} \times S \to [0, 1]$ is a state transition function where $T(s' \mid s, \bar{a})$ is the probability of transitioning into state s' given joint action $\bar{a}$ being performed in state s,
- $O : \bar{\mathcal{O}} \times S \times \bar{A} \to [0, 1]$ 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',
- $R : S \times \bar{A} \to \mathbb{R}$ is a reward function such that $R(s, \bar{a})$ is the immediate reward for performing joint action $\bar{a}$ in state s, and
- $F(\cdot \mid s, \bar{a})$ is the *communication sojourn distribution*. Each agent has a nonnegative sojourn communication time $\tau_i \in \mathbb{R}_{\geq 0}$ denoting the time until it returns to an information-sharing state, conditioned on the current state s and joint action $\bar{a}$.

2.2 Intermittent Connectivity Dynamics

The defining feature of the SDec-POMDP is the explicit modeling of intermittent connectivity via the sojourn random variable τ. Each agent in the system toggles stochastically between two information regimes:

Communication Epochs ($\tau = 0$): An agent enters a communicating state when its sojourn time resolves to zero. This may involve communication with a global *blackboard* [Erman *et al.*, 1980; Craig, 1988], or alternatively, with one of potentially many communicating subsets of agents. During a communication epoch, agents share histories and take joint actions. If $\tau_i = 0, \forall i \in I$, we collapse the distributed set of individual agent beliefs into a single shared joint belief.

Decentralized Epochs ($\tau > 0$): An agent operates in a decentralized state when its sojourn time is greater than zero. The agent relies on local action observation history at timestep t, $h_{i,t} = (a_{i,0}, o_{i,1}, \ldots, a_{i,t-1}, o_{i,t})$, possibly supplemented by prior information exchange with other agents, and acts per some decentralized policy.

This structure generalizes standard models: an SDec-POMDP with fixed $\tau_i = 0, \forall i \in I$ reduces to an MPOMDP, while $\tau_i \to \infty, \forall i \in I$ reduces to a Dec-POMDP.

2.3 Objective

The *SDec-POMDP objective* is to maximize expected reward by finding a policy encoding a favorable communication structure. Selector functions (f, g, h) govern how memories, actions, and observations inform both agent memories $\bar{m}$ and "blackboard" memory m_c. We seek an optimal blackboard policy π_c and local agent policies $\bar{\pi}$ that maximize the value function, where b^0 is the initial belief:

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

Recursive expansion $V^{\pi_c,\bar{a}}(m_c, \bar{m}, s)$ sums the immediate reward and the discounted future value across all probabilistic outcomes of communication and state transitions. This formulation relies on policy probability function $\psi(\bar{a} \mid m_c, \bar{m})$,

which selects joint actions by toggling between the centralized blackboard policy and local agent policies depending on $\bar{\tau}$. The evolution of the system's information state is governed by the memory update functions $\eta_c(m_c')$ and $\eta(m_i')$, which determine the next memory states for the blackboard and individual agents, respectively. These update functions are conditioned on selector functions (f, g, h) that act as gates—either propagating observations and actions to the shared blackboard when the communication sojourn time $\tau = 0$, or restricting them to local histories when $\tau > 0$.

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

3 Algorithm Design

Approximate RS-SDA* extends exact-optimal RS-SDA* through three approximation techniques: *interleaved offline planning and online search* via softmax-weighted voting, stage-wise *progress pruning*, and *tail value heuristic approximation* through recursive depth limits. We also consider and implement a computationally lightweight lossy clustering strategy that groups agent histories based on observation windows of size k.

3.1 Communication and Belief Splitting

RS-SDA* partitions the posterior belief b at each stage σ into decentralized b_{dec} and centralized b_{cen} components. By doing so, RS-SDA* decouples the portion of the policy that leads to a communication epoch from the expensive history-dependent decentralized search tree. The centralized probability mass is instead conditioned on the resulting joint belief.

Define a generalized trigger function Φ as the evaluation of communication sojourn distribution F within the complete transition context $\mathcal{C} = (s, \bar{a}, s', \bar{o}', b')$. Extending the communication model to $\mathcal{C}$ allows Φ to support any semi-decentralized scenario. Communication may be conditioned on:

- States: $\Phi(\mathcal{C}) = \mathbb{I}(s' \in \mathcal{S}_{\text{sync}})$
- Actions: $\Phi(\mathcal{C}) = \mathbb{I}(\bar{a} \in \mathcal{A}_{\text{sync}})$
- Observations: $\Phi(\mathcal{C}) = \mathbb{I}(\bar{o}' \in \mathcal{O}_{\text{sync}})$
- Beliefs: $\Phi(\mathcal{C}) = \mathbb{I}(\mathcal{H}(b^{\bar{a}\bar{o}'}) > \epsilon)$

or some complex combination thereof, where $\mathcal{H}(b)$ is the entropy of belief b compared against some threshold ϵ. This allows RS-SDA* to support communication conditioned on state-based triggers (spatial zones), action-based triggers (communication events), observation-based triggers (environment signals), and belief-based triggers (uncertainty thresholds) all within a single solver architecture.

Further define decentralized mass fraction ω as a function of the Bayesian posterior belief $b^{\bar{a}\bar{o}}$:

$$
\omega(b^{\bar{a},\bar{o}'}) = \sum_{s'\in\mathcal{S}} b^{\bar{a},\bar{o}'}(s')(1 - \Phi(\mathcal{C})).
$$

Recall that the belief over successor states s' is updated using the set of agent histories $\bar{h}$ of actions taken and observations received:

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

where η is a normalizing constant equal to $\Pr(\bar{o}' \mid b, \bar{a})^{-1}$, and $\Pr(\bar{o}' \mid b, \bar{a}) = \sum_{s'\in S} O(\bar{o}' \mid s', \bar{a}) \sum_{s\in S} T(s' \mid s, \bar{a}) b(s)$. Belief $b^{\bar{a}\bar{o}'}$ is then generated by applying the belief update equation $\forall s' \in S$. $b^{\bar{a}\bar{o}'}$ is partitioned into decentralized posterior $b^{\bar{a}\bar{o}'}_{dec}$ and centralized $b^{\bar{a}\bar{o}'}_{\text{cen}}$:

$$
b^{\bar{a}\bar{o}'}_{\text{dec}}(s') = \begin{cases} \frac{b^{\bar{a}\bar{o}'}(s')(1-\Phi(\mathcal{C}))}{\omega(b^{\bar{a}\bar{o}'})} & \text{if } \omega(b^{\bar{a}\bar{o}'}) > 0 \\ 0 & \text{otherwise} \end{cases}
$$

$$
b^{\bar{a}\bar{o}'}_{\text{cen}}(s') = \begin{cases} \frac{b^{\bar{a}\bar{o}'}(s')\Phi(\mathcal{C})}{1-\omega(b^{\bar{a}\bar{o}'})} & \text{if } \omega(b^{\bar{a}\bar{o}'}) < 1 \\ 0 & \text{otherwise.} \end{cases}
$$

We then apply a modified version of the RS-SDA* Bellman equation:

$$
\begin{aligned}
Q(b, \bar{a}) = &R(b, \bar{a}) + \sum_{\bar{o}'\in\mathcal{O}} \Pr(\bar{o}' \mid b, \bar{a}) \\
&\Bigg[\underbrace{\omega(b^{\bar{a}\bar{o}'}) V_{\text{dec}}(b^{\bar{a}\bar{o}'}_{\text{dec}})}_{\text{Decentralized}} + \underbrace{(1 - \omega(b^{\bar{a}\bar{o}'})) V_{\text{cen}}(b^{\bar{a},\bar{o}'}_{\text{cen}})}_{\text{Centralized}} \Bigg].
\end{aligned}
$$

3.2 Interleaved Offline Planning and Online Search

To scale beyond the small horizons of exact solvers, we interleave offline policy generation with online search, a technique typically applied in receding horizon control for POMDPs [Paquet, 2005; Ross *et al.*, 2008]. Approximate RS-SDA* expands the policy tree to some stage σ, executes a selected partial policy φ up to a communication epoch, and then re-plans from the resulting posterior belief b'. This interleaving, visualized in Figure 2, mitigates the double-exponential branching factor of Dec-POMDPs and enables aggressive pruning during the offline phase. As will be demonstrated, interleaving performs remarkably well in problems with stochastic observation models such as SDEC-TIGER and NOISY LABYRINTH.

We employ an early-commitment mechanism to avoid exhaustive search when the planner identifies a strong consensus for centralization. We periodically inspect the top-k nodes of the priority queue, computed component-wise over each stage. A centralization score vector $\bar{S}$ is found by taking the Softmax-weighted average of the binary centralization

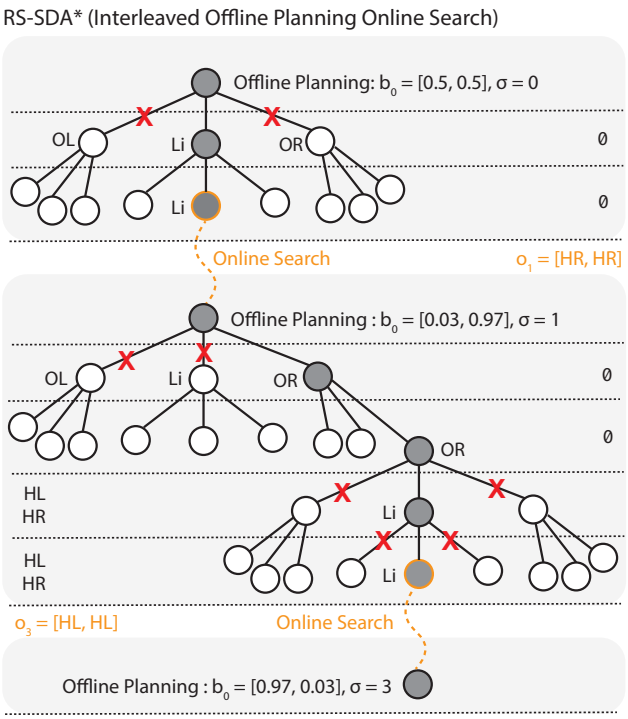


Figure 2: Illustrating Approximate RS-SDA* interleaved offline planning and online search applied to SDEC-TIGER using mixed component policies through stage $\sigma = 2$. OL, OR, and Li are agent actions and HL and HR are agent observations.

vectors $\bar{v}_\varphi$ of these k nodes, weighted by their heuristic values $V(\varphi)$ and a temperature parameter τ_{temp}:

$$w_\varphi = \frac{\exp(V(\varphi)/\tau_{temp})}{\sum_{\varphi' \in \text{Top-}k} \exp(V(\varphi')/\tau_{temp})}, \quad \bar{S} = \sum_{\varphi \in \text{Top-}k} w_\varphi \cdot \bar{v}_\varphi$$

If score S_σ exceeds a confidence threshold δ, we discard all partial policies in the queue that would result in decentralization at stage σ. This collapses the search space onto the consensus. The search continues only until the leading policy is fully expanded up to the consensus stage σ, at which point it is executed online. This ensures that the planner commits to certain communication events without wasting further computational budget on unlikely decentralized alternatives.

3.3 Progress-Based Pruning with Centralization

The solver allocates computational resources efficiently across the belief space by implementing a stage-wise progress constraint inspired by Koops *et al.* [2024] and adapted for semi-decentralized settings. The centralized policy component is treated as an additional agent and the stage-wise budget L is therefore split across $n+1$ groups. This helps ensure that search allocates resources proportionally between resolving centralized triggers and specifying decentralized agent actions. The progress of a partial policy $\text{prog}(\varphi)$ follows:

$$\text{prog}(\varphi) = \sigma(\varphi) \cdot L + \kappa \cdot \frac{L}{n+1} + c + p \cdot \left(\frac{L}{n+1} - |C_g^{\sigma(\varphi)}| \right)$$

where κ is the number of groups fully specified in the current stage $\sigma(\varphi)$, g is the group being expanded, c is the number of g's clusters that have already been assigned an action, p is the cumulative probability mass of c, and $|C_g^{\sigma(\varphi)}|$ is the total number of clusters associated with g. Critically, $\text{prog}(\varphi)$ guarantees that a fully specified policy will be found within $h \cdot L$ expansions, so long as $L \geq (n+1)\max(|C|)$.

3.4 Heuristic Approximation

Whereas exact optimal search algorithms like RS-SDA* and RS-MAA* benefit from tight, admissible heuristics to enable pruning, approximate methods benefit from computationally efficient guidance that rapidly steers the search towards promising regions of policy space. The objective is to find high-quality policies for long horizons quickly. To do so, we adapt *tail value approximation* for semi-decentralized execution. This technique splits the planning horizon into a solved "head" and an approximated "tail", the latter estimating value using relaxed heuristics like QMDP or POMDP. The depth of the head is represented by hyperparameter r. We partition this approximation into separate calculations for the centralized and decentralized policy components, weighted by the probability mass ω in each partition. Because QMDP assumes full state observability, it aligns closely with the information available to the centralized partition, rendering the heuristic tight while remaining computationally inexpensive.

4 Experiments

We evaluate Approximate RS-SDA* against RS-MAA*, Approximate RS-MAA*, and RS-SDA*. Collectively, these search techniques comprise the current state-of-the-art for decentralized and semi-decentralized multiagent planning. RS-MAA* is the decentralized lower bound and RS-SDA* with all states and joint-actions centralized is the centralized upper bound for subsequent experiments. This section considers three semi-decentralized benchmarks from the literature, including both state and action-triggered synchronization. We also present an ablation on combinations of L, r, interleaving, and heuristic across six labyrinth problems shown in Figure 3, with and without stochastic observation dynamics.

Setup. All experiments were conducted on an AMD Ryzen 9 9900X3D 12-Core Processor (4400 MHz), with timeout occurring at 20 minutes. Approximate RS-SDA* is implemented in Python 3, and code is available at https://github.com/mahdial-husseini/RSSDA to support reproducibility. We compare directly against the RS-MAA* implementation provided by Koops *et al.* [2024].

Semi-Decentralized Benchmarks. We evaluate RS-SDA* on three SDec-POMDP benchmarks: SDEC-TIGER, SDEC-MARS, and MARITIMEMEDEVAC [Al-Husseini *et al.*, 2026]. SDEC-TIGER uses action-based communication triggers and has stochastic observation dynamics; interleaved results are therefore presented in Table 1 as average values with standard errors across 1000 trials. SDEC-MARS and MARITIMEMEDEVAC use state-based communication triggers.

Labyrinths. We also test using LABYRINTH, a two-agent graph search problem, and NOISY LABYRINTH, a more challenging variant with stochastic observations. In LABYRINTH,

	decentralized				*semi-decentralized*				*centralized*	
	App. RS-MAA*		Exact RS-MAA* (lower bound)		App. RS-SDA* **(Our Approach)**		Exact RS-SDA*		Exact RS-SDA* (upper bound)	
h	value	time	value	time	value	time	value	time	value	time
SDEC-TIGER										
10	13.57	2	15.18	628	35.01 ± 1.33	<1	34.72	1	60.51	1
12	20.76	2	20.76	762	43.37 ± 1.40	<1	42.25	1	73.39	1
15	25.95	2	-	TO	53.55 ± 1.65	<1	53.55	1	92.67	1
20	28.01	3	-	TO	73.64 ± 1.86	<1	72.40	4	125.19	1
SDEC-MARS (Right Band Rendezvous)										
6	18.62	2	18.62	5	20.06	<1	20.06	2	20.07	<1
7	20.90	2	20.90	21	21.17	1	21.17	9	21.17	<1
8	22.48	2	22.48	40	23.65	1	23.65	28	23.65	<1
9	24.31	3	24.32	160	25.21	2	-	TO	25.70	<1
10	26.31	4	-	TO	28.28	2	-	TO	28.61	<1
12	32.09	4	-	TO	36.11	3	-	TO	36.11	<1
15	37.72	6	-	TO	42.63	5	-	TO	43.02	<1
20	49.37	13	-	TO	57.02	7	-	TO	57.52	<1
30	77.47	21	-	TO	86.38	10	-	TO	86.65	<1
40	100.02	25	-	TO	115.66	13	-	TO	115.85	3
50	122.56	37	-	TO	144.81	15	-	TO	144.97	4
100	234.06	89	-	TO	288.86	30	-	TO	288.97	9
MARITIMEMEDEVAC										
5	3.46	<1	3.46	1	3.48	<1	3.48	3	3.50	<1
6	3.18	<1	3.18	1	3.20	<1	3.20	28	3.20	<1
7	3.25	<1	3.27	2	6.36	<1	6.36	33	6.62	<1
8	8.03	<1	8.03	156	10.61	<1	-	TO	10.88	<1
9	10.79	<1	-	TO	12.74	<1	-	TO	13.01	<1
10	12.15	<1	-	TO	13.35	<1	-	TO	13.61	<1

Table 1: Approximate RS-SDA* performance on semi-decentralized benchmarks. TO denotes timeout (>1200s). App. RS-SDA* for SDEC-TIGER was run with 1,000 seeds.

agents execute a search and return mission: they navigate a graph to locate a hidden target node selected uniformly from the environment and then return to the start node with the target information. The agents receive a step cost of -1 and a terminal reward of $+100$ if either returns to the start node having found the target. Agents can only share information and take joint actions when within line-of-sight. NOISY LABYRINTH is a high-stakes perception and drilling problem. Agents are equipped with noisy sensors that return binary observations correlated with the target's presence at the current node. The action space is augmented with a terminal DRILL action. Agents must locate the hidden target using uncertain sensor data and commit to drilling. A correct drill yields $+100$, while drilling at a non-target location incurs a catastrophic penalty of -200. Both problems are tested using the six generated labyrinth structures in Figure 3 and the 3D DARPA Subterranean Challenge cave sites *Tunnels* and *Chamber* in Figure 4.

Hyperparameters. We consider four hyperparameters associated with our approximation techniques: stage-wise iteration limit L, recursion limit r, sliding window length k, choice of heuristic, and whether to apply interleaved offline planning and online search. This is annotated in Tables 1 and 2 using $[r]$[heuristic choice]$[k]$, $[L]$, [TI1]. Not all problems benefit from sliding window clustering, and $[k]$ and [TI1] may be left blank.

Results. Experiments on SDec-POMDP benchmarks demonstrate that Approximate RS-SDA* overcomes the scalability limitations of exact algorithms while retaining solution quality. In SDEC-MARS for $h = 100$, our solver returns a value of 288.86 in just 30 seconds, closely matching the centralized upper bound of 288.97. By contrast, the fully decentralized Approximate RS-MAA* baseline requires 89 seconds to find a significantly lower value of 234.06, illustrating the inherent value of exploiting intermittent communication opportunities. In SDEC-TIGER, Approximate RS-SDA* with

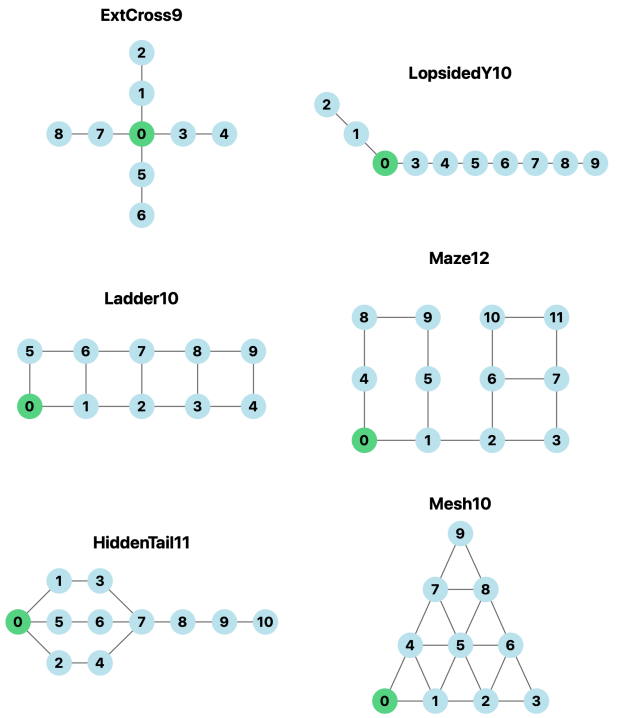


Figure 3: Six generated labyrinth structures used for Approximate RS-SDA* benchmarking. Both agents begin at the green start node.

interleaved offline planning and online search returns values that average close to the semi-decentralized exact optimal expected reward; however, each Approximate RS-SDA* run for h = 20 takes 0.01 s, whereas the exact optimal expected reward calculation requires 400 times that.

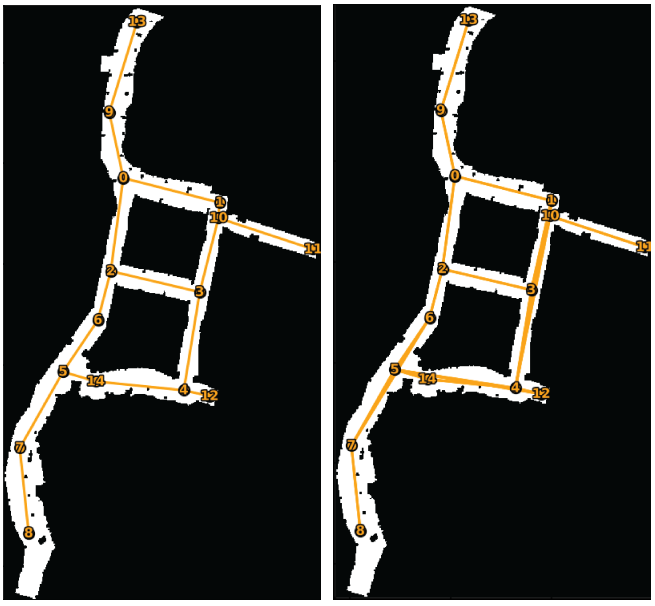
We further validate scalability on six graph-based labyrinth topologies, where we observe distinct performance trade-offs between the "Fast", "Accurate", and "Interleaved" algorithm variants. In deterministic environments, the "Fast" variant (using QMDP) is often sufficient, finding near-optimal solutions in under one second (e.g., matching the optimal 96.75 in EXTCROSS9). However, complex topologies such as MESH10 for $h = 5$ require the "Accurate" variant to escape local optima, improving values from 74.44 to 85.67 with only a marginal increase in computation time. In the challenging Noisy Labyrinth variants, where offline planning with either heuristic consistently times out due to high branching factors, the "Interleaved" variant proves valuable. By deferring long-term decisions until after online short-term partial policy execution, interleaving successfully achieves high-quality results (e.g., 68.05 ± 16.06 in NOISY LADDER10 for $h = 8$) whereas both "Fast" and "Accurate" offline baselines time out.

5 Demonstration

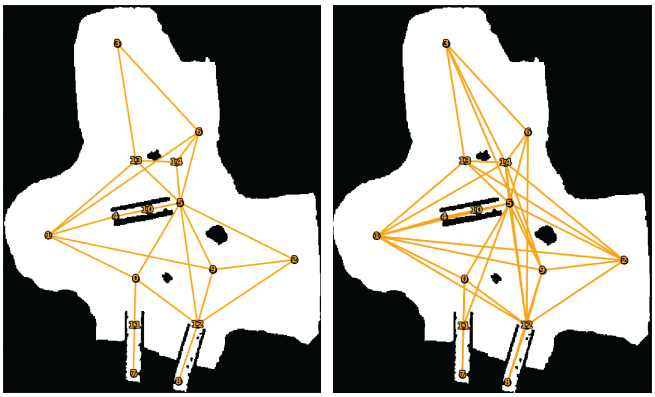
We use the *Prospector* simulator [Ward *et al.*, 2025a] to simulate caves from the DARPA Subterranean Challenge dataset, which features real world data from the Louisville Mega Cavern, a former limestone mine in Kentucky. Prospector remeshes the 3D data provided by DARPA, and provides sim-

	decentralized		*semi-decentralized*						*centralized*	
	RS-MAA*		Approximate RS-SDA* (**Our Approach**)						Exact RS-SDA*	
	Exact		*Fast*		*Accurate*		*Interleaved*		*Exact*	
	(lower bound)		1QMDP, 300		2HYB, 1000		1HYB, 300, TI1		(upper bound)	
h	value	time	value	time	value	time	avg. ± SE	time	value	time
Labyrinth ExtCross9										
6	71.25	15	84.38	<1	84.38	1	84.38	1	84.38	1
7	-	TO	84.25	<1	84.25	2	83.88	1	84.25	5
8	-	TO	96.75	<1	96.75	1	96.63	1	96.75	1
Labyrinth LopsidedY10										
5	40.78	<1	74.67	<1	74.67	<1	74.67	<1	74.67	<1
6	51.44	<1	85.67	<1	85.67	<1	85.67	<1	85.67	<1
7	51.00	<1	96.78	<1	96.78	<1	96.78	<1	96.78	<1
Labyrinth Ladder10										
5	40.78	<1	74.56	<1	74.56	<1	74.56	<1	74.67	<1
6	62.67	<1	85.56	<1	95.89	<1	84.11	<1	96.33	<1
7	62.33	<1	96.56	<1	96.44	<1	96.44	<1	96.78	<1
Labyrinth Maze12										
5	32.45	2	60.09	<1	60.09	<1	60.09	<1	60.18	1
6	59.35	2	77.64	<1	77.64	<1	77.73	1	86.90	1
7	-	TO	94.82	<1	94.82	<1	95.82	1	96.00	1
Labyrinth HiddenTail11										
4	36.80	<1	57.00	<1	57.20	<1	57.20	<1	57.20	<1
5	36.20	13	66.90	<1	66.90	<1	66.70	<1	66.90	1
6	65.90	2	86.10	<1	95.80	2	86.30	1	96.20	1
Labyrinth Mesh10										
5	52.00	<1	74.44	<1	85.67	1	85.44	3	85.67	3
6	73.89	6	74.22	<1	96.67	2	96.67	4	96.78	7
7	-	TO	85.22	<1	96.67	3	96.67	6	96.78	8
			1QMDP, 300		2HYB, 1000		1HYB, 1000, TI1			
Noisy Labyrinth ExtCross9										
7	-	TO	18.63	55	34.30	431	55.75 ± 12.26	26	-	TO
8	-	TO	19.19	44	16.68	670	56.33 ± 14.69	29	-	TO
9	-	TO	15.09	1145	18.55	216	58.19 ± 15.40	29	-	TO
Noisy Labyrinth LopsidedY10										
6	-	TO	21.73	5	32.87	68	54.92 ± 12.81	11	-	TO
7	-	TO	18.72	56	35.67	194	55.30 ± 15.18	11	-	TO
8	-	TO	-	TO	-	TO	53.57 ± 11.92	19	-	TO
Noisy Labyrinth Ladder10										
6	-	TO	50.25	25	21.61	572	44.77 ± 14.24	33	-	TO
7	-	TO	-	TO	-	TO	46.77 ± 17.27	39	-	TO
8	-	TO	-	TO	-	TO	68.05 ± 16.06	56	-	TO
Noisy Labyrinth Maze12										
6	-	TO	31.39	36	28.25	975	38.40 ± 13.92	40	-	TO
7	-	TO	35.73	120	-	TO	29.46 ± 15.23	61	-	TO
8	-	TO	-	TO	-	TO	39.71 ± 12.79	38	-	TO
Noisy Labyrinth HiddenTail11										
6	-	TO	26.41	121	-	TO	40.90 ± 13.09	77	-	TO
7	-	TO	29.82	1034	-	TO	29.35 ± 15.83	62	-	TO
8	-	TO	-	TO	-	TO	49.49 ± 9.20	96	-	TO
Noisy Labyrinth Mesh10										
6	-	TO	25.60	50	-	TO	48.67 ± 16.58	127	-	TO
7	-	TO	17.30	1192	-	TO	63.43 ± 8.23	145	-	TO
8	-	TO	-	TO	-	TO	76.78 ± 4.53	125	-	TO
			1QMDP, 300		2HYB, 1000		2HYB, 1000, TI1			
DARPA 3D Tunnels-10										
7	73.56	1	84.89	<1	84.89	<1	84.89	1	96.78	1
8	84.44	4	84.44	<1	84.00	1	84.44	1	96.78	1
9	84.33	18	95.89	<1	95.89	1	95.89	1	96.78	1
DARPA 3D Tunnels-15										
11	-	TO	79.43	5	86.71	25	79.00	31	-	TO
12	-	TO	86.43	5	86.50	29	86.29	34	-	TO
13	-	TO	86.50	5	93.50	34	91.50	34	-	TO
			1QMDP1, 300		2HYB, 1000		2HYB, 1000, TI1			
DARPA 3D Chamber-10										
7	-	TO	85.11	1	85.22	8	85.11	4	96.78	6
8	-	TO	96.22	1	96.22	8	96.11	5	96.78	6
9	-	TO	96.22	1	96.22	8	96.11	5	96.78	6
DARPA 3D Chamber-15										
9	-	TO	80.50	4	80.64	275	80.29	125	-	TO
10	-	TO	80.21	5	87.57	250	80.00	128	-	TO
11	-	TO	87.29	5	93.71	252	86.93	131	-	TO

Table 2: Approximate RS-SDA* performance on various semi-decentralized benchmarks. Interleaved results in problems with stochastic observations are averaged across 10 roll-outs for each possible target node. TO denotes timeout (>1200s).



(a) *Tunnels* traversability graph. (b) *Tunnels* coordination graph.

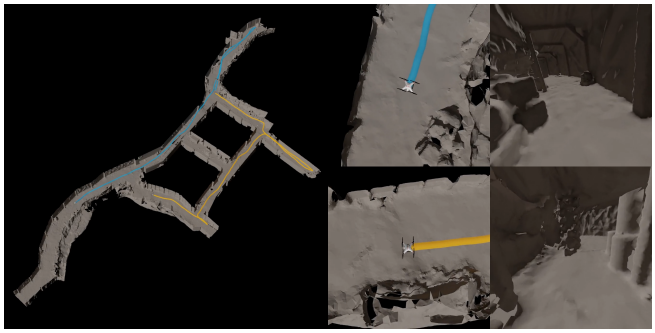


(c) *Chamber* traversability graph. (d) *Chamber* coordination graph.

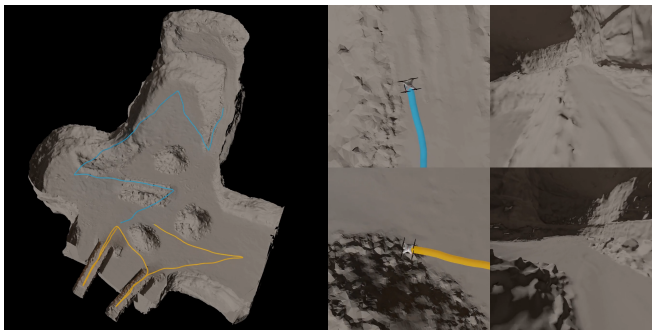
Figure 4: Fifteen node two-dimensional traversability and coordination graphs for two DARPA Subterranean Challenge cave sites: *Tunnels* and *Chamber*. The traversability graph has an edge between two nodes if they are spatial neighbors, and the coordination graph has an edge between two nodes if they have a line-of-sight connection.

ulations of other real-world caves using data gathered from caving expeditions.

The simulator provides a voxelized representation of these cave systems, manually marked waypoints of interest (Figure 4), accelerated collision and line-of-sight checking, and both 2D and 3D simulation modes. It also provides a low level Model Predictive Path Integral (MPPI) quadrotor controller [Ward *et al.*, 2025b] based on a 12 degree of freedom quadrotor dynamics model [Sabatino, 2015] which is implemented in JAX [Bradbury *et al.*, 2018] for fast, parallelized simulation.



(a) The first agent (blue) and the second agent (orange) traverse parallel shafts in the *Tunnels* cave. Both agents maintain a safe distance from the walls despite the narrow 3×3 meter shafts, avoiding collisions with struts, debris, and barrels.



(b) The first agent (blue) and the second agent (orange) perform simultaneous optimal exploration of the *Chamber* cave. The first agent explores a higher level platform, while the second agent navigates a group of rubble piles on the other side of the cave, illustrating decentralized 'divide and conquer' behavior.

Figure 5: Two-agent exploration of two cave systems using the *Prospector* simulator, featuring POV and top down views for each agent (right group), alongside an overhead view of the cave (left). Full videos are available as supplemental material.

Each quadrotor's state consists of its position, Euler angles (roll, pitch, yaw), linear velocity, and body-frame angular rates, while control inputs correspond to the angular velocities of the four rotors. The simulation time step is set to $\Delta t = 0.1\,\text{s}$. An MPPI controller [Williams *et al.*, 2017] controls the vehicle, which samples 2048 control sequences per timestep over a horizon of 32 steps. A low temperature parameter ($\lambda = 0.05$) biases samples toward high-reward trajectories, while uniform control noise $\sigma = 1.0$ encourages exploration.

A waypoint-following reward is employed to guide each quadrotor along the node sequence computed by the algorithm described in Section 3, combining a distance-to-goal term, a control-effort regularizer, and a collision penalty:

$$J_{\text{dist}} \;=\; w_d \sum_{t=0}^{T_s-1} \|\mathbf{p}_t - \mathbf{p}^\star\|_2^2 \,,$$

where $\mathbf{p}_t$ is the quadrotor's position, $\mathbf{p}^\star \in \mathbb{R}^3$ is the next waypoint's position, T_s is the number of states in the rollout, and $w_d = 1.0$ is the distance weight.

$$J_{\text{ctrl}} \;=\; w_u \sum_{t=0}^{T_a-1} \|\mathbf{u}_t\|_2 \,,$$

where $\mathbf{u}_t \in \mathbb{R}^4$ is the control input (rotor angular velocities), T_a is the number of actions in the rollout, and $w_u = 0.1$ is the control-effort weight.

$$J_{\text{col}} \;=\; c_{\text{col}}\, \mathbb{I}[\exists\, t \in \{0, \ldots, T_s - 1\} : \text{Coll}(\mathbf{p}_t, r_{\text{agent}})] \,,$$

where $\text{Coll}(\mathbf{p}, r_{\text{agent}})$ is true if the agent at position $\mathbf{p}$ intersects the environment within radius r_{agent}, $\mathbb{I}[\cdot] \in \{0, 1\}$ is an indicator function, and $c_{\text{col}} = 10^6$ is a collision penalty.

The total waypoint cost is $J = J_{\text{dist}} + J_{\text{ctrl}} + J_{\text{col}}$, and the reward is defined as the negative cost

$$r(\mathbf{x}_{0:T_s-1},\, \mathbf{u}_{0:T_a-1};\, \mathbf{p}^\star) \;=\; -\Big(J_{\text{dist}} + J_{\text{ctrl}} + J_{\text{col}}\Big).$$

All experiments are conducted with two agents operating simultaneously. We first consider the *Tunnels* environment (Figure 5a), corresponding to DARPA SubT Section 6. The policy achieves a 100% success rate over 10 randomized trials, with the quadrotors maintaining an average flight speed of $0.731\ \text{m s}^{-1}$. The entire map ($31 \times 59 \times 4$ meter extents, and an internal volume of 536 cubic meters) is explored in 20.01 seconds.

We next evaluate performance in the *Chambers* environment (Figure 5b), corresponding to DARPA SubT Section 5. The policy achieves a 100% success rate over 10 randomized trials, with the quadrotors maintaining an average flight speed of $0.775\ \text{m s}^{-1}$. The entire map ($41 \times 62 \times 11$ meter extents, and an internal volume of $11,492$ cubic meters) is explored in 26.10 seconds.

Overall, Approximate RS-SDA* prevents failures (collisions) whilst efficiently exploring these challenging 3D cave environments, demonstrating its efficacy as a semi-decentralized planning policy in a search-and-rescue context.

6 Conclusion

We present Approximate RS-SDA*, a scalable heuristic search algorithm that supports semi-decentralized problems. The results confirm that our solver finds near-optimal solutions orders of magnitude faster than exact solvers for many types of problems. The practical impact is evidenced by our demonstrations in high-fidelity DARPA Subterranean Challenge simulations, where the high-level algorithm helped efficiently navigate complex, communication-restricted cave sites. While promising, the current framework relies on predefined communication models and has not yet been evaluated on larger teams where the combinatorial belief space expands further. Future work will address these scalability constraints by considering coordination graphs for systems with three or more agents, exploring learning-based approaches for unknown or non-stationary communication dynamics, implementing learned heuristics, and investigating the theoretical interplay between sojourn communication and the sojourn control characteristic of semi-Markov decision processes.

References

[Al-Husseini *et al.*, 2026] Mahdi Al-Husseini, Kyle H Wray, and Mykel J Kochenderfer. A semi-decentralized approach to multiagent control. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2026.

[Amato *et al.*, 2013] Christopher Amato, Girish Chowdhary, Alborz Geramifard, N Kemal Üre, and Mykel J Kochenderfer. Decentralized control of partially observable markov decision processes. In *IEEE Conference on Decision and Control (CDC)*, pages 2398–2405. IEEE, 2013.

[Becker *et al.*, 2004] Raphen Becker, Shlomo Zilberstein, Victor Lesser, and Claudia V Goldman. Solving transition independent decentralized markov decision processes. *Journal of Artificial Intelligence Research*, 22:423–455, 2004.

[Bernstein *et al.*, 2002] Daniel S Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of markov decision processes. *Mathematics of Operations Research*, 27(4):819–840, 2002.

[Bradbury *et al.*, 2018] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.

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

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

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

[Koops *et al.*, 2024] Wietze Koops, Sebastian Junges, and Nils Jansen. Approximate Dec-POMDP solving using multi-agent A*. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2024.

[Oliehoek and Spaan, 2012] Frans Oliehoek and Matthijs Spaan. Tree-based solution methods for multiagent POMDPs with delayed communication. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 1415–1421, 2012.

[Oliehoek, 2013] Frans Adriaan Oliehoek. Sufficient plan-time statistics for decentralized POMDPs. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 302–308, 2013.

[Paquet, 2005] Sebastien Paquet. *Distributed decision-making and task coordination in dynamic, uncertain and real-time multiagent environments*. PhD thesis, Université Laval, 2005.

[Ross *et al.*, 2008] Stéphane Ross, Joelle Pineau, Sébastien Paquet, and Brahim Chaib-Draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Research*, 32:663–704, 2008.

[Sabatino, 2015] Francesco Sabatino. Quadrotor control: Modeling, nonlinear control design, and simulation. Master's thesis, KTH, School of Electrical Engineering (EES), Automatic Control, October 2015.

[Ward *et al.*, 2025a] Isaac Ronald Ward, Mark Paral, Kristopher Riordan, Michelle Ho, Maximilian Adang, and Mykel J Kochenderfer. Prospector: a cave simulation environment for rotorcraft. https://github.com/isaac-ward/prospector, 2025.

[Ward *et al.*, 2025b] Isaac Ronald Ward, Mark Paral, Kristopher Riordan, and Mykel J Kochenderfer. Improving the resilience of quadrotors in underground environments by combining learning-based and safety controllers. *arXiv preprint arXiv:2509.02808*, 2025.

[Williams *et al.*, 2017] Grady Williams, Andrew Aldrich, and Evangelos A Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40(2):344–357, 2017.

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