Survive and Thrive: Decentralized Multi-Agent Coordination under Attrition Risks

Nhat Nguyen, Duong Nguyen, Junae Kim, Gianluca Rizzo, and Hung Nguyen

Abstract—Decentralized online planning, such as decentralized Monte Carlo tree search (MCTS), is an attractive paradigm for cooperative multi-agent systems in information-gathering tasks. However, current MCTS algorithms implicitly assume that agents are always available and actively contributing throughout the mission. In realistic, dynamic, and volatile environments, agent attrition is common and can severely degrade performance. In this paper, we demonstrate that agent attrition can cause current decentralized MCTS methods to perform arbitrarily worse than the optimum, particularly in applications with submodular reward functions. To address this issue, we propose Attritable Monte Carlo Tree Search (A-MCTS), a decentralized MCTS algorithm that adapts quickly and efficiently to reductions in the set of active agents. Our key idea is to have each agent build its search tree using the global utility function while coordinating with teammates through a regret-matching algorithm. Our theoretical analysis and extensive simulations show that A-MCTS maintains effective coordination among agents, even in high-attrition environments. We evaluate our approach in different information-gathering problems by modeling realistic reference scenarios. Results highlight that A-MCTS substantially improves over the main competing methods regarding global utility, scalability, and robustness.

Index Terms—Monte-Carlo Tree Search, Multi-Agent Systems, Information Gathering, Autonomous Underwater Vehicles

1 Introduction

Cooperative multi-agent systems (MAS) consist of multiple autonomous agents collaborating to achieve a collective objective such as maximizing a global utility [1]. These agents are capable of communication and coordination, either directly or indirectly, to solve complicated tasks that exceed the capabilities of an individual agent. Notable applications of MAS include autonomous drone swarms for aerial surveillance or disaster relief operations, as well as teams of robots that collaborate to collect sensor data, explore unknown environments, or conduct routine inspections [2].

Decentralized planning is an emerging approach for cooperative MAS due to its scalability and robustness [3]. Unlike centralized methods, decentralized planners distribute computational effort among agents, enabling the system to scale effectively as the number of agents increases. This structure enhances responsiveness, allowing agents to react promptly to local changes, particularly in remote or post-disaster areas [4]. Additionally, decentralized approaches reduce communication overhead by relying primarily on local information exchange, making them well-suited for environments with bandwidth constraints or unreliable communication [5].

The main challenge in decentralized approaches for cooperative MAS is how to effectively coordinate agents' actions in a distributed manner – emerging from the goal of maximizing the global utility of the system. Algorithms based on Monte Carlo tree search (MCTS) are an appealing solution to address this challenge, due to their ability to efficiently explore long planning horizons, their anytime nature [6], and their excellent performance in decentralized settings [7]–[9]. Current decentralized MCTS (Dec-MCTS) methods typically have each agent optimize its action based on its *marginal contribution* rather than the global reward. This design promotes local alignment with global objectives to enhance coordination. On the other hand, existing methods also implicitly assume that all agents remain available and actively contribute throughout the mission. However, practical MAS deployments often face *attrition*—reductions in active agents due to hardware failures, energy depletion, or adversarial attacks [10]. In applications with submodular global reward structures (e.g., data collection tasks), Dec-MCTS methods can exhibit poor adaptability to attrition, resulting in suboptimal performance. This stems from the marginal contribution utility failing to reflect the absolute quality of actions. Thus, when attrition occurs, agents may inadvertently select actions that reduce global welfare while perceiving local improvements. Developing robust solutions for decentralized MAS planning capable of effectively handling agent attrition is, therefore, crucial for their successful deployment in practical applications.

The widely-used heuristic to handle attrition is to periodically reset the agents' experiences, and restart the planning process [11]–[13]. In hostile environments with high attrition rates, this technique may significantly hamper the overall performance of the system by hindering the learning process of the agents, and thus keeping the system from achieving optimal operating conditions. Therefore, how to efficiently and effectively perform online MAS planning in the presence of attrition remains a key open issue.

In this paper, we develop a novel decentralized MCTS planning algorithm that tackles the above-mentioned issue.

- *N. Nguyen and H. Nguyen are with the School of Computer and Mathematical Sciences, The University of Adelaide, SA 5005, Australia. Email: {nhatdaoanh.nguyen, hung.nguyen}@adelaide.edu.au.*
- *D. Nguyen and J. Kim are with the Defence Science and Technology Group, Australia. Email: {duong.nguyen, junae.kim}@defence.gov.au.*
- *G. Rizzo is with the HES SO Valais, Switzerland, and the University of Torino, Italy. Email: gianluca.rizzo@hevs.ch.*

In our approach, every agent grows its search tree using the global utility function while taking into account the possible actions of others. To facilitate the collaboration among agents, we propose a new coordination mechanism based on regret matching (RM) [14]. We prove that our coordination approach guarantees that the average joint action of all agents converges to a Nash equilibrium (NE) if every agent applies the same RM procedure in any cooperative game with a submodular utility function. By arriving at an NE, it is guaranteed that each agent's action is the best response to the actions of others, and no agent has the incentive to deviate unilaterally. This provides a stable self-enforcing agreement where all agents coordinate their actions in a decentralized manner effectively.

Specifically, the main contributions of our work are:

- We formulate the problem of multi-agent planning under attrition risks, wherein agent failures occur abruptly and unpredictably within their population.
- We show that, in the presence of attrition, the suboptimality of current decentralized MCTS algorithms is due to the usage of the marginal contribution combined with submodular properties of the global utility functions.
- We develop Attritable MCTS (A-MCTS), a new online decentralized planning algorithm, based on MCTS and Regret Matching, that can quickly and efficiently adapt the plan to settings where agents fail, even at high rates. We show that, by modulating the utility function for each agent and their coordination mechanism, under the assumption of submodularity, successive iterations are guaranteed to improve joint policies and deal effectively with attrition risks. We prove formally that our regret matching coordination algorithm provides a strong convergence result for approximating a pure-strategy Nash equilibrium in a fully distributed fashion. Our algorithm is anytime and robust concerning limitations in the frequency and amount of information exchanged among agents.
- We evaluate our proposed approach in two information gathering scenarios with attrition: underwater data harvesting and an infrastructure inspection task with a real-world dataset [15]. Results suggest that our solution improves substantially over the best existing approaches in terms of global utility and scalability in the presence of frequent failures.

The remainder of this paper is structured as follows. In Section 2, we discuss related work in multi-agent planning. Section 3 presents the system model and the problem formulation. Our Attritable MCTS algorithm is illustrated in Section 4. Section 5 presents some analytical results on the properties of our algorithm, which is assessed numerically in Section 6. Finally, Section 7 concludes the paper.

2 Related Work

Information-gathering problems are often modeled as sequential decision-making problems [16]. When there are multiple agents, decentralized information gathering can be viewed as a decentralized version of a partially observable Markov decision process (Dec-POMDP) [17]. The dominant approach to Dec-POMDP is to first solve the centralized, offline planning over the joint multi-agent policy space, and then push these policies to agents to execute them in a decentralized fashion [18]. These online-offline approaches are however not applicable to dynamic environments where the state of the environment is not known ahead of time. Simultaneous decentralized planning and execution solutions for Dec-POMDP exists [19], but they suffer from significant memory and computational demands to store the reachable joint state estimations of all agents [20]. Several works approximated the problem to a set of local POMDPs that are easier to solve and allow each agent to act independently [21], [22]. Alternatively, role-based abstraction was proposed to decompose the problem into a set of single-agent problems and an optimal task assignment [23], [24]. These techniques can improve computational efficiency and scalability, but at the cost of losing global optimality [25].

Recently, simultaneous distributed approaches based on MCTS have gained significant interest due to their flexibility in trading off computation time for accuracy. The key idea is for each agent to use the upper confidence bound (UCB) [6] to plan the best course of action. To implement cooperation between agents, these methods usually keep a predefined model of the teammates, which can be heuristic or machine learning trained [5], [26], [27]. However, as they are based on trained knowledge, they are unsuitable for online planning in settings that change unpredictably, such as in disastrous environments.

To address these challenges, recent approaches have been developed that rely not on prior knowledge of agents' behavior but on information sharing among them [7]–[9], [28]. Specifically, the Decentralized Monte Carlo Tree Search (Dec-MCTS) algorithm [7] employs probabilistic sampling of communicated information to predict the actions of other agents and create coordination between them. A key feature of Dec-MCTS and its subsequent variations is letting each agent optimize its actions using a local utility function, which measures only the individual agent's contribution rather than the total team reward. To deal with uncertainty arising during the mission, Dec-MCTS algorithms allow for online replanning by enabling agents to update their beliefs about the system and, if necessary, reset the search tree when changes are considered substantial. However, quantifying the *significance* of such changes remains a challenge, as no clear metric is available. Moreover, frequent resetting of the search tree may impair performance in scenarios with limited planning time or computational resources.

Under high uncertainty scenarios, a growing body of literature in the area of multi-drone systems [29]–[32] explores the significant challenges posed by agent attrition – the loss or removal of individual drones (due to mechanical failures, environmental factors, and human errors), and highlights the need to address attrition for robust system performance. In systems with attrition in which agents may fail abruptly, the approaches mentioned above do not apply, as they do not allow adjusting to attrition in agents' populations. In the present work, we show that the suboptimality of current Dec-MCTS algorithms is due to the usage of the marginal contribution combined with the submodular properties of the global utility functions.

Another body of literature on related works concerns Open Agent Systems (OASYS), in which agents can enter

and leave over time. Most solutions to OASYS are either fully or partially offline, i.e., offline planning with online execution [33], [34] or online planning with precomputed offline policies [35]. This is not feasible in applications with significant sources of uncertainty, particularly when the environment's state or mental models of the agents are unknown in advance, and when the agents' failures occur abruptly during execution. A recent work [36] proposed a fully online approach that leverages communication between agents related to their presence to predict the actions of others. This approach assumes that agents can communicate their existence implicitly. In scenarios where the communication is intermittent or agents vanish without warning, the unforeseeable failure can significantly impact coordination and planning, jeopardizing the overall performance. In addition, the computational complexity of modeling each other's presence and predicting their actions can make the system computationally intractable. Thus, it isn't directly equipped to handle sudden agent failures and may require further refinement to be viable in large-scale or highly dynamic environments.

Within the broader framework of multi-agent planning in adversarial environments, recent works consider scenarios where agents are attacked but retain partial functionality—such as the ability to communicate or relay information—or where attacks only affect the system for a finite time duration [37]–[39]. Another line of research in the multi-agent reinforcement learning (MARL) literature addresses failure models involving noisy or adversarial communication and proposed solutions robust to unreliable information exchange [40]–[42]. In contrast, our study focuses on maintaining effective multi-agent planning under sudden agent attrition, which results in the complete and permanent loss of an agent's contribution.

Our paper focuses on problems where agents experience hard failures in an abrupt and unforeseeable fashion. This unpredictable nature makes the existing techniques not directly applicable. Therefore, more research is needed to enhance the robustness and resilience of MAS in such uncertain attrition scenarios. Our work explicitly tackles this challenge by providing a new approach for online decentralized planning for multi-agents that does not require precomputed offline policies and mental models. Instead, agents reason about the actions and existence of others using directly communicated information. We employ a computationally effective game-based technique to coordinate agents, enabling adaptive decision-making in the presence of peer failures while ensuring fast convergence in polynomial time relative to system size.

This paper is an extended and substantially revised version of [43], presented at ECAI 2024, with the following key additions:

- First, we provide a strengthened theoretical analysis of the RM coordination step. We motivate the design of a cooperative coordination mechanism that converges to an NE, introduce and justify a heuristic equilibrium-selection rule, and expand the proofs of Proposition 1 and Theorem 1 to clarify the role of submodularity in guaranteeing convergence to a pure-strategy NE (PSNE). We further analyse the price of anarchy (PoA) to quantify the efficiency of the resulting equilibria relative to the global optimum.
- Second, we present a distributed and parallel implementation of the RM coordination algorithm and analyze its computational complexity, showing that convergence to an NE can be achieved in a scalable way. The algorithmic description is made more precise, with explicit definitions of the dynamically updated action sets, best-response selection, and utility computation, supported by full pseudocode for reproducibility.
- Third, the experimental evaluation is greatly expanded: in addition to larger-scale and more complex environments, we consider a practical reference scenario for oil-rig infrastructure inspection, using a real-world dataset together with an acoustic communication model.
- Finally, we conduct a deeper investigation into the optimality of the PSNE produced by RM, linking the theoretical guarantees, including PoA bounds, to observed performance in realistic scenarios, thereby strengthening the theoretical foundation and demonstrating robustness under practical operating conditions.

3 Problem Formulation

3.1 Basic Assumptions

In this paper, we consider a set $\mathcal{S}$ of S *regions of interest (ROI)* arbitrarily distributed within a volume of space, where S_k is the k-th element of the set. We assume each region is a sphere with an equal radius r, however, the formulation could extend to more complex models. These ROIs may model, for instance, an underwater wireless sensor network (UWSN) deployed to monitor a seaport exit or an oil platform complex that requires routine inspection. We assume the locations of the ROIs do not change over time.

We consider a set $\mathcal{N}$ of N autonomous agents (e.g., modeling autonomous underwater vehicles (AUV) or drones) moving within the same given volume of space and periodically visiting the ROIs to gather information from them. The agents then head to the *gathering point* of the volume, where they relay the harvested information for further processing. This formulation abstracts many multi-agent information-gathering tasks, such as informative path planning [44] or UWSN data collection [45].

We assume agents move along an undirected graph $G = (V, E)$ that is placed in the same space as the ROIs. Each vertex $v_i \in V$ represents a location, and each edge e_{ij} represents a feasible route from vertex v_i to v_j. A key property of this graph is that it traverses at least once every region of interest. This graph typically models constraints to agent trajectories due to the morphology of the monitored environment, presence of obstacles, ROI distribution, and characteristics of agent movements, among others. The specific way in which the graph is derived is thus application and context-dependent. The graph is established during system initialization and remains static throughout the task's duration. Without loss of generality, we assume that all agents have prior knowledge of the motion graph and the locations of the ROIs. Nevertheless, our framework can be naturally extended to scenarios where agents maintain only an estimated model of the environment, which is progressively refined through exploration during execution.

TABLE 1
Key notations used in the paper.

Notation	Description
$\mathcal{S}$	Set of S regions of interest
$\mathcal{N}$	Set of N agents
$\mathcal{F}$	Set of agents that fail during the mission
G	Motion graph of the agents
p_n	Path of the n-th agent
p	Joint paths for all agents
P_n	Set of all possible paths of the n-th agent
P	Collection of sets of all possible paths for all agents
$b(p_n)$	Cost of the path p_n
B_n	Budget constraint of the n-th agent's path
U_g	Global utility function
$\mathcal{T}_n$	Search tree of the n-th agent
x_n	An action sequence of the n-th agent
$\mathcal{X}_n$	Set of all possible action sequences of the n-th agent
$\hat{\mathcal{X}}_n$	Subset of M communicated action sequences of the n-th agent
$\mathcal{X}$	Set of all possible action sequences of all agents
x^*	The pure-strategy Nash equilibrium joint actions for all agents

A schematic representation of the system and an example of the motion graph are illustrated in Fig. 1.

We denote the information-gathering task as a *mission*. During a mission, each agent navigates the graph G, visiting the regions of interest to gather information. A region S_k is considered *visited* if it is traversed by an agent's path. The *path* of agent n during a mission is denoted by p_n, represented as an ordered sequence of edges $p_n = (e_n^1, e_n^2, ...)$, where consecutive edges in the path are connected by vertices in the graph. The joint paths of all agents are collectively denoted as $p = (p_1, ..., p_n, ..., p_N)$. Each path p_n is associated with a cost $b(p_n)$, and each agent has a path budget B_n such that the cost of that agent's path must not exceed this budget, i.e., $b(p_n) \leq B_n$. The budget may represent a constraint on energy, distance, or time, depending on the specific application.

At the beginning of the mission, every agent performs distributed *planning* together to decide a path for each agent. Each agent then *executes* its planned path by traversing the first edge of the path, gathering information from all ROIs for which it gets within their ranges. After that, all agents exchange updates on their intended paths and on those regions that have already been visited. Finally, they plan their residual path with the newly available information. This *planning-executing* procedure repeats until all regions have been visited, or until all agents' travel budget expires.

Each region of interest S_k is assigned a utility $U(S_k)$, representing the value of the information an agent may collect from that region. The distributed path-planning objective

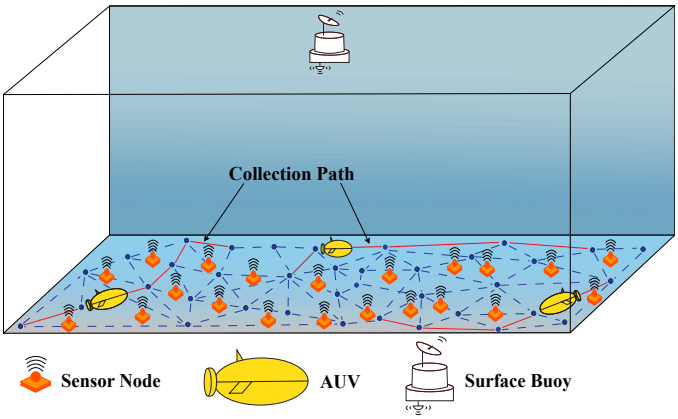

Fig. 1. Example of a multi-agent information-gathering problem for the case of data collection from UWSN scenario.

is to determine a path for each agent that maximizes the global utility of the information-gathering task, where the global utility is a function of the joint paths of all agents p, denoted $U_g(p)$. In this work, we focus on *submodular* reward functions, which capture the property of diminishing returns and naturally arise in a broad range of information-gathering applications [46], [47]

Definition 1 (Submodular set function)**.** Let $g : 2^\Omega \to \mathbb{R}$ be a set function where 2^Ω is the power set of Ω. Then g is a submodular function if for every $X, Y \subseteq \Omega$ with $X \subseteq Y$ and every $x \in \Omega \setminus Y$ the following inequality holds

$$g(X \cup x) - g(X) \geq g(Y \cup x) - g(Y) \ .$$

We assume that each agent is equipped with a wireless communication interface (e.g., cellular for drones or an acoustic modem for AUVs). The communication network is modeled as an undirected, fully connected graph, where nodes represent agents and edges denote the communication links. Each link has a transmission failure rate P_e determined by the communication model, which is application- and context-dependent and is outside the scope of this work. We assume that the planning-execution cycles can be asynchronous between agents. Thus, the duration of information exchange does not impact the performance of our algorithm, and each agent will plan based on the information it has available locally. For analytical simplicity, we assume information exchange—whether among agents or when visiting the region of interest to be instantaneous. However, our approach can be readily extended to incorporate realistic communication delays.

3.2 Multi-agent Planning under Attrition Risks

We consider scenarios where a subset $\mathcal{F}$ of the N agents fail during the mission. We focus on *hard* failures, where agents interrupt reward collection and information exchange. We assume the set of agents that fail $\mathcal{F}$ is unknown in advance, and the time at which they fail is not determined by any arbitrary criteria or distribution. Therefore, our solution does not rely on knowing its size and probability distribution. In the occurrence of a failure, all the utility collected by the failed agent is lost, i.e., it is not considered anymore in the

computation of the global utility of the mission. This models a typical setup in information gathering, in which collected data is relayed to data sinks only at the end of the mission.

Our goal is to provide an efficient planning and coordination mechanism that can quickly adapt to agent failures and maximize the global utility of the mission within the agent's budget constraint. Let $\mathcal{P} = (P_1, ..., P_n, ..., P_N)$, with P_n denoting the set of all possible paths that satisfy the budget constraint B, which starts at agent n starting position. We define the following problem:

Problem 1. (*Multi-agent planning under attrition risks)*

$$\underset{p \in \mathcal{P}}{\text{maximize}}\ U_g(p) \tag{1}$$

$$\text{Subject to:} \quad b(p_n) \leq B_n, \quad \forall n \in \mathcal{N} \tag{2}$$

$$0 \leq |\mathcal{F}| < N \tag{3}$$

Constraint (2) derives from imposing that the total path cost for the agent n is less than the travel budget B_n of each agent. Intuitively, our goal is to find a path for each agent that maximizes the global utility associated with the regions observed by all agents throughout the mission, even in scenarios where a subset $\mathcal{F}$ of agents fails. Such an optimization problem cannot be solved efficiently. Indeed, it is easy to see that Problem 1 is a variant of the well-known NP-hard Vehicle Routing Problem (VRP). More specifically, our formulation aligns with the Team Orienteering Problem (TOP) [48], a VRP variant where multiple agents aim to maximize total rewards collected from visiting a subset of locations within individual budget constraints.

Remark 1. Our formulation builds on the Team Orienteering Problem (TOP) framework [48], [49], maximizing total mission rewards within budget constraints rather than minimizing individual agent costs. This cooperative approach ensures efficient coordination and adaptability, especially in dynamic, high-attrition environments like disaster response and smart-city applications [49]. By optimizing the global objective, agents enhance mission success and reduce resynchronization costs during failures. While hybrid models that balance global and local goals are a potential future direction, our work prioritizes robust decentralized planning and real-time responsiveness, addressing critical gaps in existing TOP models.

4 Attritable MCTS with Regret Minimization

In this section, we first give a brief introduction to Monte Carlo Tree Search and its most popular decentralized version. We then show the root cause of the inefficiency of existing decentralized MCTS approaches with attrition.

4.1 Decentralized MCTS with Attrition Risk

MCTS is an online planning approach [6]. The tree $\mathcal{T}_n$ for agent n is defined such that each node s of the tree represents a state and each edge a starting from that node represents an available action. A branch from the root node to another node represents a valid action sequence. The tree is incrementally grown via a four-step process: *selection*, *expansion*, *rollout*, and *backpropagation*. Decentralized Monte Carlo Tree Search (Dec-MCTS) [7] extends the power of MCTS to MAS using intention sharing. Specifically, agent n maintains a probability mass function $q_n(x_n)$ over the set of all possible action sequences $\mathcal{X}_n$, where $x_n \in \mathcal{X}_n$ is a primitive action sequence. The intentions of other agents except agent n are denoted by q_{-n} and $\mathcal{X}_{-n}$. By taking a probabilistic sampling from the communicated intention, each agent can reconstruct the global utility. To create better coordination, rather than optimizing directly for the global utility U_g of the entire team, each agent n instead optimizes for a local *marginal contribution* utility function U_n. That is, agent n estimates the rollout score $F_n(x_n)$ for executing x_n as:

$$F_n(x_n) = U_n(x_n, x_{-n}) = U_g(x_n, x_{-n}) - U_g(x_{-n})\ , \tag{4}$$

where $U_g(x_{-n})$ is the global utility without the contribution of agent n.

We now analyze Dec-MCTS asymptotic behavior when agents fail under the assumption that the global utility is a submodular function (defined in Definition 1). In particular, at iteration t, let x_n denote the chosen action sequence of agent n and x_{-n} denote the combined sampled action sequences of other agents. Assume that at the next iteration $t+1$, a subset of agents fails. Let x'_{-n} be the combined sampled action sequences of all agents except agent n and the lost agents (i.e., that is $x'_{-n} \subseteq x_{-n}$).

Proposition 1. If the global objective function U_g is submodular, then

$$F_n^{(t+1)}(x_n) \geq F_n^{(t)}(x_n)$$

by the diminishing return property due to submodularity, where $F_n(x_n)$ is defined in (4).

Proposition 1 states that if some agents fail during the mission, the remaining agents would mistakenly perceive that the contribution of their previous actions increases. Hence, they would not be aware of the actual reduction of the global utility and update their plans. Further analysis of Dec-MCTS behavior, when agent failures occur after the algorithm has converged, is presented in Appendix A.

While using marginal contribution as the local utility function leads to suboptimal performance under agent attrition, this choice was deliberate to support and enhance the coordination mechanisms of existing decentralized MCTS methods. Fixing this issue requires a new, context-aware redesigned local utility function paired with a corresponding coordination mechanism. In the next section, we present our proposed algorithm for multi-agent planning under attrition settings of Problem 1.

4.2 Overview of the A-MCTS algorithm

We develop Attritable MCTS (A-MCTS), an online decentralized MCTS algorithm that quickly adapts to agents' attrition and efficiently coordinates action between the remaining agents. Its performance mainly relies on two key factors, including the joint-utility-guided decentralized tree search and the best response policy given the shared intentions of others. Each agent runs A-MCTS distributedly to plan for itself a path that is expected to maximize the total utility of the whole mission. Agents then execute the first planned action and observe any changes. After that, they perform

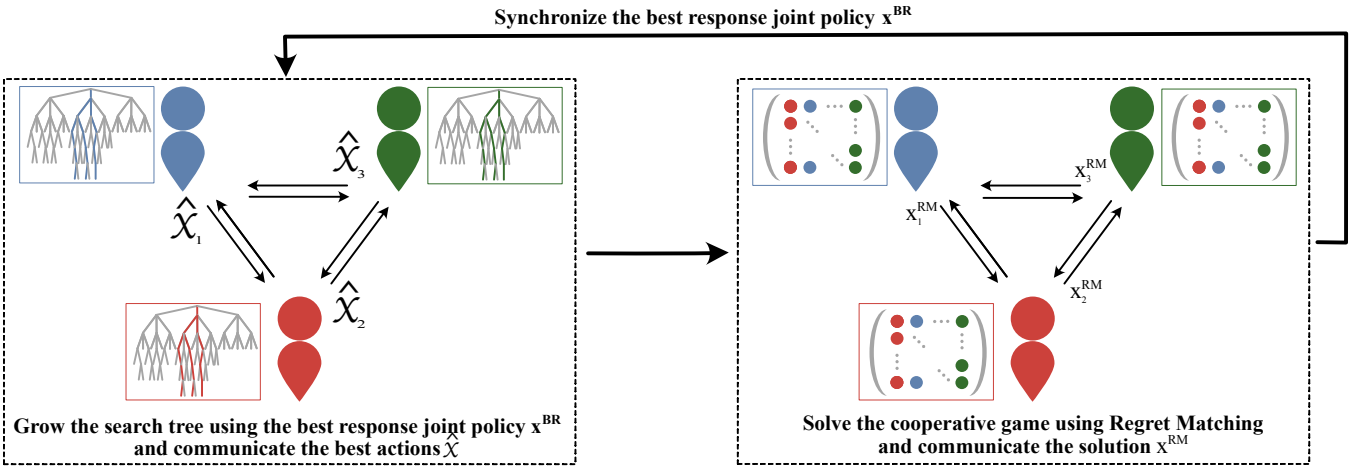

Fig. 2. Overview of the A-MCTS algorithm. Agents incrementally grow the search using the best response policy x^{BR} and communicate their best actions $\hat{\mathcal{X}}$. Regret Matching is then used to compute distributively a joint policy for the cooperative game. These solutions are synchronized and the most payoff-dominant is chosen as the best response policy x^{BR}.

Algorithm 1 A-MCTS algorithm for agent n

Input: Global objective function U_g, actions budget B_n
Output: best action sequence x_n^* for agent n

1: $\mathcal{T}_n \leftarrow$*Initialize MCTS Tree*
2: **while** computation budget not met **do**
3: $\quad\hat{\mathcal{X}}_n \leftarrow$*Select Subset From*$(\mathcal{T}_n)$
4: $\quad(\hat{\mathcal{X}}, \hat{\mathcal{X}}_{-n}) \leftarrow$*Communicate and Update*$(\hat{\mathcal{X}}_n)$
5: $\quad x^{BR} \leftarrow$*Regret Matching Coordination*$(\hat{\mathcal{X}})$ (Algo. 2)
6: $\quad$**for** fixed number of iterations **do**
7: $\quad\quad x_n \leftarrow$*D-UCT Select, Expand & Rollout*$(\mathcal{T}_n, B_n)$
8: $\quad\quad F_n \leftarrow U_g(x_n, x_{-n}^{BR})$
9: $\quad\quad \mathcal{T}_n \leftarrow$*Backpropagation*$(\mathcal{T}_n, F_n)$
10: $x_n^* \leftarrow$*Best Next Action*$(\mathcal{T}_n)$
11: **return** x_n^*

replanning from their new state and update the planned paths based on newly available information. The search tree may be pruned by removing all children of the root except the selected branch. This cycle of planning and execution continues until the travel budget expires. The pseudo-code of A-MCTS for agent n is shown in Algorithm 1.

The tree $\mathcal{T}_n$ of agent n is incrementally built over its action sequences space $\mathcal{X}_n$ while considering the possible behaviors of others $\mathcal{X}_{-n}$ (Algorithm 1, Line 6-9). In the *selection phase*, the discounted upper confidence bound on Tree (D-UCT) [7] is applied to handle the abrupt changes in reward values caused by the actions of other agents.

The key idea of our proposed algorithm is that every agent's search trees should be guided by the same utility of the joint action sequences. This is achieved by letting all agents optimize their local actions directly using the global utility U_g (Algorithm 1, Line 8). This approach enables agents to directly optimize global welfare U_g while allowing them to be aware of both the relative quality of their actions x_n and the potential reduction in global rewards caused by attrition. However, the uncertainty in other agents' plans has also been shown to degrade the overall performance when using the global objective function to optimize local actions [50]. To overcome this issue, we propose to let each agent improve its policy iteratively while assuming others keep their policies fixed.

More precisely, given a set of all possible action sequences of all agents $\mathcal{X} = (\mathcal{X}_n, \mathcal{X}_{-n})$, A-MCTS will periodically compute a "best response" set of joint action sequences that maximize the joint utility for all participants $x^{BR} := \{x_n^{BR}, x_{-n}^{BR}\}$ (Algorithm 1, Line 5). Each agent will then assume other agents coordinately determine their policies following such "best response" x_{-n}^{BR} and uses such information to compute the utility for its action sequence selection while growing the MCTS tree (Algorithm 1, Line 8). In general, the cardinality of $\mathcal{X}_n$ can be very large and it grows exponentially. To reduce the computation and communication requirements, we consider only those dynamically updated subsets $\hat{\mathcal{X}}_n \subseteq \mathcal{X}_n$ of the most promising action sequences. The set $\hat{\mathcal{X}}_n$ is chosen as the best rollouts of M fixed nodes in the search tree $\mathcal{T}_n$ with the highest discounted empirical average (Algorithm 1, Line 3). We then define the following problem:

Problem 2. (*Best joint policy for multi-agent planning*)

$$\underset{(x_1,x_2,\ldots,x_N)}{\text{maximize}} \; U_g(x_1, x_2, ..., x_N) \tag{5}$$

$$\text{Subject to:} \quad x_n \in \hat{\mathcal{X}}_n, \quad \forall n \in \mathcal{N} \tag{6}$$

The objective is to find an action profile $(x_1, x_2, \ldots, x_N)$ that maximizes the global utility $U_g(\cdot)$. Such an optimization problem has been shown to be NP-hard with a submodular utility function [51]. Seeking a Nash equilibrium (NE) (where each agent's policy is the best response to the others) that achieves a good efficiency compared to the optimal solution is more desirable in these cases [52]. A greedy algorithm is usually employed to find an approximation solution [53]. However, we will show later with simulations that greedy solutions can be substantially suboptimal even in scenarios with few agents. In the following section, we provide a distributed regret-based solution to Problem 2 that quickly and efficiently computes an NE joint policy for multi-agent systems, regardless of their complexity.

Algorithm 2 Regret Matching Coordination algorithm

Input: Global objective function U_g, joint compressed action sequences set $\hat{\mathcal{X}} = (\hat{\mathcal{X}}_n, \hat{\mathcal{X}}_{-n})$
Output: Best response joint action sequences x^{BR}

1: Every agent $n \in \mathcal{N}$ performs the following steps $2-12$
2: Initialize $\mathcal{R}$ to zeroes and p to uniformly random
3: **for** $t = 1, 2, \ldots$ **do**
4: $x^{(t)} \leftarrow$ *Sample*$(\hat{\mathcal{X}}, p)$
5: **for each** $x_{nm} \in \hat{\mathcal{X}}$ **do**
6: $\mathcal{R}_{nm} \leftarrow \mathcal{R}_{nm} + U_g(x_{nm}, x_{-n}^{(t)}) - U_g(x^{(t)})$
7: $$p(x_{nm}) \leftarrow \begin{cases} \frac{\mathcal{R}^+_{nm}}{\sum_{j=1}^{M} \mathcal{R}^+_{nm}} & \text{if } \sum_{j=1}^{M} \mathcal{R}^+_{nm} > 0 \\ \frac{1}{M} & \text{otherwise} \end{cases}$$
8: **end for**
9: **end for**
10: $x_n^{RM} \leftarrow \arg\max_{x_{im} \in \hat{\mathcal{X}}_i} [p(x_{im})], \forall i \in \mathcal{N}$
11: $x_{-n}^{RM} \leftarrow$ *Communicate and Update*(x_n^{RM})
12: **return** $x^{BR} \leftarrow \arg\max_{x \in \{x_n^{RM}, x_{-n}^{RM}\}} U_g(x)$

4.3 Regret Matching For Cooperative Coordination

We consider the distributed solution of the optimization Problem 2 where each agent decides its path based on local information and limited communication from its peers. We aim to design a decision-making method that is capable of operating and adapting with occasional communication or less, where every agent acts solely based on its local observation and does not need to constantly communicate every decision with the others. This is to guarantee that the algorithm can effectively handle the agent attrition situation described in Section 3.2. The main difficulty here is how to ensure the independent decisions of the agents lead to jointly optimal decisions for the group. To address this challenge, we formulate the problem of finding an action sequence for each agent that collectively maximizes the joint utility as a multi-agent cooperative game. We then propose a distributed mechanism, where every agent independently simulates a multi-player cooperative game based on the local information available to itself and solves the game by self-play. For this purpose, a game theory learning algorithm based on the Regret Matching technique [14] is employed to approximate the Nash equilibrium of the game.

Let $\hat{\mathcal{X}} = (\hat{\mathcal{X}}_n, \hat{\mathcal{X}}_{-n})$ denote the joint set of action sequences that are shared between all agents, and x_{nm} denote the action sequence m of agent n. In our approach, periodically, every agent independently constructs a matrix game in which the set of players contains all the active agents and the set of actions is $\hat{\mathcal{X}}$. At this stage, each agent applies the Regret Matching (RM) procedure as proposed in [14] to its estimated matrix game to compute a best response joint decision. The pseudo-code of our RM game is shown in Algorithm 2.

At each iteration t, an action $x_{nm} \in \mathcal{X}$ is sampled for each agent based on a probability distribution (Algorithm 2, Line 4). Let p denote this probability distribution where $p(x_{nm})$ is the probability for x_{nm} and $\sum_{j=1}^{M} p(x_{nj}) = 1, \forall n \in \mathcal{N}$. With $x^{(t)} := \{x_n^{(t)}, x_{-n}^{(t)}\}$, we denote the sampled set at iteration t, where $x_n^{(t)}$ is the sample action for agent n and $x_{-n}^{(t)}$ is the sampled actions for all agents except agent n. We then define the regret of agent n for not taking action m at iteration t as

$$R_{nm}^{(t)} = U_g(x_{nm}, x_{-n}^{(t)}) - U_g(x^{(t)}). \tag{7}$$

Denote $\mathcal{R}$ as the cumulative regret matrix where an element $\mathcal{R}_{nm}$ is the regret for x_{nm} and $\mathcal{R}^+_{nm} = \max\{\mathcal{R}_{nm}, 0\}$. Then, the probability distribution p used at the next iteration will be updated as

$$p(x_{nm}) = \begin{cases} \frac{\mathcal{R}^+_{nm}}{\sum_{m=1}^{M} \mathcal{R}^+_{nm}} & \text{if } \sum_{m=1}^{M} \mathcal{R}^+_{nm} > 0, \\ \frac{1}{M} & \text{otherwise.} \end{cases} \tag{8}$$

Intuitively, each agent treats its current strategy as a reference point and evaluates whether to switch based on how much better other actions appear. If an alternative action could have produced a better outcome, the agent assigns it a positive probability of being selected next time, proportional to the regret relative to its current choice. Thus, when regrets are small, the agent is less likely to change strategies, encouraging gradual and adaptive behavior.

Denote by x_n^{RM} the joint action profile with the highest probability $p(x_{nm})$ under regret matching as computed by agent n, and by x_{-n}^{RM} the corresponding profile for all other agents. These candidate profiles are exchanged among agents, and the most payoff-dominant solution is selected as the best response joint decision:

$$x^{BR} \leftarrow \arg \max_{x \in \{x_n^{RM}, x_{-n}^{RM}\}} U_g(x). \tag{9}$$

In our identical-interest setting, this rule ensures that, among the Nash equilibria reached by distributed RM, the system consistently adopts the payoff-dominant one, thereby avoiding efficiency loss. The candidate set is small, as it is drawn from the most probable RM outcomes, making the selection computationally efficient. As demonstrated later in Section 6.2.4, this selection rule achieves near-optimal performance in practice.

5 Analysis of A-MCTS

5.1 Optimality of Nash equilibrium solution

It has been shown in [54] that there exists no polynomial time algorithm to compute a pure NE in multi-agent nonzero-sum stochastic games. Hence, we employ an approximate method of finding the NE by proposing a decentralized Nash selection method based on Regret Matching for making choices in a multi-agent matrix game formulated at each decision-making state. Regret Matching is a regret-based algorithm for learning strategies in games and is often used to compute correlated equilibria in multi-agent repeated games with imperfect information. Although the regret matching technique has been widely used for non-cooperative games, its application in cooperative games, such as the problem studied in our paper with a submodular utility function, has only been recently explored [55].

In our approach, each agent's local utility is set to the global reward function, forming an *identical-interest game*—a special case of potential games [56]. In such a game, there is a strong alignment between the interests of all agents, which

guarantees that any joint action maximizing the global objective is a pure-strategy Nash equilibrium (as defined in Definition 2) [57]. By letting the agents use the global objective U_g to calculate the regrets and leveraging the submodularity property, we theoretically prove that Regret Matching converges to the approximate pure-strategy Nash solution, offering stronger guarantees than the commonly used correlated equilibrium.

Definition 2 (Pure-Strategy Nash Equilibrium)**.** A pure-strategy Nash equilibrium (PSNE) is a joint action profile $x^* = (x_n^*, x_{-n}^*) \in \hat{\mathcal{X}}$ if for all $n \in \mathcal{N}$ and all $x_n \in \hat{\mathcal{X}}_n$ such that: $U_n(x_n^*, x_{-n}^*) \geq U_n(x_n, x_{-n}^*)$.

Theorem 1. The best response joint decision x^{BR} computed using RM, under the assumption of submodular utility functions, is a PSNE solution of the matrix-game representation generated by the set of best feasible paths $\hat{\mathcal{X}}$ chosen by every agent at each decision point.

The proof of Theorem 1 is presented in Appendix B.

Remark 2. The theoretical guarantees in Proposition 1, Theorem 1, and the Nash-convergence result rely on the global reward function being submodular. This property, satisfied in many cooperative sensing and coverage tasks with diminishing returns (e.g., independent sensing, non-overlapping coverage), ensures the induced game is a potential game whose pure-strategy Nash equilibria align with high-reward joint actions. In non-submodular settings (e.g., correlated sensing with synergy, penalties for overlap, or other negative interactions), the potential-game structure may not hold, and our Nash-convergence guarantee no longer directly applies. Nevertheless, the regret-matching procedure converges to the set of correlated equilibria in any finite game with bounded payoffs [14], so the learning dynamics remain stable in that broader sense. We focus on submodular cases because convergence to a PSNE enables deterministic task allocation and predictable system performance. A PSNE also provides a stable joint strategy profile in which no agent can improve its outcome by unilaterally deviating. This stability is particularly valuable for coordinating agents with a common objective under uncertainty or limited communication.

Remark 3. The Price of Anarchy (PoA) measures the efficiency of the worst-case NE relative to the optimal outcome [57]. Given the nature of identical interest games, where every NE inherently maximizes the global objective, the PoA is generally considered to be 1, indicating no efficiency loss due to self-interested behavior. Yet, a remaining challenge is that there often exists multiple NE, and thus how to make sure the combination of these individual Nash-based strategies, which each agent independently computes, defines an optimal equilibrium. The selection of a good Nash equilibrium among the many options, known as an equilibrium selection problem, remains an open question for further investigation. Within the scope of this work, to address this dilemma, we propose that the agents synchronize their reached Nash points to identify the most payoff-dominant Nash solution. The agents then simply follow the best computed Nash equilibrium to select their decisions. In Section 6.2.4, we demonstrate via extensive simulation results that our approximate Nash-based approach achieves an overall good efficiency compared to the optimal solution and substantially improves over the main competing approaches, both in terms of convergence speed and global utility achieved.

5.2 The distributed and parallel execution of RM

In MAS, where agent loss and unreliable communication are expected to occur, a single point of failure is often unacceptable. Moreover, a centralized approach that requires a global view of the game is often intractable due to the exponential growth of the game's size and complexity. To address these challenges, we devise a distributed approach for executing the coordination algorithm.

In our approach, when an agent experiences a temporary loss of communication with its teammates, it continues operating by predicting their behaviours through simulating their likely decisions based on the most recent information received. For analytical tractability of the Regret Matching convergence, we assume instantaneous and reliable communication during the planning phase. However, in the practical implementation and evaluation, communication intermittence is handled heuristically: repeated losses of communication beyond a predefined threshold are interpreted as an agent attrition event. When this occurs, the remaining agents form a new game involving only the active participants. Unlike existing methods that reinitiate the entire planning process—often incurring significant computational overhead—our approach restarts only the cooperative game. This subproblem is considerably smaller in scale, enabling our proposed algorithm to solve it quickly and efficiently.

The distributed execution of RM enables agents to independently learn and adapt their strategies using local information, eliminating the need for a central authority or synchronous communication. Furthermore, the parallel execution of RM (with agents running the same algorithm concurrently) facilitates the exploration of multiple NE, helping to avoid local optima and identify the most payoff-dominant solutions.

5.3 Computational complexity of Algorithm 2

Regret Matching was proven to guarantee a convergence rate of $\mathcal{O}(1/\sqrt{(T)}$ after T iterations [14]. We discuss here how Algorithm 2 scales with respect to the size of the problem and the number of available actions for each agent to choose from. For matrix games, where each agent has a finite set of actions and the payoffs are given by a matrix, the RM algorithm can be implemented in polynomial time. Specifically, a matrix game with N agents and at most M actions per agent has M^N action combinations in total. Each agent has one local utility (or payoff) for each action combination, and thus it requires $N \times M^N$ integer numbers to represent all possible agents' utilities. Therefore, as the number of agents and the number of actions per agent increase, the size of the game tree grows exponentially, making it intractable to compute the entire tree in memory or time using a centralized approach.

In contrast, our proposed algorithm significantly reduces the computational complexity required to find an NE solution. At each learning time step, each agent learns only

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3636600

its utility vector (of size at most equal to M) to update its action decision policy in the next step. As a result, the total amount of queries over time required by each agent to run the algorithm will scale according to $\mathcal{O}(M \times T)$, where T is the number of iterations until convergence. Note that in the implementation of our proposed approach, each agent has to do the same calculations for other simulated agents. Thus, the total computational complexity of our solution would scale as $\sim \mathcal{O}(N \times M \times T)$, which is linearly proportional to the number of agents, the number of agents' actions, and the number of iterations until convergence. Consequently, our proposed approach can converge to an NE solution in a distributed and scalable way, making it more suitable, effective, and practical in real-world scenarios, where the agents may have access to different and asynchronous information.

5.4 Communication overhead of A-MCTS

In A-MCTS, each agent periodically selects and broadcasts a set of M action sequences to its N teammates as inputs to the Regret Matching coordination algorithm. Each agent then computes a joint action profile—containing N action sequences, one for each agent—which is shared with the others to identify the most payoff-dominant joint decision. Let each action sequence have a maximal payload size of A bytes, and let communication occur once every c planning iterations. Over a total of T iterations, the per-agent communication cost is:

$$\mathcal{O}\left((M+N) \times A \times \left\lceil \frac{T}{c} \right\rceil\right).$$

In contrast, in Dec-MCTS [7], each agent sends M action sequences and their associated probabilities to its N teammates at every planning iteration. If the maximal combined payload of a sequence and its probability is $A' > A$, the per-agent communication cost becomes:

$$\mathcal{O}\left(M \times A' \times T\right).$$

Thus, A-MCTS substantially reduces communication overhead by sending messages only at periodic intervals (every c iterations), whereas Dec-MCTS communicates at every iteration with larger messages. This leads to improved scalability and bandwidth efficiency, particularly in large multi-agent systems or bandwidth-limited environments.

6 Experimental Evaluation

6.1 Experimental Setup

6.1.1 System Components

In this section, we evaluate the performance of our A-MCTS algorithm as a function of the main system parameters. To this end, we consider a collaborative information-gathering task using multiple autonomous underwater vehicles (AUVs). To properly evaluate the proposed approach in this domain, we adopt a well-accepted underwater acoustic communication model described in detail in [58], [59]. Underwater acoustic communication typically experiences path loss due to acoustic energy transfer into heat in the medium. The empirical formula for path loss, calculated based on the distance d and frequency f, is given in [60] as

TABLE 2
Underwater acoustic communication parameters.

Parameter	Value
Transmission Frequency	13 kHz
Transmission Power	1 W
Bandwidth	1 kHz
Packet Size	256 symbols
Wind speed	2.5 m/s

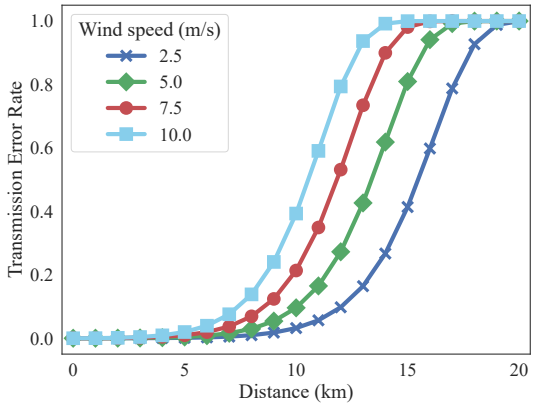

Fig. 3. Transmission failure rate for different wind speeds.

$$A(d, f) = d^k a(f)^d, \tag{10}$$

where k is the propagation loss with a practical value of 1.5. In addition, $a(f)$ is the absorption efficiency (in dB/km) which can be obtained using Thorp's relation [60] as

$$10 \log a(f) = 0.11 \frac{f^2}{1+f^2} + 44 \frac{f^2}{4100+f^2} + 2.75\text{x}10^{-4} f^2 + 0.003. \tag{11}$$

For a transmission power P and bandwidth B, the average signal-to-noise ratio (SNR) at the receiver side is then calculated as

$$\text{SNR}(d, f) = \frac{P}{A(d, f) N(f) B}, \tag{12}$$

where $N(f)$ is the total noise level considering turbulence, shipping activities, waves, and thermal noise [60]. Finally, based on the obtained SNR, we can approximate the probability of transmission failure for a packet of N symbols as

$$P_e = 1 - \left(1 - \frac{1}{4\text{SNR}}\right)^N. \tag{13}$$

The system parameters used in our simulations are given in Table 2 and Fig. 3 shows the corresponding transmission failure rate. With each communication attempt, we assume an agent only transmits its message exactly once.

For what concerns the reference scenarios, we consider two sets of experiments modeling a data collection task from UWSN and an oil rig infrastructure inspection task. We formalize these tasks as follows:

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3636600

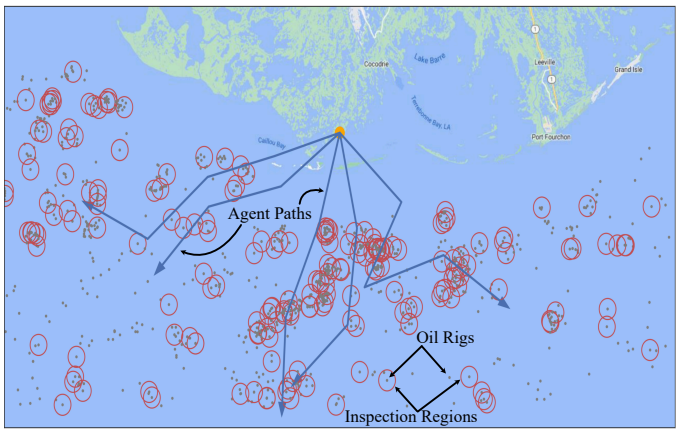

Fig. 4. Example of multi-agent information gathering task for the case of oil rigs inspection. A team of 5 agents coordinates their plans (blue paths) to inspect the set of oil rigs (red circles) in the Gulf of Mexico.

TABLE 3
Main features of the considered algorithms.

Algorithm	Utility Function	Collaboration Mechanism	Reset Planning
A-MCTS	Global utility	Regret matching	No
Cen-MCTS	Global utility	No	No
Dec-MCTS	Marginal contribution	Probabilistic sampling	No
Global-MCTS	Global utility	Probabilistic sampling	No
Reset-MCTS	Marginal contribution	Probabilistic sampling	Yes
Greedy-MCTS	Global utility	Greedy	No

- *Data collection from UWSN (DC):* In this set of experiments, we consider a scenario consisting of 200 uniformly distributed sensors in a 4 km $\times$ 4 km plane and a team of AUVs with random starting positions. We assume the sensors have a uniform transmission radius of 50 m (i.e., typical for UWSN [61], [62]). Since the data is harvested over a short distance, we assume the transmission is instantaneous and not affected by water flow. Moreover, the scale of the system ensures comprehensive communication coverage among the agents in these experiments. This is intended to provide a more thorough examination of the underlying fundamentals of the proposed algorithms.
- *Oil rig inspection (ORI):* In this set of experiments, we consider a scenario consisting of 1000 oil rigs distributed over a 200 km $\times$ 100 km region in the Gulf of Mexico. Their ground truth locations are obtained from [15]. A team of AUVs is tasked to visit 200 randomly chosen oil rigs from a range of 2.5 km. The AUVs are assumed to have the same starting position. An example of this scenario is illustrated in Fig. 4. This is an interesting real-world assessment of our algorithm in a realistic, non-uniform distribution of rewards and a larger system scale. In addition, we consider both perfect and practical communication models (i.e., as described above) for this set of experiments.

We assume the motion graph G of feasible paths in each scenario lies on the same 2-dimensional plane as the sensor and oil rig planes. The graphs are constructed using a probabilistic roadmap with a Dubins path model [63]. This model employs curves to refine the straight-line segments connecting waypoints and is extensively utilized for representing motion constraints pertinent to vehicle-like nonholonomic robots such as AUVs [64]. The corresponding graph G for the *DC* scenario consists of 400 vertices and an average of 19000 edges, and for the *ORI* scenario consists of 1000 vertices and 37000 edges.

For both scenarios, we assume the rewards for each data harvesting from sensors and inspection from oil rigs have a baseline value of 1. If a region R_k is revisited multiple times, its contribution to the global utility diminishes according to a geometric decay factor $\alpha \in [0, 1)$. Specifically, if region R_k is visited n_k times, its utility is given by:

$$U(R_k, n_k) = \sum_{i=0}^{n_k-1} \alpha^i = \frac{1-\alpha^{n_k}}{1-\alpha}. \quad (14)$$

The global utility is then defined as the sum of these utilities across all regions,

$$U_g(p) = \sum_{k=1}^{S} U(R_k, n_k), \quad (15)$$

where n_k is the number of visits to R_k under the joint paths p. This reward model captures diminishing-but-non-zero marginal gains from revisits, which are common in information gathering tasks such as pollution monitoring, disaster management, or temporal change detection. When $\alpha = 0$, the objective reduces to the only unique visits contributing to the reward, a setting widely adopted in prior works [7], [65]. Thus, our formulation generalizes the submodular property while remaining computationally efficient.

6.1.2 Baseline Methods

To perform an accurate evaluation of the performance of our A-MCTS scheme, we considered the following baselines:

- *Centralized MCTS (Cen-MCTS)*: A single search tree is built for all of the M harvesting agents with the actions of agent m are at tree depth $(m, m+M, m+2M, ...)$. The action sequences for all agents are planned offline and then executed online.
- *Dec-MCTS* [7]: It is the state-of-the-art decentralized multi-agent planning. In it, agents build their search tree with a marginal contribution utility function and adapt the same tree for online replanning.
- *Dec-MCTS with global utility (Global-MCTS)*: In this modification of Dec-MCTS, agents build their search tree with the global utility function and adapt the same tree for online replanning. We examine this variation to show that altering the utility function alone would not enhance performance against attrition.

- *Dec-MCTS with reset (Reset-MCTS)*: In this modification of Dec-MCTS, agents are assumed to be fully aware of attrition occurrences via an *oracle* and reset the tree for online replanning. We examine this variation to show that frequent resetting of the trees may adversely affect the algorithm's performance.
- *A-MCTS with greedy optimization (Greedy-MCTS)*: In this scheme, we replace the RM Coordination (Algorithm 2) in our A-MCTS with a greedy algorithm, in which every agent sequentially picks the actions that deliver the highest immediate rewards for collaboration. We examine this variation to show that greedy solutions can be substantially suboptimal in MAS coordination.

Table 3 compares the key features of the considered algorithms, i.e., the utility function, the collaboration mechanism, and whether they handle attrition via planning reset.

For all algorithms, the discounting factor is set to 0.7, and the exploration parameter is set to 0.4 (i.e., within the ranges recommended in [7] to ensure the balance between exploration and exploitation). Each agent compresses its tree into a set of 10 possible paths for every 100 iterations. Unless otherwise stated, we assume 20 agents move in the graph and the geometric decay factor α for revisit values is set to 0. We adopt two different types of planning time and travel budgets for each scenario to demonstrate the versatility of our method. Specifically, in the *DC* scenario, the default planning time is 500 iterations, and the default uniform travel budget is 9 actions. In the *ORI* scenarios, the default planning time is 60 seconds, with the default uniform travel budget is 200 km. These choices have proven sufficient to enable comprehensive coverage of all targeted rewards within the respective scenario. In realistic applications, determining the appropriate number of agents and travel budgets must account for the size of the area under surveillance and the communication range of the agents.

To model attrition in the population of agents, we assume that every agent has the same probability of failing during the mission and that the time at which each failure takes place is distributed uniformly at random throughout the mission duration. The key metric we use to evaluate the performance of the considered algorithms is the *Instantaneous reward coverage (IRC)*, i.e., the fraction of available rewards covered (i.e., collected) at a given time.

6.2 Evaluation Results

6.2.1 Performance Benchmarking

To perform a baseline evaluation of our algorithm's adaptability to agent attrition, we analyze the IRC under varying levels of failure intensity, defined as the fraction of agents that fail during the mission. Fig. 5(a)-(c) shows the improvement of the decentralized planning algorithms over the centralized method across different failure intensities and environment scenarios. As expected, the performances of all algorithms decline with increasing failure intensity, which reflects a reduction in reward coverage due to the decreased number of agents remaining in the system. Notably, when more than 50% of agents fail, Reset-MCTS surpasses Dec-MCTS, as the smaller system size requires fewer iterations for replanning. Conversely, larger systems demand more time for agents to learn the environment; thus, frequent resets impair the algorithm's performance. To further investigate this effect, we evaluate the influence of planning time on the IRC with a default failure intensity of 50%. Indeed, as demonstrated in Figure 5(d)-(f), the performance of Reset-MCTS improves as more planning time is available.

These results suggest that resetting the tree for replanning does not consistently result in improvements over non-reset methods. This is because each MCTS process begins with an exploration phase, during which agents deliberately take random actions to learn the reward distribution. Consequently, resetting the tree prematurely, without sufficient planning, leads to suboptimal joint policies derived from this exploratory phase. On the other hand, due to the use of the *marginal contribution* utility combined with a *submodular* reward function, Dec-MCTS fails to recognize the reduction of the global reward and hence is unable to adapt to failures efficiently.

Additionally, the sampling of other agents' action sequences introduces significant variance in estimating global utility, degrading coordination quality. Indeed, simply adopting the global utility function is insufficient, as Global-MCTS performs the worst across all cases. By assuming that the policies of other agents are fixed, both A-MCTS and Greedy-MCTS can overcome this instability and adapt more effectively to agent failures. Especially under perfect communication conditions, A-MCTS consistently outperforms the other approaches thanks to its regret matching coordination mechanism, which can discover superior joint policies and guide the exploration-exploitation process of the search tree better. In practical communication models, however, the performance of all decentralized algorithms deteriorates when agents are too far apart to exchange information. Moreover, A-MCTS and Greedy-MCTS exhibit comparable performance, as each agent would operate largely independently under both schemes. Despite these challenges, our proposed method maintains a sustainable 10% improvement over the centralized baseline.

It is also noteworthy how the distribution of rewards affects the performance of decentralized solutions in attrition settings. In the *DC* scenario, for instance, the uniform reward distribution requires greater coordination among agents to explore the environment efficiently. This allows decentralized methods to outperform Cen-MCTS under perfect communication conditions. Conversely, in the *ORI* scenario, the centralized method can cover a significant amount of rewards due to their concentration in a particular region. As a result, the performance gains of Dec-MCTS and its variants are substantially diminished, particularly under high attrition rates. Nevertheless, thanks to its regret matching coordination mechanism, our A-MCTS algorithm substantially maintains an advantage over other approaches, demonstrating superior adaptability to attrition risks regardless of reward distribution.

The complexity of multi-agent coordination is also significantly influenced by the number of path exchanges between agents. Increased information exchange potentially leads to better algorithm performance, albeit at the expense of greater computational resources and time. To investigate this trade-off, in Fig. 5(g)-(i) we assessed the impact of varying the number of exchanged components on the IRC with a default failure intensity of 50%. As expected, our proposed

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3636600

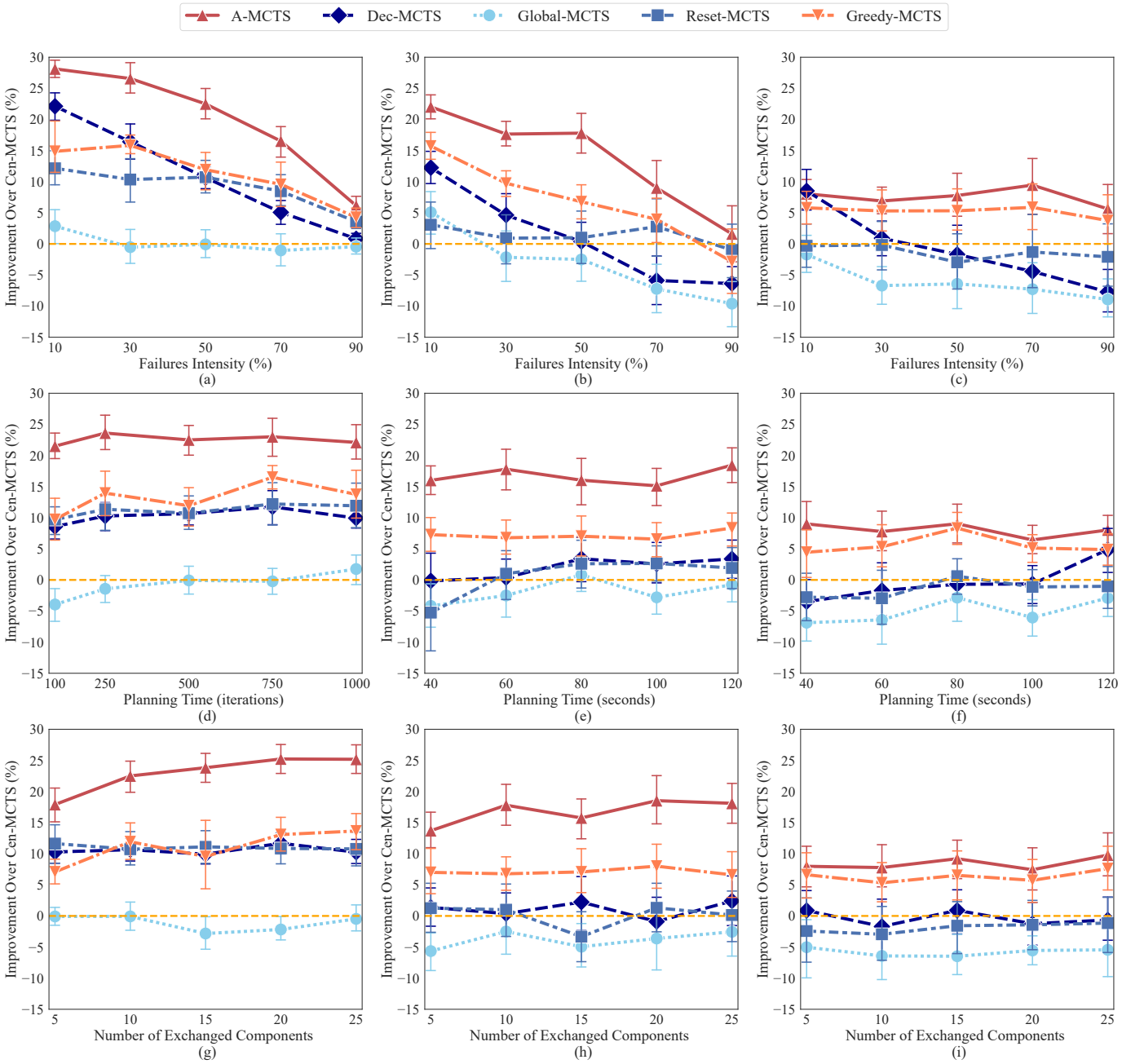

Fig. 5. Impact of failures intensity (the fraction of agents that fail) (top), planning time (middle), and number of exchanged components (bottom) on the algorithm's performance for different environment models: *DC* (left), *ORI* with perfect communication (middle), and *ORI* with practical communication (right). Results are with 95% confidence interval.

algorithm's performance improves with greater information exchange, whereas the discounted algorithms exhibit no such benefit. Indeed, with more exchanged components, the utility of the joint policy discovered by regret matching also improves. However, given the finite number of optimal policies in a multi-agent game, increasing the number of exchanged components eventually yields diminishing returns.

6.2.2 Impact of Parameter Settings

In this section, we investigate the impact of the system's key parameters on the performance of our solutions, with a special focus on its scalability and robustness. Specifically, we evaluate the impact of the travel budget B, the number of agents N, the number of rewards, and the value of the geometric decay factor α. Fig. 6 illustrates the impact of these parameters on the instantaneous reward coverage, for a default failure intensity of 50%.

As shown in Fig. 6(a)–(c), A-MCTS consistently outperforms other distributed approaches and retains a clear advantage over the centralized method, even as the action budget increases and decentralized planning becomes more challenging. A similar trend is observed when scaling the number of agents (Fig. 6(d)–(f)). However, as either the travel budget or the number of agents grows large, the

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3636600

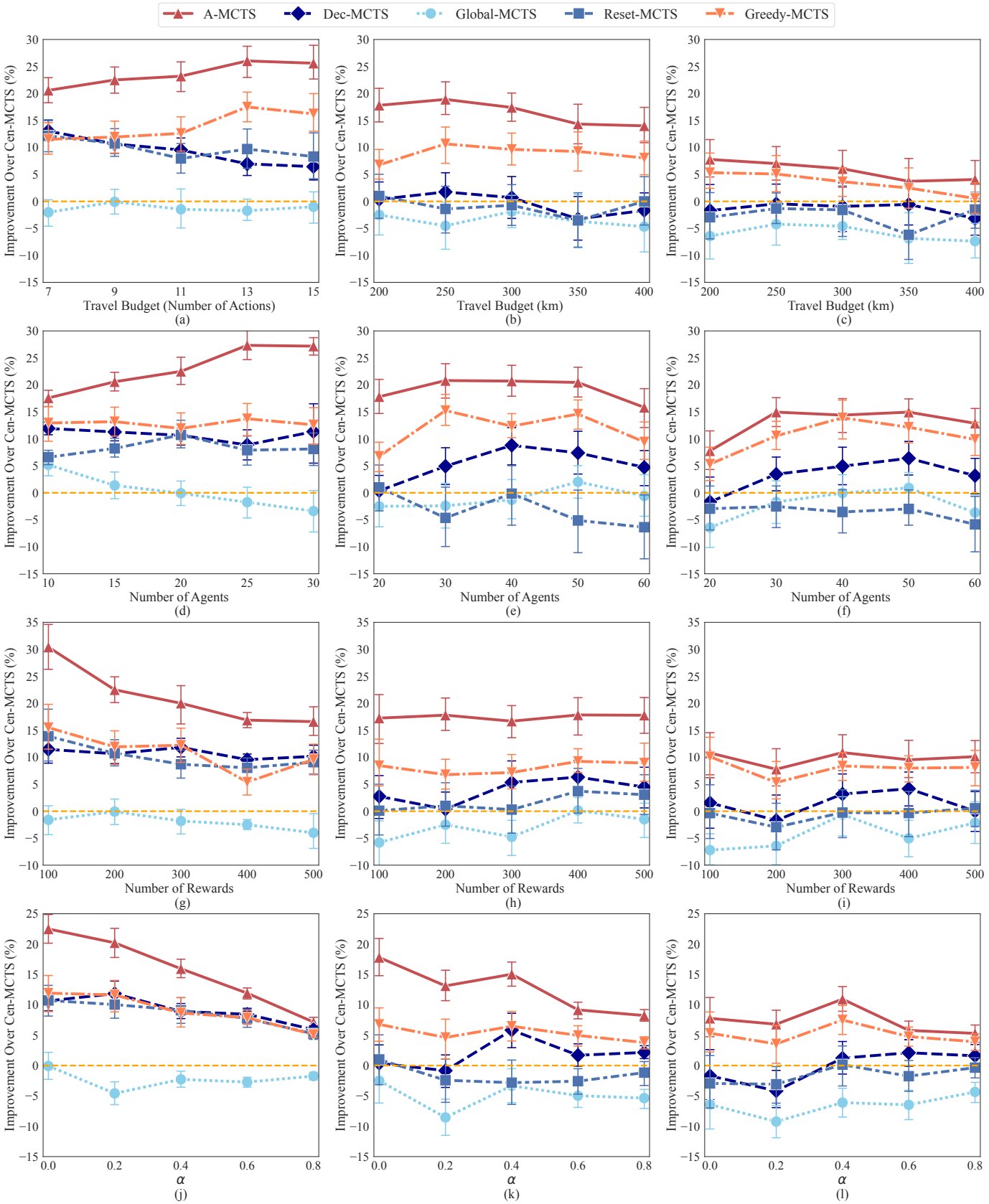

Fig. 6. Impact of travel budget (first row), number of agents (second row), number of rewards (third row), and revisit factor α (last row) on the algorithm's performance for different environment models: *DC* (left), *ORI* with perfect communication (middle), and *ORI* with practical communication (right). Results are with 95% confidence interval.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3636600

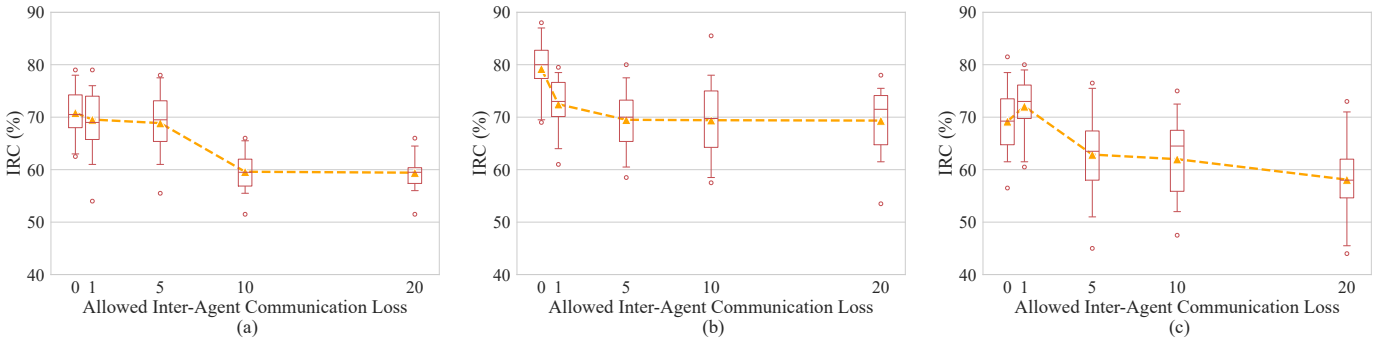

Fig. 7. Impact of allowed inter-agent communication loss on the performance of A-MCTS for different environment models: *DC* (a), *ORI* with perfect communication (b), and *ORI* with practical communication (c).

relative benefit of decentralized planning diminishes due to the finite number of rewards. In these cases, agents must also traverse edges leading to less valuable regions, for example, those farther from dense reward clusters. We further study the effect of reward density by varying the number of rewards within the same area. As expected, increasing the number of rewards enlarges the effective coverage region. Nevertheless, Fig. 6(g)–(i) shows that A-MCTS maintains consistent improvements over other methods under this scaling.

Finally, we study the impact of the geometric decay factor α that regulates the value of revisits. As shown in Fig. 6(j)-(l), the improvement of A-MCTS over other baselines decreases as α increases. This is because as α approaches 1, the diminishing-return effect is weakened, even though the submodularity property still holds. As a result, the practical severity of attrition is reduced since the revisits remain nearly as valuable as the first visits. However, even in this regime, A-MCTS still consistently outperforms all baselines, confirming its coordination and planning advantages.

Taken together, these results demonstrate that A-MCTS is a scalable and robust distributed planner for cooperative multi-agent systems under attrition risks.

6.2.3 Trade Off Between Communication Loss and Attrition for A-MCTS Analysis

In our approach, repeated communication loss serves as an indicator of agent attrition. However, in practical applications, inter-agent communication is often unreliable. If the algorithm is overly sensitive to communication loss, it can mistakenly treat delayed messages as signs of agent failures.

To investigate this issue, in this section, we examine how varying levels of tolerance to communication loss impact the performance of A-MCTS. Specifically, we parameterize this tolerance by the number of communication failures an agent must experience with another agent before being classified as lost to attrition. Fig. 7 shows the IRC against different thresholds for allowable inter-agent message loss. In the *DC* scenario, A-MCTS shows no noticeable degradation with up to 5 instances of message loss. However, in the *ORI* with a perfect communication scenario, the performance declines rapidly as the algorithm becomes more loss-tolerant. Indeed, the surviving agents are unable to detect attrition quickly enough, impairing their ability to adapt efficiently.

TABLE 4
Optimality analysis of regret matching.

Parameters	Value	PFO (%)	RNO (%)
Actions per Component	7	100	-
	9	95	96.47
	11	90	97.89
	13	85	99.06
	15	70	98.72
Number of Agents	2	100	-
	3	100	-
	4	90	96.66
	5	95	97.22
	6	95	96.47
Number of Components	10	95	96.47
	11	85	98.78
	12	85	98.81
	13	85	98.47
	14	85	98.47
Revisit Factor α	0.0	95	96.47
	0.2	80	99.78
	0.4	85	99.54
	0.6	95	99.83
	0.8	95	99.85

This is due to the increased complexity and larger scale of the *ORI* scenario, characterized by a non-uniform reward distribution. Interestingly, in the same *ORI* scenario but under a practical communication model, A-MCTS performs better when the tolerance is low. This outcome aligns with expectations, as realistic multi-agent communication is often intermittent, and a loss-tolerant algorithm helps prevent false positives regarding attrition.

6.2.4 A Closer Look At Regret Matching Behavior

In this section, we study the optimality of the NE policy computed by the regret matching algorithm in A-MCTS in the context of the multi-agent data collection from UWSN problem (the *DC* scenario). We use the following two metrics to evaluate the performance of our algorithm:

- *Probability that the NE policy is optimal in a given setting (PFO)*:

$$PFO = \frac{1}{T}\sum_{t=1}^{T} \mathbb{1}_{\{p(t)=p^*(t)\}}$$

- *Ratio between the utility of the NE policy and the optimal in a given setting (RNO)*:.

$$RNO = \frac{U_g(p(t))}{U_g(p^*(t))}$$

The optimal strategy is computed using exhaustive search. Table 4 shows the results of this study with a default number of agents of 6, number of components per agent of 10, number of actions per component of 9, and geometric decay factor α of 0. As expected, the probability of finding the optimal strategy decreases as the number of agents, components, or actions increases. This is because the game size or complexity grows exponentially with these parameters, thus potentially causing regret matching to get stuck at local optimal points. Nevertheless, even in such cases, A-MCTS consistently achieves at least 96% of the optimal value, demonstrating strong robustness. Finally, varying the revisit factor α has no noticeable effect on either PFO or RNO, confirming that our regret-matching coordination mechanism remains effective for any submodular utility function, regardless of the strength of diminishing returns.

7 Conclusions

Efficiently coordinating multi-agent planning for information gathering poses a significant challenge in real-world applications with attrition risks. In our work, we introduced a novel approach to address this issue. Our proposed algorithm, Attritable MCTS (A-MCTS), enables effective coordination among agents by allowing all agents to collaboratively optimize global utility through a new coordination technique based on regret matching and adapt to attrition. Our empirical results show that A-MCTS significantly outperforms the best existing methods regarding global utility and scalability, especially in environments with high attrition rates. As a follow-up, we plan to adapt our algorithm for applications with non-submodular reward functions, enabling its application to more general and complex environments.

The problem formulation considered in this paper is general in that we are focused on submodular reward functions, which naturally arise in a broad range of information-gathering applications. While the generalized utility model used in our experimental section captures diminishing-but-non-zero marginal gains, it remains a simplified surrogate compared with the information-theoretic objectives commonly used in high-stakes monitoring tasks (e.g., mutual information, conditional entropy, or Age of Information). Extending A-MCTS to operate under such dynamic, correlation-aware reward models, therefore, represents an important direction for future research.

Another promising avenue is to incorporate heterogeneous agents into both the system model and theoretical analysis. Our current formulation assumes homogeneous agents with identical travel budgets, motion constraints, and traversal costs. Relaxing these assumptions to account for heterogeneous vehicle capabilities and energy models would substantially enhance the applicability of our approach to real-world multi-robot systems.

References

[1] A. Dorri, S. S. Kanhere, and R. Jurdak, "Multi-agent systems: A survey," *Ieee Access*, vol. 6, pp. 28 573–28 593, 2018.

[2] J. Xie and C.-C. Liu, "Multi-agent systems and their applications," *Journal of International Council on Electrical Engineering*, vol. 7, no. 1, pp. 188–197, 2017.

[3] K. Zhang, Z. Yang, and T. Başar, "Multi-agent reinforcement learning: A selective overview of theories and algorithms," *Handbook of Reinforcement Learning and Control*, pp. 321–384, 2021.

[4] P. A. Trodden, "Robust distributed control of constrained linear systems," Ph.D. dissertation, University of Bristol, 2009.

[5] D. Claes, F. Oliehoek, H. Baier, K. Tuyls *et al.*, "Decentralised online planning for multi-robot warehouse commissioning," in *AAMAS*, 2017, pp. 492–500.

[6] L. Kocsis, C. Szepesvári, and J. Willemson, "Improved monte-carlo search," *Univ. Tartu, Estonia, Tech. Rep*, vol. 1, 2006.

[7] G. Best, O. M. Cliff, T. Patten, R. R. Mettu, and R. Fitch, "Dec-mcts: Decentralized planning for multi-robot active perception," *Int. J. Robot. Res.*, vol. 38, no. 2-3, pp. 316–337, 2019.

[8] M. Li, W. Yang, Z. Cai, S. Yang, and J. Wang, "Integrating decision sharing with prediction in decentralized planning for multi-agent coordination under uncertainty." in *IJCAI*, 2019, pp. 450–456.

[9] N. Nguyen, D. Nguyen, J. Kim, G. Rizzo, and H. Nguyen, "Multi-agent data collection in non-stationary environments," in *IEEE 23rd WoWMoM*. IEEE, 2022, pp. 120–129.

[10] G. Cybenko and R. A. Hallman, "Attritable multi-agent learning," in *Disrupt. Sci. Technol. V*, vol. 11751. International Society for Optics and Photonics, 2021, p. 117510L.

[11] O. Avner and S. Mannor, "Concurrent bandits and cognitive radio networks," in *ECML PKDD*. Springer, 2014, pp. 66–81.

[12] J. Rosenski, O. Shamir, and L. Szlak, "Multi-player bandits–a musical chairs approach," in *ICML*. PMLR, 2016, pp. 155–163.

[13] M. K. Hanawal and S. J. Darak, "Multiplayer bandits: A trekking approach," *IEEE Transactions on Automatic Control*, vol. 67, no. 5, pp. 2237–2252, 2021.

[14] S. Hart and A. Mas-Colell, "A simple adaptive procedure leading to correlated equilibrium," *Econometrica*, vol. 68, no. 5, pp. 1127–1150, 2000.

[15] B. of Ocean Energy Management, "Boem data center: Platform structures." accessed: 2024-08-13. [Online]. Available: https://www.data.boem.gov

[16] X. Yao, X. Wang, F. Wang, and L. Zhang, "Path following based on waypoints and real-time obstacle avoidance control of an autonomous underwater vehicle," *Sensors*, vol. 20, no. 3, p. 795, 2020.

[17] D. S. Bernstein, R. Givan, N. Immerman, and S. Zilberstein, "The complexity of decentralized control of markov decision processes," *Math. Oper. Res.*, vol. 27, no. 4, pp. 819–840, 2002.

[18] F. A. Oliehoek and C. Amato, *A concise introduction to decentralized POMDPs*. Springer, 2016.

[19] M. T. Spaan, G. J. Gordon, and N. Vlassis, "Decentralized planning under uncertainty for teams of communicating agents," in *AAMAS*, 2006, pp. 249–256.

[20] M. Lauri and F. Oliehoek, "Multi-agent active perception with prediction rewards," in *NeurIPS*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 13 651–13 661.

[21] L. Matignon, L. Jeanpierre, and A.-I. Mouaddib, "Coordinated multi-robot exploration under communication constraints using decentralized markov decision processes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 26, no. 1, 2012, pp. 2017–2023.

[22] P. Long, T. Fan, X. Liao, W. Liu, H. Zhang, and J. Pan, "Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning," in *ICRA*. IEEE, 2018, pp. 6252–6259.

[23] J. Capitan, M. T. Spaan, L. Merino, and A. Ollero, "Decentralized multi-robot cooperation with auctioned pomdps," *The International Journal of Robotics Research*, vol. 32, no. 6, pp. 650–671, 2013.

[24] H. Zhang, J. Chen, H. Fang, and L. Dou, "A role-based pomdps approach for decentralized implicit cooperation of multiple agents," in *2017 13th IEEE ICCA*. IEEE, 2017, pp. 496–501.

[25] M. Lauri, D. Hsu, and J. Pajarinen, "Partially observable markov decision processes in robotics: A survey," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 21–40, 2022.

[26] A. Czechowski and F. A. Oliehoek, "Decentralized mcts via learned teammate models," in *IJCAI*, 2020, pp. 450–456.

[27] S. Choudhury, J. K. Gupta, P. Morales, and M. J. Kochenderfer, "Scalable anytime planning for multi-agent mdps," in *AAMAS*, 2021, pp. 341–349.

[28] M. Li, Z. Cai, W. Yang, L. Wu, Y. Xu, and J. Wang, "Dec-sgts: Decentralized sub-goal tree search for multi-agent coordination," in *AAAI*, vol. 35, no. 13, 2021, pp. 11 282–11 289.

[29] R. Williams, B. Konev, and F. Coenen, "Multi-agent environment exploration with ar. drones," in *Advances in Autonomous Robotics Systems: 15th Annual Conference, TAROS 2014, Birmingham, UK, September 1-3, 2014. Proceedings 15*. Springer, 2014, pp. 60–71.

[30] F. F. Lizzio, E. Capello, and G. Guglieri, "A review of consensus-based multi-agent uav implementations," *Journal of Intelligent & Robotic Systems*, vol. 106, no. 2, p. 43, 2022.

[31] C. Dinelli, J. Racette, M. Escarcega, S. Lotero, J. Gordon, J. Montoya, C. Dunaway, V. Androulakis, H. Khaniani, S. Shao *et al.*, "Configurations and applications of multi-agent hybrid drone/unmanned ground vehicle for underground environments: A review," *Drones*, vol. 7, no. 2, p. 136, 2023.

[32] A. Goeckner, X. Li, E. Wei, and Q. Zhu, "Attrition-aware adaptation for multi-agent patrolling," *IEEE Robotics and Automation Letters*, 2024.

[33] J. Cohen, J.-S. Dibangoye, and A.-I. Mouaddib, "Open decentralized pomdps," in *2017 IEEE 29th ICTAI*. IEEE, 2017, pp. 977–984.

[34] M. Chandrasekaran, A. Eck, P. Doshi, and L. Soh, "Individual planning in open and typed agent systems," in *UAI*, 2016, pp. 82–91.

[35] A. Eck, M. Shah, P. Doshi, and L.-K. Soh, "Scalable decision-theoretic planning in open and typed multiagent systems," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 7127–7134.

[36] A. Kakarlapudi, G. Anil, A. Eck, P. Doshi, and L.-K. Soh, "Decision-theoretic planning with communication in open multiagent systems," in *UAI*. PMLR, 2022, pp. 938–948.

[37] B. Schlotfeldt, V. Tzoumas, and G. J. Pappas, "Resilient active information acquisition with teams of robots," *IEEE Transactions on Robotics*, vol. 38, no. 1, pp. 244–261, 2021.

[38] L. Zhou, V. Tzoumas, G. J. Pappas, and P. Tokekar, "Distributed attack-robust submodular maximization for multi-robot planning," *IEEE Transactions on Robotics*, vol. 38, no. 5, pp. 3097–3112, 2022.

[39] L. Zhou and V. Kumar, "Robust multi-robot active target tracking against sensing and communication attacks," *IEEE Transactions on Robotics*, vol. 39, no. 3, pp. 1768–1780, 2023.

[40] W. Xue, W. Qiu, B. An, Z. Rabinovich, S. Obraztsova, and C. K. Yeo, "Mis-spoke or mis-lead: Achieving robustness in multi-agent communicative reinforcement learning," in *Proceedings of the 21st AAMAS*, 2022, pp. 1418–1426.

[41] Y. Sun, R. Zheng, P. Hassanzadeh, Y. Liang, S. Feizi, S. Ganesh, and F. Huang, "Certifiably robust policy learning against adversarial multi-agent communication," in *The Eleventh ICLR*, 2022.

[42] H. L. Fung, V. Darvariu, S. Hailes, and M. Musolesi, "Trust-based consensus in multi-agent reinforcement learning systems," in *The First Reinforcement Learning Conference*, 2024.

[43] N. Nguyen, D. Nguyen, G. Rizzo, and H. Nguyen, "United we stand: Decentralized multi-agent planning with attrition," in *ECAI 2024*. IOS Press, 2024, pp. 3421–3428.

[44] G. Best, J. Faigl, and R. Fitch, "Online planning for multi-robot active perception with self-organising maps," *Autonomous Robots*, vol. 42, no. 4, pp. 715–738, 2018.

[45] N. Nguyen, D. Nguyen, J. Kim, G. Rizzo, and H. Nguyen, "Decentralized coordination for multi-agent data collection in dynamic environments," *IEEE Transactions on Mobile Computing*, pp. 1–16, 2024.

[46] M. Corah and N. Michael, "Efficient online multi-robot exploration via distributed sequential greedy assignment." in *Robotics: Science and Systems*, vol. 13, 2017.

[47] Y. Satsangi, S. Whiteson, F. A. Oliehoek, and M. T. Spaan, "Exploiting submodular value functions for scaling up active perception," *Autonomous Robots*, vol. 42, no. 2, pp. 209–233, 2018.

[48] P. Vansteenwegen, W. Souffriau, and D. Van Oudheusden, "The orienteering problem: A survey," *European Journal of Operational Research*, vol. 209, no. 1, pp. 1–10, 2011.

[49] W. Xu, Z. Xu, J. Peng, W. Liang, T. Liu, X. Jia, and S. K. Das, "Approximation algorithms for the team orienteering problem," in *IEEE INFOCOM*. IEEE, 2020, pp. 1389–1398.

[50] D. H. Wolpert, S. R. Bieniawski, and D. G. Rajnarayan, "Probability collectives in optimization," *Handbook of Statistics*, vol. 31, pp. 61–99, 2013.

[51] N. Rezazadeh and S. S. Kia, "Distributed strategy selection: A submodular set function maximization approach," *Automatica*, vol. 153, p. 111000, 2023.

[52] G. Qu, D. Brown, and N. Li, "Distributed greedy algorithm for multi-agent task assignment problem with submodular utility functions," *Automatica*, vol. 105, pp. 206–215, 2019.

[53] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions—i," *Mathematical programming*, vol. 14, no. 1, pp. 265–294, 1978.

[54] K. Berg and T. Sandholm, "Exclusion method for finding nash equilibrium in multiplayer games," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[55] D. Nguyen, L. White, and H. Nguyen, "Social optimum equilibrium selection for distributed multi-agent optimization," *arXiv preprint arXiv:2307.13242*, 2023.

[56] P. N. Brown, H. P. Borowski, and J. R. Marden, "Are multiagent systems resilient to communication failures?" *arXiv preprint arXiv:1710.08500*, 2017.

[57] P. N. Brown and J. R. Marden, "On the feasibility of local utility redesign for multiagent optimization," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 3396–3401.

[58] G. A. Hollinger, S. Yerramalli, S. Singh, U. Mitra, and G. S. Sukhatme, "Distributed coordination and data fusion for underwater search," in *2011 IEEE ICRA*. IEEE, 2011, pp. 349–355.

[59] ——, "Distributed data fusion for multirobot search," *IEEE Transactions on Robotics*, vol. 31, no. 1, pp. 55–66, 2014.

[60] M. Stojanovic, "On the relationship between capacity and distance in an underwater acoustic communication channel," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 11, no. 4, pp. 34–43, 2007.

[61] G. Han, Z. Tang, Y. He, J. Jiang, and J. A. Ansere, "District partition-based data collection algorithm with event dynamic competition in underwater acoustic sensor networks," *IEEE Trans. Ind. Informat.*, vol. 15, no. 10, pp. 5755–5764, 2019.

[62] M. Huang, K. Zhang, Z. Zeng, T. Wang, and Y. Liu, "An auv-assisted data gathering scheme based on clustering and matrix completion for smart ocean," *IEEE Internet Things J.*, vol. 7, pp. 9904–9918, 2020.

[63] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. Robot. Automat.*, vol. 12, no. 4, pp. 566–580, 1996.

[64] B. Barsky and T. DeRose, "Geometric continuity of parametric curves: three equivalent characterizations," *IEEE Comput. Graph. Appl.*, vol. 9, no. 6, pp. 60–69, 1989.

[65] C. Dornhege, A. Kleiner, A. Hertle, and A. Kolling, "Multirobot coverage search in three dimensions," *Journal of Field Robotics*, vol. 33, no. 4, pp. 537–558, 2016.

Nhat Nguyen received the BEng (Honours) degree in Electrical and Electronic at the University of Adelaide in 2020. He is currently pursuing the PhD degree with the School of Computer and Mathematical Sciences, the University of Adelaide. His research focuses on sequential decision-making algorithms for multiple-agent systems.

Duong D. Nguyen received his PhD degree in Engineering at The University of Adelaide in 2018. Following his PhD, he served as a Postdoctoral Researcher at the same institution. In 2023, he joined as a Research Scientist at the Defence Science and Technology Group, Australia. His research interests include game theory models and decision-making algorithms for autonomous systems.

Dr Junae Kim earned her Ph.D. in Computer Engineering with a focus on computer vision and machine learning from the Australian National University. Since 2013, she has been serving as an AI specialist at DSTG (Defence Science and Technology Group) in Australia, where her research centres around artificial intelligence, machine learning and cybersecurity.

Gianluca Rizzo is Associate Professor of Computer Science at Università di Torino , Italy, and Senior Research Associate at HES-SO Valais, Switzerland. Previously, he has been with Institute IMDEA Network, and Adjunct Professor at UC3M, Madrid. He received his M.Sc. in EE from Politecnico di Torino in 2001, and his PhD in Computer Science in 2008 from EPFL, Switzerland. His main research interests are in the performance evaluation of distributed systems.

Hung Nguyen is a Professor in the School of Computer and Mathematical Sciences and the leader of the Information Warfare and Advanced Cyber theme within the Defence Trailblazer, the University of Adelaide. He obtained his PhD in computer and communication sciences from EPFL. His current research interest is in models and algorithms for improving network performance, security, and resiliency, especially for IoT and wireless networks.