Robust Online Epistemic Replanning of Multi-Robot Missions

Lauren Bramblett, Branko Miloradović, Patrick Sherman, Alessandro V. Papadopoulos, and Nicola Bezzo

***Abstract*— As Multi-Robot Systems (MRS) become more affordable and computing capabilities grow, they provide significant advantages for complex applications such as environmental monitoring, underwater inspections, or space exploration. However, accounting for potential communication loss or the unavailability of communication infrastructures in these application domains remains an open problem. Much of the applicable MRS research assumes that the system can sustain communication through proximity regulations and formation control or by devising a framework for separating and adhering to a predetermined plan for extended periods of disconnection. The latter technique enables an MRS to be more efficient, but breakdowns and environmental uncertainties can have a domino effect throughout the system, particularly when the mission goal is intricate or time-sensitive. To deal with this problem, our proposed framework has two main phases: i) a centralized planner to allocate mission tasks by rewarding intermittent rendezvous between robots to mitigate the effects of the unforeseen events during mission execution, and ii) a decentralized replanning scheme leveraging epistemic planning to formalize belief propagation and a Monte Carlo tree search for policy optimization given distributed rational belief updates. The proposed framework outperforms a baseline heuristic and is validated using simulations and experiments with aerial vehicles.**

***Note*—Videos are provided in the supplementary material and also at `https://www.bezzorobotics.com/lb-iros24`.**

I. Introduction

As robotics and artificial intelligence continue to evolve, Multi-Robot Systems (MRS) have emerged as a particularly intriguing research domain. At the core of MRS research lies the concept of robots working together to achieve shared objectives. This collaboration can be seen in various applications, from search and rescue operations to underwater exploration. To collaborate effectively, robots must communicate and coordinate their activities, but challenges often arise when communication is limited or compromised. Consider a scenario in which an MRS has limited communication. In a multi-robot mission, there are typically no policies in place for limited connectivity. As such, failures or disturbances can cause the entire operation to be inefficient or compromised because robots cannot adapt to new information.

In this work, we focus on the following question: *How can we ensure cooperative and efficient behavior for task allocation when a centralized predefined plan must change at runtime?* This question is an expansion of our previous work [1], allowing for the elimination of certain limiting suppositions and making the work more suitable for practical scenarios where tasks are known but unforeseen changes in the environment or MRS may occur. Our proposed solution has two main components: 1) a centralized mission planner that accounts for intermittent rendezvous, promoting the discovery of failures and inefficiencies in the MRS if something does not go according to plan, and 2) an efficient runtime plan adaptation that leverages our recent epistemic planning research [2] to reason about the likely knowledge and intentions of others based on the current epistemic state and dynamically reassign tasks. Our proposed framework enables MRS to cooperate, given limited communication and an uncertain operating environment.

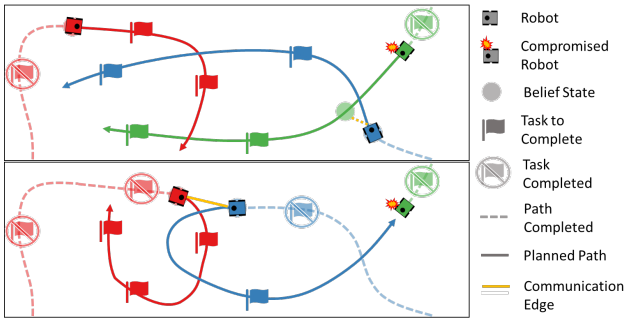


Fig. 1. Pictorial representation of the problem presented in the paper. The green robot fails, and the blue robot observes that its belief is false. The blue robot routes to share this information with the red robot, reallocating tasks in the environment before searching for the green robot.

Consider the example in Fig. 1, where three robots complete tasks based on an initial centralized plan. During disconnection, each robot maintains a set of possible *belief* states for other robots and a set of *empathy* states that other robots might believe about it. In the top frame, the blue robot realizes that its belief of the green robot is false. It then communicates this to the red robot, and consequently, the red and blue robots reallocate their tasks (bottom frame). The blue robot is assigned to locate the green robot to ensure all tasks are completed. In this manner, robots can successively reason based on their local observations.

The contributions of this work are two-fold: i) a genetic algorithm for multi-robot mission planning in a centralized manner, accounting for intermittent rendezvous at user-defined priorities, and ii) an epistemic planning framework for local replanning, utilizing a Monte Carlo tree search to maximize policy reward based on knowledge and beliefs about the system and environment. To the best of our knowledge, this is the first paper combining epistemic logic with runtime task allocation adaptations with intermittent communication. We show that our method outperforms a baseline heuristic in which robots complete their assigned tasks before backtracking to find faulty robots.

II. Related Work

Task allocation and planning are challenging problems that have attracted researchers from different disciplines.

Lauren Bramblett, Patrick Sherman, and Nicola Bezzo are with the Departments of Systems and Information Engineering and Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22904, USA. Email: {qbr5kx, ukw4tc, nb6be}@virginia.edu

Branko Miloradović and Alessandro Vittorio Papadopoulos are with the Division of Intelligent Future Technologies, Mälardalen University, Sweden. Email: {branko.miloradovic,alessandro.papadopoulos}@mdu.se

The Traveling Salesman Problem (TSP) [3], a well-studied problem in operations research, is often used to model the planning challenges encountered by a single robot. Later, this formulation was extended to include multiple vehicles (mTSP) in [4]. The mTSP is more suitable for large-scale applications but is more complex than the TSP. Several solutions have been proposed to solve this problem, such as the genetic algorithm (GA) [5], [6], which considers tasks that require specific vehicle types, and [7], which uses a consensus-based bundling algorithm for limited replanning, but few works have included communication restrictions and failures. An approach, presented in [8], utilizes an auction allocation algorithm to assign tasks but assumes enough locally connected robots to perform the assigned tasks. Other works, such as [9], address the issue of prolonged disconnections using rendezvous locations. However, this can lead to unnecessary communication and laborious backtracking. In these environments, robots may operate with outdated or incomplete information while also being aware of the possibility of misinformation. A robot may act on information that it believes to be accurate, only to discover later that it is outdated or incorrect. This can have significant consequences, particularly in critical applications such as disaster response or military operations. In [10], system failures are considered in the multi-agent policy search, but it is assumed that robots can communicate these disruptions.

In contrast, this work applies dynamic epistemic logic (DEL) [11], allowing each robot in the MRS to reason and plan using its beliefs of other robots in the system while disconnected, updating its beliefs and policy if new events are observed, and routing to communicate when necessary. DEL is a formal logic that describes how beliefs and knowledge change and has recently been integrated into robotics applications. The method presented in [12] recreates the Sally-Anne psychological test for human-robot interactions. Typical DEL-based multi-agent research uses epistemic planning for game theory-based policies [13]. We evaluate the use of DEL for an MRS application, equipping each robot with the ability to reason about the system's state, considering local observations. Our proposed solution expands on previous mTSP research. It shows that intermittent rendezvous will allow the MRS to reason about the system's state and share any new knowledge, updating its beliefs at runtime utilizing an epistemic planning framework.

III. Preliminaries

A. Notation, Communication, & Control

Consider an MRS of m robots in the set $\mathcal{A}$. We assume that all initial positions of the robots are known. The MRS's communication range is denoted as r_c, and a robot i and j can communicate when within this range.

The variable $\mathcal{V}_i \subseteq \mathcal{V}$ represents the subset of all tasks $\mathcal{V}$ assigned to a robot i. We let $\boldsymbol{x}_i(t)$ denote the state variable of the robot i that evolves according to general dynamics:

$$\boldsymbol{x}_i(t+1) = \mathbf{g}(\boldsymbol{x}_i(t), \boldsymbol{u}_i(t), \boldsymbol{\nu}_i(t))$$

where $\boldsymbol{u}_i \in \mathbb{R}^{d_u}$ and the variable $\boldsymbol{\nu}_i \in \mathbb{R}^{d_\nu}$ denote the control input and zero-mean Gaussian process uncertainty at time t. The tuple $\Omega^i = (q_i, \mathcal{V}_i)$ is called the robot i's *disposition* and is defined by the capability of the robot i and its assigned tasks. The capability q_i of robot i is informed by its kinematic specifications, such as maximum velocities and physical dimensions.

B. Epistemic Logic

For this application, the epistemic language, $\mathcal{L}(\Psi, AP, \mathcal{A})$ is obtained as follows in Backus-Naur form [14]:

$$\phi ::= H(\eta) \mid \phi \wedge \phi \mid \neg\phi \mid K_i\phi \mid B_i\phi$$

where $i \in \mathcal{A}$. $H \in \Psi$ with Ψ being a set of functions that describe the system state. η generally denotes function arguments. $\neg\phi$ and $\phi \wedge \phi$ are propositions that can be negated and form logical conjunctions, where $\phi \in AP$ and AP is a finite set of atomic propositions. $K_i\phi$ and $B_i\phi$ are interpreted as "robot i knows ϕ" and "robot i believes ϕ", respectively.

The distributed knowledge and reasoning for robots in the system are modeled using epistemic logic [12]. An epistemic state for AP is represented by the tuple $s = (W, (R_i)_{i\in\mathcal{A}}, L, W_d)$ where W is a finite set of possible worlds, $R_i \subset W \times W$ is an accessibility relation for robot i simplified to R for reference to all robots, $L : W \to AP$ assigns a labeling to each world defined by its true propositions, and $W_d \subseteq W$ is the set of designated worlds from which all worlds in W are reachable. The initial epistemic state is denoted as $s_0 = (W, R, V, \{w_0\})$. If $W_d = \{w_0\}$, s_0 is the global epistemic state. The world, w, signifies a set of true propositions that, in our application, is the disposition of each robot. The worlds that exist for the system are defined by the combinations of all possible dispositions of each robot in the MRS (e.g., task assignment, velocity). The truth of $\mathcal{L}$-formulas in epistemic states is defined with standard semantics similar to [12]:

$$\begin{aligned}
(W, R, L, W_d) &\models \phi \text{ iff } \forall w \in W_d, (W, R, L, w) \models \phi \\
(W, R, L, w) &\models \phi \text{ iff } \phi \in L(w) \text{ where } \phi \in AP \\
(W, R, L, w) &\models K_i\phi \text{ iff } \forall v \in W, \text{ if } (w, v) \in R_i \\
&\qquad \text{then } (W, R, L, v) \models \phi \\
(W, R, L, w) &\models C\phi \text{ iff } \forall v \in W, \text{ if } (w, v) \in \cup_{i\in\mathcal{A}} \\
&\qquad \text{then } (W, R, L, v) \models \phi
\end{aligned}$$

The accessibility relation R_i represents the uncertainty of robot i at run-time for a global epistemic state $s = (W, R, L, w_d)$. In this state, the robot i cannot distinguish between the actual world w_d and any other world v where $(w_d, v) \in R_i$. Consequently, robot i's knowledge is based on what is true in all of these worlds v. Sequences of relations are used to represent higher-order knowledge. For example, the statement "robot i knows that robot j knows ϕ" is true in s if and only if $s \models K_iK_j\phi$. This condition is satisfied when ϕ is true in all worlds accessible from w_d through the composite relation of R_i and R_j. The perspective of robot i is defined as $s_i = (W, R, L, \{v \mid (w, v) \in R_i; w \in W_d\}$. If s is the global state, then s_i is the perspective of robot i on s. In this work, we represent a subset of these perspectives as particles moving through the environment in the set

$$\mathcal{P}_i = \{p_{ij,b} \ \forall j \in \mathcal{A}, \forall b \in \mathcal{B}\}. \tag{1}$$

where beliefs $b \in \mathcal{B}$ are a finite set of particles for each robot i that represent how a robot j would behave given a different, but accessible, world $w \in W_d$.

Dynamic epistemic logic is expanded from epistemic logic through action models [12]. These models affect a robot's perception of an event and influence its set of reachable worlds, R_i. A robot may plan to reduce the run-time uncertainty by taking actions. We simplify the notation of the action model by referring to actions in plain language. The action library, A, is the set of actions that a robot can enact during mission execution. We express the epistemic product model as $s \otimes i : a = (W', L_i', V', W_d')$ where $s \otimes i : a$ represents the new epistemic state after the action $a \in A$ has been enacted by robot i. A planning task is represented by the tuple $\Pi = (s, A, \gamma)$. An execution policy π is a sequence of actions in A for robots in the MRS that will satisfy the common mission objective γ given an epistemic state s.

IV. Problem Formulation

In this work, we assume that all robots know the location of all tasks $\mathcal{V}$ present in the environment and the initial location of all robots in the system. Given a limited communication range r_c, this approach aims to minimize the total time to complete all tasks in the environment since robots can experience failures or disturbances during operation. We formally define our problems as:

Problem 1: ***Centralized mTSP Planner with Intermittent Communication:*** Design a strategy for an MRS to complete all tasks while weighing efficient rendezvous points. The goal is to minimize the mission's duration, considering that faults and disturbances may occur during execution, necessitating a communication and replanning mechanism.

Problem 2: ***Robust Online Replanning:*** Formulate a policy for robust online replanning of the team's operations when faults or disturbances decrease the original plan's efficiency. The policy should minimize the time to complete all tasks, considering any necessary communications with disconnected robots and the deprecated state of the system.

The mathematical formulation of Problem 1 matches the one of mTSP [15] where we seek to minimize the longest tour of any robot represented by Q. However, in our work, robots can have different starting and ending depots. We also allow robots to have no tasks assigned to them. The optimization problem is expressed in its epigraph representation, where the objective function is included in the constraints as

$$\begin{aligned} \min \quad & Q \\ \text{s.t.} \quad & \sum_{i \in \mathcal{V}^\Sigma} \sum_{j \in \mathcal{V}^\Delta} \omega_{ijs} \cdot x_{ijs} \leq Q, \qquad \forall s \in \mathcal{A}, \end{aligned} \tag{2}$$

where ω_{ijs} is the cost of traveling from task i to task j for robot $s \in \mathcal{A}$. The binary decision variable x_{ijs} defines if the robot s travels from task i to task j. The sets $\mathcal{V}^\Sigma$ and $\mathcal{V}^\Delta$ represent the inclusion of all the tasks and all the starting depots, and all the tasks and the ending depot, respectively. The goal of the optimization process is to minimize the variable Q, also known as "minMax" optimization, where we minimize all robots' maximum tour or makespan.

V. Approach

Our proposed framework is designed for an open mTSP in which robots are not required to return to their starting location; instead, each robot has a starting and ending depot. When solving the mTSP, we promote intermittent communication by rewarding robot interactions during their respective tours, allowing robots to share information or realize that the original plan has changed. To realize changes, each robot propagates belief and empathy states to allow robots to observe the local environment, reason about system operations while disconnected, and adjust local plans when necessary. For ease of discussion, let us consider two robots, i and j. From the perspective of the robot i, a *belief state*, $p_{ij,b} \in \mathcal{P}_i$, represents a possible state of a robot j and an *empathy state*, $p_{ii,b} \in \mathcal{P}_i$, is robot i's belief about itself from the perspective of other robots. With this knowledge, robot i predicts and tracks empathy states to decrease the number of locations in which robot j may search for robot i and allows the system to complete all tasks more efficiently, given new operational constraints. The diagram in Fig. 2 summarizes this architecture, where the centralized planner first routes robots to tasks in the environment, assessing the solution's fitness by minimizing the maximum tour length of a robot and rewarding intermittent communication based on a user's preferred settings. If robots disconnect, the set of belief and empathy particles, $\mathcal{P}_i$ propagates according to the sequence of actions, π_0 set by the centralized planner. If the robot locally observes system changes, epistemic planning allows each robot to determine the best series of actions to estimate the positions of lost robots, share any necessary information with other robots, and navigate to the remaining cities.

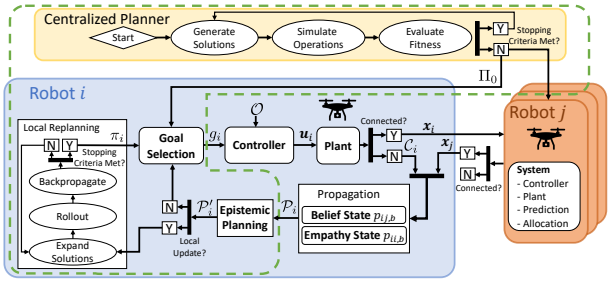


Fig. 2. Diagram of the proposed approach. The contributions of this paper are within the green box.

A. The Centralized Planning Algorithm

The centralized planner used in this work is based on a GA adapted to solve combinatorial optimization problems, specifically mTSP. Chromosomes are encoded as a set of arrays, where each array encodes a robot's plan. A graphical representation of a single chromosome is given in Fig. 3. The size of each array is equal to the sum of the number of robots and tasks, that is, $n+m$. The elements of the array are integer task IDs. Following the task chain, the robot's route can be extracted from the encoding. For example, in Fig. 3, if we look at Robot 1, we can see that the first task in its plan is 5, and the next task ID is then stored in column 5, which is Task 7. This continues until a destination depot with the ID of $n+m$, 10 in this example, is reached. The initial population is generated randomly to start with a high diversity and seeded in the feasible region of the search space.

The crossover operator is a modified version of Edge Recombination Crossover (ERX) [16]. The first step is to select two parents for crossover from the mating pool. The mating pool is generated, accounting for the crossover probability and each individual's fitness. Next, an adjacency

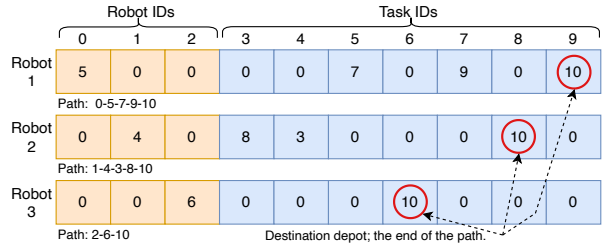


Fig. 3. Graphical representation of chromosome encoding.

matrix that contains the makeup of neighboring tasks based on the two chosen parents is constructed. We then randomly select a starting task and the selection chain continues by randomly selecting a task from a neighboring list of previously allocated tasks. We randomly select a new task if all neighboring tasks are already allocated. We apply a jump mutation that randomly changes the placement of a single task in the route, and the swap mutation selects two tasks and swaps their locations. Jump and swap mutations are invoked twice: the first for intra-robot mutations and the second for inter-robot mutations, such that there are both local and global mutation operations.

Greedy Search (GS) and 2-opt [17] are two local refinement methods implemented to reorder cities within a robot's plan resulting from the GA allocation of cities to salesmen. Local refinement methods exploit the candidate solution by reordering the list of tasks governed by the nearest-neighbor or 2-opt. In the next sections, we explain our modifications to increase the robustness of the MRS.

B. Interaction Reward Mechanism

The general rule for creating a good plan for TSP or mTSP is to have routes that do not cross. The most successful heuristic for solving these problems directly exploits this rule, e.g., 2-opt or Lin–Kernighan heuristics [18], but, in this work, we take a different approach by allowing the planner to create *interactions* between robots. Within this framework, we define an *interaction* as an event where robots are within the range r_c to exchange information. Rewarding robots who travel within r_c can create crossings in the resulting routes, contrary to [18]. However, we argue that this can benefit the overall execution time of the mission when the system does not operate as planned. This will enable robots to detect system failures faster during execution without laborious backtracking after reaching the depot.

To maximize the number of interactions among robots, we introduce a mechanism to reward the exchange of information between robots. However, maximizing the number of interactions alone is not sufficient, as each interaction's value must be taken into account. For example, exchanging information at the beginning or close to the end of a mission may not be beneficial, as little new information can be gained from these interactions. Furthermore, redundant interactions over small-time intervals should not be highly rewarded since no new information is likely to be shared. To capture this, we introduce, for every robot i, a time interval $[\tau_i^s, \tau_i^e]$ when the robot can be rewarded for interacting with other robots. The potential reward is designed to grow linearly from $t = \tau_i^s$ for $\Phi_i = (\tau_i^e - \tau_i^s)/2$ time units and to stay constant for the remaining part of the interval. Then, the potential reward function $U_i(x)$ for robot i is defined as:

$$U_i(x) = \begin{cases} x, & \text{if } 0 < x \leq \Phi_i \\ \Phi_i, & \text{if } \Phi_i < x \leq 2\Phi_i, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

However, the actual reward $\mathcal{R}_i(t)$ is assigned only if the interaction happens, according to

$$\mathcal{R}_i(t) = \begin{cases} \rho U_i(t - t_i^{lr}), & \text{if robot } i \text{ interacts at time } t \in [\tau_i^s, \tau_i^e], \\ 0, & \text{otherwise,} \end{cases} \tag{4}$$

where t is the current time, and t_i^{lr} is the time when the last reward was assigned to robot i. Furthermore, ρ is half of the average distance between tasks, and it is introduced as a weight related to the structure of the problem. This means that ρ scales with the problem instance. The total reward is then calculated as:

$$\mathcal{R}_{tot} = \sum_{t=0}^{Q} \sum_{i \in \mathcal{A}} \mathcal{R}_i(t). \tag{5}$$

However, two robots may exchange information frequently while a third robot is in no contact with them. To overcome this issue, we also introduce a penalty mechanism for robots not interacting with other robots. This mechanism requires a tunable threshold, σ, to be defined, e.g., all robots have to interact with another robot at least once before completing 50% of a given mission, i.e., $\sigma = 0.5$. The penalty for failing to do so is calculated as follows:

$$P_i = \begin{cases} (t_i^{int} - \sigma \cdot t_i^{max}) \cdot \rho & \text{if } t_i^{int} > \sigma \cdot t_i^{max}, \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

where t_i^{max} is the time required for robot i to complete its plan, and t_i^{int} is the time when robot i had its first interaction with another robot. To get the total penalty, P_{tot}, we sum up the penalties over all robots. The optimization problem (2) can now be updated with rewards and penalties as follows:

$$\min \quad Q - \mathcal{R}_{tot} + P_{tot}. \tag{7}$$

We also extend the aforementioned approach if a user requires more control over the mission makespan compared to a traditional mTSP solution. In this case, we solve a bilevel optimization problem where we first minimize Q subject to (2) and then optimize the following:

$$\begin{aligned} \max \quad & \mathcal{R}_{tot} - P_{tot} - \Delta Q \\ \text{s.t} \quad & (2);\ \Delta Q \leq \delta Q^* \end{aligned} \tag{8}$$

where ΔQ represents the difference between the solution to the upper-level optimization problem (2) represented by Q^* and the inner optimization task in (8). The user-defined variable $\delta \in [0, 1]$ represents the extent to which the typical mTSP makespan minimization can be worsened to increase interactions using (5) and (6). In this way, we have better control over the quality of the produced solution, with the mission duration upper-bounded by the user-defined limit.

C. Belief & Empathy Propagation

So far, we have explained how we developed a centralized strategy that allows intermittent interactions. Now, we transition to online adaptations, indicated by the blue section in Figure 2, to plan based on the information gained from these interactions. In our framework, each robot propagates belief states for all robots in the MRS. This allows a robot i to plan according to its beliefs about other robots and to empathize with what other robots expect robot i to do while disconnected. Each robot predicts the future states of a set of beliefs for all robots in the system and will follow the closest empathy state even if a malfunction occurs, allowing a robot only to propagate a finite number of beliefs represented by the set $\mathcal{P}_i$ in (1). A robot i defines its empathy particles as $\mathcal{P}_i^e = \{p_{ii,b}\ \forall b \in \mathcal{B}\}$ and its belief particles about other robots as $\mathcal{P}_i^r = \{p_{i,j,b}\ \forall j \in \mathcal{A}, \forall b \in \mathcal{B}\}$. If disconnected, a robot i propagates beliefs according to the last globally communicated epistemic state between robot i and robot j, moving particles based on how robots would behave given a subset of true propositions from the set AP introduced in Sec. III-B. Initially, we note that all robots know the initial position and disposition of all robots, defined by the centralized plan in Sec. V-A. Particles are propagated along the tour provided by the centralized planning algorithm. Given that all robots follow an empathy particle during exploration, we next present our strategy to update the epistemic state if changes occur at runtime.

D. Epistemic Replanning

Robots follow the centralized plan initially, but if operations do not go as planned and a robot experiences a failure or other robots communicate changes to the system, a rational belief update must occur. We formulate a belief update for this application as: (i) A robot updates its own belief given it cannot operate as expected; (ii) a robot updates its belief about another robot, given it is not traveling according to a previously expected belief; (iii) a robot communicates a belief update about a disconnected robot to a subset of connected robots within the communication range. In all these scenarios, a robot can update its execution policy of tasks in the environment, given its new belief about the MRS to complete all tasks in the environment, which is equivalent to satisfying the common goal γ from Sec. III-B. However, belief updates can have cascading effects across the MRS if new information is not communicated efficiently and on time. Therefore, we introduce a hierarchical framework to update allocations at runtime and when failures or disturbances occur and robots can no longer follow the original plan.

1) Epistemic Updates: Establishing a mechanism for logical updates is important to determine when or if a robot should find or communicate with other members of its team and how to reach a consensus on disconnected team members within a locally connected team. A dynamic epistemic logic (DEL) framework allows a robot i to succinctly share beliefs and update a robot's perspective s_i on the epistemic state s. There are two cases where updates can occur: i) when connected to all robots and ii) when expecting to connect with another robot. From the established semantics in Sec. III-B, we know $A = \{percieve(\phi), announce(\phi), complete(\phi)\}$ The action *complete* represents a robot completing a task, *perceive* symbolizes a robot observing a generic proposition ϕ about the MRS, and *announce* constitutes communication with a locally connected team. The set $\Psi = \{track\}$ is functionally interpreted for $B_i\, track(p_{ij,b})$ as the robot i knows that the robot j is tracking the belief particle b.

First, we address a belief update when robots are within communication range. We assume that because robots are cooperative, all belief updates are accepted and are only outdated if an event occurs, such as system failures or disturbances. These updates are announced such that the epistemic state from robot i's perspective is:

$$s_i \otimes announce(\Omega) = s_i' \models K_i V(\Omega^i) \bigwedge_{j \in \mathcal{A}} K_i K_j V(\Omega^j)\ \forall i \in C. \quad (9)$$

where $announce(\Omega)$ is an action symbolizing the announcement of all robots' dispositions, Ω. The notation models robot i's knowledge of the dispositions of all robots, and the function $V(\Omega^i)$ maps the dispositions of the robot i to atomic propositions in the set AP. The set $C \subseteq \mathcal{A}$ represents the set of robots within robot i's communication range. The belief particles are updated from the announcement of all states to the MRS such that

$$p_{ij,b} \leftarrow \Omega^j,\ \forall (i,j) \in \mathcal{A}^2,\ \forall b \in \mathcal{B}. \quad (10)$$

Since beliefs are shared according to (9), the particles in this set are propagated according to the dispositions of each robot. For example, in a three-robot team, if robot 1 communicates with robots 2 and 3 that it will execute robot 3's tasks, all robots would propagate a belief particle that moves robot 1 according to its assigned tasks.

The *perceive* action causes a robot to change his belief in the epistemic world. If robot i perceives that robot j is not at its believed location, it updates its epistemic state with the epistemic action *percieve*:

$$s_i \otimes i : perceive(\neg track(p_{ij,b})) \models V(\Omega^j, B_i \neg track(p_{ij,b})) \quad (11)$$

where the function V takes two arguments, mapping robot i's updated belief about robot j to an atomic proposition in AP and robot i's new epistemic state is evaluated as the epistemic product after *perceive* has been enacted. The particle propagation does not change in this case since robots may seek out other robots without knowledge of this belief update. In the event of a malfunction or fault of a robot, the robot updates its belief in the same way with $j = i$ and tracks respective empathy particle $p_{ii,b+1}$.

In this way, the knowledge of disconnected robots is not affected, nor does the robot i update its belief that a disconnected robot would know the updated information. With our epistemic states and actions defined, we now describe how these concepts can be used for planning. As stated in Sec. III-B, a planning task for the MRS is defined by the tuple $\Pi = (s, A, \gamma)$ where γ is a goal formula. In plain language, the goal formula is to complete all tasks. We define the epistemic update associated with a robot who completes an assigned task $v \in \mathcal{V}$ as:

$$s_i \otimes i : complete(v) \models V(\Omega^i, B_j complete(v))\ \forall j \in \mathcal{A}. \quad (12)$$

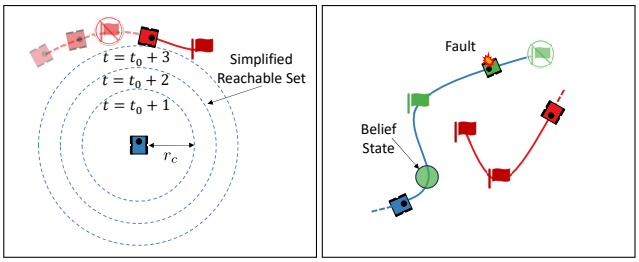
The variable Ω^i is also updated to represent the new disposition of robot i. Given that other robots are also tracking

the believed location of robot i, belief updates occur without communication, although these beliefs may be incorrect if a failure has occurred. Thus, an augmented policy that allows the MRS to achieve the common goal must be enacted.

In the event of a malfunction, we introduce two new types of tasks that allow operational robots to gather the necessary information about the system's condition and complete any unfinished tasks. These types of tasks are called *gossiping* and *finding*. Given robot i's belief and the planned interactions with other robots according to the mTSP solution (7), a robot should communicate before any planned interaction. Ensuring the completion of communication tasks (gossiping) and promptly identifying malfunctioning robots are vital steps to facilitate accurate information exchange and prevent the spread of misinformation within the system. The estimation of the interaction point can be determined using the time-based trajectory of the belief state and the reachable set of the robot involved, as depicted in Fig. 4(a). The position of robot j's belief state at a specific time t is denoted as $p_{ij,b}(t)$. To find the point where robot i's communication range intersects with the communication range of robot j's predicted location, we find the minimum timestep t_r that satisfies the equation:

$$\|\boldsymbol{x}_i(t_r) - \boldsymbol{p}_{ij,b}(t_r)\| - R(t_r) > r_c \tag{13}$$

where $\|\boldsymbol{x}_i(t_r) - \boldsymbol{p}_{ij,b}(t_r)\|$ represents the distance between the location components of the robot's position, $\boldsymbol{x}_i$ and robot i's belief about robot j, $\boldsymbol{p}_{ij,b}$. The reachable set, R, for robot i expands at every timestep based on the robot's velocity. If all belief states have been checked for a deprecated robot j, a robot i backtracks along the previously established path until robot j is located. An example is shown in Fig. 4(b), where a blue robot routes backward along the green robot's path to communicate and reallocate any remaining tasks.



(a) Planning for a dynamic task (b) Finding at an unknown location

Fig. 4. Examples of tasks generated as a result of belief updates

2) Balanced Workload Partitioning: Given the limited nature of communication in this application, robots first assign new tasks to connected robots before optimizing their path [19] so that robots do not need to maintain a connection while optimizing routes to new tasks. Robots instead partition tasks based on a balanced workload and accounting for any belief updates. For example, suppose two robots, i and j, are connected, and robot k is not within the communication range. In that case, robot i might believe that robot k is functioning according to the initial state, s_0, but robot j did not *perceive* robot k at its respective belief state $p_{ik,b}$. So robot j *announces* its belief to robot i. Robots i and j then bid on the new task, which is to find robot k. We let $\mathcal{V}_c$ be the set of tasks the connected robots must complete, and the cost function for allocating a task to a robot is user-defined (e.g., distance, time). Algorithm 1 presents the bidding mechanism used in this paper, noting that this is only instigated if a belief update about the MRS functionality has occurred (i.e., a fault or disturbance). Once the task allocations have been determined, the next step is to find the optimal tour.

Algorithm 1 Balanced Workload Partitioning

$tour_s \equiv \emptyset \quad \forall s \in C$
for each $v \in \mathcal{V}_c$ **do**
  for each $s \in C$ **do**
    $bid_s = cost(tour_s \cup v)$
  $winner = \arg\min_{s \in C}(bid_s)$
  $tour_{winner} \leftarrow tour_{winner} \oplus v$

3) Monte Carlo Tree Search: As mentioned in Sec. I, we combine Monte Carlo Tree Search (MCTS) with a DEL to implicitly coordinate plans when the system does not operate as originally intended. Referring to Sec. V-C, to limit the policy search space, each robot's state consists of believing that a robot j is following one of the particles represented in robot i's set of particles $\mathcal{P}_i$. MCTS is applied to complex games such as chess or Go to find the next best move, even in real-time [20]. To model the solution space effectively, we use the current epistemic state from a robot i's perspective s_i. The MCTS algorithm simulates changes to the epistemic state when an action is taken and is represented as $s_i' \sim s_i \otimes a$. Robots add *gossiping* or *finding* tasks based on local observations, as discussed in Sec. V-D.1 when 1) a robot experiences a fault or disturbance, or 2) a robot i observes that its epistemic belief about the state of robot j is incorrect.

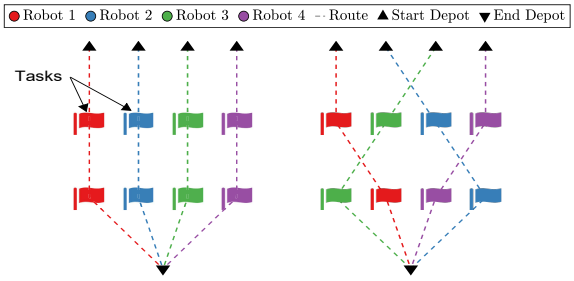
In this work, the search tree is generated by repeating the four steps – *selection*, *expansion*, *simulation*, and *back-propagation* – until a certain termination condition is met, which in this approach is a certain number of simulations. In the selection stage, a leaf node that has not yet been fully expanded is selected. We employ the upper confidence bound applied to trees (UCT) technique, which is typical in MCTS, to decide which vertex to simulate from the root node. Specifically, UTC was chosen because it has been shown to strike a good balance between exploration and exploitation [20]. Expansion occurs by randomly applying a random action or, in this case, adding a random task to a robot's route. The simulation then performs a random remaining route until termination (i.e., all of the robots' allocated tasks are performed) and then backpropagates the reward, applying the estimated value to the expanded node in the expansion step. The MCTS seeks to maximize the negative time it takes for a robot to complete all its assigned tasks, estimating the time to *find* and *gossip* with robots using the methods in Fig. 4. We summarize the MCTS simulation applied in this approach in Algorithm 2. The tour with the lowest estimated cost is the chosen execution policy, π for all the robots in the system that will satisfy γ.

To aid the reader in understanding, the proposed approach is implemented on the toy example shown in Fig. 5. We show the trivial solution to the mTSP in Fig. 5(a) and the modified mTSP solution considering the interaction reward from (5) in Fig. 5(b). In Fig. 6, we show a subset of frames from

Algorithm 2 MCTS - Simulate

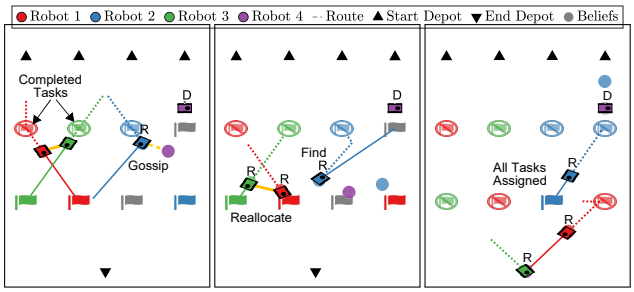
tour = child.tour(*child_rollout*)
cost = 0
for each $c \in tour$ **do**
  if *type*(*c*) = "Static Robot" **then**
    path =reverse(*particle*(*c*).*tasks*)
  else if *type*(*c*) = "Dynamic Robot" **then**
    path = find_intersect_point(*particle*(*c*).*tasks*)
  else
    path = task_location(*c*)
  cost += time_to_traverse(*path*)
reward = −*cost*

the approach in which the blue robot realizes that the purple robot is deprecated in Fig. 6(a), gossips the information to the red and green robots who reallocate remaining tasks while the blue robot is charged with finding the purple robot in Fig. 6(b). In Fig. 6(c), the blue robot has communicated with the purple robot and routes to the remaining task in the environment before returning to the home base.



(a) Ideal: 61m makespan (b) Ours: 69m makespan

Fig. 5. Ideal mTSP allocation for 4 robots is shown in (a) and (b) is the solution with our proposed method.



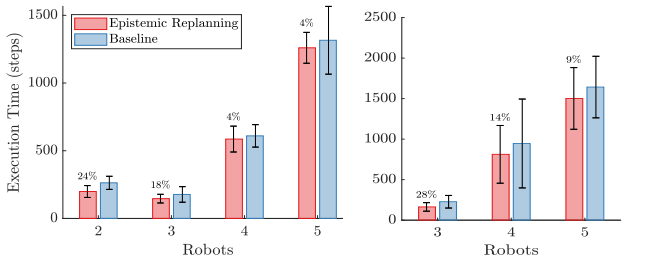
(a) (b) (c)

Fig. 6. Our approach on a toy example, where the purple robot fails, and the blue robot realizes that the purple robot is not where expected.

VI. Simulations

This section showcases the outcomes obtained through MATLAB simulations of our method executed by teams of two to five robots. The square environment used for the simulations has dimensions of 30×30, 30×30, 90×90, and 150×150 [m], and for each scenario, a total of 10, 10, 30, and 50 tasks were generated. The locations of the tasks were randomly generated for each scenario.

In our proposed approach, each robot propagates three particles. The initial maximum speed of each vehicle is 5 [m/s], and the second and third particles travel at a linear speed that is 80% and 60% of the vehicle's maximum speed, respectively. The maximum communication range is 5 [m] from the center of the robot. In our simulations, we implemented one fault for teams of two to five robots and two for teams of three to five robots, randomly occurring to any robot, causing the affected robot to track its second or third empathy particle or fail (i.e., zero velocity). Our approach was compared to a baseline heuristic in which the routes are determined by minimizing their makespan from (2) and backtracking to find team members who do not arrive at the depot when expected. We let δ equal 30% to increase the number of interactions between robots from (8) such that the makespan of our solution can be up to 30% longer than the baseline heuristic solution. We compare with this baseline to determine whether increased interactions and epistemic replanning truly improved the outcome. As shown in Fig. 7, our approach outperforms the baseline heuristic by a significant margin in all scenarios, and we note that the margin increases as the number of failures increases between Fig. 7(a) and Fig. 7(b).



(a) Scenarios with 1 failure (b) Scenarios with 2 failures

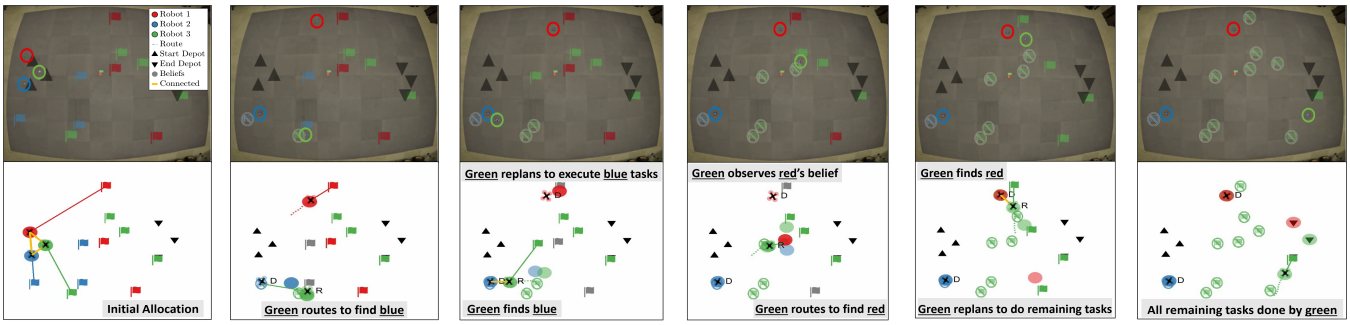
Fig. 7. Comparison of a baseline heuristic and the proposed approach.

Discussion – We emphasize that the margin of improvement is smaller as the teams become larger because information sharing becomes more inefficient as interactions between all robots become sparse. This introduces an interesting expansion outside the scope of this work for introducing optimal sub-teaming to create more efficient information sharing for static or dynamic teams during operations. In addition, not all vehicles are equally likely to fail. As vehicles age, they may become less reliable, requiring more dependable vehicles to take over or pick up additional tasks if a vehicle's operating capacity is deprecated during operations [21].

VII. Experiments

Our approach was validated through several laboratory experiments with a multi-robot team. The team consists of several Bitcraze Crazyflies that used a Vicon motion capture system for localization. Vehicles start within the communication range to complete all tasks in the environment. The experiments were carried out in a 4×5.5 [m] space with a sensing and communication range of 0.5 [m] for each robot. The results of a sample experiment with ten tasks and three Crazyflies are shown in Fig. 8.

As shown in the figure, each robot is assigned a subset of tasks in Fig. 8(a). After disconnection, the blue robot fails and the green robot observes that the blue robot is not where expected in Fig. 8(b); the green robot backtracks along the blue robot's path and finds the blue robot in Fig. 8(c). In Fig. 8(d), the green robot also observes that the red robot is not where expected. The green robot backtracks along the



(a) (b) (c) (d) (e) (f)

Fig. 8. Snapshots and results of an experimental case study.

path of the red robot and finds the red robot in Fig. 8(e). The green robot then replans all remaining tasks before ending at the depot in Fig. 8(f).

VIII. Conclusion

This paper presents a novel framework for multi-robot systems to use a modified centralized planning method to assign tasks, accounting for intermittent interactions. These interactions enable the system to use epistemic planning, which adapts to faults and disturbances by reassigning tasks based on a robot's reasoning about the system while disconnected. This method allows an MRS to disconnect and cooperatively plan based on a set of belief and empathy states if the system does not function as intended. The generalized task allocation algorithm uses these belief states to assign tasks while considering the potential need to communicate with disconnected robots, facilitating dynamic task allocation without constant communication. We show the improvement of our framework compared to a baseline heuristic over several scenarios and apply our framework to real-world experiments. Future research includes addressing the challenges of improving strategies for more complex environments. Additionally, we would like to reduce the computation time for task allocation and optimize the necessary belief propagation and interactions for a larger multi-robot system by dividing the team into sub-teams. Outdoor experiments are also on our agenda.

IX. Acknowledgements

This work is based on research sponsored by: Northrop Grumman through the University Basic Research Program, the Swedish Research Council (VR) with the PSI project No. #2020-05094, and the Knowledge Foundation (KKS) with the MARC project No. #20240011.

References

[1] L. Bramblett, S. Gao, and N. Bezzo, "Epistemic prediction and planning with implicit coordination for multi-robot teams in communication restricted environments," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 5744–5750.

[2] L. Bramblett and N. Bezzo, "Epistemic planning for multi-robot systems in communication-restricted environments," *Frontiers in Robotics and AI*, vol. 10, p. 1149439, 2023.

[3] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Journal of the operations research society of America*, vol. 2, no. 4, pp. 393–410, 1954.

[4] M. Bellmore and S. Hong, "Transformation of multisalesman problem to the standard traveling salesman problem," *Journal of the ACM (JACM)*, vol. 21, no. 3, pp. 500–504, 1974.

[5] B. Miloradović, B. Çürüklü, M. Ekström, and A. V. Papadopoulos, "GMP: A genetic mission planner for heterogeneous multirobot system applications," *IEEE Transactions on Cybernetics*, vol. 52, no. 10, pp. 10 627–10 638, 2021.

[6] M. Frasheri, B. Miloradović, L. Esterle, and A. V. Papadopoulos, "GLocal: A hybrid approach to the multi-agent mission re-planning problem," in *2023 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2023, pp. 1696–1703.

[7] J. Chen, X. Qing, F. Ye, K. Xiao, K. You, and Q. Sun, "Consensus-based bundle algorithm with local replanning for heterogeneous multi-uav system in the time-sensitive and dynamic environment," *The Journal of Supercomputing*, vol. 78, no. 2, pp. 1712–1740, 2022.

[8] M. Otte, M. J. Kuhlman, and D. Sofge, "Auctions for multi-robot task allocation in communication limited environments," *Autonomous Robots*, vol. 44, no. 3, pp. 547–584, 2020.

[9] Y. Gao, Y. Wang, X. Zhong, T. Yang, M. Wang, Z. Xu, Y. Wang, Y. Lin, C. Xu, and F. Gao, "Meeting-merging-mission: A multi-robot coordinate framework for large-scale communication-limited exploration," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 13 700–13 707.

[10] S. Al-Hussaini, J. M. Gregory, and S. K. Gupta, "Generation of context-dependent policies for robot rescue decision-making in multi-robot teams," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4317–4324.

[11] H. Van Ditmarsch, W. van Der Hoek, and B. Kooi, *Dynamic epistemic logic*. Springer Science & Business Media, 2007, vol. 337.

[12] T. Bolander, L. Dissing, and N. Herrmann, "Del-based epistemic planning for human-robot collaboration: Theory and implementation," in *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, vol. 18, no. 1, 2021, pp. 120–129.

[13] B. Maubert, S. Pinchinat, F. Schwarzentruber, and S. Stranieri, "Concurrent games in dynamic epistemic logic," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, 2021, pp. 1877–1883.

[14] D. E. Knuth, "Backus normal form vs. backus naur form," *Communications of the ACM*, vol. 7, no. 12, pp. 735–736, 1964.

[15] R. Necula, M. Raschip, and M. Breaban, "Balancing the subtours for multiple tsp approached with acs: Clustering-based approaches vs. minmax formulation," in *EVOLVE-A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation VI*. Springer, 2018, pp. 210–223.

[16] L. D. Whitley, T. Starkweather, and D. Fuquay, "Scheduling problems and traveling salesmen: The genetic edge recombination operator," in *ICGA*, vol. 89, 1989, pp. 133–40.

[17] M. Jünger, G. Reinelt, and G. Rinaldi, "The traveling salesman problem," *Handbooks in operations research and management science*, vol. 7, pp. 225–330, 1995.

[18] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Operations research*, vol. 21, no. 2, pp. 498–516, 1973.

[19] P. Mazdin and B. Rinner, "Distributed and communication-aware coalition formation and task assignment in multi-robot systems," *IEEE Access*, vol. 9, pp. 35 088–35 100, 2021.

[20] L. Kocsis and C. Szepesvári, "Bandit based monte-carlo planning," in *European conference on machine learning*. Springer, 2006, pp. 282–293.

[21] S. B. Stancliff, J. Dolan, and A. Trebi-Ollennu, "Planning to fail—reliability needs to be considered a priori in multirobot task allocation," in *2009 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2009, pp. 2362–2367.