Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling (ICAPS 2020)

A Decentralised Strategy for Heterogeneous AUV Missions via Goal Distribution and Temporal Planning

Yaniel Carreno,[†‡] Èric Pairet,[†‡] Yvan Petillot,[†] Ronald P. A. Petrick[†]
Edinburgh Centre for Robotics
[†]Heriot-Watt University, Edinburgh, EH14 4AS, United Kingdom
[‡]University of Edinburgh, Edinburgh, EH8 9AB, United Kingdom
{y.carreno, eric.pairet,y.r.petillot, r.petrick}@hw.ac.uk

Abstract

Heterogeneous multi-robot systems offer the potential to support complex missions, such as those needed for persistent autonomy in underwater domains. Such systems enable each robot to be optimised for specific tasks to better manage dynamic situations. In this context, temporal planning can generate plans to support the execution of multi-robot missions. However, the task distribution quality in the generated plans is often poor due to the strategies that existing planners employ to search for suitable actions, which do not tend to optimise task allocation. In this paper, we propose a new algorithm called the Decentralised Heterogeneous Robot Task Allocator (DHRTA) which enhances goal distribution by considering task spatial distribution, execution time, and the capabilities of the available robots. DHRTA is the first phase of our decentralised planning strategy which supports individual robot plan generation using temporal planners. Experiments illustrate the robustness of the approach and indicate improvements in plan quality by reducing the planning time, mission time and the rate of mission failures.

Introduction

Autonomous Underwater Vehicles (AUVs) present a flexible platform for addressing many types of challenging problems in the marine environment, including seabed inspection, maintenance of offshore underwater structures, and vehicle detection for defence. While such applications have previously been explored in limited contexts, long-term deployments in such settings often require a level of autonomy that is not currently available in deployed systems (Thompson and Guihen 2019).

One approach for tackling such mission complexity is to equip a single platform with the necessary hardware and software components required for all possible tasks that may arise. The resulting robots are often very expensive, and their designs often make compromises to accommodate all possible missions, leading to suboptimal systems. Alternatively, other work (Patrón, Lane, and Petillot 2009; Murphy et al. 2012; Zhang, Zhang, and Liu 2019) suggests the implementation of multi-robot systems as service-oriented agents which implement specific goals with different sensors, actuators and software capabilities. Such *heterogeneous* multi-robot systems are capable of supporting more complex missions that overcome many of the limitations of single-robot solutions, providing robustness to the overall system. However, the planning strategies used for multi-robot systems often produce suboptimal task allocations, or focus on optimising other metrics of plan quality such as makespan—the time that elapses from the start of the plan execution to the end—and planning time. In particular, limited attention has been devoted to the task allocation problem for heterogeneous agents, and the effect that this has on the overall planning problem (Zhang, Zhang, and Liu 2019; Schillinger, Bürger, and Dimarogonas 2017).

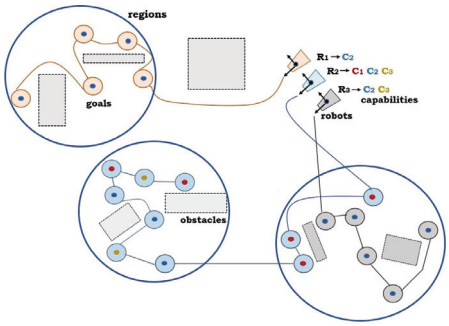

Figure 1: Mission scenario with multiple goals associated with different robot capabilities ($\mathbf{C}_1$, $\mathbf{C}_2$, and $\mathbf{C}_3$). The proposed DHRTA strategy guides task allocation to a robot set ($\mathbf{R}_1$, $\mathbf{R}_2$, and $\mathbf{R}_3$) by considering their sensory systems.

This paper proposes a novel strategy called Decentralised Heterogeneous Robot Task Allocator (DHRTA) that attempts to improve the quality of multi-robot plans by optimising goal allocation. The objective function of the DHRTA algorithm consists of two cost functions: (i) the number of solvable tasks for a robot, which refines the goal set that can be executed for each robot based on a capability analysis, and (ii) the linear combination of distance between the points of interest (POI) and the makespan of the tasks. This work builds on previous approaches that use tem-

poral planning for AUVs, such as the PANDORA project (Maurelli et al. 2016), which focused on a robot single platform, the task distribution-planning concept for multi-AUV systems defined in EUROPtus (Py et al. 2016), and the centralised Multi-Role Goal Assignment (MRGA) approach (Carreno et al. 2020) carried out as part of the ORCA Hub project (Hastie et al. 2018).

The application for this work is a fleet of heterogeneous AUVs that are responsible for the regular supervision and control of multiple offshore underwater structures. Figure 1 shows an example of the DHRTA strategy where a set of goals is distributed to a fleet of heterogeneous AUVs, taking into consideration the capabilities of the individual robots and the goal requirements. We integrate the task allocation algorithm with a temporal planner to execute multi-robot missions. We evaluate the approach by considering relevant aspects of the planning performance such as spatial goal distribution, makespan, planning time and rate of failure, and by demonstrating its applicability in real missions.

Related Work

Implementing planning strategies for multi-robot systems requires the analysis of two primary problems: *task allocation* and *task decomposition*. Task allocation is the process of assigning a set of mission goals to a group of agents such that they satisfy a set of requirements (cost functions). Task decomposition is the process of reasoning about the actions needed to achieve a set of goals, and is considered part of the mission planning stage. Multi-agent planning (MAP) has been previously addressed in approaches like (Crosby, Rovatsos, and Petrick 2013; Kvarnström 2011; Muise, Lipovetzky, and Ramirez 2015), which apply a distributed problem-solving design in place of the classical single-agent planning paradigm. However, these solvers do not typically support tasks with advanced requirements such as temporal constraints (Torreño et al. 2018), making them less attractive for the implementation of complex missions. Although several approaches (Largouët, Krichen, and Zhao 2016; Nikou et al. 2018) consider mission timing constraints to solve multi-agent problems, they use specific language representations or provide solutions to particular types of problems (e.g., low-level planning optimisation) within a global plan or a centralised planning architecture (Schillinger, Bürger, and Dimarogonas 2017), which limit their applicability in underwater applications.

Temporal planners often support language constructs that extend the Planning Domain Definition Language (PDDL) (McDermott et al. 1998), enabling the explicit representation of time to implement complex missions with multiple robots. Temporal planning problems can be modelled using PDDL2.1 (Fox and Long 2003) or other extensions that support action duration analysis. Two temporal planners are particularly promising for underwater applications: Forward-Chaining Partial-Order Planning (POPF) (Coles et al. 2010) and Optimizing Preferences and TIme-dependent Costs (OPTIC) (Benton, Coles, and Coles 2012), which have been successfully tested in real solo missions (Cashmore et al. 2014; 2015). However, there is little work addressing the multi-agent problem using temporal planners.

In the non-maritime context, there have been a number of approaches exploring the performance of PDDL planners for assistive robotics (Tran et al. 2017; Hertle and Nebel 2018) and industrial applications (Crosby and Petrick 2014). Results demonstrate that many planners often generate solutions with poor task allocation, which potentially restricts their applicability to task decomposition. Therefore, temporal planning is often combined with separate task allocation methods to improve the overall system performance. For instance, (Hertle and Nebel 2018) implement a strategy based on an auction algorithm and temporal planning. However, the approach has certain limitations around auction time and does not support domains that require concurrency. (Schneider, Sklar, and Parsons 2017) also considers an auction mechanism, however, this work does not analyse multi-robot coordination and task clustering. In addition, solutions are based on a centralised planner architecture which is not desirable in a maritime domain.

Recent research (Landa-Torres et al. 2017; Zhang et al. 2017) has considered MAP problems in the context of AUV fleets. In particular, the ScottyActivity planner (Fernandez-Gonzalez, Williams, and Karpas 2018) has been applied to task planning and trajectory optimisation with long horizons. However, the approach does not guarantee optimal action sequences and does not provide a scalability analysis. (Miloradović, Çürüklü, and Ekström 2017) propose a genetic algorithm for task allocation using a centralised mission planner which restricts its application. (Buksz et al. 2018) propose a strategy based on clustering to allocate mission goals but this method does not consider robot capabilities in the allocation. In (Py et al. 2016), task decomposition and planning are two separate modules, where planning is implemented in a decentralised architecture (on a vehicle-by-vehicle basis). However, mission tasks are distributed by the human operator which reduces the autonomy of the system. Decentralised task allocation with temporal constraints has also been addressed, such as the Temporal Sequential Single-Item auction (TeSSI) algorithm (Nunes and Gini 2015; Nunes, McIntire, and Gini 2017). TeSSI allocates tasks with time windows to cooperative robots; however, optimality depends on the number of robots and regions to explore, which limits its applicability in real applications.

While there are several architectures that support plan execution for AUV missions (McGann et al. 2008; Marques et al. 2017; Py et al. 2016), we use ROSPlan (Cashmore et al. 2015) which connects the widely used Robot Operating System (ROS) (Quigley et al. 2009) and PDDL2.1. ROSPlan allows different task planners to be embedded in a modular architecture, making it suitable for testing plan feasibility and quality while varying the underlying planning approach. ROSPlan also supports planning with concurrent actions, making it suitable for multi-robot scenarios. Prior implementations using ROSPlan have shown good performance when dealing with solo (Cashmore et al. 2014; 2015) and multi-robot (Carreno, Petillot, and Petrick 2019; Carreno et al. 2020) missions.

The main contribution of this paper is a new strategy for improving multi-robot planning by combining the capability

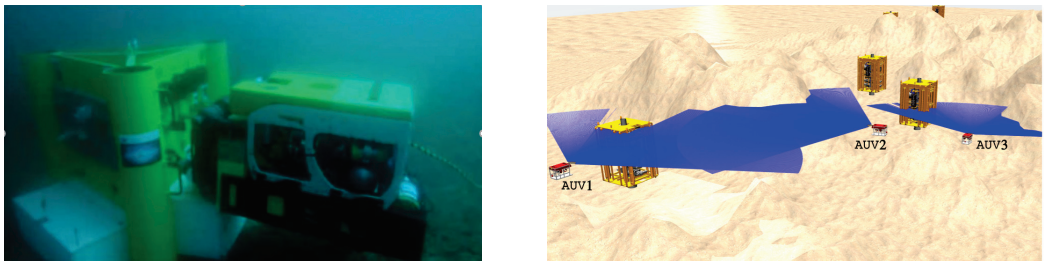

Figure 2: Real BOP structure being inspected by an AUV (left) and a simulated environment with multiple AUVs supervising BOP structures during a mission (right).

of temporal planners to implement task decomposition with a robust task allocation process. We extend the results of previous work in the underwater domain by providing a decentralised approach which can improve the quality of generated plans for a fleet of heterogeneous AUVs.

Multi-Robot Problem Description

The aim of this work is a framework that allows temporal planners to generate multi-robot plans with high task allocation quality. The proposed framework allocates mission goals to a set of heterogeneous AUVs by considering goal capability requirements, coordinates and implementation time, while providing an approach to execute plans using a decentralised architecture appropriate for underwater missions. These requirements are jointly addressed by formulating a multi-robot planning problem that considers the characteristics of the environment. That is, given a domain definition that considers the dynamics of the world and the agents' private knowledge, the approach implements a task allocation strategy that generates a set of goals for each robot, integrates the mission goals into a single robot domain definition, and generates individual robot plans using a decentralised planning method. The result is a set of temporal planning agents running in parallel to achieve each agent's plan, where each plan is a subcomponent of the full mission plan. Formally, the planning problem is defined as follows.

Definition 1 *A Multi-Agent Planning Problem is a tuple* $\Pi := \langle R, B, A, I, G, P, T \rangle$*, where* $R = \{r_1, r_2, ..., r_n\}$ *is a set of* n *heterogeneous robots,* B *is a finite set of domain propositions including robot capabilities and availability,* $A = \{a_1, a_2, ..., a_q\}$ *is a set of* q *actions which provide the means of change in the domain and are described by their preconditions and effects,* $I \subseteq B$ *defines the initial state of the propositions,* $G = \{g_1, g_2, ..., g_s\}$ *is a set of* s *goals such that* $G \subseteq B$*,* $P = \{p_1, p_2, ..., p_s\}$ *is a set of* s *goal coordinates, and* $T = \{t_1, t_2, ..., t_f\}$ *is a set of time windows. Each time window is defined using timed initial literals (TILs) which define the time* t *at which particular propositions of* B *become true/false.*

We solve the multi-agent planning problem in two parts: task allocation and task planning. The solution to task allocation distributes the goals, while the planning problem is individually solved for each agent in a decentralised manner thereby reducing the problem complexity.

As an application of this work, we consider a maritime scenario where a fleet of AUVs must complete multiple tasks located at blowout preventers (BOPs). A blowout preventer is used to control and monitor oil and gas wells to prevent the uncontrolled release of substances. These structures can be separated by long distances in a non-regular seabed which impedes the robot's operation. Figure 2 shows a BOP being inspected by an AUV (left) and a simulation environment (right) representing the inspection problem with multiple robots and structures. In this setting, we consider six types of tasks: (i) take an image of a POI, (ii) take samples of rocks or (iii) soil at a particular location, (iv) identify and observe the state of underwater structures, (v) inspect the state of a valve, and (vi) turn a valve on/off. The implementation of mission tasks is directly related to the actions a robot can execute, based on its capabilities. Each robot capability is associated with a planning action and robots typically have to perform multiple actions to successfully complete a task. As a result, our approach needs to know the set of capabilities for each robot, and jointly considers capabilities along with AUV resources and temporal constraints to define the conditions and effects of an action's implementation.

PDDL Domain Definition

Our experimental domain[1] is defined in PDDL2.1 (Fox and Long 2003) and makes use of the following **types**: `auv` defines the robot platform, `waypoint` specifies the coordinates of the POIs, which have a fixed location defined by the domain designer, `robot_sensor` refers to an AUV's sensory system, `robot_actuator` defines a robot's arm, and `valve_state` describes the state of a valve. Our work considers a set of heterogeneous AUVs.

PDDL actions and properties will typically be related to the capabilities of the specific robot platforms. Capabilities are captured by the following PDDL actions: [2]

`take_image(?auv,?poi,?camera):` a durative action which enables `?auv`'s robot sensor `?camera` to capture images on the waypoint `?poi`.

[1] Domain and problem instances are available from the DHRTA repository at https://github.com/YanielCarreno/DHRTA.

[2] We define certain parameters to have particular types: `?auv` is a parameter of type `auv`; `?poi` has type `waypoint`; `?s_soil`, `?s_rock`, `?camera`, `?sonar`, and `?cad_model` have type `robot_sensor`; `?arm` has type `robot_actuator`; and `?s1` and `?s2` have type `valve_state`.

rock_inspection(?auv,?poi,?s_rock): a durative action which enables ?auv's robot sensor ?s_rock to inspect the rocks located at waypoint ?poi.

soil_inspection(?auv,?poi,?s_soil): a durative action which enables ?auv's robot sensor ?s_soil to inspect the soil located at waypoint ?poi.

structure_id(?auv,?poi,?sonar): a durative action which enables ?auv's robot sensor ?sonar to detect and observe the structure located at waypoint ?poi.

valve_inspection(?auv,?poi,?cad_model): a durative action which enables ?auv's robot sensor ?cad_model to identify the state of the BOP located at waypoint ?poi.

turn_valve(?auv,?poi,?arm,?s1,?s2): a durative action which enables ?auv's robot actuator ?arm to turn the valve of the BOP located at waypoint ?poi from state ?s1 to ?s2.

The domain actions consider the capabilities of individual AUVs when applying actions among the fleet. Table 1 shows a list of the domain capabilities for each robot and the estimated time that it takes to perform the action associated with the capability. Our domain also includes a set of actions which are not associated with capability constraints as we assume all robots in the fleet can execute them.

navigation(?auv,?from,?to): a durative action which moves an AUV ?auv from waypoint ?from to ?to. We employ the sampling-based motion planner in (Pairet et al. 2018) to compute navigation actions which meet the AUV's kinematic capabilities, and retrieve the corresponding action duration.

refuel(?auv,?poi): a durative action for recharging ?auv's battery at waypoint ?poi. We consider multiple recharging points based on the scenario described in (Carreno, Petillot, and Petrick 2019).

broadcast_data(?auv,?poi): a durative action which allows ?auv to communicate the data recorded at waypoint ?poi.

Temporal planning is used to generate plans. Each AUV has its own domain and problem file, with the domains sharing the same information about the environment. However, the properties and goals specified in each problem file differ from one robot to another. Plans are generated in parallel by triggering multiple planning instances that generate different plans (one per robot) which contribute to the overall mission plan required to solve the multi-agent planning problem. All domain actions are deterministic. Each AUV will execute its own plan independent of the other agents' plans.

Multi-Robot Task Allocation Strategy

We address the task allocation problem in the context of a set of heterogeneous robots executing highly constrained missions. Relevant state-of-the-art temporal planners support the generation of multi-agent plans by combining forward search and partial-order construction. These frameworks apply ordering constraints among the actions during the plan search which primarily attempts to improve the plan's makespan while handling soft constraints and preferences (Benton, Coles, and Coles 2012). However, the plan's makespan optimisation does not guarantee good task allocation, particularly for missions with capability and resource constraints. In this section, we present the DHRTA approach which attempts to improve the task allocation performance of temporal planners (TP).

Capabilities	Robots with Capability	Time
take-image	R_1, R_2, R_3	10
take-rock-sample	R_1, R_4, R_5	30
take-soil-sample	R_1, R_3, R_4, R_5	30
identify-structure	R_1, R_2, R_5	20
inspect-valve	R_1, R_2, R_3, R_4	25
manipulate-valve	R_1, R_2, R_3	30

Table 1: The capabilities in the domain, the capabilities of the robots, and the execution time of actions associated with individual capabilities.

Task Allocation Algorithm

The Decentralised Heterogeneous Robot Task Allocator (DHRTA) algorithm uses two cost functions to address the task allocation problem for homogeneous fleets: (i) the number of solvable goals for each robot in different parts of the environment based on a capability analysis of the robots, and (ii) the linear combination of the task makespan and the distance between POIs. We divide the analysis into two parts: Robot Distribution and Goal Allocation.

Robot Distribution (RD): Robot Distribution attempts to find the maximum number of goals that individual robots can execute in a particular region. RD defines regions in the mission environment to deploy the robots based on the number of available robots and the goal coordinates. These regions are obtained using an updated version of the k-means algorithm (Hartigan and Wong 1979). The k-means approach decomposes goals geographically by partitioning observations according to a Voronoi diagram generated by the means. Given an initial set of k means, $m_i := \{m_1^{(1)}, \ldots, m_k^{(1)}\}$, the formal distribution is described as:

$$S_i^t = \{x_p : \| x_p - m_i^t \|^2 \leq \| x_p - m_j^t \|^2 \ \ \forall j,\ 1 \leq j \leq k\}, \tag{1}$$

where $S_i^{(t)}$ is a cluster, x_p are goal coordinates, and $m := \{m_1, \ldots, m_t\}$ is a set of means updated in each iteration. The strategy defines the goals in each cluster $\texttt{goal-a}_{S_i}$ and prevents all the tasks from being assigned to a single robot, as is the case in some auction algorithms (Nunes and Gini 2015).

Additionally, RD identifies the set of reachable goals for every AUV in the mission by comparing the capability required to implement a goal with the set of capabilities of each robot. The method allocates the task to the list $\texttt{goal-a}_r$ of achievable goals for a robot r when the agent holds the necessary capability to execute the goal. This analysis considers four inputs: (i) the set of goals G, (ii) the capabilities required to implement the goals (we assume one

capability per goal), (iii) the set of available robots R, (iv) and the set of capabilities of each robot. The results of this analysis and clustering are combined to distribute the AUVs in different regions, by allocating the robots according to the maximum number of tasks they can implement in a particular cluster. The problem is analysed as a maximisation problem (Patriksson 2015):

$$\textbf{max}\, c = \textbf{argmax} \sum_{r \in R} \sum_{S_i \in S_j} w_{rS_i} x_{rS_i}, \tag{2}$$

$$w_{rS_i} = \mathtt{D}\left(\mathtt{goal\text{-}a}_r, \mathtt{goal\text{-}a}_{S_i}\right), \tag{3}$$

where r is a robot in the set R, S_i is a cluster in the cluster set S_j, (r, S_i) is an edge, w_{rS_i} is the edge weight which is calculated using the distribution function $\mathtt{D}$ and shows the number of goals robot r can implement in cluster S_i, and x_{rS_i} is the variable on the edge. The approach finds the closest goal to each robot in the region that was assigned to it and allocates initial tasks to the agents (one task per robot), distributing the fleet in the environment. One drawback is that RD limits robot mobility to other regions, which can affect the goal solvability rate. As a result, we use a second algorithm, Goal Allocation, which removes the clustering restrictions to distribute unallocated mission goals.

Goal Allocation (GA): Goal Allocation is implemented as a vehicle routing and scheduling problem (Solomon 1987), which uses a task's makespan and the distance between the goals to distribute them across the fleet, by calculating the cost of implementing the goals for each robot:

$$\textbf{min}_r\,(\mathtt{g}) = \textbf{argmin} \sum_{g \in G} \sum_{r \in R} \textbf{cost}_r\,(\mathtt{g}), \tag{4}$$

$$\textbf{cost}_r\,(\mathtt{g}) = \gamma M^r_{max} + (1-\gamma)\left[T^r_{(g_i,g_f)} - T^r_{(g_i,0)}\right], \tag{5}$$

where $\gamma \in [0,1]$ is a weighting factor, M^r_{max} is a maximum makespan for robot r, and $[T^r_{(g_i,g_f)} - T^r_{(g_i,0)}]$ represents the distance between the position of the actual goal g_i and the final goal g_f. The distance travelled is transformed to travel time by assuming all robots are moving at the same speed. We calculate M^r_{max} by examining task makespan to date. The accumulated makespan considers the distance travelled between two POIs. The γ parameter can condition the allocation of a goal to a particular robot, giving more importance to the distance between the goals (small γ) or to the makespan (large γ). We choose γ = 0.45 as balance between distance and makespan but favouring distance. The algorithm allocates one goal at a time by calculating the cost associated with each unallocated goal for each robot. We assume a cost of infinity for goals the AUV cannot implement ($\textbf{cost}_r(\mathtt{g}) = \infty$). The goal related to the minimum cost is allocated to the robot that generates the smallest value and the goal is removed from the unallocated goal list. The robot assigned the goal then updates the costs associated with the remaining unallocated goals. This process is repeated until all goals have been allocated.

This method substantially improves goal allocation by positioning the robots in areas where they can implement a maximum number of goals, and allows AUVs to move freely in the environment to execute tasks in other regions if this is required. The output of the DHRTA algorithm is a set of goal assignments that each robot should execute (equal to the number of robots in the mission). The goals assigned to each robot are added to its PDDL problem file, using appropriate domain predicates. For instance, DHRTA might add the goal `(poi_image_taken poi20)` to the problem file of `auv1`, meaning this AUV is responsible for obtaining an image of a particular point of interest (`poi20`). This approach eliminates much of the task allocation problem from the planning process, which has the effect of reducing the search complexity for a planner to generate a plan. DHRTA distributes goals in a decentralised manner allowing robots to generate individual plans. As a result, each robot produces its own plan which contributes to the overall mission plan.

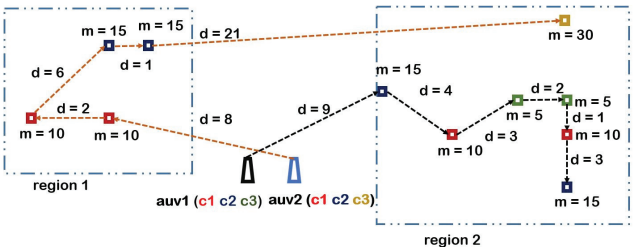

Figure 3: Example of goal allocation in a simple scenario.

Goal Decomposition Example

In this section we provide a simple example of the goal allocation strategy. Figure 3 shows two AUVs (**auv1** and **auv2**) that need to implement 11 goals in the environment. In this example, **m** represents the makespan and **d** is the distance between points. RD identifies the goals the AUVs can execute, based on their capabilities, and includes the goals in two regions (**region 1** and **region 2**). The number of goals each robot can implement in a region is calculated. In region 1, each AUV can execute 4 goals. In region 2, auv1 can execute 6 goals and auv2 can execute 5 goals. As a result, auv1 and auv2 will move to the closest goal in region 2 and region 1, respectively. The makespan is calculated by considering the distance travelled, with $\mathbf{m}(\text{auv1}) = 9$ and $\mathbf{m}(\text{auv2}) = 8$. We allocate the remaining goals using the GA algorithm. The robots generate the cost of executing all unallocated goals, where the cost of executing the closest goal to each robot is:

$$\textbf{cost}_{auv1}\,(\mathtt{g}) = \gamma\,(9+15) + (1-\gamma) \times 4,$$
$$\textbf{cost}_{auv2}\,(\mathtt{g}) = \gamma\,(8+10) + (1-\gamma) \times 2.$$

Here, the accumulated makespan considers the distance from the robot starting point to the first goal and the time to implement this goal. This analysis is applied for each unallocated goal. In this example, the lowest cost is calculated for auv2. The goal is removed from the unallocated goal list and it is assigned to auv2. The robot then updates its position and recalculates all of its costs before starting the next round of goal allocation. For auv2, the accumulated makespan increases to $\mathbf{m}(\text{auv2}) = 20$; the value for auv1 is unchanged.

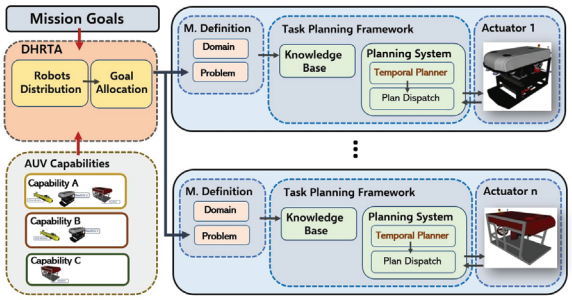

Figure 4: Framework for the DHRTA algorithm and task planning under capability constraints.

The robot allocated to a particular region has a greater chance of implementing the majority of goals in a region. However, robots from other regions can also support a robot in different clusters when (i) the robot in the region is executing goals with large makespans and robots in the neighbourhood are free, and (ii) the goal capability requirement is not in the capability list of the robot initially allocated to this region. In the example, although auv2 is initially located in region 1, it moves to region 2 to execute a goal auv1 does not have the capability to implement.

System Framework

We demonstrate the approach in simulation using the UUV Simulator (Manhães et al. 2016), with multiple heterogeneous AUVs equipped with different sensors. We integrate the temporal planners into ROS using ROSPlan. Figure 4 shows the decentralised framework with the DHRTA block, which receives data from the AUVs (number of AUVs and capabilities) and the mission goals. We implemented two ROS nodes: Robot Distribution and Goal Allocation which generate the decomposition and update the problem files. DHRTA distributes the mission goals which populate the problem file to each AUV. The robots generate and execute their plans in parallel which potentially reduces the overall mission time. Failures are addressed through replanning for individual robots when errors arise. The approach provides a generic solution which can be ported to other applications and real-world systems. Although DHRTA's output is specified as PDDL, it can also be adapted to other formats depending on the required system characteristics.

Motion Planner Strategy

We employ the motion planner proposed in (Pairet et al. 2018), which provides an efficient strategy for finding collision-free paths from a start state to a goal state while accounting for a robot's kinematic capabilities. To succeed in finding a solution to the motion planning problem, the planner exploits the robot's full kinematic model within a sampling-based strategy, which guarantees not only path feasibility, but also provides probabilistic completeness, i.e., finding a solution if one exists. Interestingly, the motion planner offers asymptotic optimality properties because it employs a shooting approach which always expands the tree from a neighbouring node with the lowest cost. This implies that the found solution will tend to the optimal solution as more computation time is given to the planner. The overall planning approach has proven to be suitable for real-world motion planning problems in the maritime domain, even for systems with limited on-board computational power.

Experiments and Results

We evaluate the DHRTA algorithm in three simulated experiments. In the first experiment, we analyse the performance of goal distribution on a particular problem. The second experiment examines the efficiency of the approach by analysing plan quality. We compare the performance of DHRTA+TP (DHRTA combined with temporal planning), with the results of three benchmark planners: TFLAP (Sapena, Onaindia, and Torreno 2015), POPF and OPTIC. We use OPTIC as the temporal planner in our DHRTA+TP strategy, since it has demonstrated promising performance in many domains. We evaluate the approach on fleets of five heterogeneous AUVs supervising up to 10 regions. The third experiment introduces a failure analysis by comparing the number of goals achieved for the robotic system using a centralised approach against our decentralised strategy. Experiments were attempted 150 times and the results show the mean and standard deviation of the data. All experiments were performed on a machine with a 4GHz processor, limiting the planner to 30 minutes of CPU for plan generation and 8GB of memory consumption.

Experiment 1: This experiment evaluates the performance of POPF, OPTIC, and DHRTA+TP using a fleet of four AUVs (R_1, R_2, R_3, R_4). The problem involves 17 goals distributed in five regions. Four different types of capabilities are required, with the capabilities corresponding to those in Table 1. The BOP platforms are concentrated in an area of five square kilometres and the initial position of the AUVs is in the centre of the environment.

Experiment 2: This experiment analyses the performance of DHRTA+TP by comparing the quality of the plans generated against those of the benchmark planners POPF, TFLAP and OPTIC. We evaluate the results of executing plans for the 12 problems with different levels of complexity (e.g., number of goals, initial battery level, etc.). Makespan and planning time are used for evaluating the results for a fleet of five AUVs.

Experiment 3: This experiment evaluates the performance of the decentralised strategy by analysing the mission failure rate. The experiment counts the number of goals achieved by the system, with task execution failing at a random time during the mission. In addition, we assume a replanning rate of success of 20% when the planner attempts to generate a new plan. The approach considers 12 sets of goals. We compare the results against a centralised planning strategy where replanning always generates a new plan for the entire robot fleet.

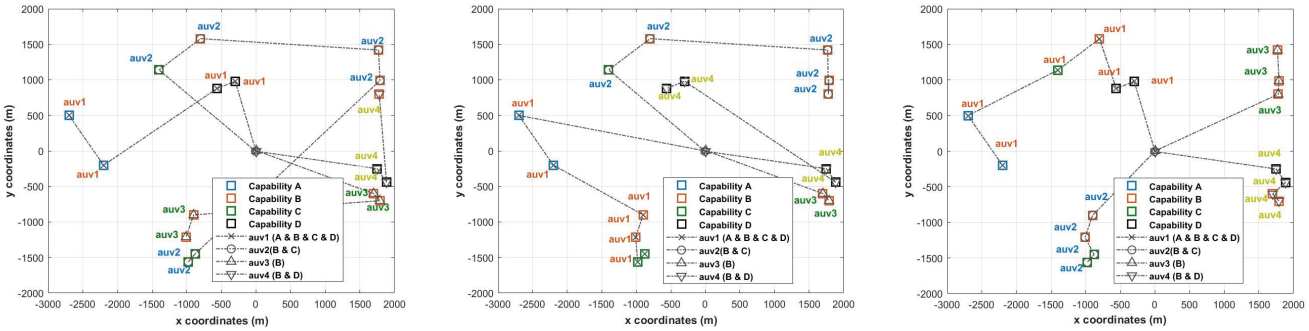

Figure 5: Goal allocation and task distribution using POPF (left), OPTIC (middle) and DHRTA+TP (right). Plans are evaluated by simulation for a set of five heterogeneous AUVs. DHRTA+TP reduces the complexity of the mission plan by considering the goal coordinates and capability requirements.

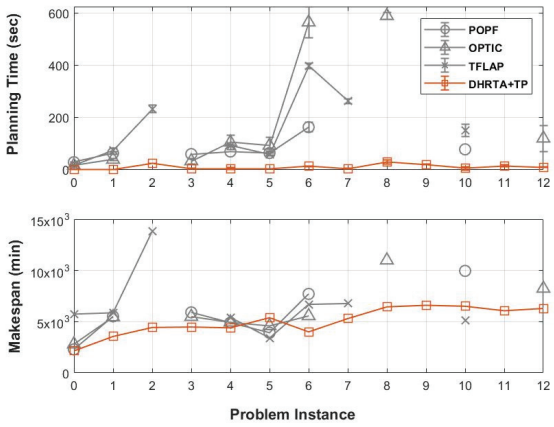

Figure 6: Planning time (top) and makespan (bottom) analysis for DHRTA+TP and benchmark planners. DHRTA improves temporal planning performance by reducing planning time, which improves the response capacity of the robotic system while keeping makespan at similar values compared with the solutions generated by the benchmark planners.

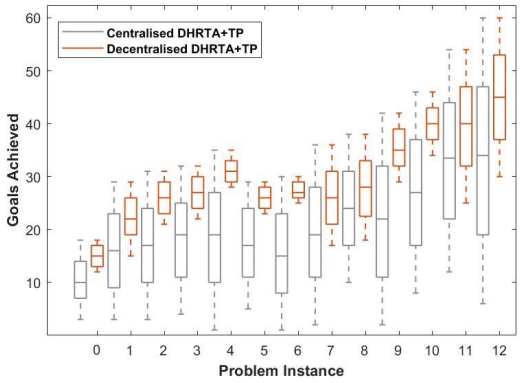

Figure 7: Number of goals achieved by DHRTA+TP during mission execution, with task failure occurring at random times. Results are compared against a centralised planner.

Results and Analysis

In Experiment 1, we compare the performance of our strategy with the results from POPF and OPTIC. Overall, we found that DHRTA+TP optimises goal distribution for the same number of robots due to the additional factors it considers: goal capability requirements, robot capabilities, task makespan and goal locations. The task allocation generated by DHRTA also results in quality improvements to the planned action schedules, as illustrated in Figure 5. DHRTA+TP effectively distributes goals among the AUVs while trying to maintain an allocation of agents at different regions to reduce the total distance travelled, energy consumed, and the possibility of robot collisions. However, the capabilities of the robots can influence this allocation and force the assignment of goals from different regions to the same robot. For instance, in the problem illustrated in Figure 5, auv1 is the only robot with capability A. Therefore, the robot moves from its initial cluster to execute goals that require capability A in other regions. In practice, DHRTA determines goal allocation by considering all the elements in the cost function when multiple robots are capable of implementing the task. Our experiments confirm this is the case with the strategy taking advantage of both the distance between the POIs and makespan. When the number of robots is smaller than the number of regions, we found the algorithm effectively considers makespan to allocate goals. POPF and OPTIC tend to produce suboptimal plans with poor task allocation in these experiments: the plans increase the distance travelled which forces robots to recharge during the mission. Results from DHRTA show that our approach optimises the use of AUVs and resources, substantially improving the general performance of the system. Similar goal allocation are obtained in the other problem sets. However, the distribution is primarily conditioned by the heterogeneity of the fleet and the capability requirements of the goals.

In Experiment 2, we evaluate plan quality by considering planning time and makespan. Planning time influences

the capacity of the robotic system to react optimally during time-sensitive tasks in AUV missions. Figure 6 (top) shows the planning time for five AUVs. We found DHRTA+TP generates solutions for all problems in less than 5 mins of planning with better makespan results compared with POPF, OPTIC and TFLAP. DHRTA+TP also produces the shortest plan generation time over all problems. The benchmark planners generate plans with long planning times, which is not desirable in real underwater applications. The systems also have to replan regularly, which means any delay in plan generation can change the state of the robot and the environment. The results demonstrate that DHRTA improves plan generation performance for a heterogeneous fleet by reducing the complexity of the planning problem at the temporal planning stage. In addition, simulations show DHRTA+TP scales well in situations with large numbers of goals and constraints, while the benchmark planners struggle to generate solvable plans in many of the 12 problems we tested: TFLAP solves 6 problems, POPF solves 7 problems, and OPTIC solves 8 problems. Of the benchmark planners, OPTIC performs best overall. DHRTA+TP also produces plans with lower makespan compared with the benchmark planners. Figure 6 (bottom) shows that in most cases DHRTA+TP results in the smallest makespan across all planners, with plans that use fewer AUVs compared to those produced by the benchmark planners.

In Experiment 3, the decentralised DHRTA+TP strategy outperforms the centralised approach, achieving the highest number of goals during the mission for all goal sets (see Figure 7). The centralised strategy is particularly affected by failures, since the planner must generate a new plan for all robots involved in the mission. Failure in the central node can also lead to situations of overall plan failure (which happened in this evaluation 80% of the time), increasing the failure rate. With the decentralised approach, failure only affects a single robot plan which needs to be replanned. The other robot plans continue unaffected. Execution continues for these agents, reducing the overall failure rate compared with the centralised approach.

Overall, the experiments demonstrate the capacity of DHRTA to improve planning performance and plan quality. The approach also works under special cases where RD finds a single cluster and the goals require multiple agents. The decentralised strategy allows us to add new goals to individual agents without affecting the execution of the overall mission. Currently, DHRTA does not focus on solving complex coordination problems. However, an analysis of multi-robot coordination is implicit in the solution. For instance, the algorithm considers the logical relationship between the goals that can be implemented for a platform. We are in the process of extending the approach to add other types of coordination which will enable agents to carry out joint tasks.

Conclusions

In this paper, we present a planner agnostic method for addressing the task allocation problem for missions involving heterogeneous multi-robot systems. We introduce an algorithm called Decentralised Heterogeneous Robot Task Allocator (DHRTA) which allocates mission goals based on task makespan, goal distance and position, and robot capabilities. We integrate DHRTA with temporal planning to generate multi-robot plans in a decentralised manner, with each robot providing a solution which contributes to the overall mission. The approach was tested and evaluated with multiple AUVs in a simulated marine environment. Results show that DHRTA improves the quality of goal allocation when compared with a set of benchmark planners, in terms of makespan, planning time and distance travelled. In addition, our strategy decreases the overall failure rate in the experimental domains. Future work aims at exploring methods for multi-robot coordination, mission replanning based on contingent planning, and new goal reallocation methods for managing catastrophic plan failures.

Acknowledgements

The authors would like to acknowledge the support of BAE Systems Surface Ships Ltd along with the EPSRC ORCA Hub (EP/R026173/1, 2017-2021, http://orcahub.org/) and consortium partners.

References

Benton, J.; Coles, A. J.; and Coles, A. 2012. Temporal planning with preferences and time-dependent continuous costs. In *Proceedings of ICAPS*.

Buksz, D.; Cashmore, M.; Krarup, B.; Magazzeni, D.; and Ridder, B. 2018. Strategic-tactical planning for autonomous underwater vehicles over long horizons. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3565–3572. IEEE.

Carreno, Y.; Pairet, È.; Petillot, Y.; and Petrick, R. P. A. 2020. Task allocation strategy for heterogeneous robot teams in offshore missions. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*.

Carreno, Y.; Petillot, Y.; and Petrick, R. P. A. 2019. Multi-vehicle temporal planning for underwater applications. In *Proceedings of ICAPS Workshop on Planning and Robotics (PlanRob)*.

Cashmore, M.; Fox, M.; Larkworthy, T.; Long, D.; and Magazzeni, D. 2014. AUV mission control via temporal planning. In *Proceedings of ICRA*, 6535–6541.

Cashmore, M.; Fox, M.; Long, D.; Magazzeni, D.; Ridder, B.; Carrera, A.; Palomeras, N.; Hurtos, N.; and Carreras, M. 2015. ROSPlan: Planning in the Robot Operating System. In *Proceedings of ICAPS*, 333–341.

Coles, A. J.; Coles, A.; Fox, M.; and Long, D. 2010. Forward-chaining partial-order planning. In *Proceedings of ICAPS*, 42–49.

Crosby, M., and Petrick, R. 2014. Temporal multiagent planning with concurrent action constraints. In *Proceedings of ICAPS Workshop on Distributed and Multi-Agent Planning (DMAP)*.

Crosby, M.; Rovatsos, M.; and Petrick, R. 2013. Automated agent decomposition for classical planning. In *Proceedings of ICAPS*, 46–54.

Fernandez-Gonzalez, E.; Williams, B.; and Karpas, E. 2018. Scottyactivity: Mixed discrete-continuous planning with convex optimization. *JAIR* 62:579–664.

Fox, M., and Long, D. 2003. PDDL2.1: An extension to PDDL for expressing temporal planning domains. *JAIR* 20:61–124.

Hartigan, J. A., and Wong, M. A. 1979. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28(1):100–108.

Hastie, H.; Lohan, K.; Chantler, M.; Robb, D. A.; Ramamoorthy, S.; Petrick, R.; Vijayakumar, S.; and Lane, D. 2018. The ORCA Hub: Explainable offshore robotics through intelligent interfaces. In *HRI 2018 Workshop on Explainable Robotic Systems*.

Hertle, A., and Nebel, B. 2018. Efficient auction based coordination for distributed multi-agent planning in temporal domains using resource abstraction. In *Joint German/Austrian Conference on Artificial Intelligence (Künstliche Intelligenz)*, 86–98. Springer.

Kvarnström, J. 2011. Planning for loosely coupled agents using partial order forward-chaining. In *Proceedings of ICAPS*.

Landa-Torres, I.; Manjarres, D.; Bilbao, S.; and Del Ser, J. 2017. Underwater robot task planning using multi-objective meta-heuristics. *Sensors* 17(4):762.

Largouët, C.; Krichen, O.; and Zhao, Y. 2016. Temporal planning with extended timed automata. In *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, 522–529. IEEE.

Manhães, M. M. M.; Scherer, S. A.; Voss, M.; Douat, L. R.; and Rauschenbach, T. 2016. UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation. In *OCEANS 2016 MTS/IEEE Monterey*. IEEE.

Marques, T.; Pinto, J.; Dias, P.; and de Sousa, J. T. 2017. Mvplanning: A framework for planning and coordination of multiple autonomous vehicles. In *OCEANS–Anchorage, 2017*, 1–6.

Maurelli, F.; Carreras, M.; Salvi, J.; Lane, D.; Kyriakopoulos, K.; Karras, G.; Fox, M.; Long, D.; Kormushev, P.; and Caldwell, D. 2016. the pandora project: A success story in auv autonomy. In *Proceedings of IEEE OCEANS 2016 - Shanghai*.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL – The Planning Domain Definition Language (Version 1.2). Technical Report CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control.

McGann, C.; Py, F.; Rajan, K.; Thomas, H.; Henthorn, R.; and McEwen, R. 2008. A deliberative architecture for AUV control. In *IEEE Int. Conf. on Robotics and Automation*, 1049–1054.

Miloradović, B.; Çürüklü, B.; and Ekström, M. 2017. A genetic mission planner for solving temporal multi-agent problems with concurrent tasks. In *International Conference on Swarm Intelligence*, 481–493. Springer.

Muise, C.; Lipovetzky, N.; and Ramirez, M. 2015. Map-lapkt: Omnipotent multi-agent planning via compilation to classical planning. *Competition of Distributed and Multi-Agent Planners (CoDMAP-15)* 14.

Murphy, R. R.; Dreger, K. L.; Newsome, S.; Rodocker, J.; Slaughter, B.; Smith, R.; Steimle, E.; Kimura, T.; Makabe, K.; Kon, K.; et al. 2012. Marine heterogeneous multirobot systems at the great eastern japan tsunami recovery. *Journal of Field Robotics* 29(5):819–831.

Nikou, A.; Boskos, D.; Tumova, J.; and Dimarogonas, D. V. 2018. On the timed temporal logic planning of coupled multi-agent systems. *Automatica* 97:339–345.

Nunes, E., and Gini, M. L. 2015. Multi-robot auctions for allocation of tasks with temporal constraints. In *AAAI*, 2110–2116.

Nunes, E.; McIntire, M.; and Gini, M. 2017. Decentralized multi-robot allocation of tasks with temporal and precedence constraints. *Advanced Robotics* 31(22):1193–1207.

Pairet, È.; Hernández, J. D.; Lahijanian, M.; and Carreras, M. 2018. Uncertainty-based online mapping and motion planning for marine robotics guidance. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2367–2374. IEEE.

Patriksson, M. 2015. *The traffic assignment problem: models and methods*. Courier Dover Publications.

Patrón, P.; Lane, D. M.; and Petillot, Y. R. 2009. Situation-aware mission planning using distributed service oriented agents in autonomous underwater vehicles. In *International Symposium on Unmanned Untethered Submersible Technology*.

Py, F.; Pinto, J.; Silva, M. A.; Johansen, T. A.; Sousa, J.; and Rajan, K. 2016. Europtus: A mixed-initiative controller for multi-vehicle oceanographic field experiments. In *International Symposium on Experimental Robotics*, 323–340. Springer.

Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; and Ng, A. Y. 2009. ROS: an open-source Robot Operating System. In *Proceedings of ICRA Workshop on Open Source Software*.

Sapena, O.; Onaindia, E.; and Torreno, A. 2015. Flap: applying least-commitment in forward-chaining planning. *AI Communications* 28(1):5–20.

Schillinger, P.; Bürger, M.; and Dimarogonas, D. V. 2017. Simultaneous task allocation and planning for temporal logic goals in heterogeneous multi-robot systems. *The International Journal of Robotics Research* 0278364918774135.

Schneider, E.; Sklar, E. I.; and Parsons, S. 2017. Mechanism selection for multi-robot task allocation. In *Annual Conference Towards Autonomous Robotic Systems*, 421–435. Springer.

Solomon, M. M. 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations research* 35(2):254–265.

Thompson, F., and Guihen, D. 2019. Review of mission planning for autonomous marine vehicle fleets. *Journal of Field Robotics* 36(2):333–354.

Torreño, A.; Onaindia, E.; Komenda, A.; and Štolba, M. 2018. Cooperative multi-agent planning: a survey. *ACM Computing Surveys (CSUR)* 50(6):84.

Tran, T. T.; Vaquero, T.; Nejat, G.; and Beck, J. C. 2017. Robots in retirement homes: applying off-the-shelf planning and scheduling to a team of assistive robots. *JAIR* 58:523–590.

Zhang, Z.; Wang, J.; Xu, D.; and Meng, Y. 2017. Task allocation of multi-auvs based on innovative auction algorithm. In *Proc. of ISCID*, volume 2, 83–88. IEEE.

Zhang, L.; Zhang, L.; and Liu, S. 2019. Role-based collaborative task planning of heterogeneous multi-autonomous underwater vehicles. *International Journal of Advanced Robotic Systems* 16(3):1729881419858536.