2024

ALGORITHMS FOR COORDINATING MULTIPLE AUTONOMOUS VEHICLES UNDER VARIOUS CONSTRAINTS WITH EMPHASIS ON WORKLOAD BALANCING

Abhishek Patil
Michigan Technological University, apatil5@mtu.edu

ALGORITHMS FOR COORDINATING MULTIPLE AUTONOMOUS VEHICLES UNDER VARIOUS CONSTRAINTS WITH EMPHASIS ON WORKLOAD BALANCING

By

Abhishek Patil

A DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

In Mechanical Engineering-Engineering Mechanics

MICHIGAN TECHNOLOGICAL UNIVERSITY

2024

This dissertation has been approved in partial fulfillment of the requirements for the Degree of DOCTOR OF PHILOSOPHY in Mechanical Engineering-Engineering Mechanics.

Department of Mechanical Engineering-Engineering Mechanics

Dissertation Co-advisor: *Dr. Jung Yun Bae*

Dissertation Co-advisor: *Dr. Myoungkuk Park*

Committee Member: *Dr. Gordon Parker*

Committee Member: *Dr. Seulchan Lee*

Department Chair: *Dr. Jason Blough*

Dedication

To Shree Guru & Dr. Jung Yun Bae

I'll always be grateful to you for giving me the opportunity to work under your guidance and supporting me throughout this awesome journey!

Contents

List of Figures

List of Tables

Acknowledgments

I sincerely thank my advisors Dr. Jung Yun Bae and Dr. Myoungkuk Park for the invaluable opportunity to work at the Intelligent Robotics and System Optimization Laboratory. I deeply appreciate their willingness to go above and beyond to help and guide me whenever needed, and I will remain indebted to them for their mentorship.

I also wish to thank my fellow lab mates, Vasishtha Sohni, Hyeseon Lee, and Zongguang Liu, for their collaborative spirit and assistance in brainstorming and troubleshooting various challenges.

Special thanks to the Department of Mechanical Engineering-Engineering Mechanics at Michigan Technological University and Graduate school Department for their provision of resources and funding, which facilitated the execution of my research.

Lastly, I am profoundly grateful to my parents, family, and friends for their unwavering support throughout my academic endeavors.

Abstract

This dissertation focuses on developing algorithms to solve the problem of coordinating multiple autonomous vehicles under various constraints, aiming to produce practical solutions for real-world applications. Built upon three journal publications addressing two coordination-related problems in different domains, this research document tackles the challenges of heterogeneity constraints and cable entanglement issues encountered by autonomous vehicle systems.

The first problem tackles task allocation and path planning for heterogeneous ground mobile vehicles operating in a 2D environment with asymmetric travel costs. By enhancing previous Primal-Dual approximation heuristic methods, novel techniques are introduced to manipulate dual variables and achieve balanced workload distribution, ultimately minimizing the maximum tour cost.

The second problem addresses the issue of tether entanglement faced by multiple tethered underwater vehicle systems navigating underwater. A multi-layer heuristic is developed by extending the Primal-Dual heuristic into a 3D environment and incorporating an additional algorithm layer to detect and resolve tether entanglements within a reasonable computation cost.

Drawing on insights gained from these two problems, a new versatile algorithm

has been developed that is applicable to a range of min-max Multiple Depot Heterogeneous Asymmetric Traveling Salesperson Problems (MDHATSPs). This novel heuristic offers enhanced computational efficiency and practicality by refining formulation and integrating critical constraints, such as cable entanglement.

This dissertation has value in providing heuristics for routing problems that involve multiple autonomous vehicles with additional constraints. Each algorithm is developed from problem formulation and considered to improve solution qualities and computation time for implementation in real-world applications.

Chapter 1

Introduction

Recent advancements in autonomous vehicle technologies have revolutionized the execution of complex and time-sensitive tasks through multi-vehicle systems. With a diverse range of autonomous vehicles, including ground mobile robots, aerial autonomous vehicles, and underwater autonomous vehicles, each possessing unique capabilities, the potential applications have expanded significantly. These technologies are undergoing development for a wide spectrum of applications, spanning manufacturing, transportation services, search and rescue missions, mapping, surveillance, inspection, maintenance, and more [1, 2, 3, 4].

However, orchestrating the systematic operations of multi-vehicle systems to accomplish complex tasks poses significant challenges due to the interrelation of numerous sub-problems. A fundamental sub-problem that has to be solved in the field is finding a tour for each vehicle so that each target is visited at least once by some vehicle and the sum of maximum travel cost is minimal. When multiple vehicles are involved, task allocation, path planning, and scheduling need to be dealt with simultaneously, which amplifies the complexity of finding effective solutions. These problems are known as vehicle routing problems, first introduced by George Dantzig and John Ramser in 1959 [5], which is NP-hard [6]. This dissertation aims to address two key problems within this field.

The first problem is coordinating multiple heterogeneous autonomous ground vehicles with different motion constraints and velocities. Having heterogeneous structures of the vehicles increases the complexity of the algorithm that provides routes satisfying all motion constraints while considering workload balancing between the vehicles. This work aims to generate good-quality solutions at a reasonable computational time while focusing on the practical aspect of the approach. As an extension of the preliminary work based on the Primal-Dual approximation heuristic, presented in [7], this work presents a new approach to manipulate the dual variables using weights to penalize the travel costs and thus explore better quality load distribution to minimize the maximum tour cost. The algorithm embraces asymmetric costs, which is more generalized than preliminary work. The

details of this problem are discussed in Chapter 2.

The second problem pertains to coordinating multiple tethered autonomous underwater vehicles while ensuring entanglement-free navigation. Despite offering advantages such as uninterrupted communication and continuous power supply, these vehicles face the risk of cable entanglement during movement, limiting their applications. While significant attention has been devoted to the navigation of standalone underwater vehicles [8], limited research has focused on tethered vehicles. Existing studies primarily concentrate on motion planning while avoiding tether entanglements between predefined start and endpoints. Addressing these limitations, this dissertation proposes a heuristic approach that integrates cable entanglement constraints into the routing problem. The approach builds upon the algorithm devised for ground vehicles, with modifications and additional strategies detailed in Chapter 3.

The knowledge gained from working on two problems led us to develop a new heuristic approach with broad applicability. The generalized nature of this approach allowed for easy adaptation to address various constraints associated with vehicle routing problems with multiple depots and min-max objectives. The new heuristic offers enhanced efficiency by swiftly addressing the limitations of the multi-layer Primal-dual-based approach. Moreover, the refined problem formulation is introduced to integrate the crucial constraint of cable entanglement for

the second problem. Subsequently, we applied the new heuristic to solve both problems, with a comprehensive discussion and analysis of the results presented in Chapter 4. Through these efforts, we aim to contribute to advancing solutions for complex optimization problems, offering practical and effective strategies to address real-world challenges.

Chapter 2

An Algorithm for Task Allocation and Planning for a Heterogeneous Multi-Vehicle System to Minimize the Last Task Completion Time

This work[9] addresses the challenge of efficiently allocating tasks and planning routes for heterogeneous multi-vehicle systems. While these approaches deal with task allocation for multi-vehicle systems, they have distinctive objectives and constraints, and none deal with the same problem as this work. Leveraging a unique

Primal-Dual formulation and a workload distribution strategy, the aim was to create a tailored solution capable of accommodating the diverse motion constraints of each vehicle while optimizing the overall system performance. A key focus was minimizing the completion time of the last task, considering heterogeneity in the system to ensure effective management of differences in speed and turning radius of the vehicles. Overall, the goal was to deliver computationally viable solutions to enhance the efficiency of multi-vehicle systems in real-world applications.

2.1 Introduction

The applications of heterogeneous multi-vehicle systems are increasing thanks to advances in autonomy and artificial intelligence in recent decades [10, 11, 12, 13, 14, 15]. However, it is still difficult to overcome some limitations of current technologies due to the dynamic and unpredictable nature of the world. For the systematic operations of multi-vehicle systems to accomplish complex tasks, four main topics should be resolved: (1) task decomposition, (2) coalition formation, (3) task allocation, and (4) task execution/planning and control [16]. These topics are correlated to each other, which makes the problems even more challenging to solve. Having heterogeneity on multi-vehicle systems significantly increases the complexity while causing more layers on the operating system regarding task decomposition and allocations [17].

Generally, heterogeneity is categorized into two parts: structural and functional heterogeneity. The structural heterogeneity typically includes the differences in vehicle designs, such as motion constraints, running speed, yaw rate, and fuel capacity. On the other hand, functional heterogeneity includes the differences in functions, such as different types of data coming from various sensors, maximum payloads, and the ability for sample collections. Sensor-related issues on each vehicle are one of the factors that make the task allocation and planning more challenging, and there are several active ongoing research [18, 19, 20, 21]. Having more heterogeneity factors increases computational loads to find an efficient operational strategy for multi-vehicle systems.

This work focuses on solving a task allocation and planning problem for multi-vehicle systems with structural heterogeneity. While various heterogeneity factors would be preferred to be considered, we deal with a problem that includes structural heterogeneity on multi-vehicle systems at this time. Specifically, we assume the system has heterogeneous mobile vehicles in 2D space, such as autonomous surface vessels or ground vehicles, with different motion constraints and running velocities. We are interested in finding paths for vehicles that complete all the given tasks within the minimum period with a given system. When the vehicles depart from distinctive locations, and the travel costs do not guarantee symmetricity, we call the problem a min-max Multiple Depot Heterogeneous Asymmetric Traveling Salesperson Problem (MDHATSP). The problem is a generalized

TSP, meaning it is an NP-hard problem [22]. As preliminary research, two vehicle problems (2DHTSP, 2DHATSP) were studied in [23], and the problem for multiple structurally heterogeneous vehicles with symmetric travel costs (MDHTSP) has been studied in [7]. At this time, we relax the symmetric travel cost condition by assuming the vehicles to be Dubin's vehicles [24] with different minimum turning radii. We put our efforts into lightening the computational loads for large problems while having good solution qualities to focus on the practical aspect of the approach.

As the research for task allocation of multi-vehicle systems is becoming more active compared to previous years, some publications are dealing with similar problems. However, as a characteristic of multi-vehicle system operational research, each publication deals with its specific scenario, which makes it difficult to deploy to other scenarios. For a recent publication, Sun and Escamilla proposed an unscented transform-based approach for a task allocation process with uncertainty in situational awareness in [25]. While dealing with functional heterogeneity, they proposed a Hungarian algorithm by focusing on handling uncertainties. Li et al. presented a hybrid large neighborhood search algorithm that solves a multiple depot Autonomous Aerial Vehicle (AAV) routing problem [26]. The article is focused on dealing with an open constraint on returning depots without considering heterogeneity. Similarly, Cho et al. presented a sampling-based tour generation

algorithm for multiple AAVs by formulating the problem into a generalized MD-HATSP [27]. While [27] dealt with the most similar problem with this work, their objective is min-sum, and there is a constraint that the vehicles must return to one of the terminal nodes. A decentralized auction algorithm for the task allocation of multi-vehicle systems under a limited communication range with a min-sum objective has been proposed in [28]. A task allocation problem of autonomous underwater vehicles problem with time and resource constraints and min-max objective is dealt with in [29]. In [30], an ant colony algorithm for a min-max MDTSP without heterogeneity has been proposed and compared the results with those of a linear program (LP) based algorithm. While these approaches deal with task allocation for multi-vehicle systems, they have distinctive objectives and constraints, and none deal with the same problem as this work. This work aims to fill the gap in the area by targeting and producing reasonable solutions within a short time for a generalized problem. This work has several unique contributions as an extension of the preliminary work presented in [7]. The heuristic in [7] is developed for a min-max MDHTSP, which only solves the problems with symmetric costs. This work has its novelty based on the following contributions. First, we present a new approach in Algorithm 1 for deciding on the dual variables W_k, which play a role as the weights on travel costs for each vehicle. Due to generalized travel costs, the algorithm is designed to embrace the asymmetricity of the costs. In addition, new pruning steps for the primal-dual heuristic have been developed in Algorithm 2

to enhance the task distribution between the vehicles. The algorithms are implemented and compared with LP solutions with integer constraints relaxed, the LP rounding method, and our work on a min-sum MDHATSP[31] to verify the effectiveness of the proposed algorithm in the perspective of the workload balancing. The real-world experiment results are added to verify the feasibility of the algorithm in the field.

The remainder of this chapter is structured as follows: In Section 2.2, we specify the problem and present the formulations. Section 2.3 presents the primal-dual heuristic approach for a min-max MDHATSP. We present the computational results in Section 2.4.

2.2 Problem Description and Formulation

This section specifies the problem of allocating tasks between vehicles in a given multiple heterogeneous ground mobile vehicles. The aim is to find a path for each vehicle that satisfies its motion constraints and completes all the given tasks by the multi-vehicle system while minimizing the maximum travel cost among the agents. This problem is formulated as a min-max Multiple Depot Heterogeneous Asymmetric Traveling Salesman Problem (MDHATSP). It is assumed that

the travel costs satisfy triangle inequalities. The vehicles depart from distinct locations and return to their depots once they have completed the assigned tasks. The vehicles are assumed to have different running velocities and minimum turning radii. The travel cost is defined as the travel time of the vehicle and is calculated by $cost_{ij}^k = d_{ij}^k / v_k$, where d_{ij}^k represents the distance of the shortest path from vertex i to j for vehicle k, and v_k represents the average running velocity of vehicle k. The vehicles are labeled as their running velocities decreased, and the minimum turning radius increased as their indices increased. Then, all travel costs will monotonically increase based on their indices, i.e., $cost_{ij}^1 \leq cost_{ij}^2 \leq, \cdots, \leq cost_{ij}^m$, $\forall \{i,j\} \in V_k$, $k = 1, \cdots, m$. For m vehicles and n tasks, the parameters and decision variables used in the formulation are described as follows:

Parameters:

$D = \{d_1, ...d_m\}$	a set of depots
$T = \{t_1, \cdots, t_n\}$	a set of tasks
$V_k = \{\{d_k\} \cup T\}$	a set of vertices for k^{th} vehicle
$E_k = \{(i,j),\ \forall i,j \in V_k\}$	a set of edges that connect all vertices in V_k
$cost_{ij}^k$	the travel cost of the edge from vertex i to vertex j for k^{th} vehicle
$\delta_k^+(S)$	the subset of the edges of E_k that entering to S from $V_k \backslash S$

Decision Variables:

x_{ij}^k the decision variable that represents whether edge (i, j) is used for the tour of k^{th} vehicle

$$x_{ij}^k = \begin{cases} 1 & \text{if edge } (i,j) \text{ is traveled by the } k^{th} \text{ vehicle} \\ 0 & \text{otherwise} \end{cases}$$

z_U^k the decision variable that represents the assignment of tasks in T for k^{th} vehicle

$$z_U^k = \begin{cases} 1 & \text{if } U \text{ contains all vertices not assigned to } 1^{st}, \cdots, k^{th} \text{ vehicle} \\ 0 & \text{otherwise} \end{cases}$$

q the maximum travel cost

The formulation for a linear program (LP) relaxation of the problem is shown below:

$$C_{LP} = \min q \tag{2.1}$$

$$\sum_{(i,j)\in\delta_1^+(S)} x_{ij}^1 \geq 1 - \sum_{T\supseteq U\supseteq S} z_U^1 \qquad \forall S \subseteq T, \tag{2.2}$$

$$\sum_{(i,j)\in\delta_k^+(S)} x_{ij}^k \geq \sum_{T\supseteq U\supseteq S} (z_U^{k-1} - z_U^k) \qquad \forall S \subseteq T,\ k = 2, \cdots, m-1, \tag{2.3}$$

$$\sum_{(i,j)\in\delta_m^+(S)} x_{ij}^m \geq \sum_{T\supseteq U\supseteq S} z_U^{m-1} \qquad \forall S \subseteq T, \tag{2.4}$$

$$q \geq \sum_{i,j\in V_k} cost_{ij}^k\, x_{ij}^k \qquad k = 1, \cdots, m, \tag{2.5}$$

$$x_{ij}^k \geq 0 \qquad \forall i, j \in V_k,\ k = 1, \cdots, m, \tag{2.6}$$

$$z_U^k \geq 0 \qquad \forall U \subseteq V_k,\ k = 1, \cdots, m-1, \tag{2.7}$$

$$q \geq 0. \tag{2.8}$$

The dual problem of the LP relaxation problem is stated below:

$$C_{dual} = \max \sum_{S \subseteq T} Y_1^+(S) \tag{2.9}$$

$$\sum_{S: e \in \delta_k^+(S)} Y_k^+(S) \leq W_k cost_{ij}^k \qquad \forall i, j \in V_k,\ k = 1, \cdots, m, \tag{2.10}$$

$$\sum_{S \subseteq U} Y_k^+(S) \leq \sum_{S \subseteq U} Y_{k+1}^+(S) \qquad \forall U \subseteq T,\ k = 1, \cdots, m-1, \tag{2.11}$$

$$\sum_{k=1,\cdots,m} W_k \leq 1 \tag{2.12}$$

$$Y_k^+(S) \geq 0 \qquad \forall S \subseteq T,\ k = 1, \cdots, m, \tag{2.13}$$

$$W_k \geq 0 \qquad k = 1, \cdots, m. \tag{2.14}$$

In this formulation, the dual variable $Y_k^+(S)$ can be interpreted as the price that the vertices in set S are willing to pay to be reachable from d_k, while W_k can be interpreted as the weight given to the k^{th} vehicle.

2.3 A Heuristic for Min-Max MDHATSP

The heuristic for a min-max MDHATSP consists of two main procedures presented in Algorithms 1 and 2. While Algorithm 1 focuses on determining W_k values for each vehicle to have better workload distribution, Algorithm 2 produces a feasible task allocation and path planning solution for the fixed W_k values. In Algorithm 1, we try to transfer the workloads from the vehicle with the maximum travel cost to other vehicles to reduce the maximum travel cost in each iteration. In the heuristic presented in Algorithm 2, we used the dual problem (2.9-2.14) to find a heterogeneous directed spanning forest (HDSF). The resulting forest will become the allocation of the tasks. As we mentioned, the algorithms treat W_k like the weights on vehicles to prioritize based on their capabilities. The weighted costs should satisfy the monotonic increase inequalities, $W_1 cost_{ij}^1 \leq W_2 cost_{ij}^2 \leq \cdots \leq W_m cost_{ij}^m \ \forall i, j \in T$, all the time to guarantee the feasibility of the algorithm. The notations below are utilized to present the algorithm details.

Algorithm Notations:

F_k	A set of edges added to the graph for the kth vehicle
$\mathcal{C}_k$	A collection of vertex sets in the graph for the kth vehicle
$Y_k(S)$	The dual variable of set S for the kth vehicle
$active_k(S)$	The variable that represents the status of $Y_k(S)$

$$active_k(S) = \begin{cases} 1 & \text{if set } S \text{ can increase its dual variable} \\ 0 & \text{otherwise} \end{cases}$$

$Cost$	A set of costs for all vehicles,$\{Cost_1, \cdots, Cost_m\}$
W	A set of all W_k, $\{W_1, \cdots, W_m\}$
$Tour$	A set of assigned paths, $\{Tour_1, \cdots, Tour_m\}$
$TourCost$	A set of travel costs of $Tour_k$, $\{TourCost_1, \cdots, TourCost_m\}$

The algorithm starts with setting W_k values as they are equally distributed. This step ensures at least one feasible solution is produced for the problem, as it should satisfy the monotonic increase inequalities in default. Once the algorithm finds a feasible solution, it iteratively runs a primal-dual heuristic (Algorithm 2) while changing W_k values to reduce the maximum travel cost. To satisfy the monotonic increasing inequalities of weighted costs, we designed the algorithm to have W_k also satisfy the monotonic increasing inequalities, i.e., $W_1 \leq W_2 \leq \cdots \leq W_m$. W_k is adjusted with a small amount ϵ (heuristically determined by the user) to share the overloaded work with other vehicles while maintaining (2.12) is tight.

Algorithm 1 *A heuristic for min-max MDHATSP*

$W_k = 1/m$ for $k = 1, \cdots, m$;
$[TourCost, Tour] = GetPartition(Cost, W)$
$G \leftarrow \max(TourCost)$
while there is no improvement in G **do**
  $[TourCost_k, k] = max(TourCost)$
  if $k > 1$ **then**
    for j= 1:$k-1$ **do**
      $W_j = W_j - \epsilon$
    end for
    for j= k:v **do**
      $W_j = W_j + \epsilon$
    end for
  else
    break,
  end if
  $W_k = \frac{W_k}{\sum_{k=1}^{m} W_k}$ for $k = 1, \cdots, m$
  $[TourCost, Tour] = GetPartition(Cost, W)$
  if $G < \max(TourCost)$ **then**
    $G \leftarrow \max(TourCost)$
  end if
end while
return $TourCost, Tour$

With every fixed W_k value, the task assignment is determined by Algorithm 2. Each vehicle has its own graph, with all targets and a depot as vertices. Initially, each vertex is in its own set, all dual variables are zero, and the edge set F_k is empty. For every iteration, the algorithm searches for the dual variable that can tighten one of the constraints (2.10) with the smallest increment. Add the corresponding edge e_k to F_k. Then, we look at the graph for this vehicle and check if any valuable changes have been made. First, if a new strongly connected component is formed but is not reachable from d_k, then let the new component be an active set. Second, if any set became newly connected to d_k, then let d_k and all reachable sets from d_k be a new inactive set. This new component's subsets should also be all deactivated, while

Algorithm 2 *[TourCost, Tour]=GetPartition(Cost, W)*

Initialization
$F_k \leftarrow \emptyset$, $\mathcal{C}_k \leftarrow \{\{v\} : v \in V_k\}$, for $k = 1, \cdots, m$
All the vertices are unmarked.
All the dual variables are set to zero.
$active_k(\{v\}) \leftarrow 1, \forall v \in V_k$, for $k = 1, \cdots, m$
$active_k(\{d_k\}) \leftarrow 0$, for $k = 1, \cdots, m$
Main loop
while there exists any active component in $\mathcal{C}_1, \cdots, \mathcal{C}_m$ **do**
  for $k = 1, \cdots, m$ **do**
    Find an edge $e_k = (i, j) \in E_k$ with $i \in C_i, j \in C_j$ where $C_i, C_j \in \mathcal{C}_k, C_i \neq C_j$ that minimizes $\varepsilon_k = \frac{W_k cost_{ij}^k - dual_k(j)}{active_k(C_j)}$.
  end for
  Let the corresponding $C_j \in \mathcal{C}_k$ be S_k while $S = \{S_1, \cdots, S_m\}$ satisfies $S_1 \supseteq S_2 \supseteq \cdots \supseteq S_m$ and all active.
  $F_k \leftarrow \{e_k\} \cup F_k$
  Increase the dual variables of S_k with amount of ε_k
  if e_k forms a new strongly connected component, and the component is not reachable from d_k, **then**
    Let the new strongly connected component be a new *active* component.
  else if e_k makes any vertex $v \in S$ reachable from d_k, **then**
    Let d_k and the all the reachable vertices from d_k be a new *inactive* component.
    if $k < m$ **then**
      Deactivate all the subsets of this component in $\mathcal{C}_{k+1}, \cdots, \mathcal{C}_m$.
    end if
    if $k > 1$ **then**
      Mark all the vertices in the supersets of this component in $\mathcal{C}_1, \cdots, \mathcal{C}_{k-1}$. Deactivate them if the corresponding components consist of all marked vertices.
    end if
  else
    Deactivate S_k.
  end if
  if there exists any inactive set without entering edge which is not connected to the depot and there exists no $S = \{S_1, \cdots, S_m\}$ can be chosen that satisfy the given conditions for any $k \in \{1, \cdots, m\}$, **then**
    Pick an inactive component for each k consisting of marked vertices with entering or outgoing edges. Combine the connected components until the new component does not have any entering edges. Let the new component be active.
  end if
end while
Pruning
Let F'_k be the resulting forest after performing reverse-deleting steps to remove all unnecessary edges.
Let P'_k be the vertices in F'_k for $k = 1, \cdots, m$.
Let P_k be the vertices that are only connected to d_k for $k = 1, \cdots, m$.
if there exist any $v \in T$ that doesn't belong to any P_k for $k = 1, \cdots, m$ **then**
  Let P_c be a set of such vertices.
  while $P_c \neq \emptyset$ **do**
    Find the closest distances to the depots for all vertices in P_c.
    Find the shortest distance. Let v_c be the corresponding vertex and d_k be the closest depot.
    $P_c \leftarrow P_c \backslash v_c; \quad v_c \rightarrow P_k$
  end while
else
  $P_k = P'_k$
end if
Find the shortest tour for P_k for $k = 1, \cdots, m$.
return *TourCost, Tour*

supersets should be all marked. Lastly, neither the first nor second happened; deactivate the component. When the algorithm proceeds further, there could be a phase where there is no active set without entering edges, but some sets are still

not connected to the depot. Then, the algorithm generates a new active component for each graph by combining some connected sets with at least one marked. The iteration will stop when all sets in the graphs are inactive.

Lemma 1. *The proposed heuristic produces a feasible plan for the given set of vehicles in which every given target is visited only once by one of the vehicles.*

Proof. In Algorithm 2, the main loop terminates when all the components are inactive. There are only three cases where the components can be deactivated. First, the component is not any part of the strongly connected components that do not have entering edges, and none of the components' vertices are reachable from the depots. Second, the component becomes reachable from its depot. Third, one of its supersets/subsets becomes reachable from its depots. As the first condition can deactivate only one component within S, the termination condition cannot be met only by the first condition. That means the second or third condition should meet at least one to terminate the main loop, implying that all components should be connected to at least one depot. The pruning steps ensure that each target is connected to only one depot if there exists any target that is connected to multiple depots. Thus, Algorithm 2 produces a feasible solution for the given set of vehicles in which every given target is visited only once by one of the vehicles. As Algorithm 1 updates W_k values while maintaining monotonic increase inequalities, the proposed heuristic produces a feasible solution to the problem.

2.4 Implementation Results

We implemented the heuristic and repeatedly performed the simulation by varying problem sizes to validate the proposed heuristic. All simulations were performed in a PC equipped with an Intel®Core™ i7-7800X CPU running at 3.5 GHz with 64 GB RAM. The numbers of vehicles and targets varied from 3 to 6 and 20 to 50, respectively. To have a standard for the produced solution qualities, we used the optimal costs for the LP relaxation problem calculated by the commercial software CPLEX [32] as lower bounds. While we have repeated the tests 50 times for each size, we have tested only the heuristic for 100 targets to estimate the computational time due to the extensive computation time of LP for large-sized problems. Using the LP solution, we also applied the LP rounding method, which assigns the target to the one with the largest partitioning variable value to compare the results based on the calculated LP relaxation costs. In addition, we have applied our previous algorithm that solves min-sum MDHATSP [31] to verify the effectiveness of the algorithm especially to reduce the last task completion time. The coordinates of depots and targets are randomly generated within a space of 3m $\times$ 3m with a uniform distribution. As we mentioned previously, the vehicles are labeled as their running velocities decrease while the minimum turning radius increases in order with the index. $cost_{ij}^{k}$ was set to the minimum travel time by calculating the

Dubin's path [24] from i to j divided by the average running velocity of k^{th} vehicle. The path within each assignment was generated using LKH [33] for both LP rounding and the proposed heuristic.

The average and maximum posteriori bounds are shown in Figure. 2.1. The posteriori bound has been calculated by $Cost_{algo}/Cost_{LP}$, where $Cost_{algo}$ represents the cost generated by an algorithm and $Cost_{LP}$ represents the optimal cost of the LP relaxation problem. As the objective of the problem is min-max, which is nonlinear, the gap between the costs of the original mixed-integer problem and the LP relaxation problem is a bit large, which means that the actual solution qualities are more reasonable than the presented numbers. As we can see from the results, the average posteriori bounds of the proposed heuristic stayed the lowest while the min-sum heuristic remained in the middle, and the LP rounding method was the highest. The worst posteriori bounds for the proposed algorithm also maintained the lowest regardless of the problem sizes.

The average and maximum computation times are shown in Table 2.1. Compared to the results of the min-sum heuristic, which was an average of 9 seconds for 6 vehicles and 50 targets, the computation time is longer for min-max cases, with an average of 35 seconds. For 10 instances of 20 vehicles and 100 targets, the algorithm produced a solution within an average of 35 minutes. Considering the fact that the algorithm can handle more generalized problems, the computation time is still in

an acceptable range for real-world operations, especially for large-sized problems. Figure 2.2 shows the results from three different algorithms for an instance of 3 vehicles and 30 targets within a 3m $\times$ 3m space.

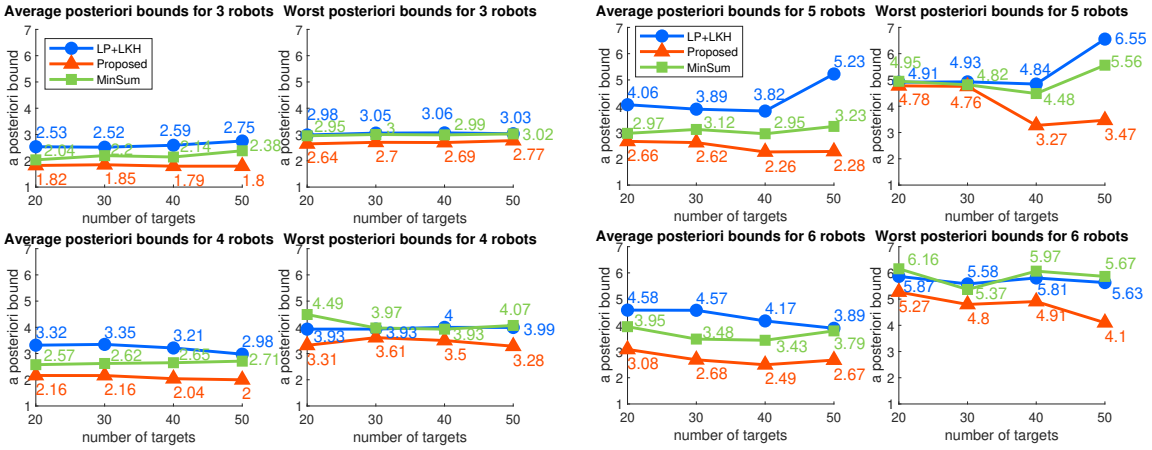


Figure 2.1: Average (**left**) and worst (**right**) posterior bounds.

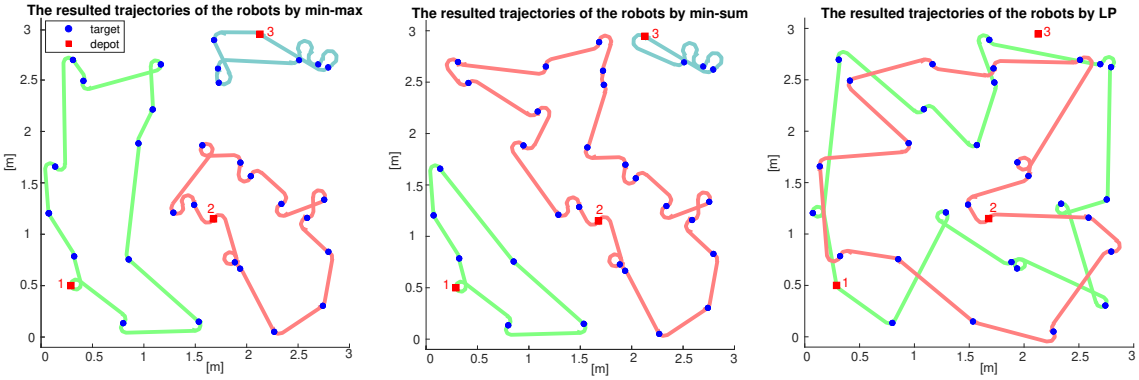


Figure 2.2: Three solutions were derived from different approaches for 3 vehicles with 30 tasks. The numbers next to the depots represent the index of the vehicles. The green, red, and blue paths represent the trajectories for vehicles 1, 2, and 3, respectively. The last task completion times for the proposed heuristic, min-sum heuristic, and LP rounding method are 8360, 11,445, and 12,309 s, respectively. The computation times for the proposed heuristic, min-sum heuristic, and LP rounding method are 2.43, 0.78, and 6495.11 s, respectively.

Table 2.1
Computation time in seconds

tasks	LP rounding	min-sum	proposed	LP rounding	min-sum	proposed
	Average with 3 vehicles			Worst with 3 vehicles		
20	4.36	0.23	0.68	6.41	0.58	1.95
30	67.39	0.65	1.77	85.97	1.07	3.38
40	619.08	1.46	4.04	888.91	1.83	7.01
50	3614.6	2.83	8.66	4922.4	3.42	12.75
	Average with 4 vehicles			Worst with 4 vehicles		
20	5.89	0.38	1.06	8.66	1.47	2.29
30	82.22	1.07	3.30	112.46	1.99	5.93
40	952.54	2.36	7.77	4183.8	3.22	12.92
50	5128.1	4.34	13.66	4434.8	5.11	24.33
	Average with 5 vehicles			Worst with 5 vehicles		
20	14.37	0.52	1.60	18.97	1.00	4.16
30	267.74	1.50	4.75	446.77	2.02	9.27
40	2960.1	3.41	11.55	4183.8	3.89	20.96
50	10920	6.67	23.88	15875	7.56	39.74
	Average with 6 vehicles			Worst with 6 vehicles		
20	14.22	0.69	2.12	17.29	1.53	5.39
30	265.13	2.14	8.00	347.92	2.59	15.97
40	3153.3	4.62	16.79	4434.8	5.52	32.63
50	15527	8.99	35.05	21225	10.32	64.30

2.4.1 Field Experiments

In addition to the simulation, we performed field experiments to verify its effectiveness in real-time applications with a small-sized problem. The multi-vehicle system consists of four ground mobile robots, Turtlebot3 Waffle Pi [34], while each robot has a different limited running velocity. The experiment site is the size of 16 ft $\times$ 12 ft and is equipped with an OptiTrack Motion Capture System (with 8 OptiTrack Prime 17W cameras) to transfer the locations of the robots in real time. The central control system is implemented in ROS [35] to navigate the robots. The

experimental setup is shown in Figure 2.4.

We have tested the proposed algorithm for a problem with 29 targets and distinctive depots for the robots. Once the task allocation and the path generation are completed, the robots immediately work on the given tasks by following the provided paths. In our experiments, unlike the simulation, we did not constrain the robots' motion to be Dubins, as the robots on the system are differential drive robots. However, the linear velocities were set to 0.1, 0.083, 0.071, and 0.063 m/sec, respectively, to include the heterogeneity in the system. To verify the effectiveness of the proposed heuristic, we have compared the results with our preliminary research, the primal-dual heuristic for min-sum MDHATSP [31].

The results from the field experiment results are shown in Figure 2.3 and Table 2.2. Figure. 2.3 shows that the vehicles were able to complete their tasks as provided by the heuristics. As shown in Table 2.2, though the min-sum heuristic ran in 0.95 seconds to complete the task allocation and path generation, the workload was overloaded to robot 3, which caused a longer last task completion time. On the other hand, the proposed heuristic ran in 3.23 seconds, which is a bit longer, but the workload was well distributed, resulting in a shorter last task completion time. However, the sum of the travel costs was better with the min-sum heuristic than with the proposed algorithm, which makes sense as it aimed to minimize the total travel costs to reduce the complexity of the problem.

Table 2.2
Computation and Travel Time in seconds of the field experiments

	Min-Max heuristic	Min-Sum heuristic
Computation	3.23	0.95
Robot1	83.3	56.7
Robot2	81.5	85
Robot3	88.7	**142.7**
Robot4	**103**	24.7
Sum of times	356.5	309.1
Completion time	106.23	143.65

The results show that the proposed approach is practical for the real-time operations of actual applications as the new heuristic deals with a more generalized problem with a better workload distribution within a reasonable computation time.

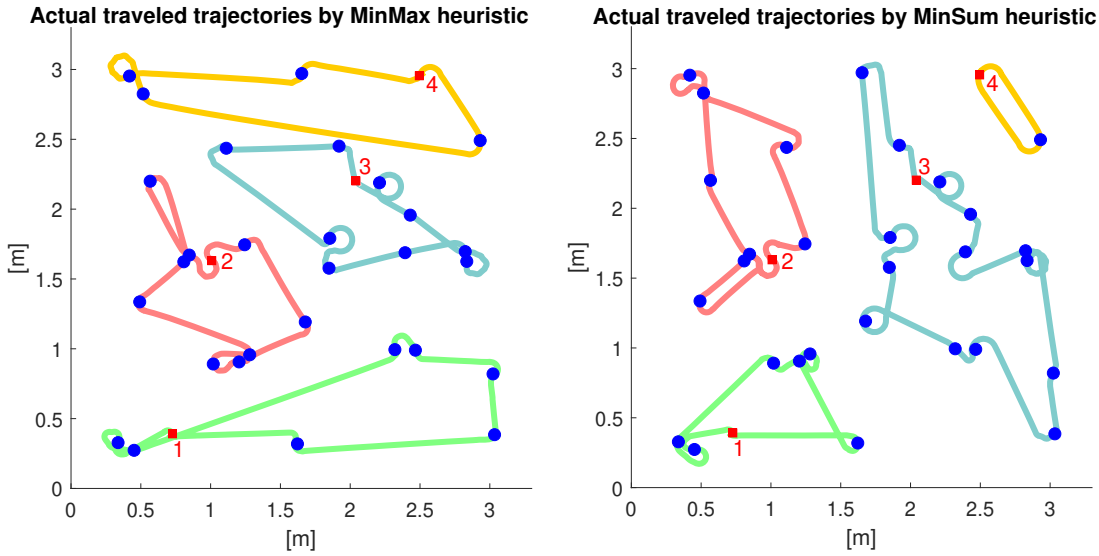


Figure 2.3: The resultant trajectories of the robots by the proposed algorithm (left) and min-sum heuristic (right)

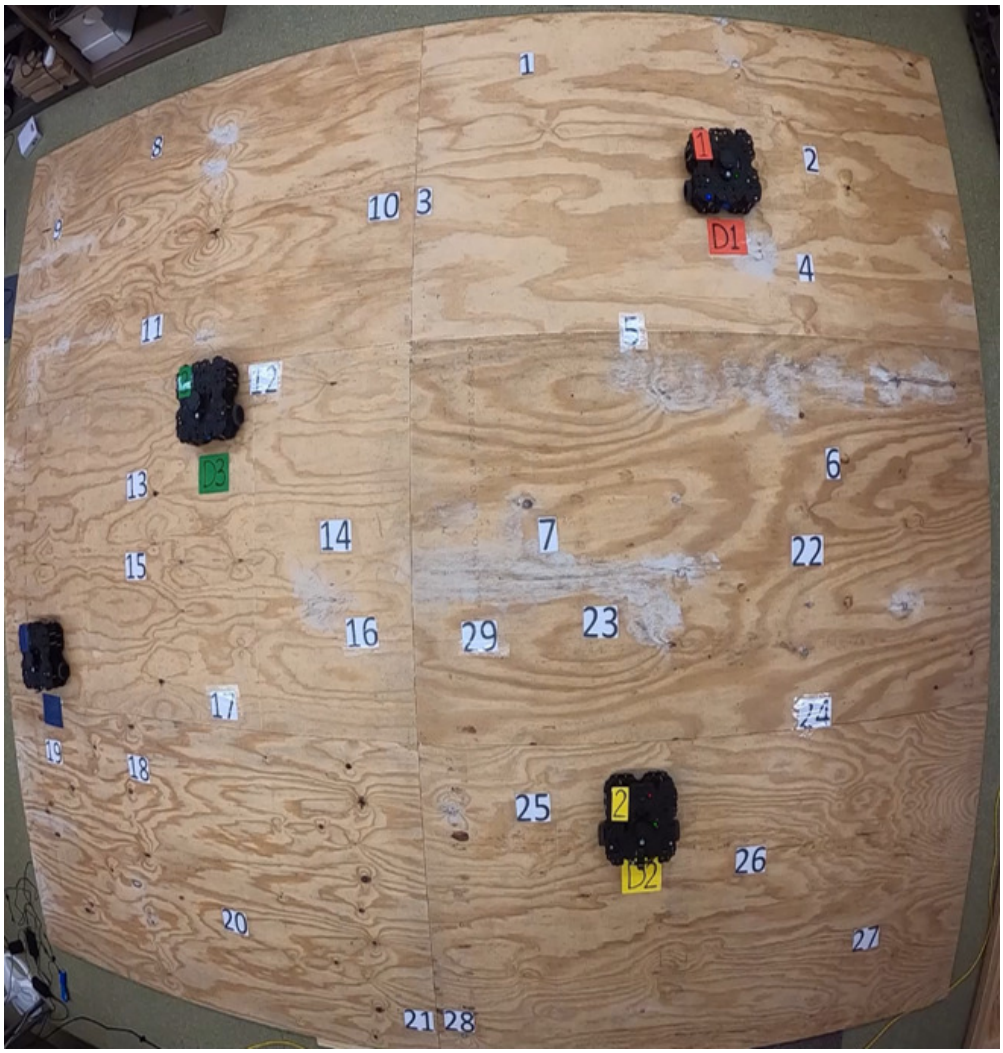


Figure 2.4: An experimental scene for the field experiments with 4 robots and 29 targets

Chapter 3

Coordinating Tethered Autonomous Underwater Vehicles toward Entanglement-Free Navigation

This work[36] addresses the persistent challenge of cable entanglement issues encountered by multiple Tethered Autonomous Underwater Vehicle (T-AUV) systems during navigation. Despite their potential for long and deep underwater operations, these entanglement problems hinder the full utilization of multiple T-AUVs. Effective planning is essential for successful large-scale operations, with meticulous attention required for task allocation, path planning, and scheduling. Recognizing the importance of overcoming these challenges, our objective was

to develop a solution that addresses cable entanglement issues while optimizing the performance of T-AUV systems. To achieve this, we developed a multi-layer heuristic approach by extending the primal-dual method to the 3D environment and incorporating a mixed approach that integrates scheduling and sectionalization methods. Ultimately, this work aims to pave the way for real-time operations of T-AUV systems, unlocking their full potential in underwater exploration and enabling critical tasks across various industries and scientific endeavors.

3.1 Introduction

Since the first Autonomous Underwater Vehicle (AUV) launched in the 1950s [37], AUVs have expanded human access to the harsh underwater environment, both for scientific research and industrial work. While autonomous navigation in aquatic environments has been focused on a single vehicle to overcome the hostile and dynamic nature of the settings, a fleet of AUVs is desirable in many underwater applications, not restricted to search, exploration, monitoring, sampling, and data collection [38, 39, 40, 41]. Proper fleet planning is required for successful mission completion in all these applications based on the AUVs' structural and functional characteristics. Planning multiple AUVs requires solving three main topics 1) task allocation, 2) path planning, and 3) scheduling, which are correlated with each other and, thus, challenging to solve.

AUVs can be categorized into two classes according to the presence of an umbilical cable. One would be a vehicle with an umbilical cable attached to a control tower called a Tethered Autonomous Underwater Vehicle (T-AUV). The second type is one without the cable, called a Stand-alone Autonomous Underwater Vehicle (S-AUV). S-AUVs include conventional AUVs, and T-AUVs include ROVs (Remotely Operated Vehicles) and hybrid AUVs. Because the umbilical cable provides a stable power source, real-time communication, and data transfer, T-AUVs benefit from long-duration working-class missions, including underwater ecosystem exploration, infrastructure inspection/maintenance, and search and rescue missions. However, umbilical entanglement can sabotage the mission for single or multi-vehicles or damage the system or other underwater elements. Murphy *et al.* deployed multiple heterogeneous AUVs at the 2011 Great Eastern Japan Earthquake and reported that the mission was usually affected by close operations and the fear of tethers tangling or even a collision[42]. Escaping from entanglement is difficult due to the dynamic environment and limited information, especially for human-operated vehicles.

Despite the crucial need for entanglement-free navigation in multiple T-AUV operations, the existing literature on this topic is limited. Specifically, Herts and Lumlsky [43] addressed the entanglement-free simultaneous motion planning problem for a highly specific scenario where each robot moves between a unique pair of start and end nodes. They employed a motion planner to compute a sequential

motion for the robots, followed by trajectories that allow for simultaneous movement. While this work represents the only available literature directly related to ours, it falls short in addressing the broader problem we tackle here, which involves task allocation of multiple targets without visiting sequences with multiple target locations and simultaneous entanglement-free path planning.

While many researchers are interested in planning for a fleet of AUVs, relatively limited methods are available [44]. There is increasing interest in advancing techniques for multiple AUVs under various conditions. For instance, [45] focused on planning the obstacle avoidance of multiple AUVs in complex ocean environments with the time coordination of simultaneous arrival. Panda et al. [46] proposed a hybrid grey wolf optimization algorithm for collision avoidance with obstacles and other vehicles. Nam [47] proposed data-gathering protocols to support long-duration cooperation by operating long-range AUVs considering energy consumption. A two-stage cooperative path planner for multiple AUVs operating in a dynamic environment that aims to minimize time consumption with simultaneous arrival while avoiding collisions [48]. A motion planner that focused on obstacle avoidance for a single AUV has been presented by McCammon and Hollinger[49]. A couple of pieces of literature focus on the motion planning of nonentangling tethers. NEPTUNE [50] solves trajectory planning for multiple robots and a single tethered robot, trajectory planning for multiple tethered robots to reach their individual targets without entanglements. While the algorithm considers 3D space,

it validated the approach with aerial vehicles. Teshnizi and Shell [51] studied a motion planner for a pair of tethered mobile robots. Zhang and Pham [52] proposed a planner that coordinates the planar robot motions to realize a given non-intersecting target cable configuration. Although some state-of-the-art techniques considered heterogeneous fleets of AUVs, no entanglement-free constraint is considered in the planning. We aim to fill the gap and build a foundation in the area by targeting to provide good approximate solutions with lighter computational loads.

In our preliminary work, the task allocation and path planning problems for multiple structurally heterogeneous autonomous ground vehicles have been studied to minimize the last job completion time [7, 9]. While working on these problems, we observed that the heuristics often produce sectioned paths due to the nature of workload balancing. The sectioned paths used in this work represent each path in a 3D space that does not intersect with other paths. Based on this observation, we propose an extended algorithm applicable to multiple AUVs by introducing an extra dimension and additional steps to ensure no entanglement happens during operations. This novel approach to the problem is rarely studied but is essential in operations for multiple T-AUVs. The proposed approach is tested extensively in simulation environments. While this work does not include field testing, it is worth noting that the Intelligent Robotics and System Optimization Laboratory (IRoSOL) at Michigan Tech possesses multiple BlueROV2 vehicles, measuring 457

mm in length, 338 mm in width, 254 mm in height, and weighing 10 kg in air. These resources offer promising opportunities for conducting field tests in the future. The remainder of this chapter is structured as follows: In Section 3.2, we specify the problem and present the formulations. Section 3.3 presents the heuristic approach to the problem. The computational results are shown in Section 3.4.

3.2 Problem Description and Formulation

In this work, our objective is to address the problem of coordinating Tethered Autonomous Underwater Vehicles (T-AUVs) in navigating a set of targets. We aim to find paths for each vehicle that satisfy the following criteria: 1) All targets are visited by at least one AUV, 2) Each path adheres to the motion constraints specific to the corresponding vehicle, 3) The tethers of the vehicles are kept at a safe distance from each other to avoid entanglement, and 4) The maximum travel cost among the vehicles is minimized. The initial setup assumes that the AUVs start at distinctive depots on the surface and return to these depots once they have visited all their assigned targets. To simplify the problem, several assumptions are made. First, we assume symmetric travel costs that adhere to the triangle inequalities. The vehicles are considered holonomic and homogeneous, with the travel cost determined as the travel time between targets using the average running velocity of the vehicles. Additionally, we assume that the cable connecting the vehicle and

the depot is a straight line managed by a tether control system without requiring extensive cable release. While these assumptions enable us to present an initial exploration of the problem, our future work aims to incorporate dynamic features of the tether shape for a more comprehensive solution. If we relax constraint 3), the problem can be formulated into a min-max Multiple Depot Heterogeneous Traveling Salesman Problem, first introduced in [7]. We use the dual formulation of the linear program relaxation of the problem to generate a minimum spanning forest, which becomes the initial task assignment. In the formulation, the following definitions are used for the variables.

Variable Definitions:

T	the set of targets
m	the number of vehicles in the cohort
d_k	the depot of the k^{th} vehicle
V_k	the set of nodes that contains targets and the depot for k^{th} vehicle
E_k	the set of edges between the nodes in V_k
$\delta_k(S)$	the subset of the edges of E_k with one end in S, and the other end in $V_k \backslash S$

In this formulation, $Y_k(S)$ can be interpreted as the prices that all targets in the set S are willing to pay to be connected to d_k, while W_k are treated like the gains for giving priority to the vehicles. Once the initial task allocation is derived, we resolve the entanglement problem. The details will be explained in the following section.

$$C_{dual} = \max 2 \sum_{S \subseteq T} Y_1(S) \tag{3.1}$$

$$\sum_{S: e \in \delta_k(S)} Y_k(S) \leq cost_{ij}^k W_k \qquad \forall i, j \in V_k,\ k = 1, \cdots, m, \tag{3.2}$$

$$\sum_{S \subseteq U} Y_k(S) \leq \sum_{S \subseteq U} Y_{k+1}(S) \qquad \forall U \subseteq T,\ k = 1, \cdots, m-1, \tag{3.3}$$

$$\sum_{k=1,\cdots,m} W_k \leq 1 \tag{3.4}$$

$$Y_k(S) \geq 0 \qquad \forall S \subseteq T,\ k = 1, \cdots, m, \tag{3.5}$$

$$W_k \geq 0 \qquad k = 1, \cdots, m. \tag{3.6}$$

3.3 A Heuristic for Scheduling Multiple Tethered Underwater Robots

As we briefly mentioned in Section 3.2, the heuristic for coordinating multiple T-AUVs consists of two main steps: 1) producing an initial task allocation and routes by relaxing the entanglement constraint and 2) detecting and resolving the possible tether entanglements. We solved the initial task allocation and routing problem with a primal-dual heuristic while following the main structure of the algorithm presented in [7] but with some revisions. The heuristic is developed based on the dual formulation (3.1)-(3.6) while iteratively changing the W_k values to improve

the workload distribution. With fixed W_k values, the heuristic runs Algorithm 3 to find a task assignment and routes. Starting from all zero dual variables, in each iteration, the dual variables $Y_k(S)$ that makes one of the dual constraints, (3.2) or (3.3), tight is increased. If one of (3.2) becomes tight, add the corresponding edge to the forest, and if one of (3.3) becomes tight, mark the corresponding set and see if they can be connected to another depot with lower cost. The main loop terminates once every target is connected to at least one of the depots. The pruning steps guarantee assigning tasks to only one of the vehicles while trying to improve workload balancing. Based on the results from fixed W_k values, the algorithm adjusts W_k values to decrease the maximum travel cost while not violating the monotonic cost increase condition, i.e., $W_1 cost_{ij}^1 \leq W_2 cost_{ij}^2 \leq \cdots \leq W_m cost_{ij}^m$ and choose the best one among the trials. When m vehicles and n targets are given, the following notations are utilized to present the algorithms.

Algorithm Notations:

R_k	The k^{th} vehicle
F_k	A set of edges in the graph of R_k
$\mathcal{C}_k$	A collection of vertex sets in the graph of R_k
$Y_k(S)$	The dual variable of set S for R_k
d_k	The depot for R_k
$dual_k(v)$	The sum of dual variables for all sets that contain vertex v
$bound_k(S)$	The sum of $Y_{k+1}(S)$

$Children_k(S)$	The vertex sets of S that exist in the graph for $R_{k+1}, \cdots, R_m$
$active_k(S)$	The variable that represents whether $Y_k(S)$ can be increased
	$active_k(S) = \begin{cases} 1 & \text{if set } S \text{ can increase its dual variable} \\ 0 & \text{otherwise} \end{cases}$
$Cost$	A set of edge costs, $\{Co_1, \cdots, Co_m\}$
$Tour$	A set of routes, $\{Tr_1, \cdots, Tr_m\}$
W	A set of all W_k, $\{W_1, \cdots, W_m\}$
S_k	The location of R_k
T_k	The heading target location from S_k
L_k^t	The estimated location of R_k between S_k and T_k
	at time t, where t is a parameter
CV_k^t	The cable vector connecting L_k^t and d_k
CP_k	The cable plane that contains d_k and two targets assigned
	to R_k in sequence
CS_{ij}	The vector connecting d_i, d_j
R_{yield}	The vehicle which is made to yield
R_{pass}	The vehicle that passes
v	The average moving velocity of the AUVs
D_z	z^{th} departure among chronologically arranged $n+m$
	scheduled departures of all vehicles

Algorithm 3 *Tour=GetPartition(Cost, W)*

1: $W_k = 1/k$ for $k = 1, \cdots, m$;
2: $[F_k, \mathcal{C}_k, active_k, Children_k] = Initialization(m, V_k, d_k)$
3: $F_k = Mainloop(F_k, \mathcal{C}_k, active_k, Children_k, Cost, W)$
4: $Tour = Pruning(F_k, Cost, W)$
5: Adjust W_k that satisfies the monotonic cost increase condition.
6: Repeat 2-4 until there is no improvement in the cost.
7: Choose the best *Tour* that produces the minimum max$(TourCost)$

Algorithm 3.1 $[F_k, \mathcal{C}_k, active_k, Children_k] = Initialization(m, V_k, d_k)$

1: $F_k \leftarrow \emptyset$, $\mathcal{C}_k \leftarrow \{\{v\} : v \in V_k\}$, for $k = 1, \cdots, m$
2: All the vertices are unmarked.
3: All the dual variables are set to zero.
4: $active_k(\{v\}) \leftarrow 1, \forall v \in V_k$, for $k = 1, \cdots, m$
5: $active_k(\{d_k\}) \leftarrow 0$, for $k = 1, \cdots, m$
6: $Children_k(\{v\}) \leftarrow \{v\}, \forall k = 1, \cdots, m-1$;

Based on the initial routes, which didn't consider entanglement constraints, we focus on detecting and resolving the possible tether entanglements. Given the initial routes, the schedule, which is each vehicle's arrival/departure time in each node, should be determined to avoid collisions and tether entanglements. The proposed approach repeatedly simulates the movement of the vehicles based on the average running speed v and processing time for each task to accomplish the mission while detecting and resolving the possible tether entanglements. The details of the algorithm have been presented in Algorithms 4-6.

Algorithm 4 presents the overall steps to detect the possible entanglements and resolve the issue. When entanglement occurs, we can observe co-planer cable vectors

Algorithm 3.2 $F_k = Mainloop(F_k, \mathcal{C}_k, active_k, Children_k, Cost, W)$

while there exists any active component in $\mathcal{C}_1$ **do**
  for $k = 1, \cdots, m$ **do**
    Find an edge $e_k = (i, j) \in E_k$ with $i \in C_i, j \in C_j$ where $C_i, C_j \in \mathcal{C}_k, C_i \neq C_j$ that minimizes
    $\varepsilon_k^1 = \frac{\{W_k cost_{ij}^k - dual_k(i) - dual_k(j)\}}{active_k(C_i) + active_k(C_j)}$.
  end for
  for $k = 1, \cdots, m-1$ **do**
    Let $\Re := \{R : active_k(R) = 1, Children(R) = \emptyset,\ R \in \mathcal{C}_k\}$. Find $\bar{R} \in \Re$ that minimizes $\varepsilon_k^2 = bound_k(\overline{C}) - Y_k(\bar{R})$
  end for
  $\varepsilon_{min} = \min(\varepsilon_1^1, \cdots, \varepsilon_m^1, \varepsilon_1^2, \varepsilon_{m-1}^2)$
  for $k = 1, \cdots, m$ **do**
    for $C \in \mathcal{C}_k$ **do**
      $Y_k(C) \leftarrow Y_k(C) + \varepsilon_{min} active_k(C)$
      $dual_k(v) \leftarrow dual_k(v) + \varepsilon_{min} active_k(C)\ \ \forall v \in C$
      if $k < m$ **then**
        $bound_k(C) \leftarrow bound_k(C) + \varepsilon_{min} |Children_k(C)|$
      end if
    end for
  end for
  if $\varepsilon_{min} = \varepsilon_k^1$ for some k **then**
    $F_k \leftarrow \{e_k\} \cup F_k$
    $\mathcal{C}_k \leftarrow \mathcal{C}_k \cup \{C_i \cup C_j\} - C_i - C_j$
    $Y_k(\{C_i \cup C_j\}) = Y_k(C_i) + Y_k(C_j)$
    if $k < m$ **then**
      $bound_k(C_i \cup C_j) \leftarrow bound_k(C_i) + bound_k(C_j)$
    end if
    if $d_k \in \{C_i \cup C_j\}$ **then**
      $active_k(C_i \cup C_j) \leftarrow 0$
      if $k < m$ **then**
        $active_{k+1}(C) \leftarrow 0\ \ \forall C \in Children_k(C_i \cup C_j)$
      end if
    else
      $active_k(C_i \cup C_j) \leftarrow 1$
    end if
  else
    $active_k(\overline{C}) \leftarrow 0$, Mark all the vertices of $\overline{C}$ with label $\overline{C}$
  end if
end while

as shown in Fig. 3.1. To inspect co-planer cable vectors for the possible entanglement, the following equation has been used:

$$(\overrightarrow{CV_i^t} \times \overrightarrow{CS_{ij}}) \cdot \overrightarrow{CV_j^t} = 0 \tag{3.7}$$

Algorithm 3.3 $Tour = Pruning(F_k, Cost, W)$

Remove all the edges in F_k that do not belong to any of the trees.
Let F'_k be the resulting forest.
Let P'_k be the vertices in F'_k for $k = 1, \cdots, m$.
$P_k \leftarrow$ {the vertices that are only connected to d_k}, $\forall k = 1, \cdots, m$
if there exist vertices that are reachable from multiple depots **then**
    Let P_c be the vertices connected to multiple depots.
    Let T_k be the minimum directed spanning tree of P_k for $k = 1, \cdots, m$.
    while P_c is not empty **do**
        Find the closest tree P_k from the vertex in P_c. Choose the one with the lowest workload if the vertex is equidistant from multiple trees.
        Remove the corresponding vertex from P_c and add to P_k.
    end while
else
    $P_k = P'_k$
end if
while there is no empty P_k **do**
    Assign the closest node to P_k
end while
Get the best $Tour$ for each P_k for $k = 1, \cdots, m$ using existing routing algorithm.

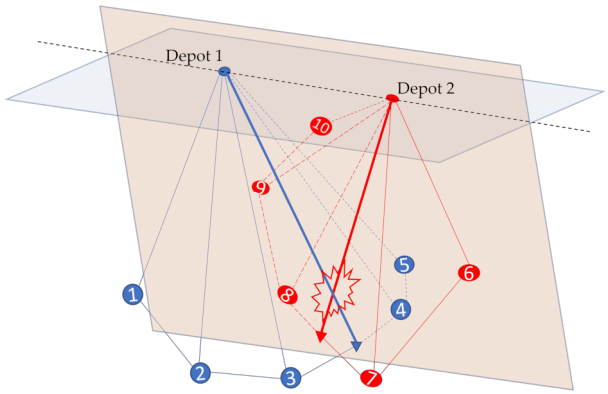


Figure 3.1: Co-planer cables at the time of entanglement

The value of t obtained upon solving (4.9) is the time after which the tethers are anticipated to entangle with each. If t is a positive real value within the range allowed by both vehicles' schedules, the cables will become a co-planner and may get entangled if the cable segments intersect. Based on this fact, Algorithm 4 tries

Algorithm 4 Possible Entanglement Management for T-AUVs

1: Initialization
2: $S_k \leftarrow$ Coordinates of the d_k for $k = 1, \cdots, m$
3: $T_k \leftarrow$ Coordinates of the first target of R_k for $k = 1, \cdots, m$
4: Q =[$1, \cdots, m$]
5: $z = 1$
6: Main Loop
7: **while** $z \leq$ the total number of departures in the mission **do**
8: **for** $\forall i \in Q$ **do**
9: $L_i^t = S_i + (T_i - S_i)/v \times t$
10: **for** j=1,. . . ,m **do**
11: $L_j^t = S_j + (T_j - S_j)/v \times t$, for $j \neq i$
12: Solve equation for t: $(\overrightarrow{CV_i^t} \times \overrightarrow{CS_{ij}}) \cdot \overrightarrow{CV_j^t} = 0$
13: **if** The value of t conforms to the range set by the time schedules of R_i and R_j respectively, and their cable segments intersect **then**
14: $[Schedules, C_1]$=TimeScheduling(R_i, R_j)
15: $[Routes, C_2]$=RouteModification(CV_i^t, CV_j^t)
16: $MC = min(C_1, C_2)$
17: **if** $MC == C_2$ **then**
18: Update routes and schedules according to the updated routes and restart from initialization.
19: **else**
20: Update the schedules and restart from initialization.
21: **end if**
22: **end if**
23: **end for**
24: **end for**
25: Q = [The next departing vehicle P associated to D_{z+1} scheduled at time t_P]
26: $S_k \leftarrow$ the location of R_k at time t_P for $k = 1, \cdots, m$
27: $T_P \leftarrow$ the immediate next target corresponds to S_P
28: $z = z + 1$
29: **end while**
30: Repeat steps 1-14 and 19-29, utilizing the TimeScheduling only. Let the cost be TC.
31: Compare TC and MC and choose the one with less cost.

to find all possible entanglement within the initial routes. For a time interval between two consecutive departures D_z and D_{z+1}, it simulates the movement of the vehicles and examines concurrent cable planes. If entanglement is not detected, it selects vehicle P associated with D_{z+1}, updates T_P and S_k, and examines new

Algorithm 5 [Routes,Time]=RouteModification(CV_i^t, CV_j^t)

1: Find nodes corresponding to the CP_i and CP_j .
2: Determine a node N_r from step 1 such that its cable vector intersects the cable plane of the other vehicle.
3: Let R_r be the vehicle corresponding to the N_r.
4: Remove N_r from the route of R_r and allocate it to another vehicle that offers the lowest maximum tour time.
5: Return the updated routes and the maximum tour time.

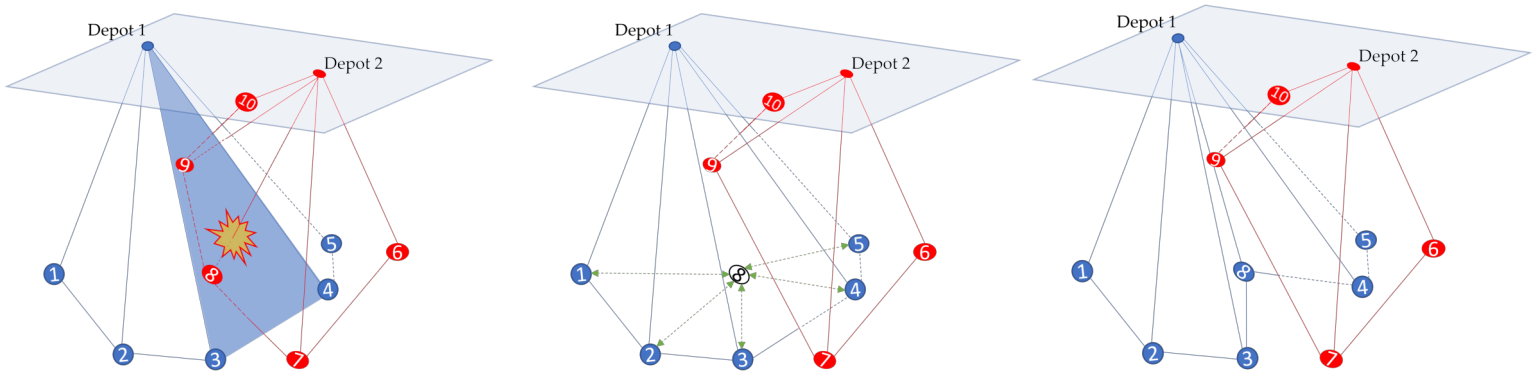
Algorithm 6 [Schedules, Time]=TimeScheduling(R_i, R_j)

1: $R_{yield} \leftarrow R_i$ and $R_{pass} \leftarrow R_j$
2: Select n_{yield} from the nodes visited by R_{yield} before the time of entanglement such that the cable vector at n_{yield} does not intersect with the cable planes of R_{pass}
3: Find a node n_{pass} in route of R_{pass} such that all the cable planes comprised of the nodes after n_{pass} never intersect with any of the cable planes of R_{yield}
4: $t_{yield} \leftarrow$ scheduled arrival time of R_{yield} at n_{yield}
5: $t_{pass} \leftarrow$ scheduled arrival time of R_{pass} at n_{pass}.
6: $W \leftarrow t_{pass} - t_{yield}$
7: $R_{yield} \leftarrow R_j$ and $R_{pass} \leftarrow R_i$ and repeat steps 1-6.
8: Choose R_{yield}, which incurs lower W
9: Return the adjusted schedules with the maximum tour time.

cable plane of P against other concurrent cable planes using (4.9). The steps are repeated until either an entanglement is detected or the time interval associated with the last departure is passed without an entanglement. If entanglement is confirmed, it either modifies the existing routes based on Algorithm 5 or adjusts the time schedules based on Algorithm 6. The heuristic chooses the approach that resolves the entanglement at the expense of lower maximum tour time and accordingly updates the schedule or routes. Consequently, the algorithm re-initiates a fresh detection for an entanglement from the beginning at time $t = 0$. As a last step, the algorithm only checks if the current solution is better than the solution with time scheduling. This step is added because 1) the schedule adjustment can

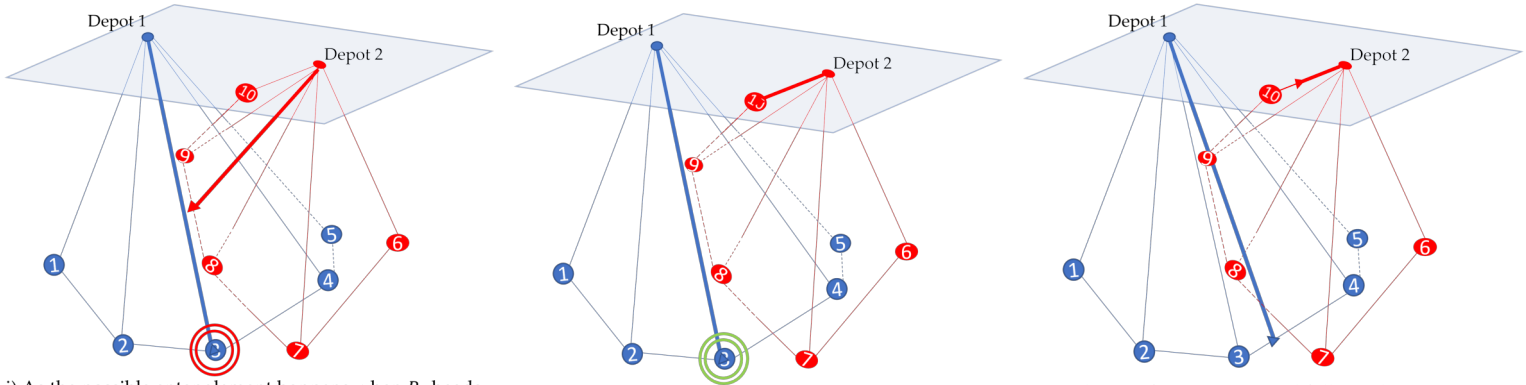
run very fast, and 2) modifying the routes at the beginning causes a drastic delay in the final result for some cases.

The route modification approach in Algorithm 5 tries to find alternate routes by reallocating the node relevant to the entanglement. For example, if the cable vector of R_a intersects with CP_b, the responsible node will be removed from R_a's route and awarded to another vehicle with the lowest maximum tour time. Fig. 3.2 shows how the approach works with an instance. In this instance, there is possible entanglement when R_1 travels between targets 3 and 4 while R_2 travels between targets 7 and 8 in the initial routes as shown in the far left figure. Thus, target 8 is removed from the route of R_2 and inserted for R_1 between targets 3 and 4.



i) R_2's cable at target 8 intersects with the cable plane CP_1. ii) Target 8 is removed from the route of R_2. iii) Target 8 is inserted between targets 3 and 4 for R_1.

Figure 3.2: An instance of the routes modification approach with 2 vehicles and 10 targets.



i) As the possible entanglement happens when R_1 heads to target 4 while R_2 already departed from target 7, R_1 loiters at target 3.

ii) R_1 loiters at target 3 until all possible entanglements are resolved, which is the time R_2 arrives at target 10.

iii) R_1 departs from target 3 and follows the route again.

Figure 3.3: The illustrations for the time scheduling approach with the same instance with Figure 3.2.

On the other hand, the time scheduling approach in Algorithm 6 adjusts the schedules while forcing one of the vehicles involved in the entanglement to loiter on the starting node on the plane involved in possible entanglement. It compares the loitering time to avoid entanglement and chooses a more efficient one. Figure 3.3 shows the rescheduling between the vehicles for the same instance in Figure 3.2. There exists a possible entanglement when R_1 travels between targets 3 and 4 while R_2 travels between targets 7 and 8. As R_2 leaves the departing target earlier than R_1, R_1 waits at target 3 until all possible entanglements are resolved. Thus, R_1 restarts to follow the route when R_2 arrives at target 10.

The proposed heuristic approach produces a feasible solution for every case for the following reasons: 1) The primal-dual heuristic that produces initial task allocation and routes guarantees a feasible solution without considering the entanglement constraint. 2) The proposed heuristic in Algorithm 4-6 ensures the removal of all the possible entanglements from the initial routes. Thus, all constraints are guaranteed to satisfy, although the heuristic may not produce an exact optimal solution.

3.4 Computational Results

The heuristic is implemented and tested in simulations with varying problem sizes for validation. All simulations were performed in a PC with an Intel®Core™ i7-7800X CPU running at 3.5 GHz with 64 GB RAM. The number of vehicles varied from 3 to 10, and the number of targets was tested for 50 and 100. The tests were repeated for 100 different instances for each problem size. The coordinates of targets and depots are randomly generated within a space of 2 m × 2 m × 3.2 m with a uniform distribution. All the depots are constrained to lie in the topmost plane of the defined space. All vehicles had the same average running speed. $Cost_{ij}^{k}$ was set to the minimum travel time by calculating the distance between i and j divided by the average running speed.

The maximum tour time among all the vehicles is considered the operation time. The experiment evaluates the efficacy of the proposed algorithm to provide entanglement-free navigation of the vehicles while aiming to curb the increase in operation time. Due to a lack of available literature on this problem, we have computed posteriori bounds based on the upper and lower bounds utilizing the initial routes produced by the primal-dual heuristic. The worst feasible solution can easily think of is one vehicle departing at a time, and all other vehicles wait until the

vehicle comes back to its depot. For this trivial approach, the operation time is calculated by adding up the respective tour time of all the vehicles, and we consider it as the upper bound. On the other hand, the initial operation time produced by the primal-dual heuristic without considering entanglement constraints is considered the lower bound, which is sometimes impossible to achieve. The equation we used to compute *Posteriori Bound* 1 is the following:

$$PB_1 = \frac{T_{algo} - T_{LB}}{T_{UB} - T_{LB}} \tag{3.8}$$

where T_{LB} represents the lower bound, T_{UB} represents the upper bound, and T_{algo} represents the operation time of the entanglement-free solution produced by the algorithm applied. While this becomes one way to estimate the qualities of solutions, we observed that the upper bound increases a lot as the problem size increases because each vehicle must wait a long time at its depot for other vehicles to complete their missions. Thus, we also computed posteriori bounds only compared with the lower bound as follows:

$$PB_2 = \frac{T_{algo}}{T_{LB}} \tag{3.9}$$

While the proposed algorithm uses a mixed approach between the rescheduling and the route modification depending upon the cost, each problem instance is also

solved using only one approach to compare the efficacy of both methods individually. The computational results are shown in Figure 3.4, and Table 3.1.

Table 3.1
Computation time in seconds

The computation time for the entire process						
vehicles	Proposed	TS	RM	Proposed	TS	RM
	Average with 50 targets			Worst with 50 targets		
3	0.56	0.48	3.54	1.39	1.34	10.81
6	1.15	1.07	3.98	2.79	2.49	9.46
10	1.97	1.83	4.08	4.04	3.17	8.66
	Average with 100 targets			Worst with 100 targets		
3	2.69	2.39	25.19	4.47	3.28	89.07
6	6.56	6.11	28.77	9.99	7.53	112.78
10	11.66	10.84	30.09	21.78	17.56	92.14
The computation time only for entanglement resolving						
vehicles	Proposed	TS	RM	Proposed	TS	RM
	Average with 50 targets			Worst with 50 targets		
3	0.1	0.02	3	0.55	0.1	10
6	0.13	0.17	3	1.12	0.17	7
10	0.28	0.13	2.38	2.50	0.26	7
	Average with 100 targets			Worst with 100 targets		
3	0.33	0.04	23	1.77	0.1	87
6	0.6	0.1	23	3.5	0.46	105
10	1.06	0.23	19	6.8	0.6	77

In Figure 3.4, the left shows *Posteriori Bound* 1, comparing the gaps to the lower bounds with gaps between upper and lower bounds. The right presents *Posteriori Bound* 2, comparing the costs with the lower bounds. The marker shows the average, while the bar shows the minimum and maximum values. In both figures, RM represents the Route Modification approach, and TS represents the Time

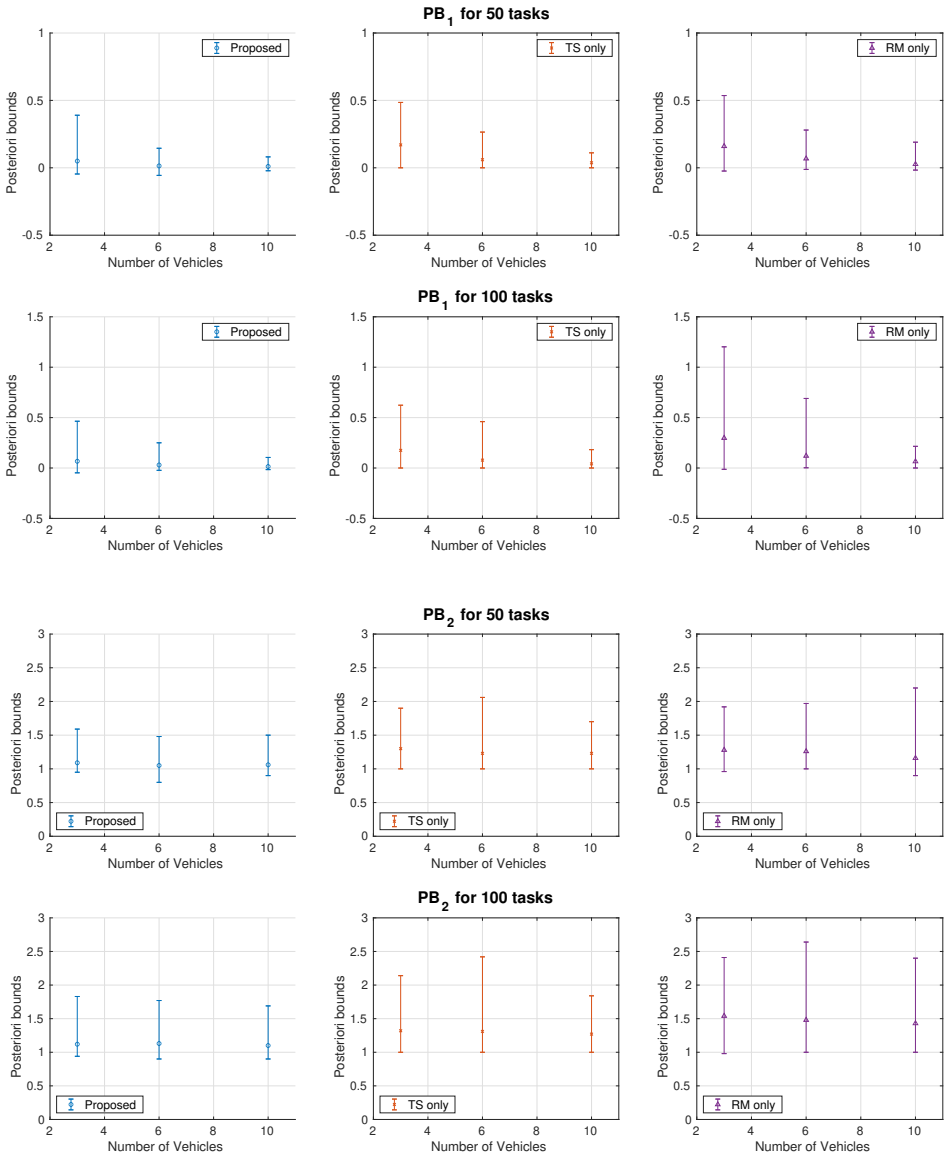
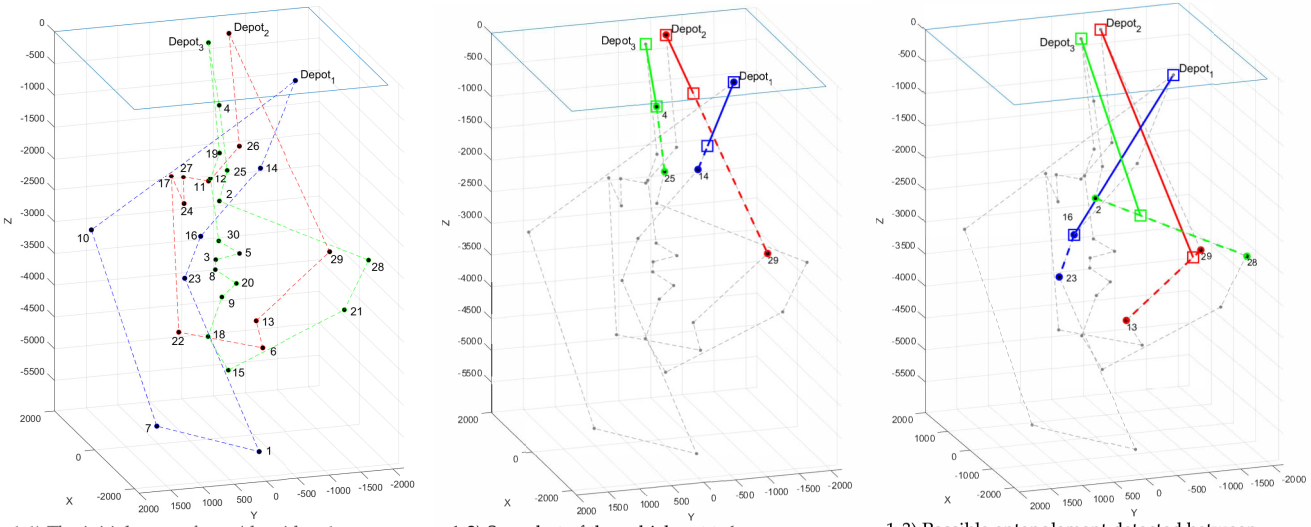


Figure 3.4: Posteriori bounds PB_1 and PB_2 of three approaches

Scheduling approach. The proposed approach is a hybrid of the two approaches. The average posteriori bounds for the proposed approach had the best solution quality among the compared methods while staying considerably closer to the minimum. This means that the results are consistent most of the time while having

some bad cases occasionally. For both posterior bounds, the minimum bounds for the proposed and route modification-only methods have negative and less than 1 values, which shows that the routes are improved by modification from the initial routes provided by the primal-dual heuristic, which is used as the lower bound. While the solution quality stayed consistent for 50 targets, the largest problem size had some increased gap from the lower bound for the proposed method. This makes sense as the number of vehicles and targets increases, and more entanglement issues can arise from the initial routes. The route modification-only method has the worst performance and the longest computation time. This method tries to remove all the possible crossing surfaces by exchanging the nodes, which could result in having an overloaded vehicle with high computation time in some cases. While the time scheduling-only method solves the problem instantly, in less than 1 second for all cases as shown in Table 3.1, the solution qualities were not as good as the proposed approach because some vehicles often need to loiter for a long time to resolve the entanglement issues. Thus, the proposed approach took advantage of both methods and produced the best solutions among the methods within a reasonable computation time. Although it compares the two methods every time it detects a possible entanglement, the proposed method's computation time is considerably shorter than the route modification-only method because changing the route at a certain iteration changes the rest of the schedules. Therefore, the number of possible entanglement changes depends on which method was chosen in the

previous step, and this affects both solution quality and computation time. The computational results show the algorithm's potential to be implemented in real-world applications, delivering an affordable solution within an average of 11.66 seconds for the largest problem size, staying less than 1.5 of the ratio from a lower bound that doesn't consider the entanglement. Lastly, Figure 3.7 shows one of the instances with 3 vehicles and 30 targets. As the possible entanglement is detected for the initial routes produced by Algorithm 3, Algorithm 4 updated the routes to avoid entanglements. As shown in Figure 3.7 of 1-1 and 2-1, some targets were exchanged between the vehicles while time scheduling was also performed in 2-2.

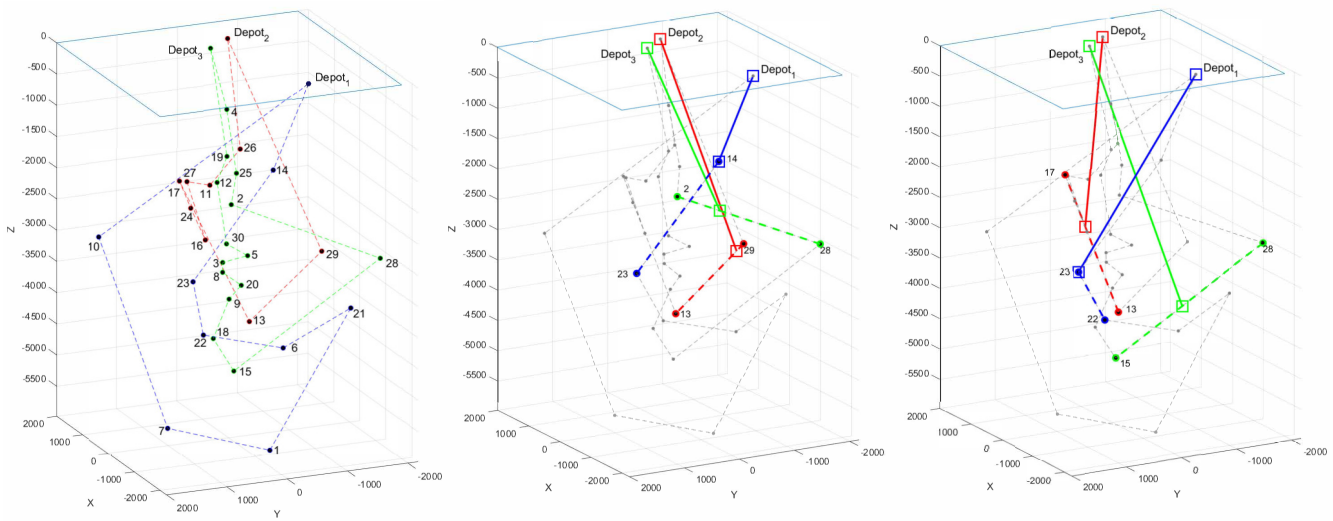


1-1) The initial routes from Algorithm 1

1-2) Snapshot of the vehicles at t=6 sec

1-3) Possible entanglement detected between R_1 and R_3 at t=25.3sec

Figure 3.5: An instance that is solved by the proposed algorithm.

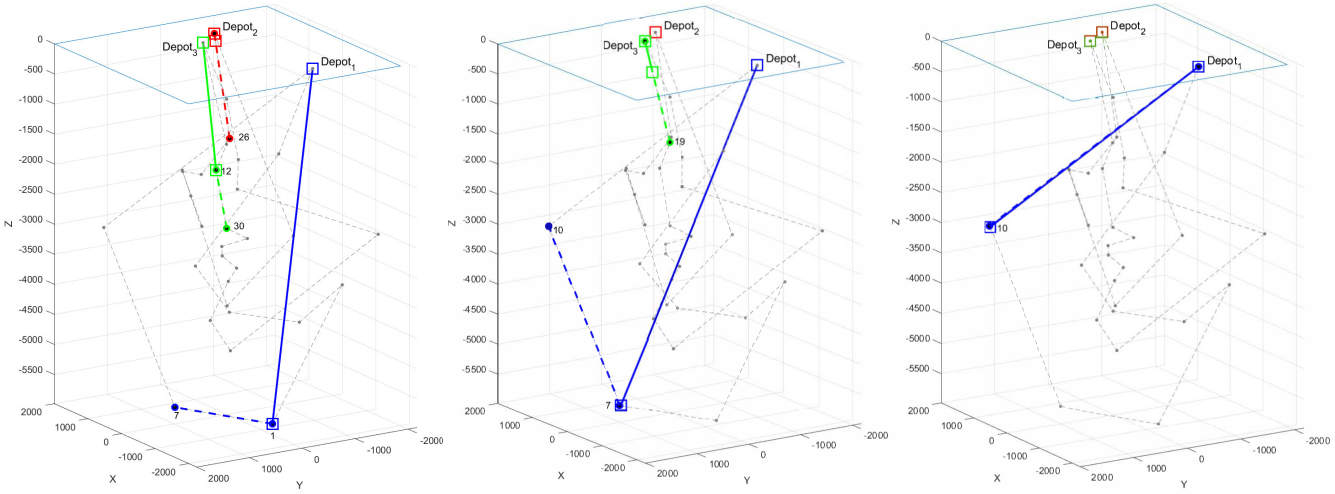


2-1) The final routes from Algorithm 2

2-2) At t=25.3 sec, R_1 waits at target 14

2-3) At t=46 sec, R_1 departs from target 14

Figure 3.6: An instance that is solved by the proposed algorithm. Continued...2



2-4) At t=103 sec, R_2 is about to finish its operation

2-5) At t=116 sec, R_3 is about to finish its operation

2-6) At t=133 sec, R_1 starts to return to its depot

Figure 3.7: An instance that is solved by the proposed algorithm. Continued...3

Chapter 4

A Novel Heuristic for a Multiple Tethered Autonomous Underwater Vehicle Routing Problem

Following the successful resolution of the cable entanglement issue, we focused on optimizing solution quality and computational efficiency. We undertook the development of a novel heuristic designed to integrate all constraints concurrently during task allocation and route planning for vehicles. We aimed to devise a versatile heuristic capable of handling diverse constraints applicable to a system of multiple autonomous vehicles, ensuring robust performance across various scenarios and applications.

4.1 Introduction

With over 70% of the Earth's surface covered by water, unlocking the potential of these submerged realms is crucial for understanding our planet's intricate ecosystems, mitigating environmental threats, and harnessing valuable resources [53, 54]. The extreme and often inaccessible nature of underwater environments has limited our ability to explore and monitor them effectively. However, by employing Autonomous Underwater Vehicles (AUVs), we can overcome these challenges and delve deeper into the mysteries of the ocean. AUVs can be broadly categorized into two classes based on their connection to a control system: tethered autonomous underwater vehicles (T-AUVs) and stand-alone autonomous underwater vehicles (S-AUVs). T-AUVs are equipped with an umbilical cable that links them to a control tower, providing stable power, real-time communication, and data transfer capabilities. Examples of T-AUVs include remotely operated vehicles (ROVs) and hybrid autonomous underwater vehicles (H-AUVs). T-AUVs are preferred for long-duration missions, such as underwater ecosystem exploration, infrastructure inspection and maintenance, and search and rescue operations due to the reliability offered by their umbilical cables. Despite their vast potential, operational challenges, particularly entanglement, hinder their efficacy, as demonstrated during the Tokyo Fire Department's search and rescue mission following a magnitude 9.0 undersea earthquake in 2011 [55]. Entanglement risks, exacerbated when multiple

T-AUVs operate in shared spaces, demand solutions for vehicle routing problems for fully autonomous navigation.

Despite the crucial need for entanglement-free routing in multiple T-AUV operations, the existing literature on this topic is limited. For multiple S-AUVs, significant attention is directed toward either formation control or path planning to achieve optimal area coverage while considering nonlinear dynamics and environmental factors. Numerous studies have explored these aspects, as documented by [8]. However, the challenges surrounding multiple T-AUVs have not been thoroughly addressed. Most of the existing research focuses on the motion planning of autonomous vehicles while avoiding entanglements of the tethers with given starting and ending nodes. Hert and Lumelsky conducted a study on an entanglement-free motion planning algorithm for multiple tethered autonomous vehicles operating in 3D space, aiming to navigate from their respective starting positions to task positions [43]. Their approach employs a simultaneous-motion planning strategy involving the determination of a sequence of vehicle movements to maximize the number of vehicles moving along straight lines toward their tasks. Subsequently, paths are generated for the vehicles based on this sequence, with the objective of minimizing the total path length. Notably, this planning method assumes a straight-line umbilical connecting the host vessel and the vehicle. While theoretically valuable with mathematical proofs on completeness and approximation ratio, this work lacks consideration of complete operational scenarios, including task

assignments and visiting sequences, which is essential for practical applications.

Recent studies have addressed routing problems for multiple tethered ground or aerial vehicles. For instance, Cao et al. introduced a non-entangling trajectory planning approach for teams comprising mobile vehicles and aerial vehicles, proposing a decentralized algorithm that ensures avoidance of entanglement through a tether-aware representation of homotopy [56]. The same team proposed a braids-based algorithm for multiple tethered aerial vehicles [57]. Zhang and Pham presented a motion planner that incorporates precedence constraints and waiting times for navigation from initial to task positions, utilizing a precedence graph to identify deadlocks [52]. While their iterative algorithm effectively removes deadlocks, it lacks a comprehensive objective function for optimization, and there's no assurance of eliminating all deadlocks. Additionally, Peng et al. investigated Non-Crossing Anonymous multi-agent path Finding (NC-AMAPF) in 2D space, aiming to determine non-crossing paths from anchor points to tasks within a workspace containing obstacles with the maximum travel cost among the vehicles minimization objective [58, 59]. Another study by Teshnizi and Shell focused on a graph search algorithm for planning paths for a pair of tethered vehicles in 2D space to prevent entanglement [51]. Several studies have also concentrated on efficient path generation for single T-AUV to ensure their umbilicals do not entangle with static obstacles in a 2D plane [49, 60, 61]. For example, McCammon et al. formulated the Non-Entangling Traveling Salesperson Problem (NE-TSP), utilizing

homotopy augmented graphs and a Mixed Integer Program model for problem resolution [49]. Moreover, literature on single-tethered vehicle path planning on the ground or in the air is more widely available, [62, 63, 64, 65, 66].

In our preliminary work, we successfully solved the challenge of a routing problem of multiple T-AUVs [36]. Our primary aim was to minimize the maximum travel cost among the vehicles while ensuring their cables did not entangle during operation. To achieve this, we developed a multi-layer heuristic, which initially relaxed the cable entanglement constraint, addressing a multiple Depot Traveling Salesman Problem (mDTSP) to generate paths and schedules for each vehicle. Specifically, we designed a primal-dual heuristic using a linear program formulation that efficiently solved a min-max mDTSP in a 3-dimensional environment. Subsequently, the next layer of heuristic identified and resolved potential entanglements within the given routes. After obtaining the routes, the heuristic simulated the movement of the vehicles along their assigned paths. Assuming a straight cable connecting a vehicle to its depot, the heuristic tracked the cable and examined its intersection with other cables as the vehicle progressed along its route. However, a drawback of the multi-layer approach is that the entanglement constraint is not considered during task allocation and path planning. Consequently, the solution quality may degrade in some cases when the second layer of the heuristic redistributes tasks among vehicles or adjusts their schedules. Furthermore, the re-routing/scheduling process increases computational time, particularly in some

cases, depending on the results from the first layer. To address these issues, this work introduces two major contributions. First, we present a modified formulation that incorporates the cable entanglement constraint, aiding in the decision-making process regarding including an edge in a vehicle's route. Second, we propose a new heuristic that simultaneously considers all constraints in the formulation. This modified formulation significantly enhances solution quality at a lower computational expanse compared to the multi-layer approach.

The remaining chapter is organized as follows: In Section 4.2, we describe the problem and present the formulation. Section 4.3 explains the heuristic approach to the problem. The computational results are presented in Section 4.4.

4.2 New Problem Formulation

Given a set of T-AUVs and designated tasks, the problem is to determine individual paths for each T-AUV while satisfying several key criteria: 1) **task Coverage**: Ensure that at least one T-AUV visits every task during the mission; 2) **Motion Constraints**: Each path must adhere to the motion constraints specific to the corresponding T-AUV, accounting for its unique capabilities and limitations; 3) **Entanglement Avoidance**: Prevent any inter-tether entanglement by ensuring that the tethers connecting the T-AUVs to their respective depots and tasks do not overlap

or cross paths; and 4) **Minimized maximum travel cost among T-AUVs**: optimizing the efficiency of the operation. The vehicles depart from their respective depots on the surface and return to the depots after visiting all assigned tasks. We assume symmetric travel costs between two tasks that satisfy the triangle inequalities. The vehicles are considered homogeneous, and travel cost is determined by the travel time between tasks, utilizing the distance between the tasks and the average running velocity of the vehicles. We also assume that the cable remains straight during the operation using a tether control system. The problem is formulated as a min-max Multiple Depot Traveling Salesman Problem. Given a set of m vehicles and n tasks, the parameters and decision variables used in the formulation are described as follows:

Parameters: .

d_k	the depot of k^{th} vehicle
D	a set of depots, $\{d_1, ...d_m\}$
T	a set of tasks, $\{t_1, \cdots, t_n\}$
V_k	a set of vertices for k^{th}, vehicle$\{\{d_k\} \cup T\}$, $\{\{d_k\} \cup T\}$
E_k	a set of edges that connect all vertices in V_k, $\{(i,j),\ \forall i,j \in V_k\}$
$cost_{ij}^k$	the travel cost of the edge from vertex i to vertex j for k^{th} vehicle
$\delta_k^+(S)$	the subset of the edges of E_k that entering into set S from $V_k \backslash S$
t_i^k	the departure time of k^{th} vehicle from task i in its route

CV_{ij}^{k} a cable vector, at any time t, that connects the k^{th} depot to its vehicle which is moving along the edge $\{i, j\}$, where $t_i^k \leq t \leq t_j^k$

Decision variables:

x_{ij}^{k} the decision variable that represents whether edge $\{i, j\}$ is used for the tour of k^{th} vehicle

$$x_{ij}^{k} = \begin{cases} 1 & \text{if edge } (i,j) \text{ is traveled by the } k^{th} \text{ vehicle} \\ 0 & \text{otherwise} \end{cases}$$

$z_{U_i}^{k}$ the decision variable that represents the assignment of tasks in T for $k+1^{th}$ vehicle

$$z_{U_i}^{k} = \begin{cases} 1 & \text{if } U_i \text{ contains task } j,\ \forall j \in T \backslash (V_1 \cup V_2 \cdots \cup V_k), \text{ such that} \\ & \text{while } k+1^{th} \text{ vehicle moves along the edge } \{i,j\},\ CV_{ij}^{k+1} \text{ do} \\ & \text{not contact with } CV_{gh}^{r}, \text{ where } r \neq k+1,\ i \neq g \text{ or } h\ ,\ j \neq g \text{ or } h \\ 0 & \text{otherwise} \end{cases}$$

q the maximum travel cost

The new binary decision variable $z_{U_i}^{k}$ identifies the set of viable tasks for the $k+1^{th}$ vehicle that can be reached from a task i within its route. It becomes 1 only if the set U_i contains all the tasks that are not assigned to any of vehicles $1, \cdots, k$, while doesn't cause any entanglement with the tasks that are assigned to the vehicles $1, \cdots, k$. This aims to establish an edge $\{i, j\}$ in the route, ensuring that the corresponding cable does not come into contact with other cables while the vehicle

traverses along the edge $\{i, j\}$. Based on the provided parameters and decision variables, the problem is formulated as a Mixed Integer Linear Program (MILP), as shown below:

$$C_{LP} = \min q \tag{4.1}$$

$$\sum_{(i,j)\in\delta_1^+(S)} x_{ij}^1 \geq 1 - \sum_{T\supseteq U_i\supseteq S} z_{U_i}^1 \qquad \forall S \subseteq T, \tag{4.2}$$

$$\sum_{(i,j)\in\delta_k^+(S)} x_{ij}^k \geq \sum_{T\supseteq U_i\supseteq S} (z_{U_i}^{k-1} - z_{U_i}^k) \qquad \forall S \subseteq T,\ k = 2, \cdots, m-1, \tag{4.3}$$

$$\sum_{(i,j)\in\delta_m^+(S)} x_{ij}^m \geq \sum_{T\supseteq U_i\supseteq S} z_{U_i}^{m-1} \qquad \forall S \subseteq T, \tag{4.4}$$

$$q \geq \sum_{i,j\in V_k} cost_{ij}^k\, x_{ij}^k \qquad k = 1, \cdots, m, \tag{4.5}$$

$$x_{ij}^k = \{0, 1\} \qquad \forall i, j \in V_k,\ k = 1, \cdots, m, \tag{4.6}$$

$$z_{U_i}^k = \{0, 1\} \qquad \forall U_i \subseteq V_k,\ k = 1, \cdots, m-1, \tag{4.7}$$

$$q \geq 0 \tag{4.8}$$

In this formulation, constraints (4.2)-(4.4) ensure that each task is assigned to one of the vehicles while not having any entanglements. (4.5) represents the min-max objective, connected to (4.1), while binary and non-negativity constraints are included in (4.6)-(4.8).

4.3 A Heuristic for Entanglement-Free Coordination of Multiple Tethered AUVs

This section introduces a new heuristic to solve the problem formulated in Section 4.2. The proposed heuristic involves two primary steps: 1) Generating entanglement-free routes using initial task allocation and 2) Iteratively redistributing the tasks among the vehicles to minimize the maximum travel cost while ensuring that the routes remain free from cable entanglement during the operation. We employed the main structure of the algorithm presented in Algorithm 7, and the functions utilized in Algorithm 7 are presented in Algorithms 8, 9, and 10. The following notations are utilized to present the algorithms.

Algorithm Notations:

R_k	Route of k^{th} vehicle
P_k	Tasks allocated to R_k
τ_k	Total travel cost of k^{th} vehicle to complete its route R_k.
N_k	A queue of nearby depots for the k^{th} depot arranged in ascending order of $\tau_z, \forall z \in N_k$
Δ_j^k	Minimum increase in travel cost of k^{th} vehicle incurred upon transferring the task j into R_k

Θ	A queue of tasks arranged in ascending order of $\Delta_j^{\beta}, \forall j \in P_\alpha$
Q_j	A queue of all vehicles arranged in ascending order of $cost_{jd_k}^k$
$\overrightarrow{CS_{kr}}$	The vector connecting d_k and d_r

Algorithm 7 Coordination of T-AUVs

$[Route, Q] = inital\_allocation(n, m, d, Cost)$
$\eta =$ **true**
$\eta_\psi =$ **true**, $\forall \psi \in \{1, 2, \cdots m\}$
while η **do**
  Let $\kappa = \{1, \cdots, m\}$
  while κ is not empty **do**
    Let $\tau = \{\tau_1, \cdots, \tau_m\}$ where $\tau_k =$ total travel cost of R_k & ψ be the vehicle with maximum $\tau_r, \forall r \in \kappa$
    $\kappa \leftarrow \kappa \setminus \psi$
    $N_\psi = find\_neighbors(R_\psi, Q, \tau)$
    for each $\gamma \in N_\psi$ **do**
      $v_h \leftarrow \max(\tau_\psi, \tau_\gamma)$
      $v_l \leftarrow \min(\tau_\psi, \tau_\gamma)$
      $[Route, \eta_\psi] = load\_transfer(v_h, v_l, \tau_{v_h}, \tau_{v_l}, Route)$
    end for
  end while
  $\eta = \eta_1 \vee \eta_2 \vee \cdots \vee \eta_m$
end while
return *Route*

The heuristic begins with generating the initial allocation of all given tasks to different depots based on the minimum travel cost each depot offers to visit a task individually. The details of the initial allocation strategy are presented in Algorithm 8, which generates entanglement-free routes and queues of depots in cost-effective order for each task.

It starts with an empty partition P_k for all vehicles. For each task j, all the depots

Algorithm 8 $[Route, Q] = initial\_allocation(n, m, d, Cost)$

$P_k \leftarrow \emptyset$ for $k = 1, \cdots, m$
for $j = 1 : n$ **do**
    Compare costs to each of the depots and find Q_j and let q be the first vehicle in Q_j.
    $P_q \leftarrow j$
end for
for $k = 1 : m$ **do**
    Find the optimal route R_k for P_k and d_k.
end for
$Route$ ={$R_1, R_2, \cdots, R_m$}
Q ={ $Q_1, Q_2, \cdots, Q_m$}
return $Route, Q$

Algorithm 9 $N_k = find\_neighbors(R_k, Q, \tau)$

$N_k \leftarrow \varnothing$
Let $P_k = R_k \backslash d_k$
for each $p \in P_k$ **do**
    Let w be the second vehicle in the Q_p
    $N_k \leftarrow \{N_k \cup w\}$
end for
Arrange N_k in ascending order of τ_z for all $z \in N_k$
return N_k

Algorithm 10 $[Route, \eta] = load\_transfer(v_h, v_l, \tau_{v_h}, \tau_{v_l}, Route)$

$\eta =$ **false**
Let $\Theta \leftarrow \varnothing$ and $P_{v_h} = R_{v_h} \backslash d_{v_h}$
for each $p \in P_{v_h}$ **do**
    Calculate $\Delta_p^{v_l}$
    $\Theta \leftarrow \{\Theta \cup p\}$
end for
for each $j \in \Theta$ **do**
    Let $\tau_{avg} = 0.5 * (\tau_{v_h} + \tau_{v_l})$
    Let $\tau_{v_l}^{'}$ be the new total travel cost of v_l after accommodating the task j
    if upon accepting task j, $|\tau_{avg} - \tau_{v_l}| > |\tau_{avg} - \tau_{v_l}^{'}|$ **and** the resultant new routes are entanglement free **then**
        Transfer j from R_{v_h} to R_{v_l}
        Update $Route$, τ_{v_h}, and τ_{v_l}
        $\eta =$ **true**
    end if
end for
return $Route, \eta$

are organized in a queue, called Q_j, arranged in ascending order of the travel cost $cost^k_{d_kj}$. Following the orders of depots into queues, tasks are assigned to the first vehicle of their respective queues.

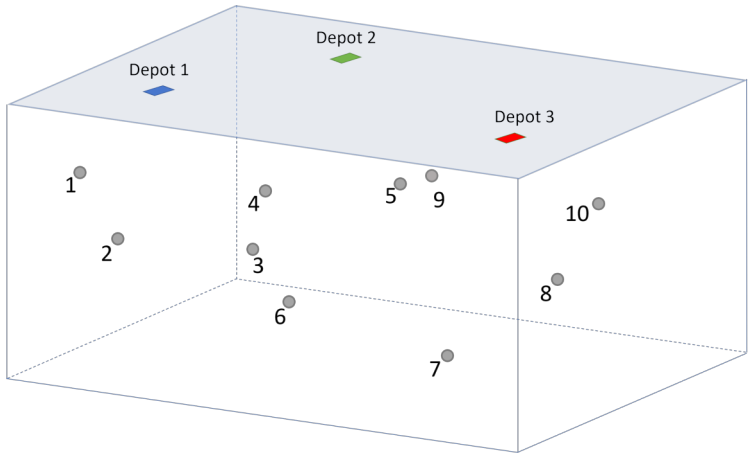


Figure 4.1: An example of initial allocation steps in Algorithm 8: (a) An example with 3 depots and 10 tasks

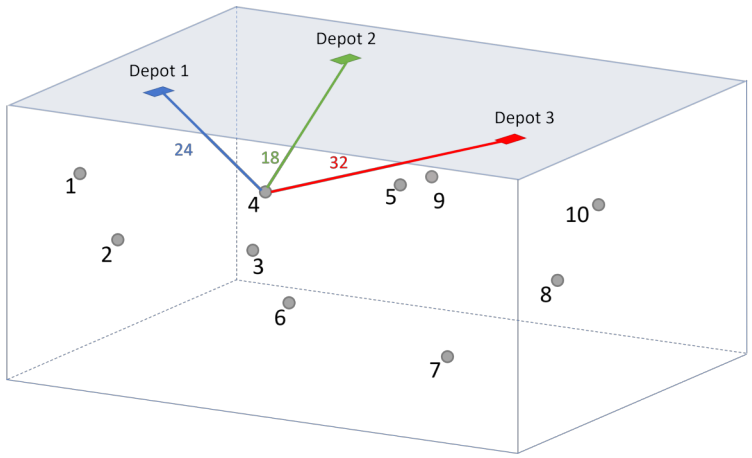


Figure 4.2: An example of initial allocation steps in Algorithm 8: (b) Comparing travel costs will give Q_4 = {2,1,3}

Figures 4.1 to 4.4 show the steps of the initial allocation with an example of 3 vehicles and 10 tasks. For task 4, Q_4 is generated based on the travel costs $cost^k_{d_kj}$ offered

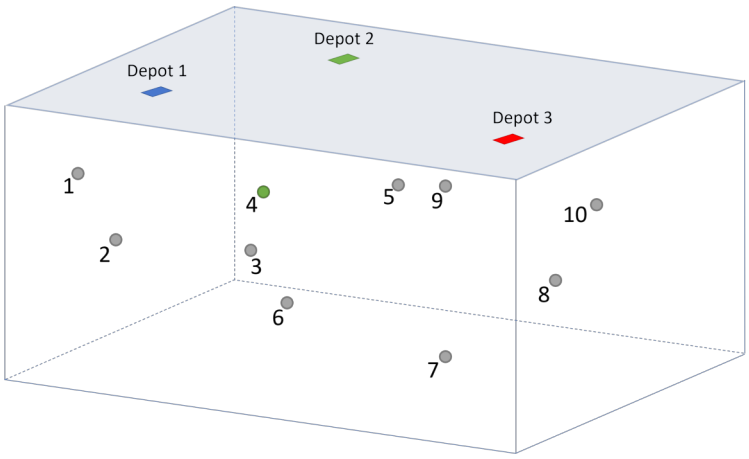


Figure 4.3: An example of initial allocation steps in Algorithm 8: (c) Task 4 is allocated to R_2 based on Q_4

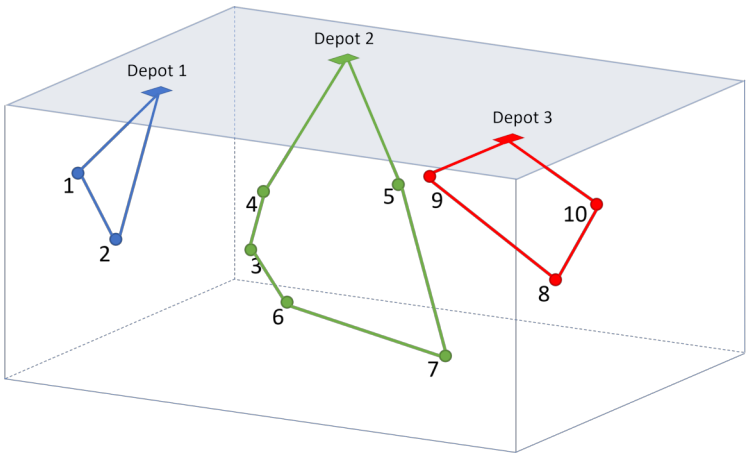


Figure 4.4: An example of initial allocation steps in Algorithm 8: (d) Initial allocation and routes

by each depot. Since d_2 offers the lowest cost, task 4 is allocated to d_2. Similarly, all the tasks are allocated to the nearest depots, and optimal routes are generated to minimize the total travel cost τ_k within the partition and thus generate the initial routes.

Once the initial entanglement-free routes are available, the heuristic iteratively re-distributes the tasks among the vehicles to minimize the maximum travel cost. In

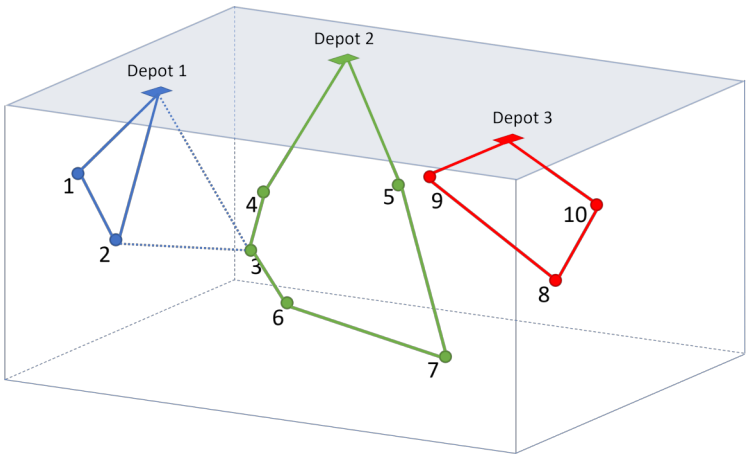


Figure 4.5: An example of target transfer without entanglement of cables in Algorithm 10: (a) Task 3 to be transferred to R_1

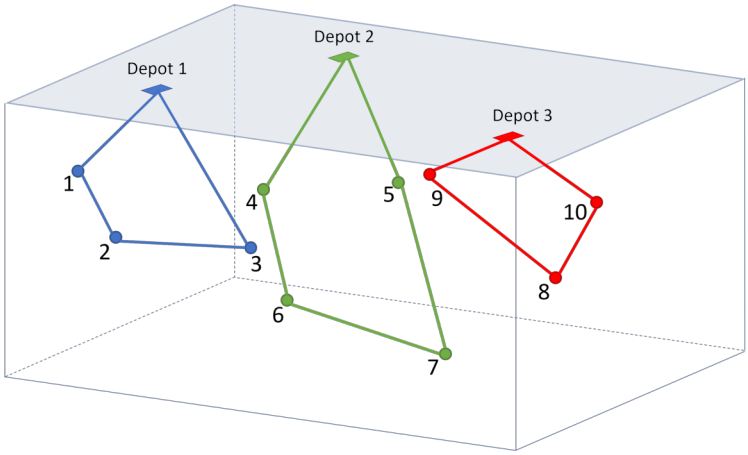


Figure 4.6: An example of target transfer without entanglement of cables in Algorithm 10: (b) Updated R_1 and R_2 do not have entanglement

each iteration, the feasibility of transferring tasks is checked for every vehicle in descending order of the total travel cost τ_k. After selecting a vehicle, the algorithm identifies suitable nearby neighbors for cost-efficient transfers of tasks. Algorithm 9 shows the details of how it generates a set of depots referred to as nearby neighbors N_k for the k^{th} depot, leveraging the previously generated queues Q_j. For each task in P_k, the respective queue of depots is referenced, and the second vehicle

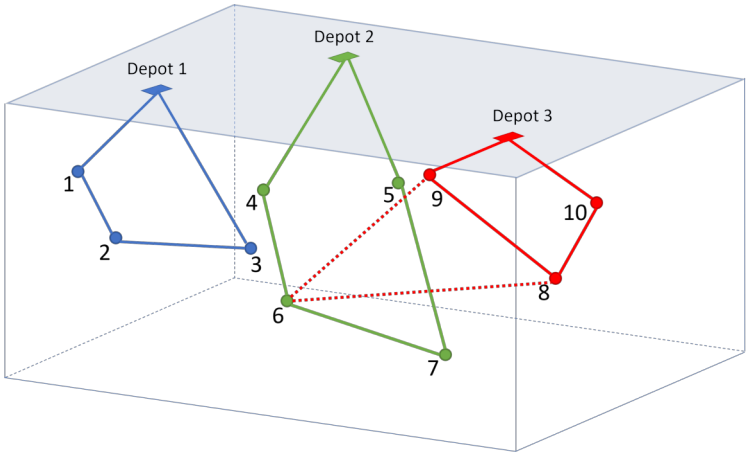


Figure 4.7: An example of target transfer result in entanglement of cables in Algorithm 10: (a) Task 6 to be transferred to R_3

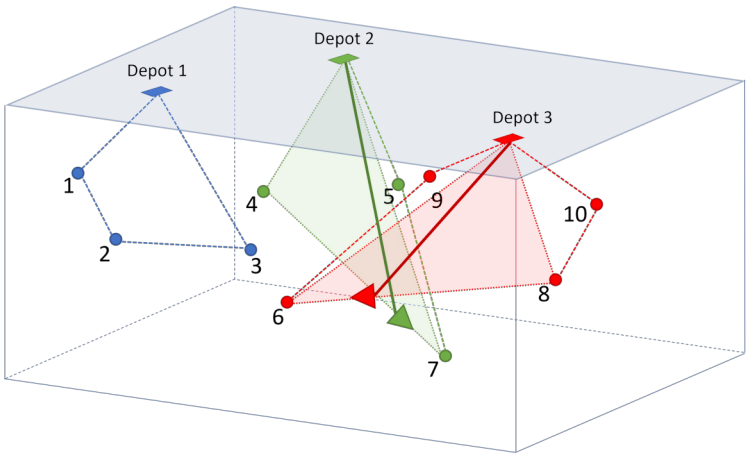


Figure 4.8: An example of target transfer result in entanglement of cables in Algorithm 10: (b) Updated R_2 and R_3 has intersecting cables

in the queue is chosen as the neighbor of k^{th} depot. The chosen neighbor is then placed into the queue of neighbors based on total travel cost. The set of nearby neighbors varies as routes evolve in each iteration, thus providing more opportunities to transfer tasks. As shown in the 4.2, since the second vehicle in Q_4 is 1, d_1 is one of the nearby neighbors for d_2.

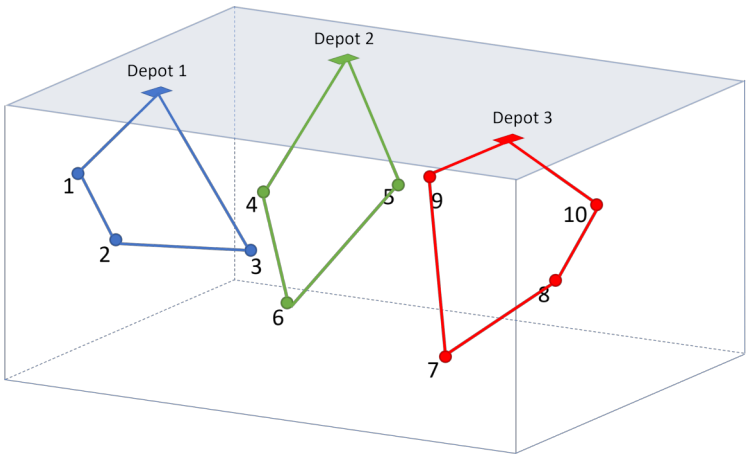


Figure 4.9: The final entanglement free routes of the example

After preparing a queue of suitable nearby neighbors, the algorithm assesses the feasibility of transferring tasks with the neighbors. Depending on the total travel costs of the vehicle and its neighbor, they are assigned the roles of releasing vehicle v_h and receiving vehicle v_l. Algorithm 10 manages the transfer of tasks between the chosen vehicle and its neighbors, provided it is permissible. All the tasks associated with v_h are organized in a queue Θ in the ascending order of the minimum increment in the total travel cost of v_l, $\Delta_p^{v_l}$ when transferring a task p from v_h to v_l. A task is eligible for transfer if both of the following two conditions are met: 1) the absolute difference between the initial average travel cost and the total travel cost of v_l decreases after transferring the task, and 2) the resulting new routes are free from cable entanglement. The first condition aims to balance the workload distribution, while the second condition ensures that cables will not be entangled due to changes in their routes.

We have outlined our method for detecting entanglement in the previous chapter,

Section 3.3. Here, we provide a concise overview of this detection approach. The algorithm simulates vehicle movements by considering their routes and average speeds. Each vehicle's path is represented by a cable sweeping a distinct plane as it moves between consecutive tasks. Consequently, when a vehicle moves away from a task, a new cable plane emerges and is assessed for intersections with other vehicles' cable planes. For a system with n tasks, the algorithm sequentially examines n cable planes for potential entanglements. To assess potential entanglements, the algorithm checks if the cable vectors sweeping these planes become co-planar and if their segments intersect. Figures 4.5 - 4.6 illustrates a scenario where the transfer of task 3 from R_2 to R_1 does not involve any entanglements, while Figures 4.7 4.8 demonstrates an instance where cable vectors CV_{68}^{3} and CV_{47}^{2} become coplanar at time t, satisfying the conditions $t_6^3 \leq t \leq t_8^3$ and $t_4^2 \leq t \leq t_7^2$, and their segments intersect. Two cable vectors become coplanar if there exists a time t that satisfies the following equation, where $t_i \leq t \leq t_j$ and $t_g \leq t \leq t_h$.

$$(\overrightarrow{CV_{ij}^{k}} \times \overrightarrow{CS_{kr}}) \cdot \overrightarrow{CV_{gh}^{r}} = 0 \tag{4.9}$$

The algorithm continues transferring tasks between vehicles in this manner until further task transfers become infeasible for each vehicle. In this problem, the travel cost is assumed to be proportional to the distance, resulting in the allocation of tasks to the nearest depot. Consequently, the cables sweep disjoint cable planes while the vehicles move between the tasks. Hence, the heuristic guarantees the

generation of at least one feasible solution. The detailed proof of the approach is explained in Lemma 2.

Lemma 2. *The proposed heuristic produces a feasible solution that is free from cable entanglement.*

Proof. The proposed heuristic starts with Algorithm 8, which produces vehicle routes by allocating tasks to their nearest depots. This becomes an initial solution, and it tries to improve the solution as much as possible. Thus, we will prove that Algorithm 8 produces an entanglement-free routing for a given problem. While the vehicles traverse the resultant routes, they cannot sweep intersecting cable plans, which we will prove by contradiction. Let's assume that the cable corresponding to a depot d_1 sweeps a cable plane $d_1 n_1 n_2$ while the vehicle moves along the edge $\{n_1, n_2\}$. Similarly, another cable corresponding to depot d_2 sweeps cable plan $d_2 n_3 n_4$. Since the tasks are allocated to the nearest depot, the following inequalities must be satisfied:

$$\overline{d_1 n_1} < \overline{d_2 n_1} \tag{4.10}$$

$$\overline{d_1 n_2} < \overline{d_2 n_2} \tag{4.11}$$

$$\overline{d_2 n_3} < \overline{d_1 n_3} \tag{4.12}$$

If these two cable planes are not disjoint, it implies that at least one edge of a cable plane will intersect with the cable plane corresponding to the other depot. Let's

assume that the edge d_2n_3 intersects with the plane $d_1n_1n_2$ at a point n_ϕ. Extend the line joining d_1 and n_ϕ, which intersects the edge n_1n_2 at a point n_ω. For n_ω, from (4.10) and (4.11) the following relationship must also be true:

$$\overline{d_1n_\omega} < \overline{d_2n_\omega} \tag{4.13}$$

Applying triangle inequality on the resultant triangles, $d_1n_\omega n_3$ and $d_2n_\omega n_3$:

$$\overline{d_1n_\omega} + \overline{n_\omega n_3} > \overline{d_1n_3} \tag{4.14}$$

$$\overline{d_2n_\omega} + \overline{n_\omega n_3} > \overline{d_2n_3} \tag{4.15}$$

Subtracting equation (4.15) from (4.14) gives following :

$$0 > \overline{d_1n_\omega} - \overline{d_2n_\omega} > \overline{d_1n_3} - \overline{d_2n_3} \tag{4.16}$$

$$\overline{d_2n_3} > \overline{d_1n_3} \tag{4.17}$$

But (4.12) and (4.17) contradict each other; therefore, intersecting cable planes can not exist. Thus, the proposed heuristic will provide a feasible solution that is entanglement-free.

4.4 Computational Results

The heuristic is implemented and tested in simulations with varying problem sizes for validation. All simulations were performed on a PC with an Intel®Core™ i7-7800X CPU running at 3.5 GHz with 64 GB RAM. The number of vehicles varied from 3 to 10, and the number of tasks was tested for 50 and 100. The tests were repeated for 100 different instances for each problem size. All the instances are carried forward from the previous work for case-to-case basis comparison. The coordinates of tasks and depots were randomly generated within a space of 2 m × 2 m × 3.2 m with a uniform distribution. All the depots are constrained to lie in the topmost plane of the defined space. All vehicles had the same average running speed. $Cost_{ij}^{k}$ was set to the minimum travel time by calculating the distance between i and j divided by the average running speed.

The maximum tour time among all vehicles is considered the operation time. The experiment evaluates the efficacy of the proposed algorithm in providing entanglement-free navigation of vehicles while aiming to minimize maximum travel costs. In our preliminary work, we solved the problem using a primal-dual based greedy heuristic, which first generates routes for each vehicle and then uses a mixed approach between the rescheduling and route modification depending

upon the cost to resolve any entanglement present in the routes. It was a multi-layer approach where the entanglement constraint was considered only after the routes were generated. In our new proposed approach, the entanglement constraint is resolved simultaneously while allocating a task to a vehicle and building the route.

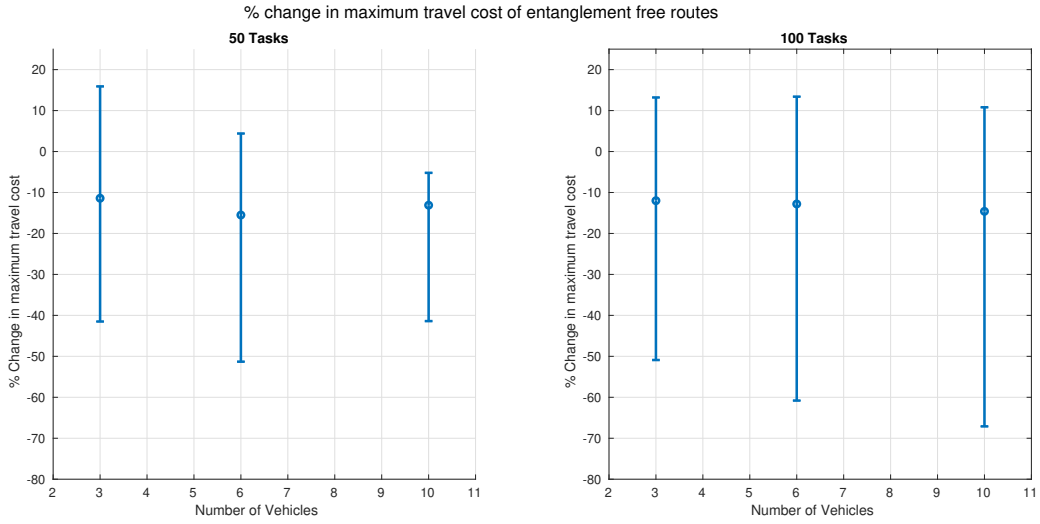


Figure 4.10: Change in maximum travel cost % for entanglement free routes generated by the multi-layer approach vs. the proposed heuristic

Figure 4.10 compares solutions of previously employed multi-layer approach and the current proposed approach. The proposed approach produces, on average, 10-15% efficient solutions for each problem size. The performance also remains consistent for larger-sized problems.

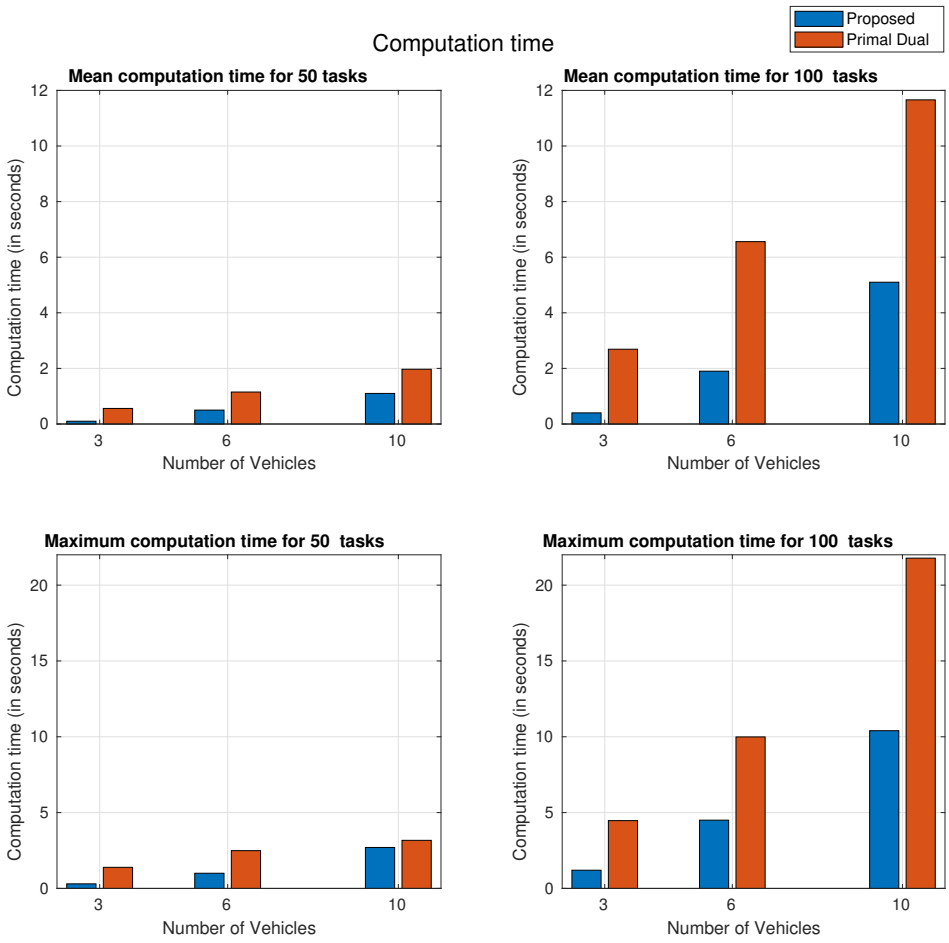


Figure 4.11: Computation time of proposed heuristic to generate entanglement-free routes

The average computation time of the proposed heuristic for the largest problem size is 5 seconds compared to 11 seconds for the previous approach, as shown in Figure 4.11.

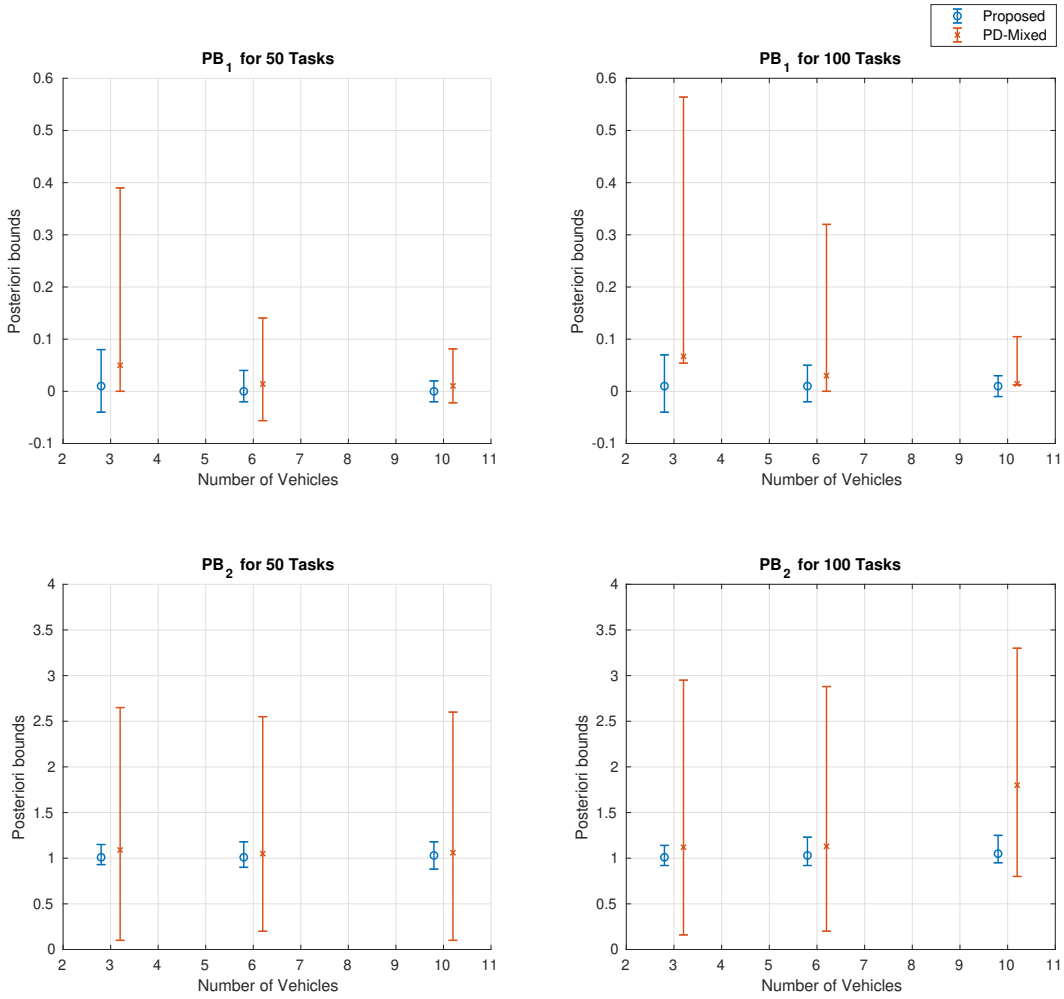


Figure 4.12: Posteriori Bounds

In the previous chapter, we defined posterior bounds, *Posteriori Bound* 1 $(PB_1, 3.8)$ and *Posteriori Bound* 2 $(PB_2, 3.9)$ to assess solution quality by comparing the maximum travel cost, T_{algo} with the trivial solution called Upper bound (UB) which is a feasible but inefficient solution and the primal-dual solution, called lower bound (LB), which is typically not possible to achieve. We compared the

Posteriori Bound 1 and *Posteriori Bound* 2 results for the proposed and multi-layer approaches shown in Figure 4.12. For the proposed approach, the average of *Posteriori Bound* 1 is closer to zero, while the average of *Posteriori Bound* 2 is closer to 1 with narrow distributions, which indicates that the solution quality produced by the proposed algorithm is close to lower bounds.

Furthermore, we slightly tweaked the heuristic and tested it to allocate tasks and path plan routes for a multi-vehicle system with heterogeneity and asymmetric cost, as discussed in Chapter 2. The solution quality of 50 instances is compared for each problem size: 5, 6 vehicles and 40, 50 tasks shown in Figure 4.13. Figure 4.14 compares the computation time.

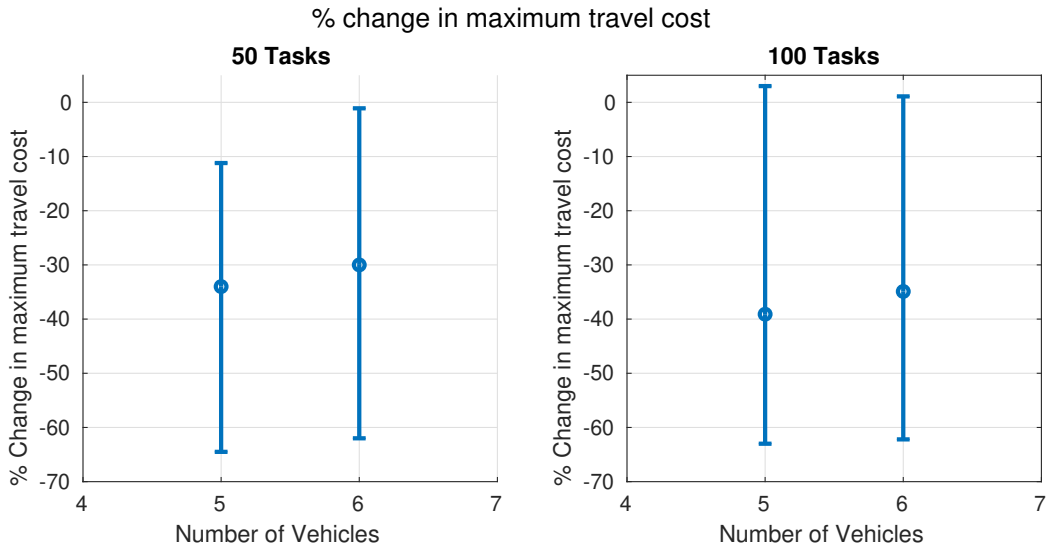


Figure 4.13: Change in maximum travel cost % for Heterogeneous system with asymmetric cost: Primal-Dual vs New Heuristics

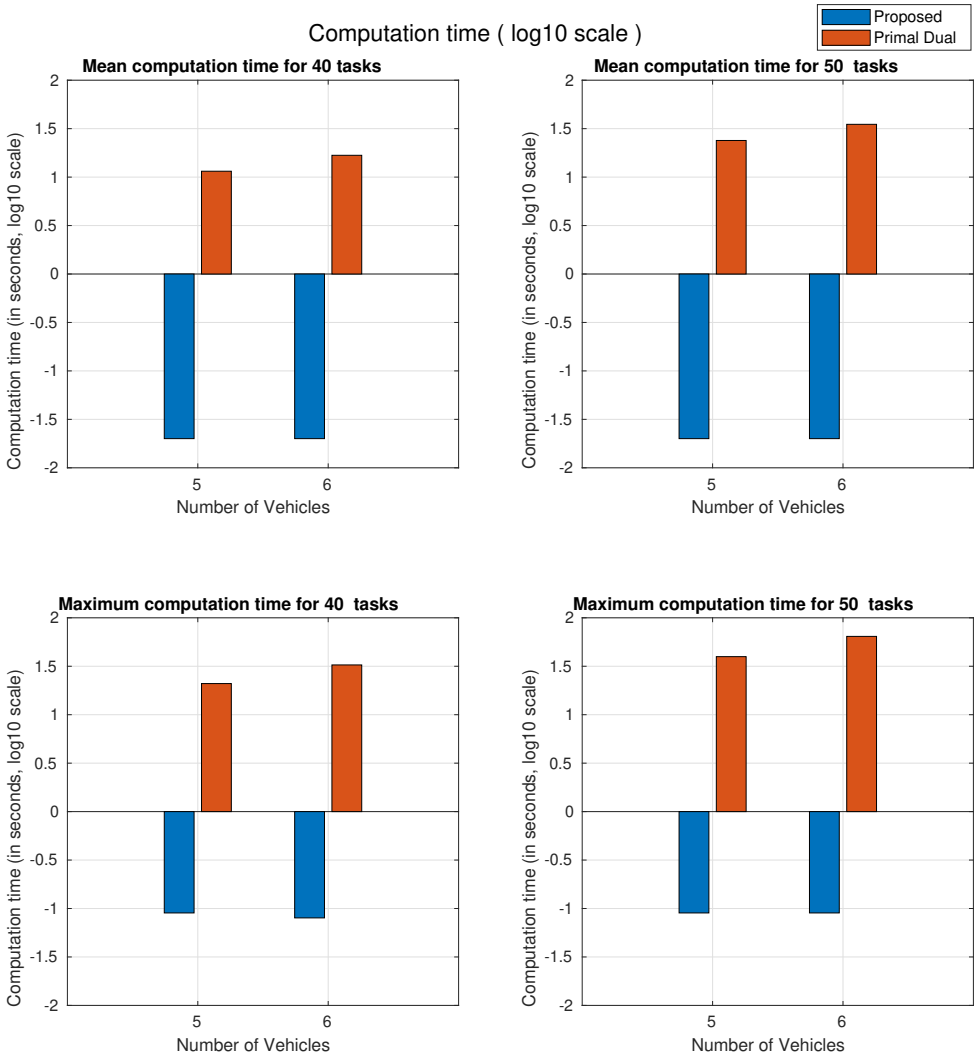


Figure 4.14: Computation time in seconds (log10 scale) for Heterogeneous system with asymmetric cost: Primal-Dual vs New Heuristics

The results show substantial improvement in the solution quality while the computation cost is reduced drastically. Routes for two instances of 5 vehicles and 50 tasks are compared in Figures 4.15 and 4.16.

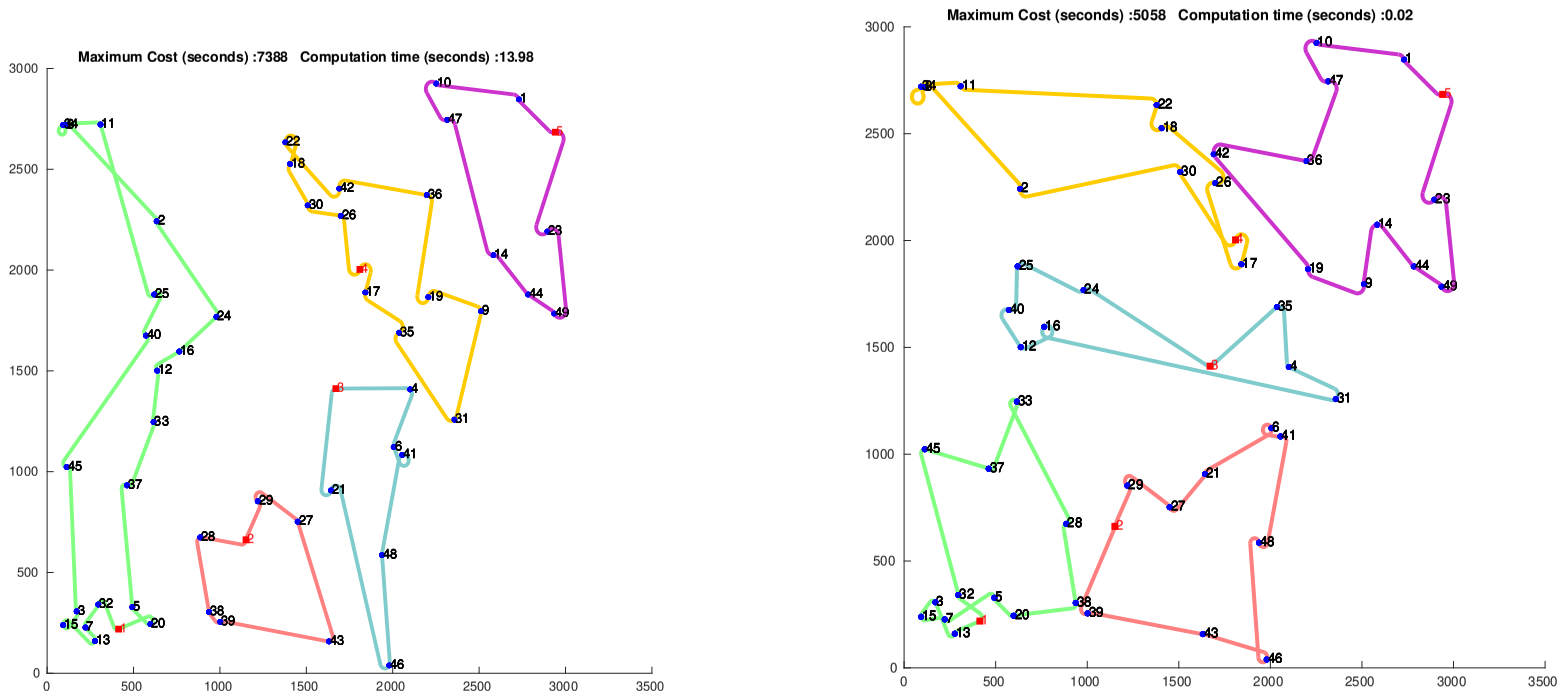


Figure 4.15: Instance 1. Left Figure: Primal-dual output. Right Figure: New heuristic output. The maximum travel cost has been reduced by 31%. The computation time for the new heuristic is 0.02 seconds compared to 14 seconds for the primal-dual approach

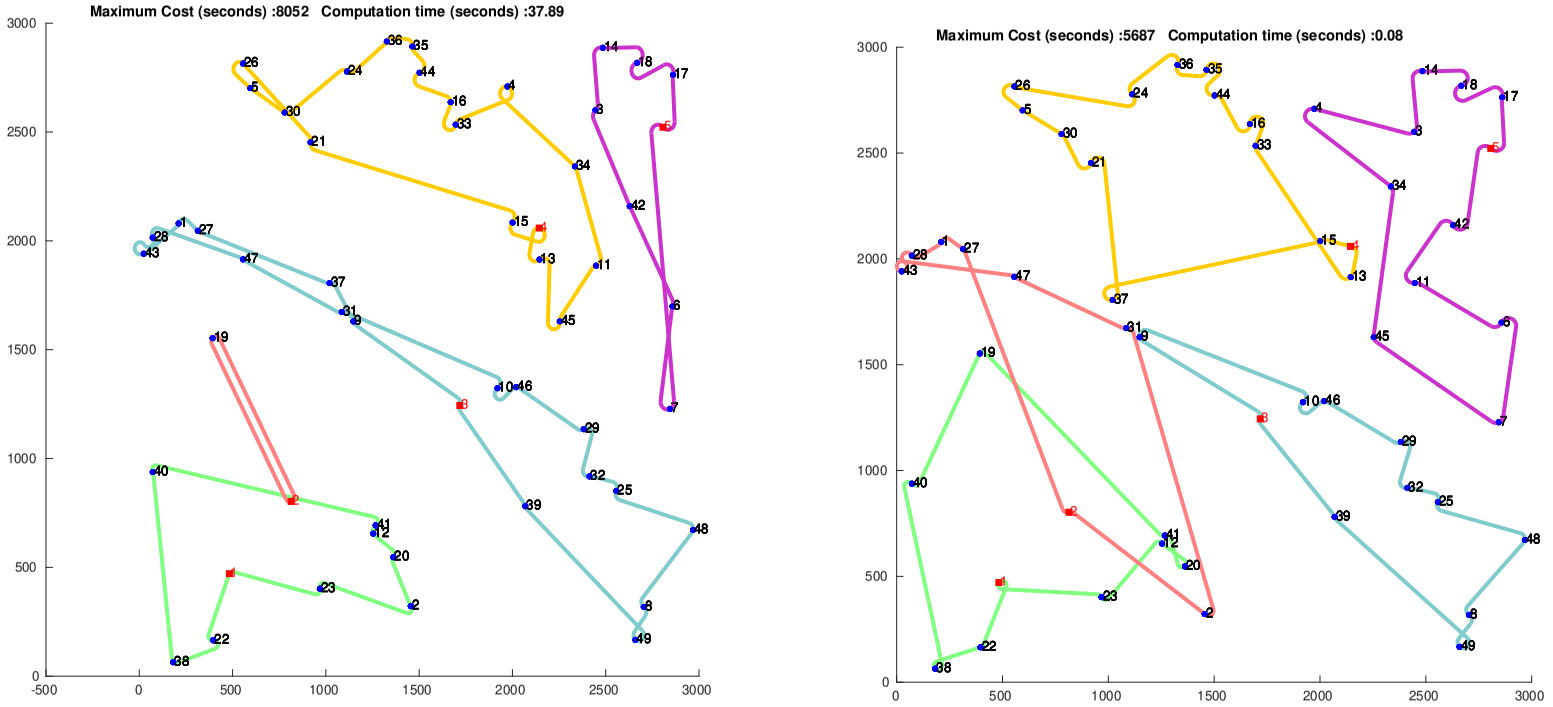


Figure 4.16: Instance 2. Left Figure: Primal-dual output. Right Figure: New heuristic output. The maximum travel cost has been reduced by 29%. The computation time for the new heuristic is 0.08 seconds compared to 38 seconds for the primal-dual approach.

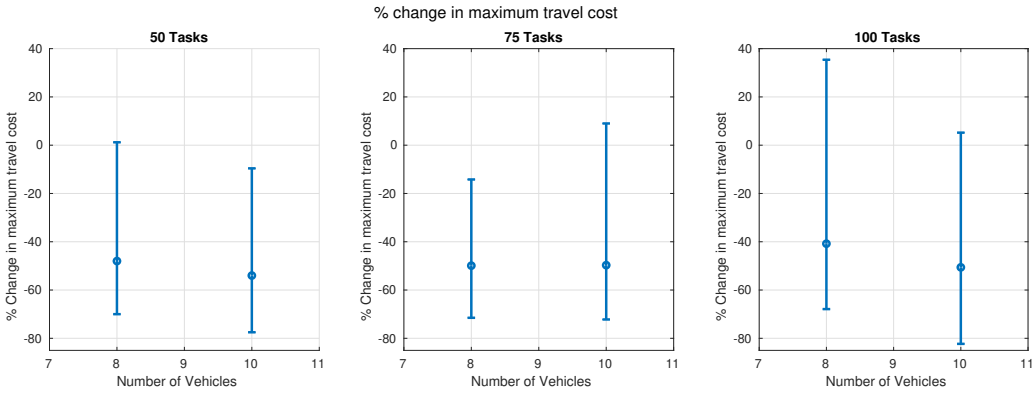


Figure 4.17: Change in maximum travel cost % for solution generated by the primal-dual based approach vs. Proposed new heuristic

The proposed heuristic is extended to solve a pick-drop problem for multiple autonomous vehicles. With minor changes in Algorithm 8 and replacing entanglement constraint with payload constraint in Algorithm 10, we tested the new heuristic on the capacity-constrained problem addressed in paper [67], which used the primal-dual based heuristic approach to solve the problem. Figure 4.17 compares solution quality of 50 instances for each of large-sized problems: 8, 10 vehicles, and 50, 75, and 100 tasks. The new heuristic produced an average of 50 % more efficient solutions compared to the previously employed Primal dual-based approach at remarkably lower computation cost, as shown in Figure 4.18.

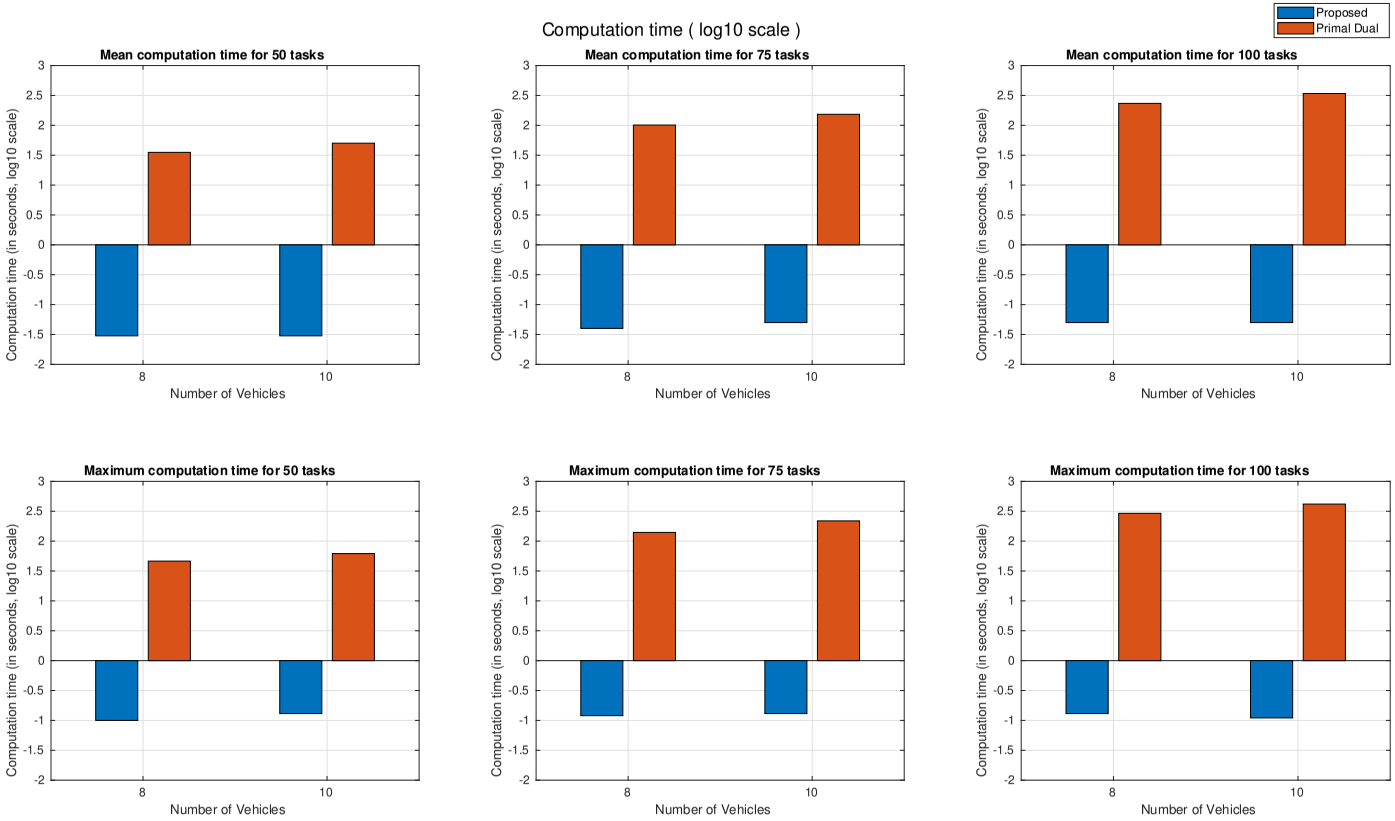


Figure 4.18: Computation time in seconds (log10 scale) for solution generated by the primal-dual based approach vs. New Heuristics

The results highlight the importance of the new heuristic by showcasing its effectiveness and adaptability across different problem sizes. The new heuristic offers a significant advancement in addressing the multi-vehicle routing problem under diverse constraints. Notably, the heuristic consistently delivers robust and reliable coordination plans capable of navigating complex scenarios. Its unique design integrates all constraints simultaneously, ensuring balanced workload distribution and leading to equitable task allocation and route planning. This capability enhances system efficiency and reduces computational costs, making it highly suitable for real-world applications where resource optimization is critical. The new heuristic offers a promising solution to optimize multi-vehicle operations and improve performance in various practical scenarios.

Chapter 5

Conclusion and Future work

This research addresses some critical challenges encountered by multiple autonomous vehicle systems in various real-world applications. The complexity of the problem multifolds when autonomous vehicles with different capabilities and constraints need to efficiently perform tasks while aiming for a balanced workload distribution. The objective of this research has been to develop heuristic methods to tackle these real-world problems effectively. A key emphasis has been placed on achieving balanced workload distribution among the vehicles while adhering to the specified constraints, which is crucial for optimizing overall system performance and successful operation.

One significant contribution of this research lies in guaranteeing practical solutions for planning cable entanglement-free navigation of multiple tethered AUVs, an area where only limited work has been done. This achievement marks a crucial step forward, providing a foundation for further advancements in addressing this challenging issue within the field. By demonstrating effectiveness in simulation environments, this research highlights the algorithm's potential for real-time applications in underwater navigation tasks. This underscores not only its adaptability but also its suitability for addressing challenges in complex and dynamic environments.

Another contribution is the development of a new heuristic approach that exhibits a combination of greedy and exploratory characteristics. This approach allows the algorithm to incorporate all constraints seamlessly and simultaneously while managing the workload balance among the vehicles. The flexibility and adaptability of this heuristic make it applicable to a wide range of scenarios, making it a valuable tool for solving min-max MDHATSP problems with diverse constraints.

Looking ahead, future research directions include further refining the heuristic approach by comparing its performance against more exact solutions and considering additional factors such as realistic cable shapes, vehicle dynamics, and hydrodynamics. While the current version considers fixed depot locations, depot locations can be determined in the problem, which adds another challenge. Additionally, efforts to enhance adaptability to diverse scenarios may involve introducing more heterogeneity among the vehicles and exploring the integration of reinforcement machine learning techniques to further optimize workload distribution.

Furthermore, extending the algorithm's applicability to optimization problems beyond the min-max MDHATSP, such as trajectory tracking controllers, presents exciting opportunities for innovation and advancement. The heuristic can solve optimization problems that may include conflicting constraints, such as cable entanglement constraints, scheduling constraints, time window constraints, etc. By continuing to explore these avenues and refining the proposed approaches, researchers can contribute to the advancement of multiple autonomous vehicle systems and their effectiveness in real-world applications.

References

[1] Kapoutsis, A. C.; Chatzichristofis, S. A.; Doitsidis, L.; De Sousa, J. B.; Pinto, J.; Braga, J.; Kosmatopoulos, E. B. *Autonomous robots* **2016**, *40*, 987–1015.

[2] Krizmancic, M.; Arbanas, B.; Petrovic, T.; Petric, F.; Bogdan, S. *IEEE Robotics and Automation Letters* **2020**, *5*(2), 798–805.

[3] Sutantyo, D.; Levi, P.; Möslinger, C.; Read, M. In *2013 IEEE International Conference on Mechatronics and Automation*, pages 456–462, 2013.

[4] Ribeiro, A.; Conesa-Muñoz, J. *Innovation in Agricultural Robotics for Precision Agriculture: A Roadmap for Integrating Robots in Precision Agriculture* **2021**, pages 151–175.

[5] Dantzig, G. B.; Ramser, J. H. *Management science* **1959**, *6*(1), 80–91.

[6] Vazirani, V. V. *Approximation algorithms*, Vol. 1; Springer, 2001.

[7] Bae, J.; Park, M. *IEEE Robotics and Automation Letters* **2021**, *6*(2), 4064–4070.

[8] Hadi, B.; Khosravi, A.; Sarhadi, P. *Journal of Intelligent & Robotic Systems* **2021**, *101*(4), 1–26.

[9] Patil, A.; Bae, J.; Park, M. *Sensors* **2022**, *22*(15).

[10] Liu, C.; Kroll, A. In *Artificial Intelligence and Soft Computing*, pages 466–474, 2012.

[11] Reinecke, M.; Prinsloo, T. In *2017 1st International Conference on Next Generation Computing Applications (NextComp)*, pages 5–10, 2017.

[12] Ozog, P.; Carlevaris-Bianco, N.; Kim, A.; Eustice, R. M. *Journal of Field Robotics* **2016**, *33*(3), 265–289.

[13] Kanistras, K.; Martins, G.; Rutherford, M. J.; Valavanis, K. P. In *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 221–234, 2013.

[14] Yuan, C.; Zhang, Y.; Liu, Z. *Canadian Journal of Forest Research* **2015**, *45*(7), 783–792.

[15] Murphy, R. R.; Kravitz, J.; Stover, S. L.; Shoureshi, R. June **2009**, *16*(2), 91–103.

[16] Rizk, Y.; Awad, M.; Tunstel, E. W. *ACM Computing Surveys (CSUR)* **2019**, *52*(2), 1–31.

[17] Kiener, J.; von Stryk, O. *Robotics and Autonomous Systems* **2010**, *58*(7), 921–929.

[18] de Oliveira, G. C. R.; de Carvalho, K. B.; Brandão, A. S. *Sensors* **2019**, *19*(5), 1049.

[19] Lin, H.-Y.; Huang, Y.-C. *Sensors* **2021**, *21*(11), 3709.

[20] Wei, Y.; Zheng, R. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pages 1–10. IEEE, 2021.

[21] Lee, K. M. B.; Kong, F.; Cannizzaro, R.; Palmer, J. L.; Johnson, D.; Yoo, C.; Fitch, R. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8685–8691. IEEE, 2021.

[22] Karp, R. 01 **1972**, *40*, 85–103.

[23] Bae, J.; Chung, W. *Sensors* **2019**, *19*(11), 2461.

[24] Dubins, L. E. *American Journal of Mathematics* **1957**, *79*(3), 497–516.

[25] Sun, L.; Escamilla, L. *arXiv preprint arXiv:2107.10350* **2021**.

[26] Li, X.; Li, P.; Zhao, Y.; Zhang, L.; Dong, Y. *IEEE Access* **2021**, *9*, 104115–104126.

[27] Cho, D.-H.; Jang, D.-S.; Choi, H.-L. *Journal of Aerospace Information Systems* **2019**, *16*(5), 168–186.

[28] Bai, X.; Yan, W.; Ge, S. S. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **2021**.

[29] Banik, S.; Rathinam, S.; Sujit, P. B. In *2018 IEEE/OES Autonomous Underwater Vehicle Workshop (AUV)*, pages 1–6, 2018.

[30] Narasimha, K. V.; Kivelevitch, E.; Sharma, B.; Kumar, M. *Swarm and Evolutionary Computation* **2013**, *13*, 63 – 73.

[31] Bae, J.; Chung, W. *International Journal of Precision Engineering and Manufacturing* **2017**, *18*(6).

[32] IBM ILOG CPLEX Optimization studio 12.10. `https://www.ibm.com/analytics/cplex-optimizer`.

[33] LKH-2.0.9. `http://www.akira.ruc.dk/~keld/research/LKH/`.

[34] Turtlebot3 waffle pi. `https://www.robotis.us/turtlebot-3-waffle-pi/`.

[35] Robot operating system (ros). `https://www.ros.org/`.

[36] Patil, A.; Bae, J.; Park, M. *Robotics* **2023**, *12*(3).

[37] Tahir, A.; Iqbal, J. 01 **2014**, *26*, 1111–1117.

[38] Schmickl, T.; Thenius, R.; Moslinger, C.; Timmis, J.; Tyrrell, A.; Read, M.; Hilder, J.; Halloy, J.; Campo, A.; Stefanini, C.; others. In *2011 Fifth IEEE Conference on Self-Adaptive and Self-Organizing Systems Workshops*, pages 120–126. IEEE, 2011.

[39] Sutantyo, D.; Levi, P.; Möslinger, C.; Read, M. In *2013 IEEE International Conference on Mechatronics and Automation*, pages 456–462. IEEE, 2013.

[40] Shkurti, F.; Xu, A.; Meghjani, M.; Higuera, J. C. G.; Girdhar, Y.; Giguere, P.; Dey, B. B.; Li, J.; Kalmbach, A.; Prahacs, C.; others. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1747–1753. IEEE, 2012.

[41] Zhou, M.; Bachmayer, R.; De Young, B. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6091–6097. IEEE, 2017.

[42] Murphy, R. R.; Dreger, K. L.; Newsome, S.; Rodocker, J.; Slaughter, B.; Smith, R.; Steimle, E.; Kimura, T.; Makabe, K.; Kon, K.; Mizumoto, H.; Hatayama, M.; Matsuno, F.; Tadokoro, S.; Kawase, O. *Journal of Field Robotics* **2012**, *29*(5), 819–831.

[43] Hert, S.; Lumelsky, V. *IEEE Transactions on Robotics and Automation* **1999**, *15*(4), 623–639.

[44] Zeng, Z.; Lian, L.; Sammut, K.; He, F.; Tang, Y.; Lammas, A. *Ocean Engineering* **2015**, *110*, 303–313.

[45] Yao, P.; Qi, S. *Science China Technological Sciences* **2019**, *62*(1), 121–132.

[46] Panda, M.; Das, B.; Pati, B. B. In *Innovation in Electrical Power Engineering, Communication, and Computing Technology;* Springer, 2020; pages 327–338.

[47] Nam, H. *IEEE Sensors Journal* **2018**, *18*(21), 8902–8912.

[48] Zhuang, Y.; Huang, H.; Sharma, S.; Xu, D.; Zhang, Q. *ISA transactions* **2019**, *94*, 174–186.

[49] McCammon, S.; Hollinger, G. A. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3040–3046. IEEE, 2017.

[50] Cao, M.; Cao, K.; Yuan, S.; Nguyen, T.-M.; Xie, L. *IEEE Transactions on Robotics* **2023**, pages 1–19.

[51] Teshnizi, R. H.; Shell, D. A. *Autonomous Robots* **2021**, *45*(5), 693–707.

[52] Zhang, X.; Pham, Q.-C. *Robotics and Autonomous Systems* **2019**, *118*, 189–203.

[53] Council, N. R. *Undersea Vehicles and National Needs;* The National Academies Press: Washington, DC, 1996.

[54] Pepin, R. O. 7 **1991**, *92*(1).

[55] Huang, Y.-W.; Sasaki, Y.; Harakawa, Y.; Fukushima, E. F.; Hirose, S. In *OCEANS'11 MTS/IEEE KONA*, pages 1–6, 2011.

[56] Cao, M.; Cao, K.; Yuan, S.; Nguyen, T.-M.; Xie, L. *IEEE Transactions on Robotics* **2023**.

[57] Cao, M.; Cao, K.; Yuan, S.; Liu, K.; Wong, Y. L.; Xie, L. *arXiv preprint arXiv:2305.00271* **2023**.

[58] Peng, X.; Solnon, C.; Simonin, O. In *CP 2021-27th International Conference on Principles and Practice of Constraint Programming*, pages 1–17, 2021.

[59] Peng, X.; Simonin, O.; Solnon, C. *Journal of Artificial Intelligence Research* **2023**, *78*, 357–384.

[60] McCammon, S.; Hollinger, G. A. *Robot Learning and Planning (RLP 2016)* **2016**, page 1.

[61] Wu, J.; Chen, D.; Xu, Y.; Chen, Y.; Lu, L. Vol. Volume 7A: Ocean Engineering of *International Conference on Offshore Mechanics and Arctic Engineering*, 2018.

[62] Kim, S.; Bhattacharya, S.; Kumar, V. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1132–1139. IEEE, 2014.

[63] Kim, S.; Likhachev, M. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4656–4663. IEEE, 2015.

[64] Salzman, O.; Halperin, D. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4161–4166. IEEE, 2015.

[65] Martinez Rocamora Jr, B.; Lima, R. R.; Samarakoon, K.; Rathjen, J.; Gross, J. N.; Pereira, G. A. *Drones* **2023**, *7*(2), 73.

[66] Dos Santos, P. H. G.; Ruiz, F.; Fagiano, L. In *2023 9th International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 391–396. IEEE, 2023.

[67] Liu, Z.; Park, M.; Bae, J. In *2023 IEEE International Conference on Electro Information Technology (eIT)*, pages 294–299, 2023.

[68] Prasad, A.; Choi, H. L.; Sundaram, S. *IEEE Transactions on Control of Network Systems* **2020**, *7*(3), 1511–1522.

[69] Helsgaun, K. *European Journal of Operational Research* **2000**, *126*(1), 106–130.

[70] Reeds, J.; Shepp, L. *Pacific Journal of Mathematics* **1990**, *145*(2), 367–393.

[71] LaValle, S. M. *Planning Algorithms;* Cambridge University Press, 2006.

[72] Patil, A.; Bae, J.; Park, M. *Sensors* **2022**, *22*(15), 5637.

[73] Bae, J.; Patil, A.; Park, M. In *INFORMS Annual Meeting 2022*, 2022.

[74] Bae, J.; Patil, A.; Park, M. In *The US Korea Conference on Science and Technology in the Wake of the Pandemic (UKC2022)*, 2022.

[75] Bae, J.; Patil, A.; Park, M. In *The US-KOREA Conference on Science, Technology, and Entrepreneurship (UKC2021)*, 2021.

[76] Patil, A.; Bae, J. In *ASEE Conference for Industry and Education Collaboration (CIEC)*, 2022.

[77] Das, B.; Subudhi, B.; Pati, B. B. *Transactions of the Institute of Measurement and Control* **2016**, *38*(4), 463–481.

[78] Park, B. S. *Ocean Engineering* **2015**, *96*, 1–7.

[79] Diehl, M.; Ferreau, H. J.; Haverbeke, N. In *Nonlinear model predictive control;* Springer, 2009; pages 391–417.

Appendix A

Letters of Permission