Communicating Intent as Behaviour Trees for Decentralised Multi-Robot Coordination

Rhett Hull[1,2], Diluka Moratuwage[1], Emily Scheide[3], Robert Fitch[1], Graeme Best[1]

***Abstract*— We propose a decentralised multi-robot coordination algorithm that features a rich representation for encoding and communicating each robot's intent. This representation for "intent messages" enables improved coordination behaviour and communication efficiency in difficult scenarios, such as those where there are unknown points of contention that require negotiation between robots. Each intent message is an adaptive policy that conditions on identified points of contention that conflict with the intentions of other robots. These policies are concisely expressed as behaviour trees via algebraic logic simplification, and are interpretable by robot teammates and human operators. We propose this intent representation in the context of the Dec-MCTS online planning algorithm for decentralised coordination. We present results for a generalised multi-robot orienteering domain that show improved plan convergence and coordination performance over standard Dec-MCTS enabled by the intent representation's ability to encode and facilitate negotiation over points of contention.**

I. Introduction

Multi-robot systems are increasingly being deployed in large-scale and unstructured outdoor environments, such as the ocean [1], underground tunnel networks [2], and farms [3]. These multi-robot teams are required to coordinate their actions so that the robots collectively achieve the objectives of the mission. It becomes increasingly difficult to achieve effective coordination in scenarios that require robots to recognise and negotiate over points of contention, such as junctions that lead to high-reward regions. Ideally, this decision making should be decentralised, such that each robot is making online decisions based on locally-available information while communicating their "intent"—that is, the planned motion policy of the robot—as the communication infrastructure permits.

Decentralised Monte Carlo tree search (Dec-MCTS) [4] has emerged as a powerful decentralised coordination algorithm with demonstrated success in application domains ranging from agriculture [5] to surveillance [6] and onboard platforms including ground [5, 7, 8], aerial [6], and marine [9] robots. In Dec-MCTS, each robot asynchronously cycles between three steps: (1) incrementally plan this robot's actions using a new variant of Monte Carlo tree search,

This research is supported in part by the Commonwealth of Australia and the Centre for Advanced Defence Research in Robotics and Autonomous Systems, Australia.

[1]Robotics Institute, University of Technology Sydney, NSW, Australia. rhett.c.hull@student.uts.edu.au, {diluka.moratuwage, robert.fitch, graeme.best}@uts.edu.au.

[2]Defence Science and Technology Group, Department of Defence, Australia. rhett.hull1@defence.gov.au.

[3]Collaborative Robotics and Intelligent Systems (CoRIS) Institute, Oregon State University, USA. scheidee@oregonstate.edu.

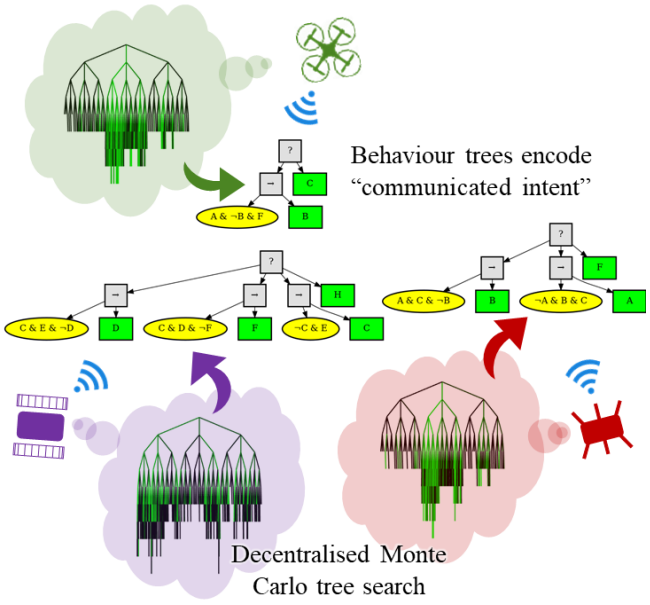

Fig. 1. Concept diagram illustrating the proposed decentralised coordination algorithm. Each robot incrementally searches for promising action sequences using a variant of Monte Carlo tree search (MCTS). The MCTS tree is periodically compressed into the form of a behaviour tree that is conditional on the intended actions of other robots, which facilitates negotiation over recognised points of contention. These behaviour trees are communicated asynchronously between robots and used to update the plans of each robot to collectively improve mission performance.

(2) represent this intended plan in a compressed form, and (3) communicate this plan to and from other robots.

In [4], the communicated intent representation in the second step is proposed to be a probability distribution over action sequences. This representation results in a higher likelihood of convergence compared to communicating just the single-best action sequence. However, to facilitate decentralised optimisation of these probability distributions, this representation assumes independence between the action selection of each robot, which limits the effectiveness in scenarios involving points of contention requiring negotiation. Here, we seek an alternative representation that overcomes these challenges while still being in a concise form suitable for low-bandwidth communication infrastructure.

We propose employing behaviour trees (BTs) as a new representation for expressing intent to be communicated between robots during multi-robot coordination. BTs are a data structure for conveniently representing policies that have gained wide-spread use in computer games and a growing popularity in robotics [10]. BTs encode an adaptive policy and can provide real-time feedback to human operators

regarding behaviour selection and system state [2]. Fig. 1 illustrates our proposed algorithm in action where BTs are generated and communicated between robots.

In order to incorporate BTs as the communication representation within Dec-MCTS, two key algorithmic challenges need to be addressed that we solve here. Firstly, the current intent of a robot needs to be extracted from the partially-expanded MCTS search tree; we achieve this by finding promising action sequences in the tree, evaluating them with respect to other robots' actions, and encoding them as a concise BT through algebraic logic simplification. Each generated BT is a policy that is conditional on the intended actions of other robots that are recognised as points of contention. Secondly, the plan of a robot should be directly influenced by the communicated intent of other robots; we achieve this by evaluating the BTs of other robots within the reward calculations of the MCTS rollouts.

We present simulated experiments for a problem domain formulated as a generalisation of the multi-robot orienteering problem. This problem domain is representative of important information gathering problems, including coverage, exploration, and logistics. We formulate a problem instance that requires the robots to negotiate over a contentious junction. Our results show improved coordination performance over standard Dec-MCTS and baseline methods. Overall, the proposed method enables the robots to effectively negotiate over a point of contention and adapt to the changing intent of other robots to find high-performing joint plans.

II. Related Work

Decentralised algorithms are typically more suitable than centralised algorithms for multi-robot systems operating in environments without fixed communication infrastructure. Dec-POMDP based approaches typically involve centralised offline computation that enables decentralised execution [11]. The advantage of instead performing decentralised computation and execution is that it typically allows robots to adapt online to unforeseen events, and many multi-robot systems in practice follow this paradigm [2, 12–15]. The Dec-MCTS algorithm [4] is a popular approach for this paradigm, and is generally applicable to a wide range of robotics problems. A key step in Dec-MCTS and other decentralised planners is communicating intended plans between robots [16].

We investigate using BTs to facilitate this communication of intent. A BT (as illustrated in the figures throughout this paper) is a directed tree structure that models an adaptive policy [10, 17]. *Condition* nodes (ellipses) are evaluated as true (success) or false (failure) based on observations of the system and the environment. An *action* node (rectangle) describes a behaviour that is executed when the node becomes active and evaluates as success, running, or failure, as defined by the implementation of the behaviour. *Fallback* nodes ("?") are true (success) if any of their children return success, while *sequence* nodes ("→") are false (failure) if any of their children return failure. Determining which nodes are active is performed with a recursive depth-first process controlled by the logic of *fallback* and *sequence* nodes. The left-most children of an active *fallback* node are active up to the first child evaluated as true/success or running. Similarly, the left-most children of an active *sequence* node are active up to the first false/failure or running.

BTs have been commonly used in computer games and have recently gained popularity in robotics due to the ease of design and online introspection [2, 17–22]. BTs used in practice today are mostly hand-designed [2, 10], which typically requires extensive time and domain expertise. Automatic generation techniques have been proposed for providing effective BTs with minimal human input, such as using an offline simulator [20, 23–26] or online synthesis [19, 27]. While a key step of our proposed algorithm is the automatic generation of BTs, the context is different here such that the BT needs to compactly and accurately encode the policy represented by a partially-expanded MCTS search tree, and thus we require a new BT generation approach. BTs have also been used in multi-robot contexts [2, 28–30], but not for facilitating the communication of intent during decentralised planning as we propose in this paper.

MCTS [31, 32] is becoming increasingly popular for online planning in robotics. The most common MCTS variant is the upper-confidence bounds applied to trees (UCT) algorithm [33]. UCT performs an asymmetric expansion of a search tree using a best-first policy that generalises the UCB1 policy [34], and is said to balance between exploration and exploitation. A key component of Dec-MCTS [4] is a new UCT variant that accounts for a changing reward distribution by generalising D-UCB [35], which is particularly relevant for decentralised multi-robot planning, and we also employ in our proposed approach.

III. Problem Formulation

Consider a set of robots $\mathcal{R}$ where each robot $r \in \mathcal{R}$ plans a discrete action sequence $\boldsymbol{x}^r = (x_1^r, x_2^r, ..., x_N^r) \in \mathcal{X}^r$ with the aim of maximising a known global reward function g that encodes the problem at hand, e.g. information gathering or task allocation. A cost c_i^r is associated with each action x_i^r. Each robot has a cost budget C^r such that all feasible action sequences have $\sum_i^N c_i^r \leq C^r$; $\mathcal{X}^r$ defines the set of all action sequences that satisfy this cost constraint. The collection of action sequences for the team of robots is denoted $\boldsymbol{x} = \{\boldsymbol{x}^1, \boldsymbol{x}^2, ..., \boldsymbol{x}^{|\mathcal{R}|}\}$.

Problem 1 (Decentralised multi-robot coordination): We define the multi-robot coordination problem as follows: find the set of action sequences $\boldsymbol{x}^*$ that maximises g; i.e.,

$$\boldsymbol{x}^* = \underset{\{\boldsymbol{x}^1, \boldsymbol{x}^2, ..., \boldsymbol{x}^{|\mathcal{R}|}\}}{\arg\max} \; g\left(\boldsymbol{x}^1, \boldsymbol{x}^2, ..., \boldsymbol{x}^{|\mathcal{R}|}\right). \tag{1}$$

This problem is to be solved in a decentralised manner, such that each robot r computes the plan $\boldsymbol{x}^r$ while considering the reward function g and the *intent* of other robots. Since the reward function and other robots' intent may change unpredictably, planning should be performed online. Additionally, it is necessary for robots to regularly communicate their intent, but this communication may be subject to delays or bandwidth limitations. The representation for these intent messages are defined as part of the proposed solution.

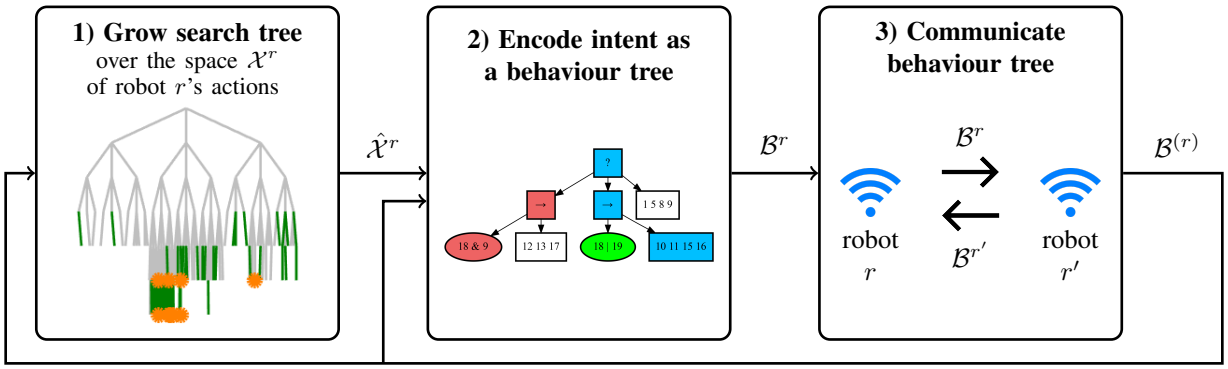

Fig. 2. Overview of the algorithm running on-board robot r. 1) The search tree is expanded by adding new actions (green). Periodically, the set of best nodes (orange) is selected. 2) Encode robot r's best action sequences into a BT by taking into account other robots' BTs. 3) If possible, the BT is communicated to other robots, and received BTs from other robots are used in subsequent planning iterations.

IV. Decentralised Coordination Facilitated by Behaviour Trees

We propose a solution to the decentralised multi-robot planning problem using BTs to communicate intent between robots. Our algorithm combines the non-myopic planning capabilities of MCTS with the expressiveness and interpretability of BTs. We extend the well-known Dec-MCTS [4] coordination algorithm with a new BT-based representation for communicating the *intent* of each robot.

As highlighted in Problem 1, robots coordinate their actions by communicating their intent. We must find a suitable encoding of intent so that it accurately conveys the current intent described by the planner, be compact for efficient communication, and be interpretable by the receiving robot (and, ideally, human operators). Therefore, we select BTs as the representation due to their expressiveness and interpretability. However, for BTs to be a useful representation of intent, we require methods for automatically generating such BTs within the planning loop of a decentralised planner.

This section begins by providing an overview of our coordination algorithm that combines Dec-MCTS with BTs, followed by our algorithm for encoding the current intent as a BT suitable for communication, and then a brief analysis.

A. Dec-MCTS with BT Intents

Our algorithm consists of three main steps, depicted in Fig. 2 and further described in Algorithm 1: 1) incrementally expand the robot's search tree, 2) encode this robot's intended plan as a BT, and 3) communicate the BT to other robots. During the GrowTree phase, the MCTS tree $\mathcal{T}^r$ is grown incrementally whilst considering the BTs of other robots during Simulate. In the encoding phase, we select a set of the most promising solutions in $\mathcal{T}^r$, along with the BTs of other robots $\mathcal{B}^{(r)}$ and generate the BT representation of the robot's intent, $\mathcal{B}^r$. Note, we use the superscript notation (r) to refer to the set $\mathcal{R} \setminus r$, where $\setminus$ denotes the set difference. During the communication, robots send their respective BTs to one another and receive BTs from all other robots.

Algorithm 2 outlines the grow tree algorithm used to incrementally expand the MCTS search tree. At line 2, we select a candidate node to expand (line 3) from the tree. Due to the expected breakpoints in reward as a result of updating behaviour trees, we use the discounted upper confidence trees (D-UCT) selection policy [4] as it yields similar analytical guarantees to standard MCTS in the context of changing reward distributions. We then run the default policy until the action cost budget C^r is exhausted in line 4 to yield an action sequence $\boldsymbol{x}^r$. In line 5, we extract the action sequences of other robots $\boldsymbol{x}^{(r)}$ by iteratively evaluating BTs in a randomised order, i.e. $\boldsymbol{x}^j \leftarrow \mathcal{B}^j(\boldsymbol{x}^i)$, $\boldsymbol{x}^k \leftarrow \mathcal{B}^k(\boldsymbol{x}^i \cup \boldsymbol{x}^j)$, $\boldsymbol{x}^l \leftarrow \mathcal{B}^l(\boldsymbol{x}^i \cup \boldsymbol{x}^j \cup \boldsymbol{x}^k)$ and so on. We do this iterative evaluation of other robots' plans to make planning more reactive to the intent of other robots, in comparison to Dec-MCTS where each probability distribution is jointly optimised but independently sampled. Given the action sequences, we can evaluate the local utility of the robot f^r, as defined on line 6, rather than g directly, as this is less sensitive to uncertainty in other robots' plans [36]. Finally we update the node statistics (mean and count) through backpropagation (line 7), which influences the subsequent selection step. To select an action to execute (line 8), the robots perform a random order round robin evaluation of their BTs in a similar manner to line 5.

Algorithm 1 Overview of Dec-MCTS with BTs

input: Objective g, computation budget B^r and cost budget C^r
output: action sequence $\boldsymbol{x}^r$ for robot r
1: $\mathcal{T}^r \leftarrow$ initialise MCTS tree
2: **while** B^r is not exceeded at iteration n **do**
3: $\quad \mathcal{B}^{(r)} \leftarrow$ CommunicateReceive
4: $\quad \mathcal{T}^r \leftarrow$ GrowTree$(\mathcal{B}^{(r)}, \mathcal{T}^r, g, C^r)$ ▷ See Alg. 2.
5: $\quad \mathcal{B}^r \leftarrow$ EncodeToBT$(\mathcal{T}^r, \mathcal{B}^{(r)})$ ▷ See Alg. 3.
6: $\quad$ CommunicateTransmit$(\mathcal{B}^r)$
7: **end while**
8: **return** $\boldsymbol{x}^r \leftarrow$ RoundRobinSelect$(\boldsymbol{x}^{(r)'})$

B. Encoding Intent with BTs

To convert the MCTS tree $\mathcal{T}^r$ into a BT we need to find a set of conditions and actions that sufficiently represents the intent of the current robot with respect to other robots. For compactness and computational efficiency, building a representation of the entire tree $\mathcal{T}^r$ is impractical; as such,

Algorithm 2 GROWTREE for robot r using Monte Carlo tree search with the D-UCT action selection policy.

input: $\mathcal{B}^{(r)}$, $\mathcal{T}^r$, g, C^r
output: $\mathcal{T}^r$
1: **for** fixed number of iterations **do**
2: $i_{d-1} \leftarrow$ SELECT($\mathcal{T}^r$) ▷ D-UCT policy
3: $[i_d, \mathcal{T}^r] \leftarrow$ EXPAND($\mathcal{T}^r, i_{d-1}$)
4: $\boldsymbol{x}^r \leftarrow$ SIMULATE($\mathcal{T}^r, \mathcal{B}^{(r)}, i_d, g, C^r$)
5: $\boldsymbol{x}^{(r)} \leftarrow$ EXTRACTSEQUENCEFROMBT($\mathcal{B}^{(r)}$, $\boldsymbol{x}^r$)
6: $f^r \leftarrow g(\boldsymbol{x}^r \cup \boldsymbol{x}^{(r)}) - g(\boldsymbol{x}^r_\emptyset \cup \boldsymbol{x}^{(r)})$ ▷ Local utility
7: $\mathcal{T}^r \leftarrow$ BACKPROPAGATE($\mathcal{T}^r, i_d, f^r$)
8: **end for**
9: **return** $\mathcal{T}^r$

our BT represents a sparse set of action sequences $\hat{\mathcal{X}}^r$ corresponding to nodes in $\mathcal{T}^r$ with the highest expected reward. For each action sequence $\boldsymbol{x}^r$, we generate a set of conditions based on the set of anticipated action sequences of other robots $\boldsymbol{x}^{(r)}$, and apply algebraic logic simplification to find a concise BT representation. The logic simplification results in an adaptive policy that spans the entire space $\hat{\mathcal{X}}^{(r)}$, seeded by the conditions associated with $\hat{\mathcal{X}}^r$.

We detail our approach to automatic generation of BTs as follows with reference to Algorithm 3. The algorithm consists of three steps: 1) create a lookup table of conditions and actions, 2) build a set of truth tables for each unique action, and 3) apply algebraic logic simplification to create a BT subtree. We repeat the last two steps to fill in the entire tree, with the subtrees connected with a fallback root node.

Constructing the LUT: First, we build a lookup table (LUT) containing (condition, action) pairs, where conditions are the action sequences of other robots $\boldsymbol{x}^{(r)}$ and the action a is defined as the action sequence $\boldsymbol{x}^r$ for robot r that is associated with $\boldsymbol{x}^{(r)}$. The LUT is filled in by first selecting a set of the nodes with the highest expected reward in the MCTS tree (line 3). Then, we evaluate the response of a random subset of other robots $(r)'$ based on their communicated BTs (line 5), using the same approach as the rollout phase (see Sec. IV-A). Then, the current robot's best action is selected from $\hat{\mathcal{X}}^r$ with the highest local utility with respect to $\boldsymbol{x}^{(r)'}$ (line 6). These actions are added as a pair to the LUT (line 7). Since the BT evaluation order is randomised, the output action sequence $\boldsymbol{x}^{(r)}$ is not deterministic.

Converting the LUT to a logical expression: Secondly, we convert this LUT into a BT (lines 9-25). We achieve this by looking at each unique a in the LUT sequentially and building a truth table (line 10-17). Each entry in the truth table for a given a consists of three elements: "true" for conditions that correspond to this a, "false" for conditions that correspond to actions that have not yet been considered, and "don't-care" for conditions that do not correspond to any a or correspond to an action that has already been considered ($\mathcal{V}$). We repeat this for all unique a sequentially, with conditions corresponding to previously considered a becoming "don't-cares" for subsequent truth tables. We do this as a BT is evaluated left to right, and therefore subtrees

Algorithm 3 Procedure ENCODETOBT for robot r, which converts robot r's current *intent* as a BT.

input: Current MCTS tree $\mathcal{T}^r$, received BTs $\mathcal{B}^{(r)}$
output: Robot r's BT, $\mathcal{B}^r$
1: $\mathcal{B}^r \leftarrow \{(?)\}$ ▷ Initialise BT with fallback as root
2: $\mathcal{L} \leftarrow \{\}$ ▷ LUT of (condition, action) pairs
▷ Create LUT of conditions $\boldsymbol{x}^{(r)}$ and actions $\boldsymbol{x}^r$
3: $\hat{\mathcal{X}}^r \leftarrow$ SELECTSETOFSEQUENCES($\mathcal{T}^r$)
4: **for** n_i iterations **do**
5: $\boldsymbol{x}^{(r)'} \leftarrow$ EXTRACTSEQSFROMBTS($\mathcal{B}^{(r)'}$, $\boldsymbol{x}^r$)
6: $\boldsymbol{x}^r \leftarrow \arg\max_{\boldsymbol{x}^r \in \hat{\mathcal{X}}^r} f^r(\boldsymbol{x}^r \cup \boldsymbol{x}^{(r)'})$
7: $\mathcal{L} \leftarrow \mathcal{L} \cup \{(\boldsymbol{x}^{(r)'}, \boldsymbol{x}^r)\}$
8: **end for**
▷ Build BT from unique actions in the LUT
9: $\mathcal{V} \leftarrow \emptyset$ ▷ Evaluated actions to propagate "don't cares"
10: **for** each unique action sequence $\boldsymbol{x}^r \in \hat{\mathcal{X}}^r$ **do**
▷ Truth table T maps conditions to Boolean logic
11: $T(c) \leftarrow$ *don't care*, $\forall c \in \mathcal{C}$
12: **for** $(c, a) \in \mathcal{L}$ **do**
13: **if** $a = \boldsymbol{x}^r$ **then** ▷ Current action sequence
14: $T(c) \leftarrow$ *true*
15: **else if** $a \notin \mathcal{V}$ **then**
16: $T(c) \leftarrow$ *false*
17: **end if**
18: **end for**
19: $\mathcal{V} \leftarrow \mathcal{V} \cup \{\boldsymbol{x}^r\}$ ▷ Add action as "evaluated"
20: $l \leftarrow$ ESPRESSO(T) ▷ Simplify truth table logic
21: **if** $l =$ *true* **then** ▷ Default action
22: $\mathcal{B}^r \leftarrow$ ADDDEFAULTACTION($\mathcal{B}^r, \mathbf{x}^r$)
23: **else** ▷ Add subtree
24: $\mathcal{B}^r \leftarrow$ ADDSEQUENCESUBTREE($\mathcal{B}^r, l, \mathbf{x}^r$)
25: **end if**
26: **end for**
27: **return** $\mathcal{B}^r$

are only activated if all previous conditions are evaluated to false; this makes the representation more compact rather than having redundant conditions.

Logical expression into a BT subtree: For each truth table we use the ESPRESSO [37] 2-level logic minimiser (line 19) to reduce the truth table into a logic formula l. This logic formula is appended as a subtree to BT $\boldsymbol{B}^r$. Each of these subtrees is defined as sequence node ($\rightarrow$) with a condition (l) and action ($\boldsymbol{x}^r$); this sequence node is connected to a fallback root node ($?$). When l is simply "true" (when we evaluate the last unique action sequence $\boldsymbol{x}^r$), this node becomes the rightmost default action node in the BT. We repeat these two steps to build the entire tree.

C. Analysis

As our proposed algorithm extends Dec-MCTS [4], the main analytical result in [4] carries over to our proposed algorithm. Our tree search algorithm in Sec. IV-A applies the same D-UCT algorithm except using behaviour trees, which provides guarantees for maintaining the exploration-exploitation trade-off during node selection in scenarios where the reward function is changing due to the changing

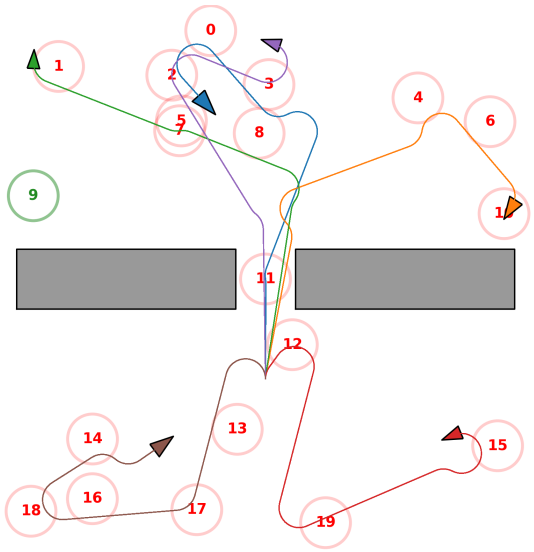

Fig. 3. Generalised team orienteering scenario with 6 robots and 25 discs. Unvisited discs are shown in green, visited discs are shown in red, and obstacles are shown in grey. Triangular markers represent the current state of the robot and the line represents the path traversed.

intents of other robots. The time complexity of the BT encoding step is dominated by the logic simplification step; 2-level logic minimisation is NP-complete, but polynomial-time approximations can be employed as optimality for this step is not necessary for our algorithm.

V. Experiments: Multi-Robot Generalised Orienteering

We evaluate the performance of our algorithm against the generalised team orienteering problem [4], where a team of robots aim to maximally visit a number of regions of interest (discs) within a given distance budget, as shown in Fig. 3. When a robot visits a disc, it collects the associated reward; revisiting the disc more than once yields no additional reward. This formulation is motivated by robot coverage, exploration, and inspection tasks, which are often formulated with a similar reward structure [4, 38].

A. Comparison Methods

To assess the effectiveness of our proposed approach, we compare it to several alternative methods:

- *Dec-MCTS with BTs*: is our proposed algorithm.
- *Dec-MCTS*: standard Dec-MCTS [4] using the top 10 action sequences evaluated so far to create the probability distribution.
- *Greedy*: in a randomised round-robin manner, each robot selects the nearest unvisited disc within its travel budget until all robots exhaust their travel budget.
- *Independent*: each robots runs the UCB variant of MCTS [39] without coordination or communication.

B. Experimental Setup

Fig. 3 illustrates the environment, with all robots starting in the same location and obstacles are arranged to partition the environment into two spaces. This scenario creates a

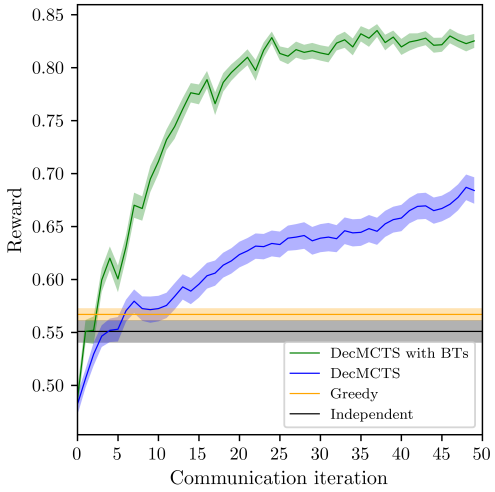

Fig. 4. Comparison of Dec-MCTS with Behaviour Trees against comparison methods for the generalised team orienteering problem. Solid lines represent the average score over 100 repeated trials in the environment and the shaded region denotes the standard error.

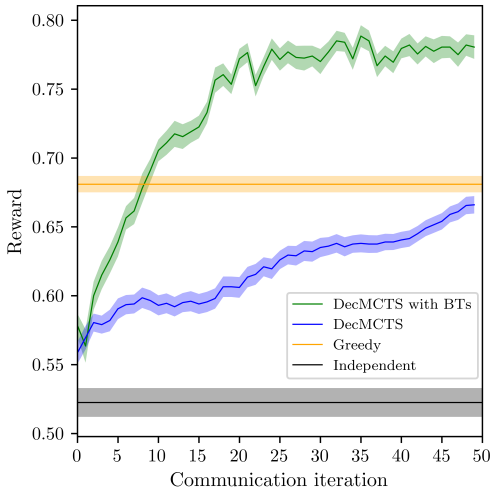

Fig. 5. Comparison of Dec-MCTS with and without BTs for a larger instance of the generalised team orienteering problem tested against 100 randomly generated environments with 20 discs and 6 robots. Solid lines denote the mean reward whilst shaded regions denote the standard error.

junction that forces the robots to negotiate whether to traverse the passageway. Each robot follows a Dubin's motion model and plans over a probabilistic roadmap (PRM) of 400 nodes with a fixed travel budget. While robots plan over the PRM, the actions represented in the BT are the discs visited by each robot. 50 iterations of Algorithm 1 are performed, each with 50 iterations of GrowTree and 1 communication broadcast. For D-UCB in Dec-MCTS we use a discount factor of 0.75 and exploration-exploitation constant of $\sqrt{2}$. *Independent* MCTS, Dec-MCTS and our approach have an equivalent compute budget of 2500 GrowTree iterations.

C. Results

We evaluate our algorithm over a smaller problem instance depicted in Fig. 6 of 20 discs and 4 robots over 100 trials. The results, depicted in Fig. 4 demonstrate our algorithm's superiority over standard Dec-MCTS when initialised

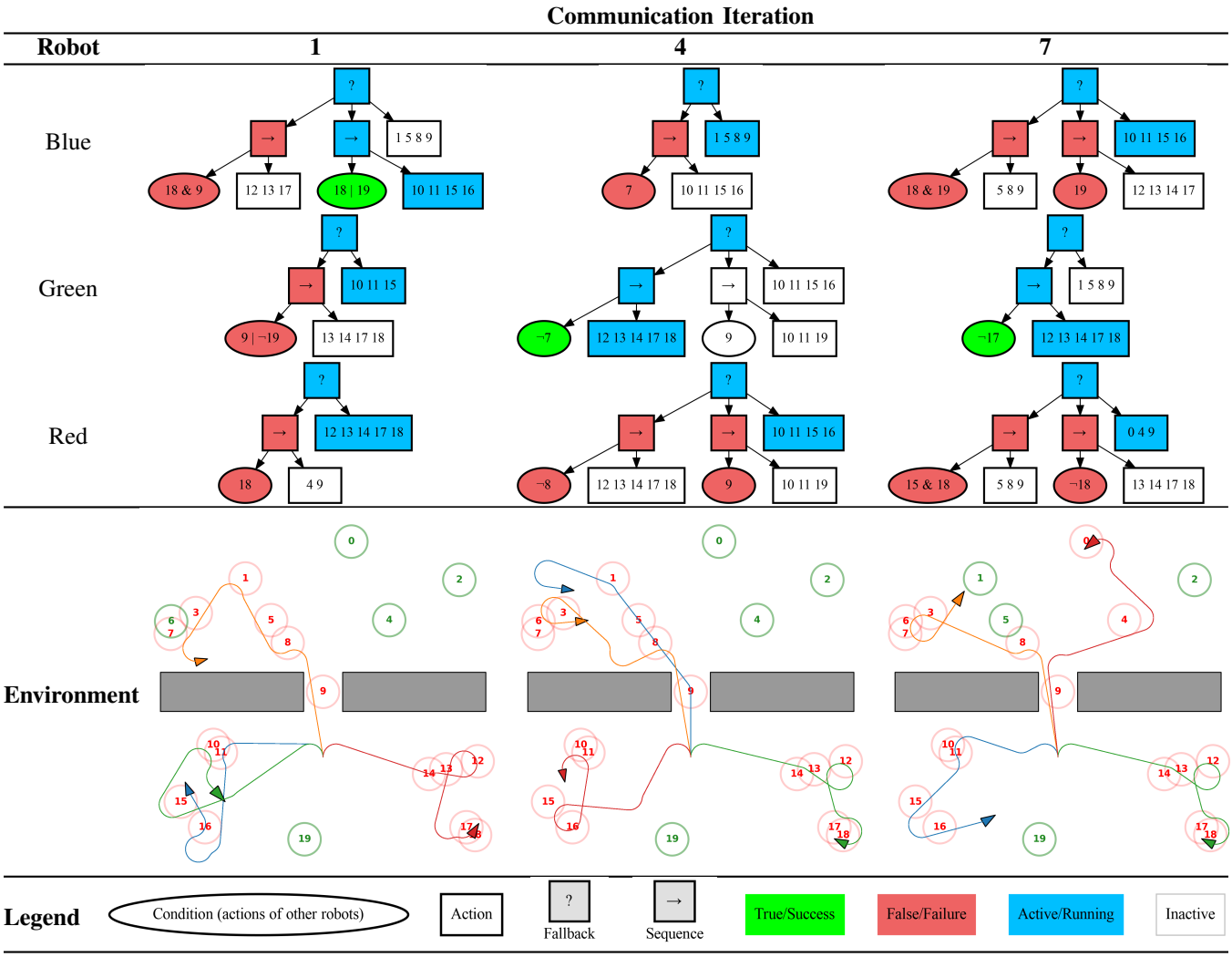

Fig. 6. Evolution of BTs of the blue, green and red robots at communication iteration 1, 4 and 7. Refer to the legend for the state of the behaviour tree. Note, the BTs are evaluated in round-robin randomised order (refer to Algorithm 1).

with identical starting conditions. As expected, *independent* exhibits the poorest performance due to the absence of explicit coordination between robots. *Greedy* performs marginally better due to the heuristic-based coordination. Dec-MCTS only marginally exceeds both *independent* and *greedy* benchmarks under 50 communication rounds whilst the proposed approach performs approximately 25% better than Dec-MCTS. We attribute this gain in performance to our explicit modelling of the intent of other robots with respect to the current robots' actions.

To further understand the behaviour of our algorithm, we inspect the BTs communicated between robots at iterations 1, 4 and 7 in Fig. 6. In general, the robots intuitively conditioning their actions based on geographical distance; e.g., the green robot at iteration 7 has its leftmost subtree that says "if no other robot is doing disc 17, then do discs 12, 13, 14, 17 and 18". We also observe implicit coordination occurring; e.g., the leftmost subtree of the red robot at iteration 7 indicates that if discs 18 and 19 are covered, then it is likely that the region below disc 9 is covered by other robots, hence it should visit discs above 9.

Our approach also demonstrates targeted communication to negotiate over points of contention. For example, the green and red robots share nearly identical trees at iteration 4; with conditions deciding which of the lower LHS and RHS of the region to cover. In iteration 7, the blue and red robot now share nearly identical trees whilst the green robot prunes the lower LHS region and now has two distinct solutions above and below the narrow passage.

We further demonstrate our algorithm on a larger orienteering instance with 6 robots and 20 discs over 100 trials in Fig. 5. The results are consistent with Fig. 4, with Dec-MCTS with BTs outperforming Dec-MCTS, and only our approach outperformed the greedy baseline.

VI. Conclusion and Future Work

Our approach presents many options for future research. Our BT-based intent representation could be further developed by the notion of action "preference" and time to handle tightly-coupled coordination problems. In addition our approach presents a promising opportunity to develop a communication scheme that only targets points of contention.

REFERENCES

[1] S. McCammon, G. Marcon dos Santos, M. Frantz, G. Best, R. K. Shearman, J. D. Nash, J. A. Barth, J. A. Adams, and G. A. Hollinger, "Ocean front detection and tracking using a team of heterogeneous marine vehicles," *Journal of Field Robotics*, vol. 38, no. 6, pp. 854–881, 2021.

[2] G. Best, R. Garg, J. Keller, G. Hollinger, and S. Scherer, "Multi-robot, multi-sensor exploration of multifarious environments with full mission aerial autonomy," *International Journal of Robotics Research*, 2023.

[3] C. Lytridis, V. G. Kaburlasos, T. Pachidis, M. Manios, E. Vrochidou, T. Kalampokas, and S. Chatzistamatis, "An overview of cooperative robotics in agriculture," *Agronomy*, vol. 11, no. 9, p. 1818, 2021.

[4] G. Best, O. Cliff, T. Patten, R. R. Mettu, and R. Fitch, "Dec-MCTS: Decentralized planning for multi-robot active perception," *International Journal of Robotics Research*, vol. 38, no. 2-3, pp. 316–337, 2019.

[5] F. Sukkar, G. Best, C. Yoo, and R. Fitch, "Multi-robot region-of-interest reconstruction with Dec-MCTS," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2019.

[6] A. J. Smith, G. Best, J. Yu, and G. A. Hollinger, "Real-time distributed non-myopic task selection for heterogeneous robotic teams," *Autonomous Robots*, vol. 43, no. 3, pp. 789–811, 2019.

[7] M. Dalmasso, A. Garrell, J. E. Domínguez, P. Jiménez, and A. Sanfeliu, "Human-robot collaborative multi-agent path planning using Monte Carlo tree search and social reward sources," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2021.

[8] C. Yoo, S. Lensgraf, R. Fitch, L. M. Clemon, and R. Mettu, "Toward optimal FDM toolpath planning with Monte Carlo tree search," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2020.

[9] G. D'Urso, J. J. Heon Lee, O. Pizarro, C. Yoo, and R. Fitch, "Hierarchical MCTS for scalable multi-vessel multi-float systems," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2021.

[10] M. Colledanchise and P. Ögren, *Behavior trees in robotics and AI: An introduction*. CRC Press, 2018.

[11] S. Omidshafiei, A.-A. Agha-Mohammadi, C. Amato, S.-Y. Liu, J. P. How, and J. Vian, "Decentralized control of multi-robot partially observable Markov decision processes using belief space macro-actions," *International Journal of Robotics Research*, vol. 36, no. 2, pp. 231–258, 2017.

[12] M. Corah and N. Michael, "Distributed matroid-constrained submodular maximization for multi-robot exploration: Theory and practice," *Autonomous Robots*, vol. 43, no. 2, pp. 485–501, 2019.

[13] Z. Xu, R. Fitch, J. Underwood, and S. Sukkarieh, "Decentralized coordinated tracking with mixed discrete-continuous decisions," *Journal of Field Robotics*, vol. 30, no. 5, pp. 717–740, 2013.

[14] N. Atanasov, J. L. Ny, K. Daniilidis, and G. J. Pappas, "Decentralized active information acquisition: Theory and application to multi-robot SLAM," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2015, pp. 4775–4782.

[15] S. Kim, M. Corah, J. Keller, G. Best, and S. Scherer, "Multi-robot multi-room exploration with geometric cue extraction and circular decomposition," *IEEE Robotics and Automation Letters*, vol. 9, no. 2, pp. 1190–1197, 2024.

[16] S. A. Gielis Jennifer and P. Amanda, "A critical review of communications in multi-robot systems," *Current Robotics Reports*, vol. 3, no. 4, pp. 213–225, 2022.

[17] M. Iovino, E. Scukins, J. Styrud, P. Ögren, and C. Smith, "A survey of behavior trees in robotics and AI," *Robotics and Autonomous Systems*, vol. 154, p. 104096, 2022.

[18] M. Iovino, J. Förster, P. Falco, J. J. Chung, R. Siegwart, and C. Smith, "On the programming effort required to generate behavior trees and finite state machines for robotic applications," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2023, pp. 5807–5813.

[19] E. Safronov, M. Colledanchise, and L. Natale, "Task planning with belief behavior trees," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2020.

[20] E. Scheide, G. Best, and G. A. Hollinger, "Behavior tree learning for robotic task planning through Monte Carlo DAG search over a formal grammar," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2021, pp. 4837–4843.

[21] C. I. Sprague, Ö. Özkahraman, A. Munafo, R. Marlow, A. Phillips, and P. Ögren, "Improving the modularity of AUV control systems using behaviour trees," in *Proc. IEEE/OES Autonomous Underwater Vehicle Workshop (AUV)*, 2018.

[22] A. Wathieu, T. R. Groechel, H. J. Lee, C. Kuo, and M. J. Matarić, "RE:BT-Espresso: Improving interpretability and expressivity of behavior trees learned from robot demonstrations," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2022, pp. 11 518–11 524.

[23] M. Colledanchise, R. Parasuraman, and P. Ögren, "Learning of behavior trees for autonomous agents," *IEEE Transactions on Games*, vol. 11, no. 2, pp. 183–189, 2018.

[24] D. Perez, M. Nicolau, M. O'Neill, and A. Brabazon, "Evolving behaviour trees for the Mario AI competition using grammatical evolution," in *Proc. European Conference on the Applications of Evolutionary Computation*, 2011, pp. 123–132.

[25] J. Styrud, M. Iovino, M. Norrlöf, M. Björkman, and C. Smith, "Combining planning and learning of behavior trees for robotic assembly," in *Proc. Int. Conf. on Robotics and Automation*, 2022, pp. 11 511–11 517.

[26] B. Banerjee, "Autonomous acquisition of behavior trees for robot control," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2018.

[27] C. Gao, Y. Zhai, B. Wang, and B. M. Chen, "Synthesis and online re-planning framework for time-constrained behavior tree," in *Proc. IEEE Int. Conf. on Robotics and Biomimetics (ROBIO)*, 2021, pp. 1896–1901.

[28] M. Colledanchise, A. Marzinotto, D. V. Dimarogonas, and P. Ögren, "The advantages of using behavior trees in mult-robot systems," in *Proc. International SYmposium on Robotics*, 2016.

[29] Q. Yang, Z. Luo, W. Song, and R. Parasuraman, "Self-reactive planning of multi-robots with dynamic task assignments," in *Proc. IEEE Int. Symp. on Multi-Robot and Multi-Agent Systems*, 2019, pp. 89–91.

[30] S. S. O. Venkata, R. Parasuraman, and R. Pidaparti, "KT-BT: A framework for knowledge transfer through behavior trees in multirobot systems," *IEEE Transactions on Robotics*, 2023.

[31] C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. Cowling *et al.*, "A survey of Monte Carlo tree search methods," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 4, no. 1, pp. 1–43, 2012.

[32] M. Świechowski, K. Godlewski, B. Sawicki, and J. Mańdziuk, "Monte Carlo tree search: A review of recent modifications and applications," *Artificial Intelligence Review*, vol. 56, no. 3, pp. 2497–2562, 2023.

[33] L. Kocsis and C. Szepesvári, "Bandit based Monte-Carlo planning," in *Proc. Eurpoean Conference on Machine Learning*, 2006, pp. 282–293.

[34] P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time analysis of the multiarmed bandit problem," *Machine Learning*, vol. 47, no. 2, pp. 235–256, 2002.

[35] A. Garivier and E. Moulines, "On upper-confidence bound policies for switching bandit problems," in *Proc. Int. Conf. on Algorithmic Learning Theory*, 2011, pp. 174–188.

[36] D. H. Wolpert, S. R. Bieniawski, and D. G. Rajnarayan, *Handbook of Statistics 31: Machine Learning: Theory and Applications*. Elsevier, 2013, ch. Probability collectives in optimization, pp. 61–99.

[37] R. K. Brayton, G. D. Hachtel, C. McMullen, and A. Sangiovanni-Vincentelli, *Logic minimization algorithms for VLSI synthesis*. Springer Science & Business Media, 1984, vol. 2.

[38] 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.

[39] P. Coquelin and R. Munos, "Bandit algorithms for tree search," in *Proc. Conference on Uncertainty in Artificial Intelligence*, 2007, pp. 67–74.