Cooperative task assignment for multiple vehicles

Bai, Xiaoshan

Cooperative Task Assignment for Multiple Vehicles

Xiaoshan Bai

university of groningen

The research described in this dissertation has been carried out at the Faculty of Science and Engineering, University of Groningen (RUG), The Netherlands.

The work was supported by China Scholarship Council (CSC).

Cover design: Wendy Bour || Ipskamp Printing
Printed by Ipskamp Printing
Enschede, The Netherlands

ISBN (book): 978-94-034-1211-5
ISBN (e-book): 978-94-034-1210-8

university of groningen

Cooperative Task Assignment for Multiple Vehicles

PhD thesis

to obtain the degree of PhD at the
University of Groningen
on the authority of the
Rector Magnificus Prof. E. Sterken
and in accordance with the decision by the College of Deans.

This thesis will be defended in public on

Monday 12 November 2018 at 12:45 hours

by

Xiaoshan Bai

born on 18 September 1987
in Hubei, China

Supervisors
Prof. M. Cao
Prof. J.M.A. Scherpen

Assessment committee
Prof. B. Jayawardhana
Prof. H. Li
Prof. W. Ren

Acknowledgments

The completion of this thesis is with the support of many people to whom I am deeply indebted. First of all, I especially thank my first advisor Prof. Ming Cao, who not only gave me an opportunity to carry out my PhD research at the University of Groningen (RUG), but also patiently guided me during the PhD research. Without his supervision and many valuable suggestions and comments on my research, the thesis could not have been completed. I also owe great thanks to my second advisor Prof. Jacquelien Scherpen, who provided a pleasant research environment for the Discrete Technology and Production Automation (DTPA) group. I am grateful for her reading and commenting on the thesis.

I am grateful to Prof. Weisheng Yan (Northwestern Polytechnical University, NWPU), whose keen insight into engineering problems triggered my initial curiosity and interest in the research topic of this thesis. Thank you so much for your guidance and countless support during my study in NWPU. I would like to give my thanks to Prof. Shuzhi Sam Ge (National University of Singapore, NUS), who greatly inspired me during my visiting study in NUS. He guided me on how to logically think and deal with problems not only on research, but also related to my daily life. In his lab, students are greatly encouraged to help each other learn and grow together, and to bravely share their ideas with the other lab members. I also thank Prof. Brian D. O. Anderson (Australia National University) for several meaningful discussions, which guided the main direction for conducting my PhD research. I am deeply impressed by his rigorous attitude and huge enthusiasm for research.

I give my sincere thanks to the thesis reading committee members, Prof. Bayu Jayawardhana, Prof. Huiping Li, and Prof. Wei Ren for reading the thesis and providing valuable comments.

I would like to thank our colleagues. Prof. Bayu, a role model for me, is usually the first one working in office on weekday mornings. I also thank Prof. Claudio De Persis for freely sharing with me how to conduct research, how to guide

students, and the suggestions on how to look for postdoctoral positions. I admire Prof. Yutao Pei very much for his energetic status and intelligence, which deeply impressed me during the last year's group outing. He is a versatile person who not only does good research but also is good at outdoor activities. I also deeply thank Karen Meyer, Frederika Fokkens and Johanna M. Tinga for their contribution to the DTPA, which linked the group participants together just as family members. Great appreciation to Karen for helping me prepare the documents for applying the PhD position with the RUG. I deeply thank Frederika for her good night greetings on each weekday, which really warmed us. Martin Stokroos and Simon Busman, thank you for organizing the DTPA lab.

Special thanks to Martijn Dresscher for his kindness. I like the potatoes fried by Martijn very much. He also shared the monitor of his desktop computer with me, which greatly increased my efficiency at work. I am very glad that he found a job deeply interesting to him before his graduation, and I hope that I can attend his defense ceremony. Thanks Martijn, Xiaodong and Marco Vasquez Beltran for sharing the office with me in DTPA. I give my sincere gratitude to Nelson Chan for translating the summary of this thesis into Dutch and Alain Govaert for the final proofreading. Nelson is a very friendly colleague who maintains traditional Chinese hospitality. Alain is so humorous that one can always have a good laugh when talking with him whatever the discussed topic is. I also want to thank Dong (Tony) Xue for many constructive discussions both on research and daily life topics, which directly contributed to our CDC paper. I am so happy that he recently defended his PhD thesis and hope he finds a promising position soon. I also give my thanks to Yuzhen Qin, who is humorous and intelligent. I remember clearly that Yuzhen received me at the Groningen train station when I first came to Groningen and he also led me around to the supermarkets to buy some daily use products. I deeply thank Mingming Shi for discussing and deducing the principles for optimal control problems, and it is really enjoyable to play table tennis and badminton with him. Special thanks to Liangming Chen and Hongyu Zhang for being my paranymphs. I like to discuss things with Liangming, who is so smart that he always has some good ideas. The days when we traveled together to Paris, Giethoorn and Van Gogh Museum are really good memories to me. Thank you Hongyu for scheduling the bus to take our Chinese students from RUG to Enschede to attend the 2018 Spring Festival Gala hosted by the China Federation of Students and Scholars in the Netherlands. Thank you Carlo Cenedese for organizing the lab sections for the Robotics course. Yu Kawano, Pablo Borja, Ning Zhou, Mengbin Ye, and Liangming, thanks for your companionship while working together in DTPA at evenings. Congratulations to Yu for finding a promising faculty position in your home country. Thank you Michele Cucuzzella for the good morning greetings on each weekday morning, which is really a good habit. The conversations with Matthijs de Jong, Shuai Feng, Erik Weitenberg, Rully Tri Cahyonoe, Zaki Almuzakki, and Os-

car Portolés Marín also broadened my vision. Thank you Sebastian Trip, Danial Senejohnny, Chris de Jonge, Tjerk Stegink, Henk van Waarde, Alessandro Luppi, and Monica Rotulo for bringing interesting presentations on your research work in our group meetings. Tobias Van Damme and Agung Prawira Negara are thoughtful coworkers who usually gathered the group members for lunch. I also thank Yuzhen and his wife Shanshan Ge, Lulu Gong and his wife Feifei Wang, Miao Guo and his wife Bei Tian, Weijia Yao and his girlfriend Sha Luo for dinner invitations to experience traditional Chinese foods provided by you. Thank you Junjie Jiao, Jiajia Jia, Cheng Wang, and Hongyun Liu for bringing many happy moments to our group. I welcome our new colleagues Tábitha Esteves, Vahab Rostampour, Krishna Kosaraju, Zhongqi Sun, Rafael Cunha, Emin Martirosyan, Carmen Chan, Lanlin Yu, and Guopin Liu to join our DTPA group.

I would also like to thank several former colleagues. Thank you Sietse Achterop for organizing the lab experiments and the technical support for setting up our computers. Prof. Jie Huang, Prof. Qingkai Yang, Prof. Yiwen Qi, and Dr. Zhiyong Sun are thanked for sharing their ideas on some interesting research topics with me when they were in DTPA. Jie, Qingkai and Zhiyong are not only intelligent, but also hardworking. It is no wonder that they are so successful in their research careers. I also owe thanks to Prof. Jing Guo for sharing her research experience with me, which encouraged me immensely. I really enjoyed playing badminton with Prof. Chuanjiang Li during his short visit at DTPA. Also I thank Pouria Ramazi and Anton Proskurnikov for their creative thinking both on research and life. I deeply thank the newly graduated Hadi Tagvafard and his considerable help during the last three years. He is so nice that whenever we needed some assistance we would turn to him for help, and I really wish him every success in his postdoctoral life at Leiden University. It was enjoyable playing table tennis with Zakiyullah Romdlony, whose skill is so great that he always wins the match. It is really nice that he defended his PhD thesis successfully and took care of his son during the study. D. Bao Nguyen's laughing face also appears in my memory, and wish him every happiness in his current job. I also thank Yuri Kapitanyuk for his constructive suggestions and comments to us whenever we gave our presentations. He is really a versatile man who always spares no efforts in sharing his expertise with us.

I also thank Long Ma, Yiqun Ma, Fangying Chen, Yingfen Wei, Xiaoying Xi, Yichen Li, Yuan Zou, and Na Li for their company when we lived in the same apartment in Groningen. I cannot forget our happy gatherings held in the apartment where we usually cooked together. Long is so versatile that he does good research while fully enjoying various kinds of outdoor activities. Many thanks to Liqiang Lu and his wife Yuanyuan Wang, Sheng He, Chen Yang, Bin Liu, Yizou Wang, Keni Yang, and Yehan Tao for taking the English course with me. Great appreciation to Mr. Yao Zhang, Jiaxi Zhang, Bing Han and his wife Haoxiao Zuo, Youhai Li and his wife Jingjing Wang, Rui Yan, Xuanbo Feng, Yue Sun, Rongge

Zhang, Tianle Xu, Yiyang Dong, Junyue Xue, Huifang Yin, Chenming Peng, Cheng Chen, Emily, Suxiao Li, Rui Wu, Yuan Wu, Chunshuang Qu, and Tiancen Hu for having so many gatherings together in Groningen, which are valuable memories to me. Special thanks to Man Zhang for countless encouragement and support for my study in RUG.

I also thank Chenglong Deng for organizing the weekly badminton activities in ACLO Sportcentrum for us, which really enriched our lives in RUG. Congratulations to Chenglong for graduating from RUG and I wish you a smooth application to the following master study. Great thanks to Huatang Cao, Yifei Fan and his wife Yihui Wang, Lulu, Liangming, Weijia, Mengbin, Yi Yu, Qian Wang, Feng Yan, Yangyang Guo, Prof. Ning Ding, Xuewen Zhang, Chengyong Xiao, Kailan Tian, Huan Liu, Keni Yang, Yonglian Chen, Kaisheng Zhen, Xin Jiang, Zhiwen Wang, Xiaocui Wang, Zhibo Li, Yuzhen Feng, Qi Xu, and Gang Ye for playing badminton with me, which is really enjoyable.

Last but not the least, I would like to express my love and my gratitude to my family for their constant support and encouragement.

Xiaoshan Bai
Groningen
October, 2018

Contents

Chapter 1
Introduction

THIS thesis studies the task assignment problem for multiple dispersed vehicles (sometimes also taken as mobile robots) to efficiently visit a set of target locations in challenging environments posing constraints on vehicles, such as the winds or currents in a drift field, a limited communication range of the vehicles, and the precedence constraints that specify which targets need to be visited before which other targets. For the task assignment of multiple vehicles in a time-invariant drift field, we first investigate the path planning algorithm for a vehicle to travel between two prescribed locations in the drift field with the minimum travel time. Then, we look into the task assignments for the vehicles to visit a set of target locations in a time-invariant drift field with and without obstacles. For the task assignment of vehicles with limited communication range, we study the cooperative strategies for a group of dispersed heterogeneous vehicles to efficiently visit a set of target locations while having an arbitrary communication network, which in particular is not required to be connected. For the task assignment of vehicles under precedence constraints, we integrate topological sorting techniques with assignment algorithms to enable a team of heterogeneous vehicles to quickly visit a set of target locations respecting every precedence constraint.

1.1 Multi-vehicle task assignment problem

The increasing automation level of human being's activities both in civil and military applications poses higher requirement on the intelligence of the various autonomous vehicles/robots, such as unmanned aerial vehicles (UAVs) [57, 68], unmanned marine vehicles [11, 63], ground vehicles/robots [8, 76] and sensor networks [19, 70]. Among these applications, task assignment for one or multiple vehicles has been an important research area due to its high theoretical value and wide engineering applications, as shown in Fig. 1.1. Generally speaking, task assignment for vehicles is to properly assign a certain subset of tasks to each individual vehicle such that the vehicles can complete the whole mission at a minimal cost while satisfying every given constraint.

The tasks required to be performed by the vehicles can be discrete, such as package carrying and delivering, and the patrolling of a set of target locations

Figure 1.1: (a) Robots are exploring on Mars [18]; (b) The Swarm-bots used for environmental mapping [35].

of interest, or continuous, such as the persistent surveillance of chosen area. In particular, the task assignment for one or multiple vehicles to visit a set of target locations, while minimizing some objective such as the vehicles' maximum travel time [72] and total travel distance [81], has wide applications in logistics, terrain mapping, environmental monitoring, and disaster rescue [4, 12, 72, 81]. The problem can be taken as a variant of the traveling salesman problem (TSP) or the vehicle routing problem (VRP), which are both NP-hard [47, 48]. The TSP focuses on designing one route with the minimum length for a salesman/vehicle to visit a set of dispersed customers while the VRP aims at employing multiple vehicles to efficiently deliver products/packages to a set of dispersed customers.

As researchers design, build, and employ cooperative multi-vehicle systems, they invariably encounter the fundamental question: which vehicle should execute which task in order to achieve the global goal cooperatively [31]? This question must be answered, even for relatively simple multi-vehicle systems, and the importance of task allocation grows with the complexity, in size and capability, of the vehicle system under study [15]. From the decision-making standpoint, task assignment approaches are either centralized (a single control vehicle makes task allocation decisions for all vehicles) [15, 20] or distributed, yet cooperative (each vehicle or each small coalition formed by several vehicles decides which tasks to do themselves) [49, 50]. Centralized implementations are generally more accurate (i.e., yielding high-quality, even optimal solutions) than distributed schemes and rely on the information gathered by all the vehicles. However, they entail non-negligible communication overhead, and have poor fault tolerance and scalability properties. In comparison, the distributed approaches do not require such a powerful central decision-maker with the tradeoff that they make the solution more complicated. However, distributed control strategies find their growing ap-

plications in complex networked systems under information constraints, such as limited sensing range and limited bandwidth in communication [54].

1.2 Multi-vehicle task assignment in challenging environments

This thesis addresses the coordination of a fleet of dispersed vehicles to efficiently visit a set of target locations in challenging environments. The motivation of investigating the multi-vehicle task assignment in challenging environments is discussed in this section.

Generally speaking, three components require special attention in the framework of the multi-vehicle task assignment problem: environmental factors such as winds or currents in a drift field affecting the motion dynamics of the vehicles, the communication network of the vehicles governing the vehicles' interaction, and the service demands coming from the target locations requiring how the targets will be visited. First, the environmental factors such as winds or currents in a drift field have a direct influence on the motion dynamics of the vehicles, where different rates of change of the navigation angle of a vehicle generally result in different times for the vehicle to travel through two prescribed locations. Second, the communication network of the initially dispersed vehicles might not be connected due to the vehicles' limited communication range, so the vehicles are subject to local, and possibly outdated, information. Finally, the service demands such as the sequence priorities on visiting the target locations have a great influence on how to construct the vehicles' paths to visit all the target locations. In particular, this thesis studies the multi-vehicle task assignment in challenging environments from the three aspects listed below.

1.2.1 Multi-vehicle task assignment in a drift field

When a vehicle's motion is affected by external disturbance such as winds or currents in a drift field, the multi-vehicle task assignment problem consists of two sub-problems, namely how to optimally navigate a vehicle from one location to a target location and how to determine the sequences for the vehicles to visit the target locations. Deeply coupled with the target locations' assignments, the planning of the vehicles' paths between two given locations need to be carried out before or at the same time when the task assignment is performed because some off-line designed paths may not be optimal or even not be feasible in the presence of time-varying or strong time-invariant drift fields [73]. For path planning, the vehicle's dynamics need to be taken into account to design robust cooperative

behavior among multiple vehicles. A properly designed navigation control is necessary to minimize the time for each vehicle to travel between two given locations in the drift field. The path planning algorithm provides the cost matrix for the target assignment, and generates routes once the target locations are assigned to the vehicles. Hence, it is of practical importance to investigate the vehicles' path planning in the multi-vehicle task assignment problem.

1.2.2 Multi-vehicle task assignment under limited communication range

One of the most important factors in decision making for a multi-vehicle system is the information constraint incurred by a complicated environment especially in an underwater environment; existing research has considered limited communication range, bandwidth and time delay [61, 81]. When designing cooperative strategies for multi-vehicle task assignment, not many existing works have addressed communication constraints, such as the vehicles' limited communication range. The partial knowledge of the surrounding environment caused by the vehicles' disconnected communication topology and their limited sensing range could lead to task conflicts between different vehicles, thus resulting in resource waste; for example, several communication disconnected vehicles may move toward the same target which could be sufficiently handled by one single vehicle. When the vehicles' communication is not perfect as those in [72, 81], the main challenge in the studied multi-vehicle task assignment problem is to design what information should be carried by each vehicle, what information carried by each vehicle should be communicated to its directly communication-connected (CC) vehicles through local communication, how to merge the received local, possibly outdated, information, and how to coordinate the vehicles in each CC vehicle subgroup to guarantee that the overall tasks will be completed.

1.2.3 Precedence-constrained multi-vehicle task assignment

In recent years, parcel delivery to customers/targets is facing new challenges as e-commerce has grown vastly [58] where the benefit of using micro drones as additional support for package delivery has been identified [27]. Consequently, some leading retailers or distributors such as Amazon and DHL have planned to employ micro drones for small package deliveries. However, micro drones are subject to short operation range and small payload capacity which greatly restrict their efficiency to function in an autonomous delivery network [53]. To overcome the limitations, some investigation has been done to consider a heterogeneous team consisting of one carrier truck and one micro drone with complementary capabilities [53, 60, 69, 77]. For logistic scheduling, another challenge is that

some customers/targets can have priority over the others to be served due to their urgency or importance. In these cases, the precedence constraints on the visiting sequence of customers have to be respected, and the positioning of one customer in the sequence is directly affected by the customers which are required to be served earlier. Thus, we investigate the precedence-constrained task assignment problem for a truck and a micro drone to deliver packages to a set of dispersed customers subject to the sequence priorities on visiting the target locations.

1.3 Outline and main contributions of this thesis

This thesis is divided into three main parts in which the multi-vehicle task assignment problem in challenging environments is investigated subject to constraints, from respectively the environment, the vehicles and the targets/customers.

In Chapter 2, we study the multi-vehicle task assignment problem where several dispersed vehicles need to visit a set of target locations in a time-invariant drift field while trying to minimize the vehicles' total travel time. It is assumed that the amplitude of the vehicles' velocity is greater than that of the currents in the drift field for all locations, which makes it possible for the vehicles to reach every target location. A path planning algorithm is first designed, which leads to the minimum time for a vehicle to travel between two prescribed locations in the drift field. Then, several clustering-based task assignment algorithms are proposed, where two of the algorithms guarantee that all the target locations will be visited within a computable maximal travel time. This maximal time is at most twice of the optimal when the travel cost matrix is symmetric.

In Chapter 3, as a follow-up of Chapter 2, we investigate the multi-vehicle task assignment problem where a fleet of dispersed vehicles is used to visit a set of target locations in a time-invariant drift field with obstacles. The vehicles have different capabilities, and each kind of vehicles needs to visit a certain type of target locations; each target location might have the demand to be visited more than once by different kinds of vehicles. The objective is to visit all the target locations while minimizing the vehicles' total travel time. A path planning method has been designed to enable the vehicles to move between two prescribed locations in a drift field with minimal time while avoiding obstacles. We show that this task assignment problem is NP-hard, and propose an auction-based distributed task assignment algorithm to assign the target locations to the vehicles using only local communication.

In Chapter 4, we look into the task assignment for heterogeneous vehicles under limited communication range, where each vehicle initially has the position information of all the targets and of those vehicles that are within its limited communication range, and each target demands a vehicle with some specified capa-

bility to visit it. A decentralized auction algorithm is proposed, which guarantees that all the target locations are visited in a finite time irrespective of the vehicles' communication range, and the vehicles' total travel distance is within twice of the optimal when the vehicles are initially communication-connected. We illustrate that a longer communication range of the vehicles does not necessarily lead to a better performance of the algorithm, which can also be the case for other decentralized algorithms due to the lack of global information.

In Chapter 5, we study the precedence-constrained task assignment for a team of heterogeneous vehicles to deliver packages to a set of dispersed customers subject to precedence constraints that specify which customers need to be visited before which other customers. A truck and a micro drone with complementary capabilities are employed where the truck is restricted to travel in a street network and the micro drone, restricted by its loading capacity and operation range, can fly from the truck to perform the last mile package deliveries. The objective is to minimize the time to serve all the customers while respecting every precedence constraint. This task assignment problem is shown to be NP-hard, and a lower bound on the optimal time to serve all the customers is constructed by using tools from graph theory. We design several task assignment algorithms integrated with a topology sorting technique, which show the superior performances compared with popular genetic algorithms.

We conclude the thesis and provide discussions for possible future work in Chapter 6.

1.4 Selected publications

Journal papers:

- X. Bai, W. Yan, M. Cao. "Clustering-Based Algorithms for Multivehicle Task Assignment in a Time-Invariant Drift Field." *IEEE Robotics and Automation Letters*, 2(4): 2166-2173, 2017.
- X. Bai, W. Yan, M. Cao, S. S. Ge. "Distance optimal task assignment for heterogeneous robots under limited communication range." *IEEE Transactions on Cybernetics*. *Under review*, 2018.
- X. Bai, M. Cao, W. Yan, S. S. Ge. "Efficient routing for precedence-constrained package delivery for heterogeneous vehicles." *IEEE Transactions on Automation Science and Engineering*. *Under review*, 2018.
- X. Bai, W. Yan, M. Cao, D. Xue. "Distributed multi-vehicle task assignment in a time-invariant drift field with obstacles." *IET Control Theory & Applications*. *Under review*, 2018.

Conference papers:

- X. Bai, W. Yan, M. Cao, J. Huang. "Task assignment for robots with limited communication." *36th IEEE Chinese Control Conference (CCC)*, pp. 6934-6939, Da lian, China, Jul. 2017.
- X. Bai, W. Yan, M. Cao, D. Xue. "Heterogeneous multi-vehicle task assignment in a time-invariant drift field with obstacles." *56th IEEE Conference on Decision and Control (CDC)*, pp. 307-312, Melbourne, Australia, Dec. 2017.

Conference abstracts:

- X. Bai, M. Cao. "Time optimal task assignment for robotic networks with limited communication range." *Benelux Meeting on Systems and Control*, Soesterberg, Netherlands, Mar. 2016.
- X. Bai, M. Cao. "Clustering-based algorithms for multi-vehicle task assignment in a time-invariant drift field." *Benelux Meeting on Systems and Control*, Spa, Belgium, Mar. 2017.

1.5 Preliminaries

We present some basics on algebraic graph theory and several definitions, which will be used in the subsequent chapters.

1.5.1 Graph theory

This section introduces some useful notations from graph theory, and more details can be found in [10].

A graph $\mathbb{G} = (\mathcal{V}, \mathcal{E})$ consists of a set of vertices $\mathcal{V}$, and a set of edges $\mathcal{E}$. A *directed* graph, or a digraph for short, is a graph where each edge in $\mathcal{E}$ is denoted by an ordered pair of vertices. Let $(i, j), i, j \in \mathcal{V}$, denote an edge which starts at vertex i and ends at vertex j. In this thesis, we will only consider *simple* graphs, i.e. graphs that do not contain self-loops $(i, i), \forall i \in \mathcal{V}$. A graph $\mathbb{G}$ is *undirected* if $(i, j) \in \mathcal{E} \Leftrightarrow (j, i) \in \mathcal{E}$ for each pair of $i, j \in \mathcal{V}$. It is straightforward that undirected graphs can be treated as special directed graphs.

A *walk* W in a graph is an alternative sequence of vertices and edges, say $v_0, e_1, v_1, e_2, \ldots, e_n, v_n$, where $e_i = (v_{i-1}, v_i), 0 < i \leqslant n$. The walk from v_0 to v_n is denoted by $v_0 v_1 \ldots v_n$. This walk is called a *trail* if all the edges are distinct. A trail whose endvertices coincide (a closed trail) is a *circuit*. A circuit in a graph $\mathbb{G}$ containing all the edges is said to be an *Euler circuit* of $\mathbb{G}$, and a graph is *Eulerian*

if it has an Euler circuit. Note that a *path* is a walk with distinct vertices. If a walk $W = v_0 v_1 \dots v_n$ is such that $n \geqslant 3, v_0 = v_n$, and the vertices $v_i, 0 \leqslant i < n$, are distinct from each other, then W is said to be a *cycle*. A graph is *acyclic* if it does not contain any cycles. A graph is *connected* if there is a path from i to j for each pair of $i, j \in \mathcal{V}$. A graph $\mathbb{G}$ is a tree if it is acyclic and connected. A *Hamiltonian path* in a graph is a path containing every vertex of the graph, and a *Hamiltonian cycle* is a cycle containing every vertex of the graph. If we consider weights on the edges of a graph, the weighted graph can be described by $\mathbb{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ with the vertex set $\mathcal{V}$, the edge set $\mathcal{E}$, and the map $\mathbb{W} : \mathcal{E} \to \mathbb{R}$ defines the *weight* for each edge in $\mathcal{E}$. Such weights might represent, for example, lengths, costs or utilities, etc.

1.5.2 Related definitions

We first introduce the definition of the arborescence of a digraph.

Definition 1. *(Arborescence [34]) An arborescence is a digraph with a single root in which, there is exactly one directed path from the root to every other vertex.*

Based on Definition 1, we extend the concept of arborescence with a single root to a general one with several roots.

Definition 2. *(Generalized arborescence) A generalized arborescence is a digraph with several roots in which, there is exactly one directed path from one and only one of all the roots to every non-root vertex.*

Chapter 2

Multi-vehicle task assignment in a time-invariant drift field

THIS chapter studies the multi-vehicle task assignment problem where several dispersed vehicles need to visit a set of target locations in a time-invariant drift field while trying to minimize the vehicles' total travel time. Using optimal control theory, we first design a path planning algorithm to minimize the time for each vehicle to travel between two given locations in the drift field. The path planning algorithm provides the cost matrix for the target assignment, and generates routes once the target locations are assigned to a vehicle. Then, we propose several clustering strategies to assign the targets, and we use two metrics to determine the visiting sequence of the targets clustered to each vehicle. Mainly used to specify the minimum time for a vehicle to travel between any two target locations, the cost matrix is obtained using the path planning algorithm, and is in general asymmetric due to time-invariant currents of the drift field. Let the weight of a directed edge between two vertices be the minimum travel time between them respecting the orientation. We show that one of the clustering strategies can obtain a min-cost arborescence of the asymmetric target-vehicle graph where the sum of all the edge weights of the arborescence is minimum. Using tools from graph theory, a lower bound on the optimal solution is found, which can be used to measure the proximity of a solution from the optimal. Furthermore, by integrating the target clustering strategies with the target visiting metrics, we obtain several task assignment algorithms. Among them, two algorithms guarantee that all the target locations will be visited within a computable maximal travel time, which is at most twice of the optimal when the cost matrix is symmetric. Finally, numerical simulations show that the algorithms can quickly lead to a solution that is close to the optimal.

2.1 Introduction

When a vehicle's motion is affected by external disturbance such as winds or currents in a drift field, the multi-vehicle task assignment problem consists of two sub-problems, namely how to assign subtasks as sequences of target locations to individual vehicles and how to navigate a vehicle from its initial location to a tar-

get location optimally. There are some research works considering both the target assignment and path planning for the employed vehicles [23, 37, 84]. By simply requiring moving in straight lines between prescribed locations, Han and Chung [37] employed an autonomous underwater vehicle (AUV) to optimally visit several target points considering the ocean currents and obstacles. Furthermore, to enable multiple AUVs to visit several target points in the time-varying (in a discrete time scale) 3-D underwater environment, Zhu *et al.* [84] employed the velocity synthesis approach to enable each AUV to reach its targets along the shortest path and used the self-organizing map neural network to realize the multi-AUV target point assignment. Grid-modeling based graph methods were designed by Eichhorn [23] for vehicle path planning in a time-varying environment. Delmerico *et al.* [16] used active aerial exploration for robot path planning through an unknown terrain for search and rescue missions. The path planning methods minimizing the travel distance between two given locations in [37, 84] do not necessarily lead to the minimal travel time between the locations. More importantly, since the metric matrix representing the minimal travel time between the target locations is in general asymmetric, the existing algorithms, e.g. the Prim algorithm [43], may fail to guarantee their performances.

In our work [80], the multi-AUV routing problem was studied in temporally piecewise constant ocean currents aiming at minimizing the vehicles' total travel time. In addition, time-optimal coverage control of multiple vehicles in a drift field was studied in [85] where the time-optimal paths were generated over a sequence of discrete time instants. In this chapter, we investigate the task assignment problem for which several dispersed vehicles need to visit a set of target locations in a time-invariant drift field while trying to minimize the vehicles' total travel time. To solve the problem, we first design a path planning method to deal with the vehicle path planning in currents. Then, we propose several clustering strategies to assign the target locations to the vehicles, and we use two metrics to put the target locations assigned to each vehicle in an ordered sequence. Our main contributions are as follows. Firstly, based on the accessible area analysis and optimal control theory, the proposed planning algorithm can generate the time-optimal path for a vehicle to travel between two prescribed locations in a drift field, which provides the travel cost matrix to be used later for the task assignment. Secondly, a lower bound on the optimality of the solution to the task assignment problem with the asymmetric travel cost matrix is achieved using one of the proposed clustering strategies. As the task assignment problem is NP-hard [41, 43], the lower bound can be used to approximately measure the quality of a solution. Lastly, we have studied how the asymmetric travel cost matrix caused by the drift field influences the performances of different clustering algorithms. Two novel algorithms, in the form of integrating the clustering strategies with the target-inserting metrics, guarantee that the vehicles' total travel time to visit all the target locations is within a

reasonable computable upper bound, which, when the cost matrix is symmetric, is twice of the optimal.

The rest of this chapter is organized as follows. In Section 2.2, the formulation of the task assignment problem is given. Section 2.3 presents the path planning algorithm which generates the optimal navigation control law, and in Section 2.4 several target clustering strategies and two target inserting metrics are discussed. We present the simulation results in Section 2.5 and conclude the chapter in Section 2.6.

2.2 Problem formulation

Consider a fleet of m homogeneous vehicles initially randomly distributed in a planar time-invariant drift field. They need to visit n target locations while trying to minimize their total travel time. The vehicles are not required to return to their initial locations after visiting the targets (namely we are considering a variation of the open vehicle routing problem [26]), and their net speed is affected by the speed of the currents in the drift field.

2.2.1 Formulation as an optimization problem

We use the vector $\vec{v}_c = [v_{cx}, v_{cy}]^T$ to describe the drift velocity of the time-invariant field with respect to some coordinate system fixed to the ground. Note that $\vec{v}_c$ changes with locations. We assume that the vehicles are driven by constant thrust, and consequently their velocity $\vec{v}$ is with constant speed v relative to the field [71, 73]. Since the dimension of the drift field is significantly larger than the vehicles' size, we assume that the vehicles are free of turning ratio constraints. The kinematics of each vehicle are

$$\dot{x} = v\cos\psi + v_{cx}, \qquad \dot{y} = v\sin\psi + v_{cy}, \tag{2.1}$$

where $[x, y]^T$ is the vehicle's position and ψ is the vehicle's navigation angle.

We label the target locations by $1, \ldots, n$, and let $\mathcal{T} = \{1, \ldots, n\}$ be the set of these indices. Let $\mathcal{R}$ denote the set of indices of all the vehicles' initial locations, namely $\mathcal{R} = \{n+1, \cdots, n+m\}$, $m \leqslant n$. For each pair of distinct $i, j \in \mathcal{T} \cup \mathcal{R}$, let $t(i, j)$ denote the minimal time for a vehicle to travel from i to j using a properly designed navigation control. Let σ_{ij} be the path-planning mapping that maps the indices $i \in \mathcal{T} \cup \mathcal{R}$ and $j \in \mathcal{T}$ of the starting and ending locations of a vehicle to a binary value, which equals one if and only if it is planned that there exists a vehicle travels from location i to j. So $\sigma_{ii} = 0$ for all $i \in \mathcal{T} \cup \mathcal{R}$. Then, the problem

is to minimize the vehicles' total travel time for visiting all the target locations

$$f = \sum_{i \in \mathcal{R} \cup \mathcal{T}, j \in \mathcal{T}} t(i,j)\sigma_{ij}, \tag{2.2}$$

subject to

$$\sum_{i \in \mathcal{R} \cup \mathcal{T}} \sigma_{ij} = 1, \qquad \forall\, j \in \mathcal{T}; \tag{2.3}$$

$$\sum_{j \in \mathcal{T}} \sigma_{ij} \leqslant 1, \qquad \forall\, i \in \mathcal{T} \cup \mathcal{R}; \tag{2.4}$$

$$\sum_{i,j \in \mathcal{S}} \sigma_{ij} \leqslant |\mathcal{S}| - 1, \qquad \forall\, \mathcal{S} \subseteq \mathcal{T}, |\mathcal{S}| \geqslant 2. \tag{2.5}$$

Constraint (2.3) ensures that each target location is visited once and only once; (2.4) ensures that each target's and vehicle's initial location is departed at most once; and (2.5) guarantees that there is no subtour among the target locations.

Remark 2.1. *If ignoring the effect of the field currents on the speed of the vehicles, the task assignment problem just presented reduces to the uncapacitated multi-depot open vehicle routing problem with the symmetric travel cost matrix [45]. We refer the interested reader to [45] for detailed discussions on the relationship between a standard vehicle routing problem and the multi-depot open vehicle routing problem.*

After formulating the task assignment problem as a constrained minimization problem, we present in the following section a component of the path planning that is critical for solving the overall optimization problem.

2.3 Path planning algorithm given the starting and target locations

To plan the optimal path that minimizes the time for a vehicle to travel through two prescribed locations in a given field with currents, we first look at the accessible region of a vehicle starting from an arbitrary location. Then using optimal control theory, we construct the navigation rule that guides a vehicle to travel between two given locations following the path using the minimum time.

2.3.1 Accessible region analysis

As before, $\vec{v}_c$ is used to denote the velocity of the currents, which changes with location; its amplitude is v_c. As in (2.1), the vehicle's velocity relative to the field

is $\vec{v}$ with the amplitude v. Similarly, we use $\vec{v}_{net}$ to denote the vehicle's net velocity with amplitude v_{net}. Obviously, the vehicle can reach all locations of the field given enough time if $v > v_c$. For this reason, we make this standing assumption for the rest of the chapter.

Assumption 2.1. *It holds for all locations of the field and all time that $v > v_c$.*

Consequently, with this assumption, each vehicle can travel from any given location to any given other target location and the travel time depends on the path planned and the associated navigation rule, which will be discussed in the following subsection.

2.3.2 Optimal navigation law

We now show how to navigate a vehicle between any two given locations of a time-invariant drift field with the minimum travel time.

Lemma 2.2. *Under Assumption 2.1, for a vehicle with kinematics (2.1), to travel with the minimum time between any given starting and ending locations of the time-invariant drift field with the current velocity $\vec{v}_c$, the rate of change of the vehicle's optimal navigation angle ψ^* must satisfy*

$$\dot{\psi^*} = -\frac{\partial v_{cx}}{\partial y}\cos^2\psi^* + (\frac{\partial v_{cx}}{\partial x} - \frac{\partial v_{cy}}{\partial y})\sin\psi^*\cos\psi^* + \frac{\partial v_{cy}}{\partial x}\sin^2\psi^*. \quad (2.6)$$

Proof. Let t_0 and t_f be the vehicle's starting and finishing times respectively. Then to minimize the vehicle's travel time, is to minimize the objective function

$$J = \int_{t_0}^{t_f} dt = t_f - t_0.$$

Define the corresponding Hamiltonian to be

$$\begin{aligned} H(t,[x,y]^T,\lambda,\psi) &= 1 + \lambda^T[\dot{x},\dot{y}]^T \\ &= 1 + \lambda_1(v\cos\psi + v_{cx}) + \lambda_2(v\sin\psi + v_{cy}), \end{aligned} \quad (2.7)$$

where $\lambda = [\lambda_1, \lambda_2]^T$ is the two-dimensional Lagrangian multiplier. From Pontryagin's minimum principle of variational analysis in optimal control theory [40, P188], it must be true that the optimal Lagrangian multiplier λ^* and the optimal

navigation angle ψ^* satisfy

$$\dot{\lambda}^* = -\frac{\partial H}{\partial [x,y]^T}, \tag{2.8}$$

$$0 = \frac{\partial H}{\partial \psi^*}. \tag{2.9}$$

Since (2.9) holds for all $t \geqslant t_0$, the time derivative of its right-hand side must also be zero. So we have

$$\dot{\lambda}_1^* \sin\psi^* + \lambda_1^* \dot{\psi}^* \cos\psi^* = \dot{\lambda}_2^* \cos\psi^* - \lambda_2^* \dot{\psi}^* \sin\psi^*.$$

Combining with what can be obtained from (2.8)

$$\begin{aligned} \dot{\lambda}_1^* &= -\lambda_1^* \frac{\partial v_{cx}}{\partial x} - \lambda_2^* \frac{\partial v_{cy}}{\partial x}, \\ \dot{\lambda}_2^* &= -\lambda_1^* \frac{\partial v_{cx}}{\partial y} - \lambda_2^* \frac{\partial v_{cy}}{\partial y}, \end{aligned}$$

the optimal navigation control ψ^* must satisfy (2.6) when $t_f - t_0$ is minimized.

□

Because of Assumption 2.1, we know that a solution ψ, and thus the optimal solution ψ^*, always exist. Lemma 2.2 gives a necessary condition on $\dot{\psi}^*$; what remains to be determined is the initial orientation $\psi^*(0)$. After knowing $\psi^*(0)$ and $\dot{\psi}^*$, the optimal navigation angle $\psi^*(t)$, $t > 0$, can be determined through the integration of $\dot{\psi}^*$ over t. However, to determine $\psi^*(0)$ with the initial location $[x_0, y_0]^T$ and the finishing location $[x_f, y_f]^T$, one needs to solve the two-point boundary problem:

$$\begin{cases} x_f = x_0 + \int_{t_0}^{t_f} [v\cos(\psi^*(0) + \int_{t_0}^{t} \dot{\psi}^* d\tau) + v_{cx}] dt, \\ y_f = y_0 + \int_{t_0}^{t_f} [v\sin(\psi^*(0) + \int_{t_0}^{t} \dot{\psi}^* d\tau) + v_{cy}] dt. \end{cases} \tag{2.10}$$

The solution $\psi^*(0)$ to (2.10) in general can be found numerically using the shooting method [59]. As becomes clear later in an example, the structure of the current velocity $[v_{cx}, v_{cy}]^T$ in the field can be utilized to simplify the computation.

Let i, j and k be three arbitrary different locations. We first give some property on the optimal travel time matrix of the task assignment problem.

Lemma 2.3. *The minimum travel times for a vehicle to travel between two locations can be asymmetric; the minimum travel times between any three locations i, j and k satisfy the inequality $t(i,k) \leqslant t(i,j) + t(j,k)$.*

Proof. We first prove that the minimum travel times between two locations are in general asymmetric. To simplify the proof, we consider the case when the currents

$\vec{v}_c$ are spatially invariant. Thus, the optimal navigation angle for a vehicle to travel from i to j is constant according to Lemma 2.2. In other words, the direction of the net velocity for the vehicle to travel from i to j is directly towards j and the magnitude of the net velocity is constant, and vice versa. Consequently, the optimal travel times $t(i,j)$ and $t(j,i)$ are asymmetric as long as the magnitudes of the two net speeds are different since the travel distances are both the Euclidean distance between i and j. The magnitudes of the net speeds for a vehicle to travel from i to j and from j to i are the same only when $\vec{v}_c$ is perpendicular to the vector pointing from i to j. Thus, the first half of the statement is proved. (Using the path planning method, we give an example in Fig. 2.1 to show the property of the asymmetric minimum travel times between two locations.)

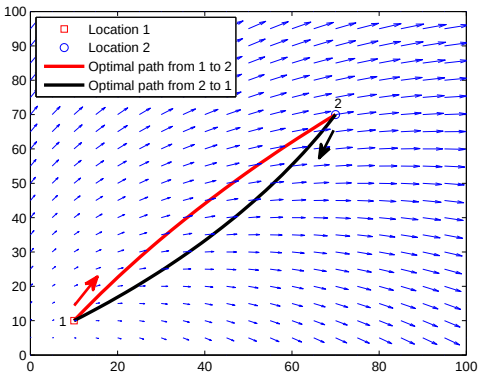

Figure 2.1: Optimal path planning for a vehicle with $v = 1$ to travel between two locations in the drift field $\vec{v}_c = 10^{-2}[0.3x + 0.2y, -0.2x + 0.3y]^T$ where the minimum travel times are $t(1,2) = 73.0058$s and $t(2,1) = 103.3586$s.

Under the optimal navigation law (2.6), a vehicle takes the minimum travel time $t(i,k)$ to travel from i to k. It is obvious that only if j is located on the optimal path from i to k, one has $t(i,k) = t(i,j) + t(j,k)$. On the other hand, $t(i,k)$ would not be the minimum travel time should $t(i,k) > t(i,j) + t(j,k)$, since the navigation law shown in Lemma 2.2 is time optimal. The proof is complete.

□

2.4 Task assignment algorithms

2.4.1 Target clustering strategies

In this subsection, three strategies are presented to cluster the target locations to the vehicles based on the optimal travel time matrix $C = (t(i,j))_{i \in \mathcal{R} \cup \mathcal{T}, j \in \mathcal{R} \cup \mathcal{T}}$

obtained from the path planning algorithm.

Voronoi clustering

Inspired by the coverage control study where each vehicle can reach any point of its partitioned area with the shortest travel time among all the vehicles [85], we first propose the Voronoi clustering strategy assigning each target k to the vehicle $j^\star$ such that

$$j^\star = \underset{j\in\mathcal{R}}{\operatorname{argmin}}\, t(p_j, k), \tag{2.11}$$

where p_j is the index of vehicle j's initial position and $t(p_j, k)$ is the minimum travel time for vehicle j to visit target k from p_j. In the case when a target location is on one of the boundaries of the Voronoi areas, it is randomly clustered to one of the vehicles whose Voronoi areas share the boundary.

Extended Voronoi clustering

In the task assignment problem, the vehicles need to visit *all* the target locations which is different from the coverage control problem [85] where a vehicle in essence only visits one target (although its location is unknown beforehand). In other words, Voronoi clustering might lead to assignment unfairness to the target locations. Thus, we extend the Voronoi clustering strategy by assigning each target according to the locations of those targets already assigned and the locations of all the vehicles.

Let $\mathcal{T}_j$ contain the indices of those targets that have already been assigned to vehicle j, which is initialized as $\{p_j\}$. The target set $\mathcal{T}^u$ is used to contain the indices of those unclustered targets, which is initialized as $\mathcal{T}$. Then, the first target $k^\star$ in $\mathcal{T}^u$ to be clustered and its assigned vehicle $j^\star$ are determined by

$$(j^\star, k^\star) = \underset{i\in\mathcal{T}_j, j\in\mathcal{R}, k\in\mathcal{T}^u}{\operatorname{argmin}} t(i, k), \tag{2.12}$$

where the targets already assigned to the vehicles affect the clustering of the remaining targets. After clustering target $k^\star$, $\mathcal{T}^u$ is updated to

$$\mathcal{T}^u = \mathcal{T}^u \setminus \{k^\star\}, \tag{2.13}$$

while the targets assigned to vehicle $j^\star$ are updated to

$$\mathcal{T}_{j^\star} = \mathcal{T}_{j^\star} \cup \{k^\star\}. \tag{2.14}$$

Marginal-cost-based clustering

In this subsection, a marginal-cost-based clustering strategy is designed which determines the visiting sequence of a target during its clustering process.

Let o_j be the route containing the ordered targets already assigned to j, which is initialized as $\{p_j\}$. Then, the first target $k^\star$ in $\mathcal{T}^u$ to be clustered, its assigned vehicle $j^\star$ and the inserting position $q^\star$ are

$$(k^\star, j^\star, q^\star) = \underset{k\in\mathcal{T}^u, j\in\mathcal{R}, 1<q\leqslant|o_j|+1}{\operatorname{argmin}} \{t(o_j \oplus_q k) - t(o_j)\}, \tag{2.15}$$

where the operation $o_j \oplus_q k$ inserts target k at the qth position of o_j and $|o_j|$ is the length of o_j. Target k is inserted to the end of o_j if $q = |o_j| + 1$, and $t(o_j)$ denotes the total travel time for vehicle j to visit all the targets in o_j.

2.4.2 Target-visiting metrics

For targets clustered by the Voronoi clustering and extended Voronoi clustering in §2.4.1, their visiting sequence is not determined. Putting the target locations assigned to each vehicle into a sequence to minimize the vehicle's travel time is in fact the traveling salesman problem (TSP) [3]. In this subsection, we design two target-visiting metrics: the nearest inserting principle and smallest marginal cost principle.

Nearest inserting principle

The first metric is the nearest inserting principle where vehicle j always inserts an unordered target location in $\mathcal{T}_j$ with the smallest travel time into the end of its route. Let o_j be the route containing the ordered targets already inserted, which is initialized as $\{p_j\}$. The target set $\mathcal{T}_j^u$ is used to contain the targets in $\mathcal{T}_j$ that have not been inserted into o_j. Then, the first target in $\mathcal{T}_j^u$ to be inserted for vehicle j is

$$k^\star = \underset{i=o_j(|o_j|), k\in\mathcal{T}_j^u}{\operatorname{argmin}} t(i,k), \tag{2.16}$$

Then, $\mathcal{T}_j^u$ and o_j are updated as

$$\mathcal{T}_j^u = \mathcal{T}_j^u \setminus \{k^\star\}, \quad o_j = o_j \oplus_{|o_j|+1} k^\star. \tag{2.17}$$

The inserting procedure continues until all the targets in $\mathcal{T}_j$ are inserted into vehicle j's target list o_j.

Algorithm 1 The Extended Voronoi clustering for achieving a min-cost generalized arborescence (MCGA) of a directed graph.

Input: Locations of targets in $\mathcal{T}$ and vehicles in $\mathcal{R}$, the travel time matrix C for digraph $\mathcal{G}$.
Output: An MCGA of $\mathcal{G}$.

```
Initialize MCGA ← R.
while T ≠ ∅ do
    (j*, p*) ← argmin_{(j,p)∈MCGA×T} t(j, p).
    Add p* in MCGA and connect it with j* using an edge with weight t(j*, p*).
    T ← T \ {p*}.
end while
```

Smallest marginal cost principle

The other one is the smallest marginal cost principle, which determines the first target $k^\star$ in $\mathcal{T}_j^u$ to be inserted and its visiting sequence $q^\star$ for each vehicle j by

$$(k^\star, q^\star) = \underset{1<q\leqslant|o_j|+1, k\in\mathcal{T}_j^u}{\operatorname{argmin}} \{t(o_j \oplus_q k) - t(o_j)\}. \tag{2.18}$$

Then, $\mathcal{T}_j^u$ and o_j are updated by

$$\mathcal{T}_j^u = \mathcal{T}_j^u \setminus \{k^\star\}, \quad o_j = o_j \oplus_{q^\star} k^\star. \tag{2.19}$$

The inserting procedure continues until $\mathcal{T}_j^u$ is empty.

2.4.3 Correctness of the proposed strategies

Let $\mathcal{G}$ be a digraph whose vertices contain all the vehicles' initial positions and the target locations. The weight for a directed edge is the minimum time for a vehicle to travel from the starting vertex to the ending vertex if at least one vertex represents a target location, and is otherwise zero. Compared with the Prim algorithm used to find a minimum spanning tree for a undirected graph [62], we use the extended Voronoi clustering strategy proposed in §2.4.1 to obtain a min-cost generalized arborescence (MCGA) for the digraph $\mathcal{G}$ where the sum of all the edge weights of the arborescence is minimum. The procedure to achieve an MCGA is shown in Algorithm 1. Let f_a be the sum of all the edge weights of an MCGA of $\mathcal{G}$, and f_o be the optimal objective value in (2.2). Then, we first investigate some property of the optimal solution to the problem with an asymmetric travel cost matrix.

Lemma 2.4. *It holds that* $f_a \leqslant f_o$.

Proof. We first prove the statement when m, the number of all the vehicles, is one. In this case, only one vehicle needs to visit all the target locations, which

is a variant of the TSP [36]. An optimal route for the vehicle leaving from its initial position to visit all the targets is in fact an arborescence of $\mathcal{G}$ according to Definition 1. As f_a is the cost of the min-cost arborescence, $f_a \leqslant f_o$.

When $m > 1$, from the definition of the generalized arborescence in Definition 2, the optimal solution of the problem is also a generalized arborescence of $\mathcal{G}$, in which both the outdegree and indegree of each vertex is at most one. As f_a is the sum of all the edge weights of the min-cost generalized arborescence, the proof is complete.

□

Removing the zero cost edges from the MCGA to get an arborescence for each vehicle and duplicating each directed edge of the arborescence but with the opposite direction, we can construct a Eulerian graph [10] for each vehicle (this is inspired by the multi-vehicle algorithm [67]). Let f_{da} be the sum of all the edge weights of the arborescences after duplicating their directed edges.

Lemma 2.5. *The optimal total travel time f_o is upper bounded by $f_o \leqslant f_{da}$, where $f_{da} = 2f_a$ if the travel cost matrix is symmetric.*

Proof. For the first statement, similar to the multi-vehicle algorithm operating on undirected graphs [67], we can obtain a TSP tour for each vehicle based on the corresponding Eulerian graph. As the directed edges satisfy the inequality in Lemma 2.3, the total travel time of each vehicle is at most the sum of all the edge weights of the duplicated arborescence for each vehicle. Thus, the total travel time of all the vehicles is not greater than the sum of all the edge weights of the duplicated generalized arborescence. As the total travel time of each feasible solution is an upper bound for the optimal solution, the first statement is proved.

When the travel cost matrix is symmetric, the minimum travel times between any two vertices in $\mathcal{G}$ are the same. Thus, $f_{da} = 2f_a$ as f_{da} is the sum of all the edge weights of the duplicated generalized arborescence.

□

Using the extended Voronoi clustering strategy in Algorithm 1, we obtain a min-cost arborescence for each vehicle. Then, we can utilize the target-inserting metrics in §2.4.2 to put the targets on each arborescence into sequence. Integrating the extended Voronoi clustering strategy with the smallest marginal cost principle, we obtain a task assignment algorithm, called EVM for simplification. Let f_{EVM} be the vehicles' total travel time of the solution resulting from EVM.

Theorem 2.6. *The* EVM *guarantees that $f_{\text{EVM}}/f_o \leqslant f_{da}/f_a$.*

Proof. The proof is conducted by induction. The solution resulting from EVM has the same target assignment compared with that of the duplicated min-cost generalized arborescence as they use the same target clustering strategy (2.12). Let

f_{da}^j be the sum of all the edge weights of the duplicated arborescence for vehicle j. Then, $f_{da} = \sum_{j\in\mathcal{R}} f_{da}^j$. The first target $k^\star$ to be inserted in o_j is determined by (2.18) for EVM. It is straightforward to see that the first target inserted in o_j is the same as the first target inserted in the min-cost arborescence for vehicle j according to line 3 of Algorithm 1. Thus, $f_{\text{EVM}}^{j1} \leqslant f_{da}^{j1}$ as $f_{\text{EVM}}^{j1} = f_{da}^{j1}$, where the superscripts 1 and j are associated with the total travel time for vehicle j to visit the first target inserted in o_j.

Now suppose the first $|\mathcal{T}_j| - 1$ targets inserted in o_j and those inserted in the arborescence for vehicle j are the same and $f_{\text{EVM}}^{j|\mathcal{T}_j|-1} \leqslant f_{da}^{j|\mathcal{T}_j|-1}$, where $\mathcal{T}_j$ contains all the targets in the end assigned to vehicle j. As the inequality specified in Lemma 2.3 holds for the optimal travel times between the vertices in $\mathcal{G}$ and according to (2.18), for EVM the marginal travel time incurred by inserting the last target k into o_j is

$$\begin{aligned}
\delta f_{\text{EVM}}^j &= \min_{1<q\leqslant|o_j|+1}\{t(o_j \oplus_q k) - t(o_j)\} \\
&= \min\{\min_{q\leqslant|o_j|-1}(t(o_j^q,k) + t(k,o_j^{q+1}) - t(o_j^q,o_j^{q+1})), \\
&\quad t(p_j,k) + t(k,o_j^1) - t(p_j,o_j^1), t(o_j^{|o_j|},k)\} \\
&\leqslant \min_{q\leqslant|o_j|-1}\{t(o_j^q,k) + t(k,o_j^q), t(k,o_j^{q+1}) + t(o_j^{q+1},k)\}, \qquad (2.20)
\end{aligned}$$

where p_j is the index of vehicle j's initial position and o_j^q is the qth target on o_j. On the other hand, considering the travel time cost on duplicating the edge of the min-cost arborescence, the minimum travel time incurred by inserting the last target k into the arborescence for vehicle j is

$$\delta f_{da}^j = t(k,o_j^{q^\star}) + \min_{q\leqslant|o_j|} t(o_j^q,k), \qquad (2.21)$$

where $q^\star = \text{argmin}_{q\leqslant|o_j|}\, t(o_j^q,k)$. It then follows that $\delta f_{\text{EVM}}^j \leqslant \delta f_{da}^j$.

Combining (2.20), (2.21) and $f_{\text{EVM}}^{j|\mathcal{T}_j|-1} \leqslant f_{da}^{j|\mathcal{T}_j|-1}$, we get

$$\begin{aligned}
f_{\text{EVM}}^{j|\mathcal{T}_j|} &= f_{\text{EVM}}^{j|\mathcal{T}_j|-1} + \delta f_{\text{EVM}}^j \\
&\leqslant f_{da}^{j|\mathcal{T}_j|-1} + \delta f_{da}^j. \qquad (2.22)
\end{aligned}$$

As $f_{da}^j = f_{da}^{j|\mathcal{T}_j|-1} + \delta f_{da}^j$, it holds that $f_{\text{EVM}}^{j|\mathcal{T}_j|} \leqslant f_{da}^{j|\mathcal{T}_j|}$ for each vehicle j. Thus, $\sum_{j\in\mathcal{R}} f_{\text{EVM}}^j \leqslant \sum_{j\in\mathcal{R}} f_{da}^j$, which proves $f_{\text{EVM}} \leqslant f_{da}$. Combining with $f_a \leqslant f_o$ in view of Lemma 2.4 and $f_{\text{EVM}} \leqslant f_{da}$, we have $f_{\text{EVM}}/f_o \leqslant f_{da}/f_a$.

□

Theorem 2.6 gives an upper bound of the worst case performance of EVM

compared with an optimal solution with the asymmetric travel cost matrix, which extends the upper bound result in [43] for the problem with the symmetric travel cost matrix. Furthermore, based on Lemma 2.5, the upper bound f_{da}/f_a is 2 if the travel cost matrix is symmetric which is the same as in [43]. We now investigate the property of the marginal-cost-based clustering strategy (MC) in §2.4.1 which directly puts the target locations assigned to each vehicle into the sequence during their assignment. Let f_{MC} be the vehicles' total travel time of a solution resulting from MC.

Theorem 2.7. *The task assignment algorithm* MC *guarantees that the total travel time* $f_{\text{MC}} \leqslant f_{\text{EVM}}$.

Proof. The proof is carried out by induction. The algorithm MC assigns the target locations according to (2.15), while EVM is based on (2.12). One can check that the first targets chosen by the two algorithms are the same. Thus, $f^1_{\text{MC}} \leqslant f^1_{\text{EVM}}$, where the superscript 1 means the vehicles' total travel time after assigning the first target.

Now suppose the first $|\mathcal{T}|-1$ targets assigned by the two algorithms are the same and $f^{|\mathcal{T}|-1}_{\text{MC}} \leqslant f^{|\mathcal{T}|-1}_{\text{EVM}}$. From the inequality in Lemma 2.3 and according to (2.15), for MC the marginal travel time incurred by inserting the last target k is

$$\begin{aligned} \delta f_{\text{MC}} &= \min_{j\in\mathcal{R}, 1<q\leqslant|o_j|+1} \{t(o_j \oplus_q k) - t(o_j)\} \\ &\leqslant \min_{1<q\leqslant|o_{j^\star}|+1} \{t(o_{j^\star} \oplus_q k) - t(o_{j^\star})\}, \end{aligned} \tag{2.23}$$

where $j^\star = \operatorname{argmin}_{i\in o_j, j\in\mathcal{R}} t(i,k)$ is determined by (2.12) and o_j contains those targets already assigned to vehicle j. On the other hand, EVM assigns the last target k as

$$\delta f_{\text{EVM}} = \min_{1<q\leqslant|o_{j^\star}|+1} \{t(o_{j^\star} \oplus_q k) - t(o_{j^\star})\}, \tag{2.24}$$

where $j^\star = \operatorname{argmin}_{i\in o_j, j\in\mathcal{R}} t(i,k)$. Thus, $\delta f_{\text{MC}} \leqslant \delta f_{\text{EVM}}$.

Combining (2.23), (2.24) and $f^{|\mathcal{T}|-1}_{\text{MC}} \leqslant f^{|\mathcal{T}|-1}_{\text{EVM}}$, we get

$$\begin{aligned} f_{\text{MC}} &= f^{|\mathcal{T}|-1}_{\text{MC}} + \delta f_{\text{MC}} \\ &\leqslant f^{|\mathcal{T}|-1}_{\text{EVM}} + \delta f_{\text{EVM}}. \end{aligned} \tag{2.25}$$

As $f_{\text{EVM}} = f^{|\mathcal{T}|-1}_{\text{EVM}} + \delta f_{\text{EVM}}$, it holds that $f_{\text{MC}} \leqslant f_{\text{EVM}}$.

□

The proposed clustering-based algorithms can be applied to the task assignment problem for vehicles in time-varying drift fields by assigning the target loca-

tions to the vehicles based on a time-varying travel cost matrix which needs to get updated as the drift field changes. This obviously will affect the visiting sequence of target locations, and as a result, the performance of the solution constructed by the clustering-based algorithms at any given time can only be guaranteed for a limited time thereafter.

Now we have presented all the theoretical results of this chapter, in the following section, we carry out simulation studies.

2.5 Simulations

One can obtain four task assignment algorithms after integrating the target clustering strategies with the target-inserting metrics: integrating the Voronoi clustering strategy with the nearest principle (VN); integrating the Voronoi clustering strategy with the smallest marginal cost principle (VM); integrating the extended Voronoi clustering strategy with the nearest principle (EVN); and integrating the extended Voronoi clustering strategy with the smallest marginal cost principle (EVM). As the marginal-cost-based clustering strategy (MC) directly determines the targets' visiting sequence during their assignment, it is already a task assignment algorithm. Integrating with the proposed path planning method, the existing task assignment algorithms can be used to solve the task assignment problem. The proposed clustering-based algorithms are compared with a GA which is a popular heuristic algorithm for VRP [46]. The GA encodes each target as a numbered gene and inserts $m-1$ marker genes into the target genes. Then, each chromosome represents a candidate solution to the task assignment problem. The GA employs the widely used tournament selection because of its efficiency and simplicity, which preserves gene diversity while guaranteeing all individuals might be selected [2].

Monte Carlo simulations are carried out to test the proposed algorithms, where all the experiments have been performed on an Intel Core $i5-4590$ CPU 3.30 GHz with 8 GB RAM, with algorithms compiled by Matlab under Windows 7. The solution quality of each algorithm is quantified by

$$q \quad = \quad \frac{f}{f_a}, \tag{2.26}$$

where f is the objective value in (2.2) and f_a is the sum of all the edge weights of an MCGA of the target-vehicle digraph $\mathcal{G}$. Since $f_a \leqslant f_o$, from Lemma 2.4 where f_o is the vehicles' total travel time of an optimal solution, a value of the ratio q closer to 1 means a better performance of the solution.

The algorithms are tested on the task assignment problem for multiple vehicles with $v = 1$ in a square drift field with edge length 10^3m and $\vec{v}_c = 10^{-3}[0.3x + 0.2y, -0.2x + 0.3y]^T$. The number of chromosomes in the GA is empirically set to

Table 2.1: The average solution quality q of the algorithms (A) on 400 scenarios for the task assignment problem under different instances (I) where $n50m10$ means 10 vehicles need to visit 50 target locations.

I \ A	VN	VM	EVN	EVM	MC	GA
$n50m10$	1.8641	1.5099	1.6811	1.3222	1.1581	1.4626
$n100m10$	2.0078	1.5877	1.7956	1.3725	1.2077	1.7987
$n110m10$	2.0090	1.5770	1.7955	1.3730	1.2159	1.8968
$n120m10$	2.0180	1.5888	1.8059	1.3792	1.2264	1.9993
$n120m12$	2.0333	1.6067	1.7750	1.3662	1.2076	2.0869
$n120m14$	2.0481	1.6188	1.7499	1.3575	1.1918	2.1669
$n120m16$	2.0570	1.6318	1.7293	1.3468	1.1774	2.2309
$n120m18$	2.0607	1.6399	1.7127	1.3338	1.1660	2.2963
$n120m20$	2.0592	1.6418	1.7003	1.3276	1.1562	2.3777

be 120, and the crossover rate and mutation rate for the GA are 0.9 and 0.1 respectively. The GA terminates at the maximal iteration number 350. Several instances $n50m10$, $n100m10$, $n110m10$, $n120m10$, $n120m12$, $n120m14$, $n120m16$, $n120m18$ and $n120m20$ are generated where $n50m10$ means 10 vehicles need to visit 50 target locations. For each instance, 400 scenarios of the initial positions of the targets and vehicles are randomly generated. The average q of the algorithms on each instance is shown in Table 2.1, and the corresponding average computation time for each algorithm is listed in Table 2.2. Firstly, Table 2.1 shows VM performs better than VN, and EVM performs better than EVN. The four algorithms first cluster the target locations to the vehicles, and then employ the nearest inserting principle or the smallest marginal cost principle to put the target locations assigned to each vehicle into sequence. In other words, the target locations have the same assignment for VM and VN, and for EVM and EVN. Thus, the better performances of VM over VN and EVM over EVN imply that the smallest marginal cost principle is more effective than the nearest principle to put the target locations into sequence. The reason lies partly in the fact that for each vehicle, the smallest marginal cost principle leads to the ordering of the target locations after computing the incurred travel cost at all possible positions on the vehicle's route; in contrast, the nearest principle is myopic in the sense that it inserts the target location with the minimal incurred cost at the end of the vehicle's route.

Secondly, Table 2.1 shows EVM performs better than VM, and EVN performs better than VN, which reflects the advantage of the extended Voronoi clustering strategy over the Voronoi clustering strategy. The reason is that the former strategy clusters a target using both the vehicles' initial locations and the clustered targets' locations, while the later only uses the vehicles' initial locations. Finally,

Table 2.2: The corresponding average computation time (s) for the algorithms (A) to get the solution to the task assignment problem under different instances (I).

I \ A	VN	VM	EVN	EVM	MC	GA
$n50m10$	0.0279	0.0281	0.0418	0.0423	0.0326	86.7761
$n100m10$	0.0500	0.0574	0.0984	0.1410	0.0590	140.6570
$n110m10$	0.0519	0.1130	0.0988	0.2904	0.0722	150.0892
$n120m10$	0.2381	0.2424	0.3665	0.4288	0.2493	163.2563
$n120m12$	0.2415	0.2440	0.4363	0.4434	0.2593	172.2250
$n120m14$	0.2480	0.2483	0.4453	0.4472	0.2616	176.7572
$n120m16$	0.2510	0.2541	0.4537	0.4566	0.2821	183.0387
$n120m18$	0.2563	0.2571	0.4655	0.4685	0.2913	197.1980
$n120m20$	0.2585	0.2597	0.4733	0.4744	0.3048	226.9434

MC performs better than EVM as shown in Table 2.1, which verifies Theorem 2.7. The reason is that MC considers the overall incurred travel cost for all the vehicles when clustering a target, which makes use of the complete geographic information on the vehicles' locations and the clustered targets' locations. In addition, the q of VM, EVN, EVM and MC shown in Table 2.1 is stably below twice of the optimal for each instance, which displays the efficient and robust performances of the algorithms. However, the GA's performance deteriorates as the size of the problem increases as shown in the last column of Table 2.1. Finally, in Table 2.2 the mean computation time of the proposed algorithms does not increase too much compared with the GA, which shows the proposed algorithms can scale with the problem size. The reason is that the clustering-based algorithms are heuristic in the sense that they assign the targets based on the geographic locations of the vehicles as well as the locations of those targets already clustered. The small computation time implies in particular the efficiency of the proposed algorithms when the solution's quality is improved by location-based clustering.

To further evaluate the solution quality q for the 400 scenarios of each instance, the Wilcoxon signed-rank test [39] is carried out in a two-tail test with the 5% significance level for each pair of the algorithms. It is clear that the q of 400 scenarios on each instance differ significantly between the proposed algorithms (q from left to right corresponds to MC $\rightarrow$ EVM $\rightarrow$ VM $\rightarrow$ EVN $\rightarrow$ VN). This implies the algorithms have an increasingly better performance as VN $-$ EVN $-$ VM $-$ EVM $-$ MC. The Wilcoxon signed-rank test shows that the performance of the GA deteriorates as the problem size increases, which is better than VM and worse than EVM for $n50m10$, better than VN, worse than VM and the same as EVN for $n100m10$, better than VN and worse than EVN for $n110m10$, worse than EVN and the same as VN for $n120m10$, and worse than VN for the remaining instances.

The results of the Wilcoxon signed-rank test are consistent with what is shown in Table 2.1.

2.6 Conclusion

In this chapter, we have investigated the task assignment problem in which multiple dispersed vehicles need to efficiently visit a set of target locations in a time-invariant drift field. A path planning method has been first designed which enables the vehicles to move between two prescribed locations in a drift field with minimal time. The travel cost matrix resulting from the path planning method provides the route information for the target location assignment. In addition, several target clustering strategies and two target inserting principles have been proposed. The target clustering strategies assign the target locations to the vehicles based on the travel cost matrix while the target inserting principles put the target locations assigned to each vehicle into sequence. Integrating the clustering strategies and the target inserting principles, we have obtained several algorithms which can efficiently solve the target visiting problem. The chapter's practical contributions are threefold: First, the deduced optimal navigation law provides a guiding principle for navigating vehicles in the presence of winds or currents. Second, the definition of the generalized arborescence helps to clarify for practitioners the space of feasible solutions to the task assignment problem. Third, the four clustering-based strategies offer a set of tools for different scenarios in logistic applications.

Chapter 3

Multi-vehicle task assignment in a time-invariant drift field with obstacles

THIS chapter studies the task assignment problem in which a fleet of dispersed vehicles needs to visit a set of target locations in a time-invariant drift field with obstacles while trying to minimize the vehicles' total travel time. The vehicles have different capabilities, and each kind of vehicles needs to visit a certain type of target locations; each target location might have the demand to be visited more than once by different kinds of vehicles. We prove that the problem is in fact NP-hard. To find approximate solutions for such a challenging problem, we first design a path planning algorithm to minimize the time for a single vehicle to travel between two given locations through the drift field while avoiding obstacles. The path planning algorithm provides the travel cost matrix for the target assignment, and generates routes once the target locations are assigned to the vehicles. Then, we propose an auction-based distributed task assignment algorithm to assign the target locations to the vehicles using only local communication. The task assignment algorithm guarantees that all the demands of the targets will be satisfied within a total travel time that is at most twice of the optimal one when the cost matrix is symmetric. Finally, numerical simulations show that the algorithm can lead to solutions close to the optimal.

3.1 Introduction

There are some research works considering the path planning for the employed vehicles to avoid obstacles [32, 33, 38, 64]. In [32], distance functions were studied for robot path planning in the presence of obstacles where the main principle is to express obstacle avoidance in term of distances between the obstacles and the part of the robot that may potentially interface into the obstacles. The path planning of multiple robots with kinodynamic constraints along specified paths was addressed in [64] by first solving the two-point boundary value problems on the minimum and maximum possible traversal times that satisfy the kinematics and dynamics constraints. Later on, a steering potential function was designed in which the robots' headings to the goal and obstacles, and the distance to the goal are used to compute a potential field to navigate the robots [38]. The angular ac-

celeration of a robot is controlled by the potential field to steer the robot towards its goal and away from obstacles. The time-optimal path planning for a Dubins car operating in its workspace while avoiding collision with obstacles was investigated in [33]. However, little attention has been paid to the path planning for vehicles in a drift field with obstacles, let alone coordinating multi-vehicles to compete for some tasks in a drift field with obstacles.

In the previous chapter, we have studied the multi-vehicle task assignment problem in a drift field without obstacles. In this chapter, we investigate the task assignment problem for which a set of target locations in a time-invariant drift field with obstacles needs to be visited by several dispersed vehicles as energy-efficiently as possible. The vehicles are heterogeneous in the sense that each kind of vehicles carrying one specified sensor needs to visit a certain type of targets, and some target locations might need to be visited by vehicles with different capabilities in order to e.g. measure ocean salinity, temperature and conductivity. To solve the problem, we design a path planning method and a distributed task assignment algorithm (DTAA). Our main contributions are as follows. Firstly, the path planning algorithm can generate the time-optimal path for a vehicle traveling between two prescribed locations in a time-invariant drift field with obstacles, which provides the cost matrix to be used later for the task assignment. Secondly, we prove that the task assignment problem is NP-hard, and achieve a tight lower bound on the optimal solution by using tools from graph theory. The NP-hardness property of the problem means un-affordable long computation time is needed to get the optimal solution when the numbers of vehicles and tasks grow. Thus, the tight lower bound is critical to measure the performance of a task assignment algorithm. Lastly, the DTAA is fully distributed where every vehicle cooperates with the communication-connected local neighbors while guaranteeing that all the visiting demands of the targets are satisfied within the total travel time, which, when the cost matrix is symmetric, is at most twice of the optimal.

The rest of this chapter is organized as follows. In Section 3.2, the formulation for the heterogeneous task assignment problem is given. Section 3.3 presents the path planning algorithm which generates the cost matrix for the task assignment, and in Section 3.4 the task assignment problem is analyzed. Section 3.5 presents the DTAA and Section 3.6 analyzes the performance of DTAA. We present the simulation results in Section 3.7 and conclude the chapter in Section 3.8.

3.2 Problem formulation

Consider m dispersed aerial/marine vehicles that need to visit a set of n target locations in a planar time-invariant drift field with obstacles while trying to minimize the vehicles' total travel time. Each vehicle has one specified capability to

visit a certain kind of targets and several vehicles might have the same capability. Some targets need to be visited more than once by vehicles with different capabilities but at most once by vehicles with the same capability.

3.2.1 Formulation as an optimization problem

We use the vector $\vec{v}_c = [v_{cx}, v_{cy}]^T$ with magnitude v_c to denote the drift velocity of the field with respect to some coordinate system fixed to the ground. Note that $\vec{v}_c$ varies with locations. The vehicles' velocity $\vec{v}$ is assumed with constant speed v relative to a static drift field [73]. We assume that the vehicles are free of turning ratio constraints since the dimension of the drift field is significantly larger than the vehicles. The kinematics of each vehicle are governed by (2.1).

Let $\mathcal{T} = \{1, \ldots, n\}$ be the labeling of the target locations, and $\mathcal{R} = \{1, \ldots, m\}$ be the labeling of the vehicles. We use $\mathcal{V}$ to denote the set of indices of all the vehicles' initial locations, namely $\mathcal{V} = \{n+1, \cdots, n+m\}$, where each vehicle has one capability $a_j \in \mathcal{A}$ with $|\mathcal{A}| \leqslant m$. Let $\mathcal{R}_q, q \in \{1, \cdots, |\mathcal{A}|\}$, be the vehicle set that contains the indices of those vehicles with the same capability where $\mathcal{R} = \cup_{q=1}^{|\mathcal{A}|} \mathcal{R}_q$ and $\mathcal{R}_q \cap \mathcal{R}_p = \varnothing$ for $\forall q \neq p$. The binary value y_{jq} is employed which is one if target j requires to be visited by one vehicle from $\mathcal{R}_q$ and otherwise zero. Let $C = (t(i,j))_{(m+n)\times(m+n)}, i, j \in \mathcal{T} \cup \mathcal{V}$, denote the cost matrix where $t(i,j)$ specifies the time for a vehicle to travel from i to j in the drift field. The time $t(i,j)$ to be minimized is derived from a properly designed path planning algorithm where $t(i,i) = 0$ for each i. Let σ_{ijk} be the path-planning mapping that maps the indices $i \in \mathcal{T} \cup \mathcal{V}$ and $j \in \mathcal{T}$ of the starting and ending locations of a vehicle $k \in \mathcal{R}$ to a binary value, which equals one if and only if it is planned that vehicle k directly travels from location i to j and otherwise zero. The minimization of the vehicles' total travel time for satisfying all the visiting demands of the target locations is to minimize

$$f = \sum_{i \in \mathcal{T} \cup \mathcal{V}, j \in \mathcal{T}, k \in \mathcal{R}} t(i,j)\sigma_{ijk}, \tag{3.1}$$

subject to

$$\sum_{i \in \mathcal{T} \cup \mathcal{V}, k \in \mathcal{R}_q} \sigma_{ijk} = y_{jq}, \qquad \forall\, j \in \mathcal{T}, \forall\, \mathcal{R}_q \subseteq \mathcal{R}; \tag{3.2}$$

$$\sum_{j \in \mathcal{T}, k \in \mathcal{R}_q} \sigma_{ijk} \leqslant 1, \qquad \forall\, i \in \mathcal{T} \cup \mathcal{V}, \forall\, \mathcal{R}_q \subseteq \mathcal{R}. \tag{3.3}$$

Constraint (3.2) ensures that the visiting demand of each target is satisfied; (3.3) means that each target location and each vehicle's initial location is departed at most once for each kind of vehicles $\mathcal{R}_q \subseteq \mathcal{R}$.

After formulating the task assignment problem as a constrained minimization problem, we present in the following section the path planning algorithm that is critical for solving the overall optimization problem.

3.3 Path Planning Algorithm

In this section, we conduct a path planning algorithm based on grid modeling for a vehicle to efficiently travel through two prescribed locations in a drift field while avoiding any obstacle. The operation environment is first evenly divided into grids of small square cells where the cells occupied by obstacles are marked as unfeasible areas. The square cells are labeled by ascending integers from $1, 2, 3, \cdots$. Potential moving directions for a vehicle located in a cell are shown in Fig. 3.1, where the angle θ_{net} of the net velocity for a vehicle to move to the neighboring cell $i \in \{1, \cdots, 8\}$ is $(i-1)\pi/4$. Let the edge length of each square cell be l. Then, the corresponding distance is l for the vehicle to move to its neighboring cells with odd number labels as shown in Fig. 3.1, while the distance is $\sqrt{2}l$ to move to those with even number labels.

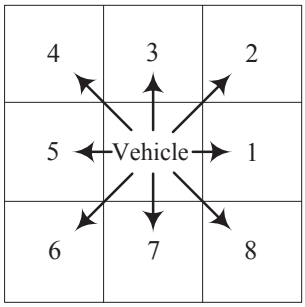

Figure 3.1: The possible moving directions for a vehicle to travel to its neighbor cells.

The feasibility of moving into a neighbor cell depends not only on whether obstacles occupy the cell, but also on the strength and orientation of currents at the vehicle's location. The currents within each square cell are assumed to be the same, which is natural when the operation area is discretized with high resolution. We obtained the magnitude of the optimal net velocity of an underwater vehicle in [80], and calculated the rate of change of the vehicle's optimal navigation angle in a drift field in (2.6) under the assumption that $v > v_c$. Here, we generalize the analysis. Let v_{net} be the magnitude of the net velocity $[\dot{x}, \dot{y}]^T$ of the vehicle with the motion dynamics of (2.1), and θ_c be the heading of the currents $\vec{v}_c$. Then, in

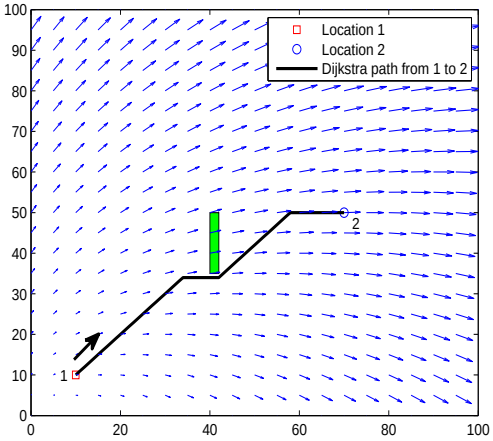

Figure 3.2: Optimal path planning for a vehicle with $v = 1$ to travel between two locations in the drift field $\vec{v}_c = 10^{-2}[0.3x + 0.2y, -0.2x + 0.3y]^T$ with an obstacle colored in green where the travel time resulting from the Dijkstra algorithm is $t(1, 2) = 66.5618$s.

view of (2.1), it holds that

$$v_{net}^2 - 2v_c v_{net} cos(\theta_{net} - \theta_c) - (v^2 - v_c^2) = 0. \tag{3.4}$$

According to the discriminant of roots of a quadratic equation, the feasible solution v_{net} to (3.4) exists when

$$|sin(\theta_{net} - \theta_c)| \leqslant v/v_c. \tag{3.5}$$

Based on (3.5) and the obstacle information, one can determine whether a vehicle can move to a certain neighbor cell. The time for the vehicle to travel from its location to a neighbor cell is the division of the travel distance by v_{net} which can be obtained by (14) in [80]; the time is infinite if the neighbor cell is impossible for the vehicle to visit due to obstacles or currents.

For each cell, we first use (3.4), (3.5) and the obstacle information to calculate the time for a vehicle to move from the cell to each neighbor cell. Then, a weighted digraph whose vertices contain the indices of all the cells is constructed, where the weight for a directed edge is the time for a vehicle to travel from the starting vertex to the ending vertex. For a given source vertex in the graph, Dijkstra's algorithm finds the shortest path between that node and every other node [55, P196-206]. Thus, once the weighted digraph containing the indices of all the cells

is constructed, Dijkstra's algorithm can find the path with the minimum travel time between any two given locations in $\mathcal{T} \cup \mathcal{V}$, which leads to the travel cost matrix C for (3.1). We show the performance of the path planning algorithm for a vehicle to travel through two given locations in a drift filed with an obstacle in Fig. 3.2, where the vehicle avoids the obstacle effectively while adjusting its route to the currents.

Remark 3.1. *The cost matrix C is in general asymmetric, namely $t(i,j) \neq t(j,i)$ for $\forall i \neq j$, and satisfies $t(i,k) \leqslant t(i,j) + t(j,k)$ as Dijkstra's algorithm finds the path with the minimum travel time between any two locations i and k.*

3.4 Problem analysis

3.4.1 Proof of NP-Hardness

Consider a digraph $\mathcal{G} = (V, E, C)$ that consists of a set of vertices $V = \mathcal{T} \cup \mathcal{V}$, a set of directed edges E, and the travel cost matrix C that contains the weight of each edge in E. We first introduce the multiple traveling salesman problem (MTSP) which is to determine a set of routes for m salesmen who all start from and return to a home city/depot to visit a set of target locations once for each target. It is a relaxation of the vehicle routing problem (VRP) without the capacity constraint [6]. In VRP, a fleet of vehicles initially located at one or several depots is required to optimally transport the products to a set of dispersed customers and then returns to the respective starting depots, and the VRP is NP-hard [75]. If the vehicles do not need to return to the starting depots, the VRP is called the open VRP which is also NP-hard [26]. When the number of the vehicles is one, the MTSP is reduced to the traveling salesman problem (TSP) and the open MTSP is then an open TSP. In graph theory, the open TSP determines a Hamiltonian path in an undirected or directed graph that connects in sequence each vertex exactly once, and the TSP involves determining a Hamiltonian cycle. Determining whether such paths and cycles exist in graphs is the Hamiltonian path problem, which is NP-complete [30, 199-200]. The requirement of returning to the starting city does not change the computational complexity of the problem. So the open MTSP (OMTSP) is NP-hard as well [28].

Proposition 3.2. *The task assignment problem (3.1) is NP-hard.*

Proof. We first prove the NP-hardness of (3.1) with homogeneous vehicles by showing that: (i) every instance of the OMTSP can be reduced to an instance of (3.1) with homogeneous vehicles in polynomial time and (ii) an optimal solution to (3.1) with homogeneous vehicles provides an optimal OMTSP solution.

Let $\mathcal{G}' = (V', E', C')$ with $|V'| = n + 1$ being an arbitrary input to the OMTSP, where n is the number of dispersed target locations. To prove (i), we give a

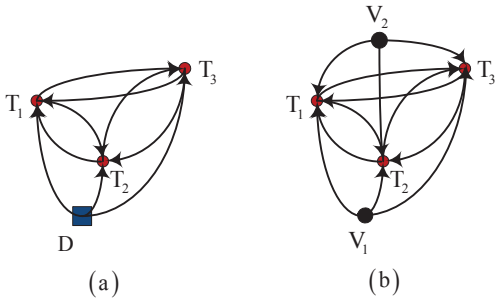

Figure 3.3: A transformation from the OMTSP on graph $\mathcal{G}'$ to the task assignment problem with homogeneous vehicles on graph $\mathcal{G}$ where (a) $\mathcal{G}' = (V', E', C')$ and (b) $\mathcal{G} = (V, E, C)$.

polynomial-time transformation of $\mathcal{G}'$ into an input of $\mathcal{G} = (V, E, C)$ to (3.1) with homogeneous vehicles, shown in Fig. 3.3 where $\{T_1, T_2, T_3\}$ is the target set, D is the depot, and $\{V_1, V_2\}$ is the vehicle set.

Let graph $\mathcal{G}$ first inherit all the target vertices of $\mathcal{G}'$. Then, the vertices representing the dispersed vehicles are added in $\mathcal{G}$ as V_1 and V_2. Now for each edge (D, T_i) in E' with the weight $t'(D, T_i)$, add a directed edge from each vehicle vertex V_j to the target T_i as (V_j, T_i) with the weight $t(V_j, T_i) = t'(D, T_i)$. Let $t(V_j, V_k) = 0, \forall j, k \in \mathcal{R}$, and the weights of other edges of $\mathcal{G}$ be the same as $\mathcal{G}'$. Thus, the transformation from $\mathcal{G}'$ to $\mathcal{G}$ is complete, which is the input to (3.1) with homogeneous vehicles.

To prove (ii), we show that an optimal solution to the task assignment problem with homogeneous vehicles corresponds to an optimal OMTSP solution. After the transformation of $\mathcal{G}'$ to $\mathcal{G}$, it is straightforward to see that an optimal solution to (3.1) with homogeneous vehicles is also optimal to the OMTSP based on the edge weights of the graph shown in Fig. 3.3 (a) and Fig. 3.3 (b). Thus, (3.1) with homogeneous vehicles is NP-hard.

When the vehicles are heterogeneous and some target locations need to be visited more than once by vehicles with different capabilities, the task assignment problem (3.1) can be divided into $|\mathcal{A}|$ sub-problems where subproblem $q \in \{1, \cdots, |\mathcal{A}|\}$ contains the target locations in $\mathcal{T}_q$ needed to be visited by the vehicles in $\mathcal{R}_q$. In view of the NP-hardness of each sub-problem with homogeneous vehicles, the task assignment problem is NP-hard. The proof is complete.

□

We add the following remark to emphasize why the problem we study is difficult in nature.

Remark 3.3. *A target location might belong to several subproblems when it needs to be visited more than once by vehicles with different capabilities.*

3.4.2 A lower bound on the optimal solution

It can be costly to solve (3.1) optimally due to the NP-hardness of the problem. As a consequence, it is natural to design heuristic algorithms to find sub-optimal solutions. Then, one issue arises on how to evaluate the level of optimality of a sub-optimal solution as the optimal is typically unknown. In this section, a lower bound on the minimum total travel time for the vehicles to satisfy all the visiting demands of the target locations while avoiding any obstacle is constructed. As some target locations might need to be visited more than once by vehicles with different capabilities, we first decouple the problem by dividing all the target locations into $|\mathcal{A}|$ subgroups $\mathcal{T}_q, q \in \{1, \cdots, |\mathcal{A}|\}$, where $\mathcal{T} = \cup_{q=1}^{|\mathcal{A}|}\mathcal{T}_q$ and the targets within the subgroup $\mathcal{T}_q$ need to be visited by the vehicles from vehicle set $\mathcal{R}_q$.

Let $\mathcal{G}_q$ be a digraph whose vertices contain the indices of the vehicles in $\mathcal{R}_q$ and the target locations in $\mathcal{T}_q, \forall q \in \mathcal{A}$. The weight for a directed edge is the minimum time for a vehicle to travel from the starting vertex to the ending vertex if at least one vertex represents a target location and otherwise zero. We extend the Prim algorithm [62], used to find a minimum spanning tree for an undirected graph, to obtain an MCGA for the digraph $\mathcal{G}_q$. The procedure to achieve an MCGA is shown in Algorithm 1. Let f_q^a be the sum of all the edge weights of an MCGA of $\mathcal{G}_q$, and f_q^o be the minimum total travel time for the vehicles in $\mathcal{R}_q$ to visit all the target locations in $\mathcal{T}_q$. We set $f^a = \sum_{q=1}^{|\mathcal{A}|} f_q^a$, and employ f^o as the minimum total travel time for (3.1). Then, we first present a lower bound on f^o.

Theorem 3.4. *The optimal total travel time has a lower bound $f^a \leqslant f^o$.*

Proof. We first prove that the optimal total travel time for the vehicles in $\mathcal{R}_q$ to visit all the targets in $\mathcal{T}_q$ is lower bounded by the sum of all the edge weights of an MCGA of $\mathcal{G}_q$, i.e., $f_q^a \leqslant f_q^o$. When the number of all the vehicles in $\mathcal{R}_q$ is one, the vehicle needs to visit all the target locations in $\mathcal{T}_q$ which is an open TSP problem where the vehicle does not return to its initial location. An optimal open TSP route for the vehicle to visit all the targets is in fact an arborescence of $\mathcal{G}_q$ according to Definition 1. As f_q^a is the sum of all the edge weights of a min-cost arborescence, the total travel time for the optimal open TSP route satisfies $f_q^a \leqslant f_q^o$.

When $m > 1$, from the definition of the generalized arborescence in Definition 2, the optimal solution of the problem is also a generalized arborescence of $\mathcal{G}_q$. As f_q^a is the sum of all the edge weights of an MCGA, $f_q^a \leqslant f_q^o$.

As $\mathcal{R} = \sum_{q=1}^{|\mathcal{A}|} \mathcal{R}_q$ where $\mathcal{R}_q \cap \mathcal{R}_p = \varnothing$ for $\forall q \neq p$, we get $f^o = \sum_{q=1}^{|\mathcal{A}|} f_q^o$. In view of $f_q^a \leqslant f_q^o$, it holds that $\sum_{q=1}^{|\mathcal{A}|} f_q^a \leqslant \sum_{q=1}^{|\mathcal{A}|} f_q^o$. Thus, the proof is complete. □

Having done the problem analysis, we construct a distributed task assignment algorithm in the next section.

3.5 Distributed task assignment algorithm

In this section, we present the distributed algorithm DTAA that relies on both an auction mechanism and a marginal-cost-based target ordering principle. We satisfy the visiting demands of the target locations by assigning the targets in each $\mathcal{T}_q, \forall q \in \mathcal{A}$, to the vehicles in $\mathcal{R}_q$ through using the following distributed auction mechanism and the target ordering principle.

3.5.1 Distributed auction mechanism

The first phase of the DTAA is the distributed auction mechanism which is used to assign the targets within each $\mathcal{T}_q$ to the vehicles in $\mathcal{R}_q$. The auction mechanism works under all connected topologies of the communication network among all the vehicles. Each vehicle j in $\mathcal{R}$ carries an information tuple $\mathcal{I}_j = \{j, p_j, a_j, c_j, \beta_j, o_j\}$, where j is the vehicle's unique identifier, p_j is the index of vehicle j's initial position, $a_j \in \mathcal{A}$ is j's capability, c_j is a $|\mathcal{T}|$-tuple where its rth component c_{jr} stores the minimal incurred time known by vehicle j for the vehicle kept in β_{jr} to visit target r in $\mathcal{T}_q$, and o_j keeps the target locations already assigned to vehicle j which is initialized to be $\{p_j\}$.

Let the target set $\mathcal{T}_q^u$ contain the indices of those unassigned targets in $\mathcal{T}_q$, which is initialized as $\mathcal{T}_q$. During the auction process, each vehicle j first updates its c_{jr} by bidding for every target $r \in \mathcal{T}_q^u$ itself and initializes the β_{jr} as

$$c_{jr} = \min_{k \in o_j} t(k, r), \qquad \beta_{jr} = j, \tag{3.6}$$

where c_{jr} is infinite if vehicle j does not have the capability to visit r. We enable all the vehicles to bid for each target in $\mathcal{T}_q^u$ instead of only using vehicles in $\mathcal{R}_q$ as the vehicles in $\mathcal{R}_q$ might not be communication-connected. The motivation is that through a certain number of information updates through communicating with those vehicles outside of $\mathcal{R}_q$, the vehicles in $\mathcal{R}_q$ know the minimum time incurred to visit each target in $\mathcal{T}_q^u$.

Then, for $\forall r \in \mathcal{T}_q^u$, c_{jr} and β_{jr} are updated by vehicle j through local communication with directly connected neighbor vehicles in $\mathcal{R}_N^j$ as

$$c_{jr} = c_{i^\star r}, \qquad \beta_{jr} = \beta_{i^\star r}, \tag{3.7}$$

where $i^\star = \operatorname{argmin}_{i \in \mathcal{R}_N^j} c_{ir}$. After at most $|\mathcal{R}|$ synchronized iterative communications, c_j and β_j reach consensus for every vehicle in $\mathcal{R}$. Then, the first target $r^\star$ in $\mathcal{T}_q^u$ to be assigned is

$$r^\star = \operatorname*{argmin}_{r \in \mathcal{T}_q^u} c_{jr}, \tag{3.8}$$

Algorithm 2 Distributed auction mechanism for vehicle $j \in \mathcal{R}$.

Input: The directly communication-connected vehicle set $\mathcal{R}_N^j$, cost matrix C, target set $\mathcal{T}_q$ and vehicle j' position index p_j.
Output: All the targets in $\mathcal{T}_q$ are assigned.

$\mathcal{T}_q^u \leftarrow \mathcal{T}_q$.
$o_j \leftarrow \{p_j\}$.
while $\mathcal{T}_q^u \neq \varnothing$ **do**
  for $\forall r \in \mathcal{T}_q^u$ **do**
    $c_{jr} \leftarrow \min_{k \in o_j} t(k, r)$,
    $\beta_{jr} \leftarrow j$.
  end for
  $s \leftarrow 0$.
  while $s < |\mathcal{R}|$ **do**
    Send c_j and β_j to every vehicle $i \in \mathcal{R}_N^j \setminus \{j\}$.
    Receive c_i and β_i from every vehicle $i \in \mathcal{R}_N^j \setminus \{j\}$.
    for $\forall r \in \mathcal{T}_q^u$ **do**
      $i^\star \leftarrow \operatorname{argmin}_{i \in \mathcal{R}_N^j} c_{ir}$,
      $c_{jr} \leftarrow c_{i^\star r}$,
      $\beta_{jr} \leftarrow \beta_{i^\star r}$.
    end for
    $s \leftarrow s + 1$.
  end while
  $r^\star \leftarrow \operatorname{argmin}_{r \in \mathcal{T}_q^u} c_{jr}$.
  if $j = \beta_{jr^\star}$ **then**
    $o_{\beta_{jr^\star}} \leftarrow o_{\beta_{jr^\star}} \cup \{r^\star\}$.
  end if
  $\mathcal{T}_q^u \leftarrow \mathcal{T}_q^u \setminus \{r^\star\}$.
end while

which is assigned to vehicle $\beta_{jr^\star}$. Then, the targets already assigned to vehicle $\beta_{jr^\star}$ and the unassigned target set $\mathcal{T}_q^u$ are updated by

$$o_{\beta_{jr^\star}} = o_{\beta_{jr^\star}} \cup \{r^\star\}, \qquad \mathcal{T}_q^u = \mathcal{T}_q^u \setminus \{r^\star\}. \tag{3.9}$$

The auction process continues until the unassigned target set $\mathcal{T}_q^u$ is empty. The distributed auction mechanism applied to each vehicle j for bidding the targets in $\mathcal{T}_q$ is shown in Algorithm 2.

3.5.2 Target locations' ordering principle

After the auction operation, all the target locations in $\mathcal{T}$ have been assigned where the target locations assigned to vehicle $j \in \mathcal{R}$ are kept in o_j. Putting the target locations assigned to each vehicle into a sequence to minimize the vehicle's total travel time is in fact the open TSP problem. Let λ_j, initialized as $\{p_j\}$, be the route with the ordered target locations for vehicle j, and o_j^u contain the unordered

targets in o_j where o_j^u is initialized as $o_j \setminus \{p_j\}$. In this subsection, we design the marginal-cost-based target ordering principle to construct λ_j, which determines the first target $r^\star$ in o_j^u to be inserted in λ_j and its visiting sequence $k^\star$ for vehicle j by

$$(r^\star, k^\star) = \underset{1<k\leqslant|\lambda_j|+1, r\in o_j^u}{\operatorname{argmin}} \{t(\lambda_j \oplus_k r) - t(\lambda_j)\}, \tag{3.10}$$

where the operation $\lambda_j \oplus_k r$ inserts target r at the kth position of λ_j. Target r is inserted to the end of λ_j if $k = |\lambda_j| + 1$, and $t(\lambda_j)$ denotes the total travel time for vehicle j to visit all the targets in λ_j. Then, route λ_j and the unordered target set o_j^u are updated to

$$\lambda_j = \lambda_j \oplus_{k^\star} r^\star, \qquad o_j^u = o_j^u \setminus \{r^\star\}. \tag{3.11}$$

The target inserting process continues until the unordered target set o_j^u is empty.

Now we discuss in more detail why our proposed algorithm DTAA works.

3.6 Performance analysis of DTAA

In the above section, the task assignment problem is decoupled by dividing all the target locations into $|\mathcal{A}|$ subgroups: the targets within each subgroup $\mathcal{T}_q$ need to be visited by the vehicles from vehicle set $\mathcal{R}_q$, where $\mathcal{R}_q \cap \mathcal{R}_p = \varnothing$ for $\forall q, p \in \mathcal{A}, q \neq p$. We now analyze the performance of the DTAA.

3.6.1 Convergence performance

Removing the zero cost edges connecting each two vehicle vertices after obtaining the MCGA of each $\mathcal{G}_q, \forall q \in \mathcal{A}$, we get an arborescence for each vehicle in $\mathcal{R}_q$. We now present some convergence property of the DTAA.

Theorem 3.5. *The targets in $\mathcal{T}_q, \forall q \in \mathcal{A}$, assigned to each vehicle in $\mathcal{R}_q$ by the DTAA converge to the targets in the vehicle's arborescence resulting from the MCGA of $\mathcal{G}_q$ after at most $|\mathcal{T}_q||\mathcal{R}|$ synchronized communication iterations.*

Proof. We first prove that the DTAA produces a stable assignment after at most $|\mathcal{T}_q||\mathcal{R}|$ synchronized iterations for $\mathcal{G}_q$. There are $|\mathcal{T}_q|$ targets to be assigned to $|\mathcal{R}_q|$ vehicles. The DTAA uses the distributed auction mechanism shown in Algorithm 2 to assign the targets in $\mathcal{T}_q$ to the vehicles in $\mathcal{R}_q$. After at most $|\mathcal{R}|$ iterations of synchronized communications as shown in lines 10 to 17 of Algorithm 2, all the vehicles in $\mathcal{R}$ reach consensus on the bid list c_j keeping the minimum travel cost for the vehicles to visit each unassigned target in $\mathcal{T}_q^u$ and on the β_j recording

the corresponding vehicles with the minimum travel time. Then, at line 19 of the algorithm, the target $r^\star$ with the smallest travel cost among the unassigned target set $\mathcal{T}_q^u$ is chosen. At line 20, we check whether j is the vehicle with the smallest travel cost to visit $r^\star$ and update the targets assigned to j according to line 21 if j wins $r^\star$. Afterwards, the auction process continues with the unassigned target set $\mathcal{T}_q^u$ being updated as line 23 of Algorithm 2. Thus, all the targets in $\mathcal{T}_q$ are assigned to the vehicles in $\mathcal{R}_q$ after at most $|\mathcal{T}_q||\mathcal{R}|$ synchronized communication iterations.

The proof for the convergence of the targets assigned to each vehicle guided by the DTAA to those of the targets in the vehicle's arborescence resulting from the MCGA is as follows. From the analysis just now, for the DTAA the target with the smallest travel cost among the unassigned targets in the current $\mathcal{T}_q^u$ is assigned to the vehicle with the smallest travel cost after at most $|\mathcal{R}|$ synchronized communications. That process is in fact the same as the target election operation in lines 3 and 4 of Algorithm 1. The arborescence for each vehicle in $\mathcal{R}_q$ is obtained by breaking the zero-weighted edges connecting arbitrary two vehicles in the MCGA of $\mathcal{G}_q$ obtained by Algorithm 1. Herein, the statement is proved.

□

3.6.2 Worst-case performance guarantee

Duplicating each directed edge of the arborescence for each assigned vehicle in $\mathcal{R}_q$ but with the opposite direction, we can construct a Eulerian graph for each vehicle (this is inspired by the multi-vehicle algorithm [67]). Let f_q^{da} be the sum of all the edge weights of all the arborescences for the vehicles in $\mathcal{R}_q$ after duplicating their directed edges where $f^{da} = \sum_{q=1}^{|\mathcal{A}|} f_q^{da}$.

Lemma 3.6. *It holds that* $1 \leqslant f^{da}/f^o$, *and* $f^{da}/f^o \leqslant 2$ *when the travel cost matrix is symmetric.*

Proof. We first prove that $1 \leqslant f_q^{da}/f_q^o$. To prove the statement, similar to the multi-vehicle algorithm operating on undirected graphs [67], we can obtain an open TSP route for each vehicle based on the corresponding Eulerian graph. As the directed edges satisfy the inequality in Remark 3.1, the total travel time of each vehicle in $\mathcal{R}_q$ is at most the sum of all the edge weights in the duplicated arborescence for the vehicle [67]. Thus, the total travel time of all the vehicles in $\mathcal{R}_q$ is not greater than the sum of all the edge weights of the duplicated generalized arborescences for the vehicles in $\mathcal{R}_q$. As the total travel time of each feasible solution is an upper bound for the optimal solution, we get $f_q^o \leqslant f_q^{da}$. As $f^{da} = \sum_{q=1}^{|\mathcal{A}|} f_q^{da}$ and $f^o = \sum_{q=1}^{|\mathcal{A}|} f_q^o$, it holds that $f^o \leqslant f^{da}$.

When the travel cost matrix of $\mathcal{G}$ is symmetric, $\mathcal{G}_q$ is symmetric. Thus, $f_q^{da} = 2f_q^a$ as f_q^{da} is the sum of all the edge weights of the duplicated generalized ar-

borescence. In view of $f_q^a \leqslant f_q^o$ in the proof of Theorem 3.4, it follows that $\sum_{q=1}^{|\mathcal{A}|} f_q^{da} \leqslant 2\sum_{q=1}^{|\mathcal{A}|} f_q^o$. The proof is complete.

□

After the targets in $\mathcal{T}_q$ being assigned to the vehicles in $\mathcal{R}_q$ for each $q \in \mathcal{A}$, the marginal-cost-based target ordering principle proposed in §3.5.2 is used to put the targets into sequence. These two steps lead to the DTAA. Let f_q^{DTAA} be the total travel time for the vehicles in $\mathcal{R}_q$ guided by the DTAA to visit all the targets in $\mathcal{T}_q$ where the vehicles' total travel time to satisfy all the visiting demands of the targets in $\mathcal{T}$ is f^{DTAA}. We are now able to present the worst performance guarantee of the DTAA.

Theorem 3.7. *The task assignment algorithm DTAA guarantees that* $f^{DTAA} \leqslant f^{da}$.

Proof. We first prove that $f_q^{DTAA} \leqslant f_q^{da}, \forall q \in \mathcal{A}$. The proof is conducted by induction. In Theorem 3.5, the assignment of the targets in each $\mathcal{T}_q$ resulting from the DTAA is shown to converge to that of the generalized min-cost arborescence of $\mathcal{G}_q, \forall q \in \mathcal{A}$. Let f_{qj}^{da} be the sum of all the edge weights of the duplicated arborescence for vehicle $j \in \mathcal{R}_q$ that can visit targets in $\mathcal{T}_q, \forall q \in \mathcal{A}$. Then, $f_q^{da} = \sum_{j\in\mathcal{R}_q} f_{qj}^{da}$. The first target $r^\star$ to be inserted in o_j for the DTAA is determined by (3.8). It is straightforward to see that $r^\star$ is the same as the first target inserted in the arborescence for vehicle j according to line 3 of Algorithm 1. Thus, $f_{qj1}^{DTAA} \leqslant f_{qj1}^{da}$ as $f_{qj1}^{DTAA} = f_{qj1}^{da}$, where the subscripts 1 and j are associated with the total travel time for vehicle j to visit the first target ordered in λ_j.

Now suppose the first $|o_j| - 1$ targets inserted in o_j and those inserted in the arborescence for vehicle j are the same and $f_{qj|o_j|-1}^{DTAA} \leqslant f_{qj|o_j|-1}^{da}$, where o_j contains all the targets in the end assigned to vehicle j. As the inequality specified in Remark 3.1 holds for the optimal travel times between the vertices in $\mathcal{G}$ and according to (3.10), for the DTAA the marginal travel time incurred by inserting the last target r of o_j into λ_j is

$$\begin{aligned}
\delta f_{qj}^{DTAA} &= \min_{1<k\leqslant|\lambda_j|+1}\{t(\lambda_j \oplus_k r) - t(\lambda_j)\} \\
&= \min\{\min_{k\leqslant|\lambda_j|-1}(t(\lambda_j^k, r) + t(r, \lambda_j^{k+1}) - t(\lambda_j^k, \lambda_j^{k+1})), \\
&\quad t(p_j, r) + t(r, \lambda_j^1) - t(p_j, \lambda_j^1), t(\lambda_j^{|\lambda_j|}, r)\} \\
&\leqslant \min_{k\leqslant|\lambda_j|-1}\{t(\lambda_j^k, r) + t(r, \lambda_j^k), t(r, \lambda_j^{k+1}) + t(\lambda_j^{k+1}, r)\}, \quad (3.12)
\end{aligned}$$

where p_j is the index of vehicle j' initial location and λ_j^k is the kth target on the ordered target list λ_j. On the other hand, considering the travel time cost on duplicating the edge of the min-cost arborescence, the minimum travel time incurred by inserting the last target r into the arborescence for vehicle j is

$$\delta f_{qj}^{da} = t(r, \lambda_j^{k^\star}) + \min_{k \leqslant |\lambda_j|} t(\lambda_j^k, r), \tag{3.13}$$

where $k^\star = \text{argmin}_{k \leqslant |\lambda_j|} t(\lambda_j^k, r)$. It then follows that $\delta f_{qj}^{DTAA} \leqslant \delta f_{qj}^{da}$.

Combining (3.12) and (3.13) and in view of $f_{qj|o_j|-1}^{DTAA} \leqslant f_{qj|o_j|-1}^{da}$, we get

$$\begin{aligned} f_{qj|o_j|}^{DTAA} &= f_{qj|o_j|-1}^{DTAA} + \delta f_{qj}^{DTAA} \\ &\leqslant f_{qj|o_j|-1}^{da} + \delta f_{qj}^{da}. \end{aligned} \tag{3.14}$$

As $f_{qj}^{da} = f_{qj|o_j|-1}^{da} + \delta f_{qj|o_j|}^{da}$, it holds that $f_{qj|o_j|}^{DTAA} \leqslant f_{qj|o_j|}^{da}$ for each vehicle j. Thus, we get $\sum_{j \in \mathcal{R}_q} f_{qj}^{DTAA} \leqslant \sum_{j \in \mathcal{R}_q} f_{qj}^{da}$, which proves $f_q^{DTAA} \leqslant f_q^{da}$.

As $\mathcal{R} = \sum_{q=1}^{|\mathcal{A}|} \mathcal{R}_q$ where $\mathcal{R}_q \cap \mathcal{R}_p = \varnothing$ for $\forall q \neq p$, it is straightforward to observe that $f^{DTAA} = \sum_{q=1}^{|\mathcal{A}|} f_q^{DTAA}$. Thus, $f^{DTAA} \leqslant \sum_{q=1}^{|\mathcal{A}|} f_q^{da}$. The proof is complete.

□

Remark 3.8. *Theorem 3.7 gives the worst case performance of DTAA compared with the optimal solution as f^{da}/f^o, which generalizes the upper bound for the task assignment problem with a symmetric cost matrix in [43]. Furthermore, based on Lemma 3.6 and Theorem 3.7, the upper bound f^{da}/f^o is 2 if the cost matrix is symmetric.*

Now we have presented all the theoretical results of this chapter. In the following section, we carry out simulation studies.

3.7 Simulations

In this section, Monte Carlo simulations are carried out to test the proposed algorithms compared with a GA which is a popular heuristic algorithm for the VRP [46]. For each pair of target subset $\mathcal{T}_q$ and the corresponding vehicle subset $\mathcal{R}_q$, $q \in \{1, \cdots, |\mathcal{A}|\}$, the GA encodes each target in $\mathcal{T}_q$ as a numbered gene and inserts $|\mathcal{R}_q| - 1$ marker genes into the target genes. Then, each chromosome represents a candidate solution to the assignment of the targets in $\mathcal{T}_q$. The GA employs the widely used tournament selection because of its efficiency and simplicity, which preserves gene diversity while guaranteeing all individuals might be selected [2]. The number of chromosomes in the GA for each $\mathcal{T}_q$ is empirically set as $3(|\mathcal{T}_q| + |\mathcal{R}_q|)$, and the crossover rate and mutation rate for the GA are 0.9 and 0.1. The GA terminates at the maximal iteration number 350. All the experiments have been performed on an Intel Core $i5 - 4590$ CPU 3.30 GHz with 8 GB RAM, with the algorithms compiled by Matlab under Windows 7. The solution quality of

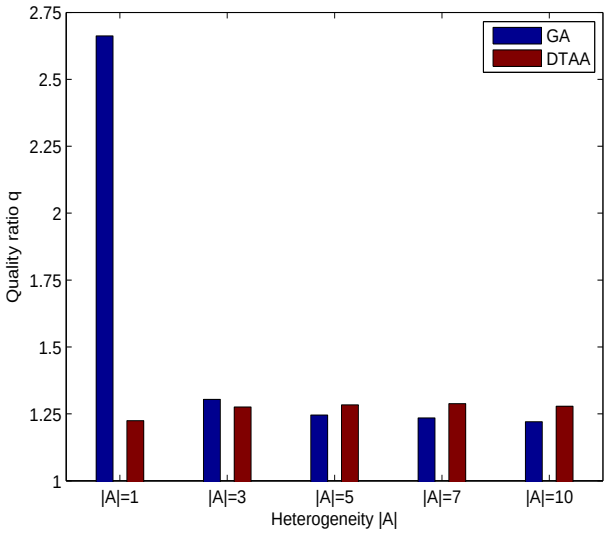

Figure 3.4: The average performance of the algorithms for 10 vehicles under different heterogeneities $|\mathcal{A}|$ to satisfy the visiting demands of 50 target locations in the drift field.

each task assignment algorithm is quantified by

$$q = \frac{f}{f^a}, \tag{3.15}$$

where f is the objective value in (3.1) and f^a is a lower bound on the optimal solution of the problem as shown in Theorem 3.4.

The task assignment algorithm DTAA and GA are first integrated with the designed path planning algorithm to assign 10 vehicles under different heterogeneities $|\mathcal{A}|$ with $v = 1$ to satisfy the visiting demands of 50 target locations in a square drift field with edge length 10^3m and $\vec{v}_c = 10^{-3}[0.3x + 0.2y, -0.2x + 0.3y]^T$. In the field, there are three static rectangular obstacles whose positions are $\{(x, y)|(x, y) \in [150\ 300] * [100\ 120] \cup [400\ 420] * [350\ 500] \cup [600\ 750] * [600\ 620]\}$. Under each $|\mathcal{A}| \in \{1, 3, 5, 7, 10\}$, 50 scenarios for each set of the initial positions of the targets and vehicles are randomly generated while the positions are located outside the obstacles. The visiting demands of each target and the capability of each vehicle are randomly generated while the vehicle capabilities can satisfy all the targets. The average q of the solutions resulting from the algorithms under different $|\mathcal{A}|$ are shown in Fig. 3.4, and the average variance of q and the average computation time of the algorithms are shown in Table 3.1 and Table 3.2, respec-

Table 3.1: The average variances of the solution qualities q of the algorithms on 50 scenarios for the task assignment problem under different instances (I) and different heterogeneities $|\mathcal{A}|$ where $n50m10$ means 10 vehicles need to satisfy the visiting demands of 50 target locations.

I \ $\lvert\mathcal{A}\rvert$		1	3	5	7	10
$n50m10$	GA	0.4975	0.0210	0.0036	0.0025	0.0015
	DTAA	0.0035	0.0030	0.0024	0.0024	0.0014
$n70m10$	GA	0.0515	0.1015	0.0453	0.0257	0.0018
	DTAA	0.0026	0.0017	0.0012	0.0013	0.0009
$n90m10$	GA	0.0297	0.1086	0.0700	0.0570	0.0262
	DTAA	0.0025	0.0013	0.0011	0.0009	0.0013

Table 3.2: The average corresponding computation time (s) of the algorithms to get the solution to the task assignment problem under different heterogeneities $|\mathcal{A}|$ and different instances (I).

I \ $\lvert\mathcal{A}\rvert$		1	3	5	7	10
$n50m10$	GA	37.04	41.12	49.66	58.27	72.74
	DTAA	0.19	0.32	0.46	0.60	0.82
$n70m10$	GA	65.06	81.63	105.94	128.53	156.09
	DTAA	0.29	0.48	0.71	0.93	1.25
$n90m10$	GA	104.55	134.19	172.45	213.34	273.92
	DTAA	0.38	1.19	2.29	3.20	4.40

tively. First, the average q of the DTAA displayed in Fig. 3.4 is within 1.3 times of the optimal under different $|\mathcal{A}|$ while the average q of the GA decreases when increasing $|\mathcal{A}|$, which shows stable performance of the DTAA. When $|\mathcal{A}|$ increases, the number of targets within each target subset $\mathcal{T}_q, q \in \{1, \cdots, |\mathcal{A}|\}$, and the number of vehicles within each vehicle subset $\mathcal{R}_q$ generally decrease, thus leading to a smaller problem size for assigning the targets in each $\mathcal{T}_q$ to the vehicles in $\mathcal{R}_q$. The reason that GA generally performs better when $|\mathcal{A}|$ increases might due to the resulting smaller problem size. Second, the DTAA far betters than the GA when $|\mathcal{A}| = 1$ while the GA is competitive with the DTAA with the increase of $|\mathcal{A}|$. However, Table 3.2 shows that the average computation time of the GA increases with the increase of $|\mathcal{A}|$, which far exceeds those of the DTAA. The phenomenon indicates that the DTAA is more scalable than the GA. The computation time increases with a larger $|\mathcal{A}|$ as some target locations need to be visited more than once by vehicles with different capabilities and more visiting demands might occur when increasing $|\mathcal{A}|$. Thus, more computation time is needed to find solutions when increasing $|\mathcal{A}|$. The average variances of the solution qualities q of the DTAA is also

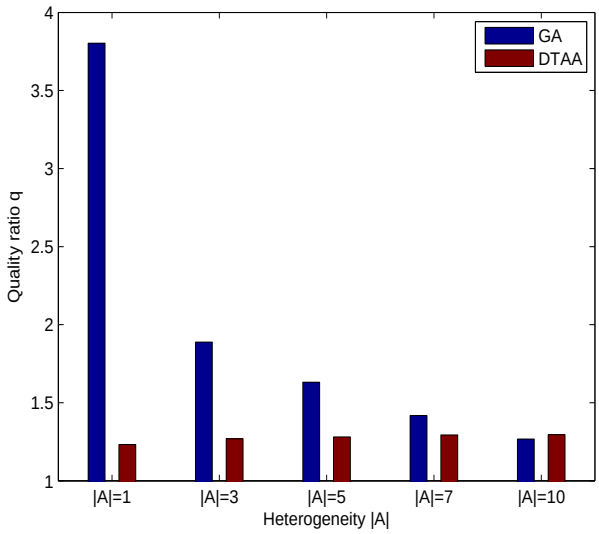

Figure 3.5: The average performance of the algorithms for 10 vehicles under different heterogeneities $|\mathcal{A}|$ to satisfy the visiting demands of 70 target locations in the drift field.

relatively smaller than those of the GA under different $|\mathcal{A}|$ for $n50m10$ as shown in Table 3.1, which implies the stable feature of the DTAA. The instance $n50m10$ means 10 vehicles need to satisfy the visiting demands of 50 target locations.

We have also tested the DTAA and the GA on the instances with larger problem sizes where 10 vehicles under different $|\mathcal{A}|$ need to satisfy the visiting demands of 70 and 90 target locations respectively in the same drift field. For each instance, simulations on 50 scenarios have been performed for the problem with each $|\mathcal{A}| \in \{1, 3, 5, 7, 10\}$. The average q of the solutions are shown in Fig. 3.5 and Fig. 3.6, and the variances of q and the average computation times of the algorithms are shown in Table 3.1 and Table 3.2, respectively. The average q of the DTAA under each $|\mathcal{A}|$ shown in Fig. 3.5 and Fig. 3.6 are within 1.5 times of the optimal as those in Fig. 3.4, which displays the satisfying and stable performance of the DTAA. The changing trends of the average q of the GA shown in Fig. 3.5 and Fig. 3.6 are generally the same as those shown in Fig. 3.4, which decreases with the increase of $|\mathcal{A}|$. However, the GA becomes worser with an increase of problem size from $n70m10$ to $n90m10$, where the DTAA far betters the GA under each $|\mathcal{A}|$ as shown in Fig. 3.6. In addition, Table 3.2 shows that the mean computation times of the DTAA on $n70m10$ and $n90m10$ do not increase too much compared with those on $n50m10$ while that is not the case for the GA, which shows the scalability of the DTAA. Furthermore, in Table 3.1 the variances of q of the algorithms decrease

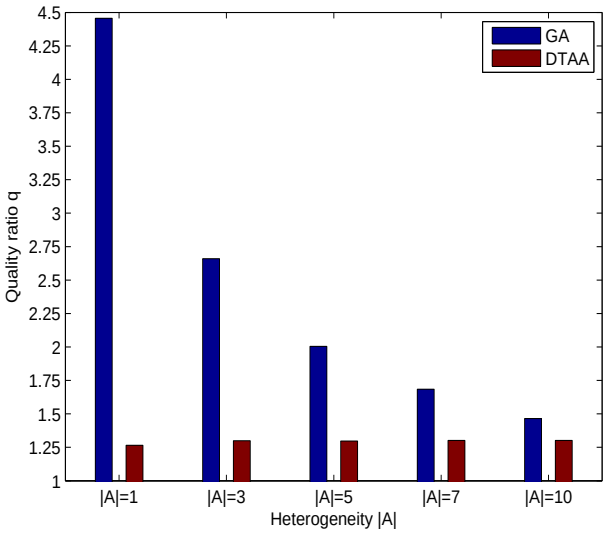

Figure 3.6: The average performance of the algorithms for 10 vehicles under different heterogeneities $|\mathcal{A}|$ to satisfy the visiting demands of 90 target locations in the drift field.

when $|\mathcal{A}|$ increases, which might due to less cooperation among the vehicles when increasing their heterogeneities.

3.8 Conclusion

This chapter has investigated the task assignment problem in which multiple dispersed heterogeneous vehicles need to satisfy the visiting demands of a set of target locations in a time-invariant drift field with obstacles. We have proved the NP-hardness of the task assignment problem. A path planning method has been designed to enable the vehicles to move between two prescribed locations in a drift field with minimal time while avoiding any obstacle. The travel cost matrix resulting from the path planning method provides the route information for the target location assignment. In addition, a distributed task assignment algorithm has been proposed which enables the vehicles to cooperate based on local information. The task assignment algorithm guarantees that all the visiting demands of the targets are satisfied within the total travel time, which is at most twice of the optimal one when the cost matrix is symmetric.

Chapter 4

Communication-constrained multi-vehicle task assignment

THIS chapter investigates the task assignment problem in which multiple dispersed heterogeneous vehicles/robots need to visit a set of target locations while trying to minimize the vehicles' total travel distance. Each vehicle initially has the position information of all the targets and of those vehicles that are within its limited communication range, and each target demands a vehicle with some specified capability to visit it. We propose a decentralized auction algorithm which first employs an information consensus procedure to merge the local information carried by each communication-connected vehicle subnetwork. Then, we apply a marginal-cost-based strategy to construct conflict-free target assignments for the communication-connected vehicles. When the communication network of the vehicles is not connected, we demonstrate that the vehicles' total travel distance might in fact increase when their communication range grows, and more importantly such a somewhat counterintuitive fact holds for a range of algorithms. Furthermore, the proposed algorithm guarantees that the total travel distance of the vehicles is at most twice of the optimal when the communication network is initially connected. Finally, Monte Carlo simulation results demonstrate the satisfying performance of the proposed algorithm.

4.1 Introduction

Task assignment for a team of vehicles to visit a set of targets is to assign each target to some properly chosen vehicle, which can be achieved by either centralized or decentralized algorithms [41]. More specifically, the vehicle routing problem (VRP), which is NP-hard, has given rise to a number of centralized algorithms, including exact algorithms [7, 52], and heuristic algorithms [24, 66]. The VRP setup considered in [75] is that a fleet of vehicles needs to deliver products from one or several depots to a group of dispersed customers. Furthermore, a vacancy chain scheduling was developed to formalize vehicle interactions in [15]. A genetic algorithm was integrated with the Dubins car model in [21] for multi-UAV target assignment where the UAVs' limited turning radius was considered. Centralized algorithms can achieve optimal or near-optimal assignments; however, in dynamic

environment they require persistently reliable communication between the central server and each vehicle to obtain global information. Note that some multi-vehicle task assignment problems have been shown to be NP-hard [41], where high computation capacity is demanded for the central server. Consequently, centralized methods are not scalable with the growing numbers of vehicles and/or targets, and not suitable for the situation where vehicles are subject to local, and possibly outdated, information due to imperfect communication.

In parallel, decentralized algorithms have also been developed to enable each vehicle to plan its route [1, 17], and there are some investigation on task assignment for robots under limited communication ability [29, 72, 81, 83]. When robots are under limited communication range and potential malfunctions, a cooperative backoff adaptive scheme was proposed in [29] for the task allocation where the completion of one task might require the cooperation of multiple robots. In [72], monotonic algorithms were proposed to minimize the time when the last target location was occupied by a robot under limited communication range. Later on, decentralized algorithms were designed in [81] to minimize the robots' total travel distance until every target location was occupied by one robot subject to limited communication and sensing ranges. However, the numbers of robots and targets in [72] and [81] are equal, and consequently it is always logical for a robot to stop moving upon reaching its target. In recent years, auction-based methods, known to be computationally efficient, are popular for solving the task assignment problem [13, 18, 43, 44, 51, 67]. For the static task assignment of homogeneous robots, the auction-based allocation algorithm proposed in [43] guarantees that the robots' total travel cost is at most twice of the optimal assuming all the robots are *communication-connected* (CC). In [44], several Euclidean-supported auction algorithms were designed for optimizing different team objectives for which the theoretical performance of the algorithms has been provided accordingly. In [18], survey on the market-based multi-robot coordination approaches was presented where the main principles on auctions are discussed. Luo *et al.* [51] presented a distributed auction-based algorithm for the multi-robot task assignment where the tasks have to be completed within given deadlines. The auction-based algorithm has an approximation ratio of 2 for the problem in which the payoff for completing a task is independent of the others. Under the assumption of connectivity of the communication network, an auction algorithm, namely the consensus-based bundle algorithm (CBBA), was designed for multi-agent task allocation [13]. A robot using CBBA bids for a task based on the cost incurred by inserting the task into the robot's target bundle. Thus, the assignment of a task is heavily affected by the task's inserting sequence. As a result, CBBA needs to release all the tasks after the out-bid task in the bundle list. Furthermore, the performance of CBBA is guaranteed if the cost function of assigning each task leads to diminishing marginal gain, which means that the value of a task does not increase as other tasks are added

to the bundle list before this task [13]. In [83], the proposed heuristic distributed algorithm (HDA) changes the assignment of a target based on the significance and the marginal significance of a target where the significance of a target is defined as the cost saved by removing the target from the corresponding position of the ordered target list containing the target, and the marginal significance of a target for a vehicle, currently not containing the target, is the smallest cost for inserting the target into the current ordered target list of the vehicle. For each target on the route of a vehicle, the vehicle will release the target if the significance of the target is larger than its marginal cost while another vehicle with the smallest marginal significance for inserting the target will take over the target. The HDA has shown its better performance over the CBBA for search and rescue task assignment of multiple heterogeneous vehicles in [83].

Motivated by the existing literature just mentioned, our research focuses on the more realistic situation when the communication topology of the vehicles dynamically changes and might not be connected due to the vehicles' movement, and the number of targets is greater than that of vehicles. To be more precise, a fleet of randomly dispersed heterogeneous vehicles under limited communication range needs to visit several target locations while trying to minimize their total travel distance. Each target location requires to be visited by a vehicle with a special capability. As in [72] and [81], each vehicle is assumed to know at the beginning the positions of all the targets, and the positions of those vehicles that are within its limited communication range. Different from the auction-based task assignment algorithms discussed above, the communication network of the vehicles here can be arbitrary, and in particular is not required to be connected. Since the communication network of the vehicles might not be connected, the challenge is how to make those CC vehicles cooperate to ensure all the targets to be visited based on their knowledge of the local, possibly outdated, environmental information. The other interesting question we try to answer is whether a longer communication range of the vehicles always brings a better performance of the vehicles. A novel decentralized auction algorithm (DAA) is proposed which exploits the marginal inserting cost concept. The main contributions of this chapter are as follows. Firstly, for each CC vehicle subgroup that appears as the vehicles move around, DAA leads to the non-increasing projected total travel distance while guaranteeing all the target locations to be visited using the vehicles' local information. For DAA, we investigate what information should be carried by each vehicle, what information carried by each vehicle should be communicated to its directly CC vehicles through local communication, for each vehicle how to merge the received local, possibly outdated, information, and how to coordinate the vehicles in each CC vehicle subgroup to guarantee that all the targets will be visited within a computable maximal total travel distance. This is an extension to the existing auction-based task assignment studies, which in general assume that the commu-

nication network of the vehicles is always connected. Secondly, we find that the vehicles might have a greater total travel distance when they have a longer communication range, which holds for a range of algorithms. Lastly, it is analytically proven that the solution resulting from DAA is the same as some locally centralized auction algorithms, and the total travel distance of the vehicles is upper bounded by twice of the optimal when the communication network is initially connected.

The rest of this chapter is organized as follows. In Section 4.2, the formulation of the task assignment problem is presented. Section 4.3 studies the decentralized auction algorithm. We present Monte Carlo simulation results in Section 4.4 and conclude the chapter in Section 4.5.

4.2 Problem Statement

Consider a set of randomly dispersed heterogeneous vehicles $\mathcal{R} = \{1, \cdots, m\}$ that is employed in the plane to efficiently visit a set of target locations $\mathcal{T} = \{1, \cdots, n\}$ where $n > m$. Each target q has an attribute $a^q \in \mathcal{A}$ while each vehicle j has one ability $a_j \in \mathcal{A}$ with $|\mathcal{A}| \leqslant m$. Target q can be visited by vehicle j if $a^q = a_j$ as in [78] and [83]. Several targets might have the same attribute and several vehicles might have the same ability, while the union of all the vehicles' abilities and the union of all the targets' attributes are the same $\mathcal{A}$ so that all the targets can be visited. Each vehicle is initially unaware of the existence of other vehicles employed in the environment; however, it has the position information of all the targets in a map and can sense the positions of the vehicles within its limited sensing range, which for simplicity is assumed to be the same as the communication range.

The binary variable y_{qj} is used to represent whether target q is assigned to vehicle j. Each vehicle stops moving when knowing all of its assigned targets have been visited, and restarts to move once new assignment arrives through communicating with other moving vehicles. We assume that all the vehicles move with the unit speed. Then, the objective of the task assignment problem is to minimize

$$f = \sum_{j \in \mathcal{R}} d_j, \tag{4.1}$$

subject to

$$\sum_{j \in \mathcal{R}^q} y_{qj} \geqslant 1, \qquad \forall\, q \in \mathcal{T}, \tag{4.2}$$

where d_j is the total travel distance of vehicle j when all the targets are visited and $\mathcal{R}^q$ contains the indices of all the vehicles having the ability matching the attribute of target q.

Remark 4.1. *When $|\mathcal{A}| = 1$, the task assignment problem is a variant of the VRP [75]. When $|\mathcal{A}| = m$, each vehicle differs from the other vehicles. Thus, the problem can be reduced to m independent subproblems each being a variant of the traveling salesman problem [36] if the vehicles are not subject to a limited communication range.*

4.3 Auction algorithm

In this section, we propose auction-based algorithms for solving the task assignment problem. As the vehicles move around, the corresponding communication topology of the network dynamically changes because the vehicles are subject to limited communication range. We use $r > 0$ to denote the vehicles' communication range, and use the binary variable $g_{jk}(t)$ to describe whether vehicle j can directly communicate with vehicle k at time t, namely

$$g_{jk}(t) = \begin{cases} 1, & \text{if } d(p_j(t), p_k(t)) \leqslant r, \\ 0, & \text{otherwise}, \end{cases} \tag{4.3}$$

where $p_j(t)$ is the index of the position where vehicle j is located at time t, and $d(p_j(t), p_k(t))$ is the distance between the two vehicles.

To deal with the limited communication range, each vehicle $j \in \mathcal{R}$ carries an information tuple $\mathcal{I}_j(t) = \{j, p_j(t), a_j, \beta_j(t), c_j(t), o_j(t), d(o_j(t)), u_j(t), s_j(t)\}$ with $\mathcal{I}_j(t) \in \mathbb{R}^{5n+4}$, where j is the vehicle's unique ID, $a_j \in \mathcal{A}$ is its ability, $\beta_j(t)$ is an n-tuple whose qth component $\beta_{jq}(t)$ stores the ID of that vehicle assigned to target q, $c_j(t)$ is an n-tuple whose qth component $c_{jq}(t)$ stores the incurred distance for vehicle $\beta_{jq}(t)$ to visit target q, $o_j(t)$ is an n-tuple, which keeps the indices of all the ordered targets assigned to vehicle j at time t, $d(o_j(t))$ is the projected total distance for vehicle j to visit all the targets on the route $o_j(t)$, $u_j(t)$ is an n-dimensional vector initialized as $u_j(0) = (\mathbf{0}_n)^T$ where $u_{jk}(t)$ stores the latest time when a task reassignment is performed for a CC vehicle subgroup containing vehicles j and k, and $s_j(t)$ is an n-dimensional zero-one vector recording the targets' status, namely

$$s_{jq}(t) = \begin{cases} 1, \text{ if vehicle } j \text{ knows target } q \text{ has been visited}, \\ 0, \text{ otherwise}. \end{cases} \tag{4.4}$$

4.3.1 Centralized auction algorithm

For each newly CC vehicle subnetwork, we first propose a centralized auction algorithm (CAA) where all information tuples carried by the vehicles within the subnetwork are first relayed to a local leader vehicle having the most one-hop

neighbours. If there are several such potential leading vehicles, the leader is randomly chosen from those vehicles.

Then, after collecting all available information, the local leader vehicle of the CC vehicle subnetwork $\mathcal{R}_i(t)$ updates $s_j(t)$ for each vehicle j within the subnetwork by

$$s_{jq}(t) = \vee_{k\in\mathcal{R}_i(t)} s_{kq}(t),\ \forall j \in \mathcal{R}_i(t),\ \forall q \in \mathcal{T}, \tag{4.5}$$

where $\vee$ denotes the logical "or" operator, $\mathcal{R}_i(t)$ is the subgroup i in which vehicles are CC at time t. Then, based on $s_j(t)$, $o_j(t)$ is first updated by deleting the targets known to have been visited,

$$o_j(t) = o_j(t) \setminus \{q\},\ \text{if } s_{jq}(t) = 1, \forall q \in o_j(t). \tag{4.6}$$

Afterwards, the projected travel distance $d(o_j(t))$ of vehicle j is updated considering all the targets in $o_j(t)$ to be consecutively visited by vehicle j.

If vehicle subgroup $\mathcal{R}_i(t)$ has no new members compared with $\mathcal{R}_i(t-1)$ or the vehicles in $\mathcal{R}_i(t)$ have recently been CC with the vehicles in the same $\mathcal{R}_i(t)$ for a task reassignment, the vehicles within the group perform their assigned tasks without task reassignment. Otherwise, let the set $\mathcal{R}_{ij}(t) \subseteq \mathcal{R}_i(t)$ contain those vehicles in $\mathcal{R}_i(t)$ who own ability $a_j \in \mathcal{A}$. To deal with how to make CC vehicles in $\mathcal{R}_{ij}(t)$ coordinate based on their carried local information, a cooperative strategy is designed. We use $\mathcal{R}^1_{ij}(t) \subseteq \mathcal{R}_{ij}(t)$ to identify those vehicles in $\mathcal{R}_{ij}$ who have never been CC with any other vehicle with the ability a_j before time t, and define $\mathcal{R}^2_{ij}(t)$ to be $\mathcal{R}_{ij}(t) \setminus \mathcal{R}^1_{ij}(t)$. Then, the targets to be assigned to the vehicles in $\mathcal{R}_{ij}(t)$ are those in the set

$$\mathcal{T}_{ij}(t) = \begin{cases} o_p(t), & \text{if } \sum_{k\in\mathcal{R}^2_{ij}(t)} d(o_k(t)) \geqslant d(o_p(t)) \\ & \text{or } \mathcal{R}^2_{ij}(t) = \varnothing, \text{ when } \mathcal{R}^1_{ij}(t) \neq \varnothing, \\ \bigcup\limits_{k\in\mathcal{R}^2_{ij}(t)} o_k(t), & \text{otherwise,} \end{cases} \tag{4.7}$$

where $p = \operatorname{argmin}_{k\in\mathcal{R}^1_{ij}(t)} d(o_k(t))$. When the total travel distance incurred for visiting all the targets on the routes of the vehicles in $\mathcal{R}^2_{ij}(t)$ is larger than that incurred by visiting all the targets in $o_p(t)$, it holds that $\sum_{k\in\mathcal{R}^2_{ij}(t)} d(o_k(t)) \geqslant d(o_p(t))$. What is more, if $\mathcal{R}^2_{ij}(t) = \varnothing$ and $\mathcal{R}^1_{ij}(t) \neq \varnothing$, $\mathcal{T}_{ij}(t)$ is set to $o_p(t)$, and otherwise $\mathcal{T}_{ij}(t)$ contains all the unvisited targets on the routes of the vehicles in $\mathcal{R}^2_{ij}(t)$. The targets to be divided by all the vehicles in $\mathcal{R}_i(t)$ are the union of all

the targets in each $\mathcal{T}_{ij}(t)$,

$$\mathcal{T}_i(t) = \bigcup_{j \in \mathcal{R}_i(t)} \mathcal{T}_{ij}(t). \tag{4.8}$$

Afterwards, if any vehicles $j, k \in \mathcal{R}_i(t)$ satisfy $a_j = a_k$ and $u_{jk}(t) \neq \max u_k(t)$, CAA iteratively assigns the targets in $\mathcal{T}_i(t)$ to the vehicles in $\mathcal{R}_i(t)$ as follows. Firstly, CAA determines the minimum marginal travel distance $c_{jq}(t)$ for inserting each target q with attribute $a^q = a_j$ of the current unassigned target set $\mathcal{T}_i(t)$ into the current $o'_j(t)$, initially set as an empty vector, by

$$c_{jq}(t) = \begin{cases} d(p_j(t), q), \text{ if } o'_j(t) = \varnothing, \\ \min\limits_{l \leqslant |o'_j(t)|+1} \{d(o'_j(t) \oplus_l q) - d(o'_j(t))\}, \text{ otherwise}, \end{cases} \tag{4.9}$$

where $o'_j(t) \oplus_l q$ means to insert target q at the lth position of the target list $o'_j(t)$ if $o'_j(t) \neq \varnothing$. Target q is inserted to the end of $o'_j(t)$ if $l = |o'_j(t)| + 1$ where $|o'_j(t)|$ is the length of $o'_j(t)$, and $c_{jq}(t) = \infty$ if $a^q \neq a_j$ which means vehicle j lacks the ability to visit target q. Then, the n-tuple marginal cost c_j for each vehicle $j \in \mathcal{R}_i(t)$ is updated as

$$c_{jq}(t) = \min_{k \in \mathcal{R}_i(t)} c_{kq}(t), \forall q \in \mathcal{T}_i(t). \tag{4.10}$$

Then, CAA determines that vehicle $j^\star \in \mathcal{R}_i(t)$ wins the target $q^\star \in \mathcal{T}_i(t)$ by

$$(j^\star, q^\star) = \operatorname*{argmin}_{(j,q) \in \mathcal{R}_i(t) \times \mathcal{T}_i(t)} c_{jq}(t). \tag{4.11}$$

Vehicle $j^\star$ will update its $o'_{j^\star}(t)$ as

$$o'_{j^\star}(t) = \begin{cases} \{q^\star\}, \text{ if } o'_{j^\star}(t) = \varnothing, \\ o'_{j^\star}(t) \oplus_{l^\star} q^\star, \text{ otherwise}, \end{cases} \tag{4.12}$$

where $l^\star = \operatorname{argmin}_{l \leqslant |o'_{j^\star}|+1} \{d(o'_{j^\star}(t) \oplus_l q^\star) - d(o'_{j^\star}(t))\}$. Meanwhile, $\mathcal{T}_i(t)$ is updated to

$$\mathcal{T}_i(t) = \mathcal{T}_i(t) \setminus \{q^\star\}. \tag{4.13}$$

The target assignment procedure continues until $\mathcal{T}_i(t)$ is empty.

The objective for the leader vehicle of each $\mathcal{R}_{ij}(t)$ at time t is to minimize

$$f'_{ij}(t) = \sum_{k \in \mathcal{R}_{ij}(t)} d(o'_k(t)), \tag{4.14}$$

Algorithm 3 CAA for the leader vehicle s in each newly CC vehicle subgroup $\mathcal{R}_i(t)$.

Input: Each vehicle $j \in \mathcal{R}$ carries a matrix $\mathcal{X}_j \in \mathbb{R}^{n*(3n+3)}$ where $\mathcal{X}_j(j,:) = \{j, p_j(t), a_j, o_j(t), u_j(t), s_j(t)\}$ and $\mathcal{X}_j(p,:) = (\mathbf{0}_{3n+3})^T$ if $p \neq j$.
Output: Every target in the cooperative target set $\mathcal{T}_i(t)$ is assigned.

$\lambda \leftarrow 0$, and $\mathcal{R}_s^c(t) \leftarrow \mathcal{R}_s^d(t)$, where for $\forall k \in \mathcal{R}_s^d(t)$, $d(p_s(t), p_k(t)) \leqslant r$.
while $\lambda < n$ **do**
  Send $\mathcal{X}_s$ and $\mathcal{R}_s^c(t)$ to each $k \in \mathcal{R}_s^d(t) \setminus \{s\}$, and receive $\mathcal{X}_k$ and $\mathcal{R}_k^c(t)$ from k.
  $\mathcal{R}_s^c(t) \leftarrow \mathcal{R}_s^c(t) \bigcup \mathcal{R}_k^c(t), \forall k \in \mathcal{R}_s^d(t)$.
  for every row $w \in \{1, ..., n\}$ **do**
    if $\mathcal{X}_s(w,:) = (\mathbf{0}_{3n+3})^T$ and $\mathcal{X}_k(w,:) \neq (\mathbf{0}_{3n+3})^T$ **then**
      $\mathcal{X}_s(w,:) \leftarrow \mathcal{X}_k(w,:)$.
    end if
  end for
  $\lambda \leftarrow \lambda + 1$.
end while
$\mathcal{R}_i(t) \leftarrow \mathcal{R}_s^c(t)$, and vehicle s calculates $\mathcal{T}_i(t)$ by updating $\mathcal{X}_s$ through (4.5) to (4.8).
if $\exists k, j \in \mathcal{R}_i(t)$ satisfy $a_j = a_k$ and $u_{jk}(t) \neq \max u_k(t)$ **then**
  for the leader vehicle s **do**
    $o_j'(t) \leftarrow \varnothing, \forall j \in \mathcal{R}_i(t)$.
    while $\mathcal{T}_i(t) \neq \varnothing$ **do**
      Use (4.9) where $o_j'(t)$ is used to update $c_{jq}(t), \forall j \in \mathcal{R}_i(t), \forall q \in \mathcal{T}_i(t)$.
      $(j^\star, q^\star) \leftarrow \mathrm{argmin}_{(j,q) \in \mathcal{R}_i(t) \times \mathcal{T}_i(t)} c_{jq}(t)$.
      if $o_{j^\star}'(t) = \varnothing$ **then**
        $o_{j^\star}'(t) \leftarrow \{q^\star\}$.
      else
        $l^\star \leftarrow \mathrm{argmin}_{l \leqslant |o_{j^\star}'|+1} d(o_{j^\star}'(t) \oplus_l q^\star)$, $o_{j^\star}'(t) \leftarrow o_{j^\star}'(t) \oplus_{l^\star} q^\star$.
      end if
      $\mathcal{T}_i(t) \leftarrow \mathcal{T}_i(t) \setminus \{q^\star\}$.
    end while
    Calculate $f_{ij}'(t)$ based on the new assignment $o_j'(t)$.
  end for
  if $f_{ij}'(t) > f_{ij}(t)$ **then**
    The leader vehicle informs each $j \in \mathcal{R}_i(t)$ to keep its previous route $o_j(t)$.
  else
    $\lambda \leftarrow 0$, and the leader vehicle s updates $\mathcal{X}_s$ with $o_j'(t)$ and $u_j(t)$, $\forall j \in \mathcal{R}_i(t)$, and each other vehicle j updates $\mathcal{X}_j$ with $\mathcal{X}_j = \mathbf{0}_{n*(3n+3)}$.
    while $\lambda < |\mathcal{R}_i(t)|$ **do**
      for every vehicle $j \in \mathcal{R}_i(t)$ **do**
        Send $\mathcal{X}_j$ to each vehicle $k \in \mathcal{R}_j^d(t) \setminus \{j\}$, and receive $\mathcal{X}_k$ from k.
        for every row $w \in \{1, ..., n\}$ **do**
          if $\mathcal{X}_j(w,:) = (\mathbf{0}_{3n+3})^T$ and $\mathcal{X}_k(w,:) \neq (\mathbf{0}_{3n+3})^T$ **then**
            $\mathcal{X}_j(w,:) \leftarrow \mathcal{X}_k(w,:)$.
          end if
        end for
      end for
      $\lambda \leftarrow \lambda + 1$.
    end while
    Every vehicle $j \in \mathcal{R}_i(t)$ uses the updated $\mathcal{X}_j$ to update its $o_j(t)$ and $u_j(t)$.
  end if
end if

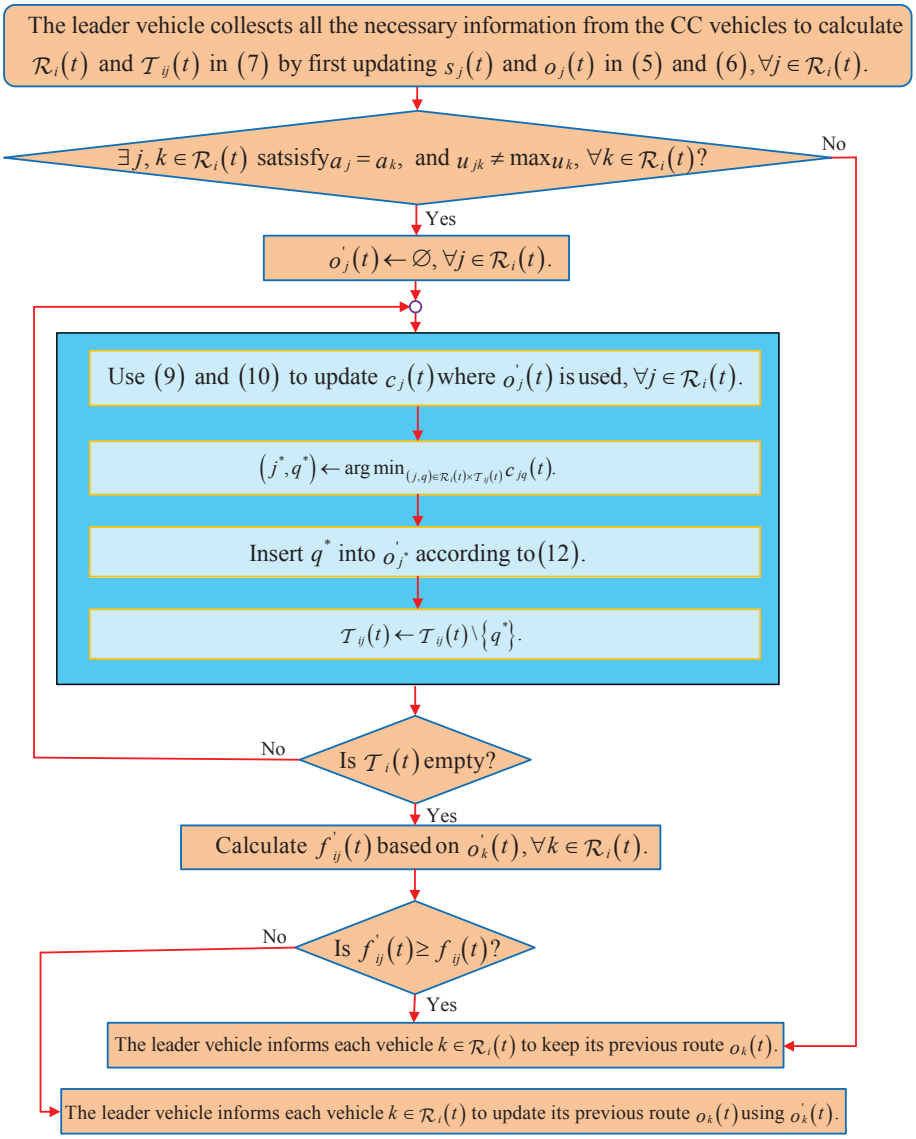

Figure 4.1: The flowchart of the centralized auction algorithm CAA.

where $d(o_k'(t))$ is the total travel distance for vehicle k to visit all the targets in its ordered target list $o_k'(t)$. The algorithm CAA used by the leader vehicle to assign the cooperative targets in $\mathcal{T}_{ij}(t)$ to the vehicles in $\mathcal{R}_{ij}(t)$ is shown in Algorithm 3, and the flowchart of CAA is shown in Fig. 4.1. For the task assignment problem,

the longest travel distance of a vehicle occurs when the vehicle visits all the target locations that it can visit without cooperation with any other vehicle. Thus, each vehicle j initially inserts all the targets it can visit into its route $o_j(t)$ by using the CAA where $\mathcal{R}_i(t) = \{j\}$ and $\mathcal{T}_i(t) = \mathcal{T}$.

For the CC vehicles in $\mathcal{R}_{ij}(t)$, they keep their previous target assignment if the total travel distance $f'_{ij}(t)$ resulting from the new assignment is greater than the previous total travel distance $f_{ij}(t)$. Otherwise, once a locally centralized task assignment has been completed in $\mathcal{R}_{ij}(t)$, each vehicle $k \in \mathcal{R}_{ij}(t)$ updates its route $o_k(t)$ using $o'_k(t)$. Then, the leader vehicle updates $u_{jk}(t), \forall j, k \in \mathcal{R}_i(t)$, to

$$u_{jk}(t+1) = \begin{cases} t, \text{ if } f'_{ij}(t) < f_{ij}(t), \\ u_{jk}(t), \text{ otherwise.} \end{cases} \tag{4.15}$$

4.3.2 Decentralized auction algorithm (DAA)

In this section, we design a decentralized auction algorithm (DAA) where each vehicle within a newly CC vehicle group updates its information tuple based on direct communication with its neighbors. DAA consists of three phases, which are information consensus for CC vehicles, target assignment for the vehicles, and assignment check. For each vehicle j, it initially inserts all the targets it can visit into its target list $o_j(t = 0)$ by using CAA shown in Algorithm 3 where $\mathcal{R}_i(t) = \{j\}$ and $\mathcal{T}_i(t) = \mathcal{T}$.

The first phase of DAA, shown in Algorithm 4, is to make each vehicle in $\mathcal{R}_i(t)$ reach consensus on the cooperative targets $\mathcal{T}_i(t)$ and the indices of the CC vehicles in $\mathcal{R}_i(t)$ based on their carried information matrix $\mathcal{X}_j \in \mathbb{R}^{n*(n+4)}$ where the jth row of $\mathcal{X}_j$ is $\mathcal{X}_j(j,:) = [j, a_j, u_j(t), p_j(t), d(o_j(t))]$ and $\mathcal{X}_j(p,:) = (\mathbf{0}_{n+4})^T$ if $p \neq j$. We use $\mathcal{X}_j(w, 1:l), w \in \{1, ..., n\}, l \in \{1, ..., n+4\}$, to denote the first to the lth elements of the wth row of $\mathcal{X}_j$, and $\mathcal{X}_j(:, 1:l)$ is the sub-matrix containing the first l columns of the matrix $\mathcal{X}_j$. Let the set $\mathcal{R}^d_j(t)$ contain all those vehicles that are directly CC with vehicle j at time t, where the distance $d(p_j(t), p_k(t))$ between vehicle $j \in \mathcal{R}_i(t)$ and any vehicle $k \in \mathcal{R}^d_j(t)$ is no larger than their communication range r, and initialize $\mathcal{R}^c_j(t)$ to $\mathcal{R}^d_j(t)$ as shown in line 1 of Algorithm 4. In Phase 1 of DAA, each vehicle j first updates its $s_j(t)$ and $\mathcal{R}^c_j(t)$ according to lines 1 to 12 shown in Algorithm 4. Then, in line 14 of Phase 1, vehicle j updates its projected total travel distance $d(o_j(t))$ after deleting in $o_j(t)$ the targets known to have been visited in line 13. Afterwards, vehicle j checks whether some of the CC vehicles have the same ability and whether they have not been CC recently in line 15 of the Phase 1. If both are yes, vehicle j continues communicating with other vehicles in $\mathcal{R}^d_j(t)$ to get the $p_k(t)$, and the updated $d(o_k(t))$ for each $k \in \mathcal{R}_i(t)$ as shown in lines 15 to 26 of Algorithm 4. Let the set $\text{IDs}_j = \{p\}$ if

Algorithm 4 DAA Phase 1 for information consensus of each $j, \forall j \in \mathcal{R}$ at time t.

Input: Vehicle j carries a matrix $\mathcal{X}_j \in \mathbb{R}^{n*(n+4)}$ with $\mathcal{X}_j(j,:) \leftarrow \{j, a_j, u_j(t), p_j(t), d(o_j(t))\}$, an array $\mathcal{R}_j^c(t) \in \mathbb{R}^n$, $\mathcal{I}_j(t)$ and r.
Output: The cooperative target set $\mathcal{T}_i(t)$ and CC vehicle set $\mathcal{R}_i(t)$.

$\lambda \leftarrow 0$, and $\mathcal{R}_j^c(t) \leftarrow \mathcal{R}_j^d(t)$ where for $\forall k \in \mathcal{R}_j^d(t)$, $d(p_j(t), p_k(t)) \leqslant r$.
while $\lambda < n$ **do**
  Send $\mathcal{X}_j(:, 1:(n+2))$, $\mathcal{R}_j^c(t)$ and $s_j(t)$ to each vehicle $k \in \mathcal{R}_j^d(t) \setminus \{j\}$.
  Receive $\mathcal{X}_k(:, 1:(n+2))$, $\mathcal{R}_k^c(t)$ and $s_k(t)$ from each vehicle $k \in \mathcal{R}_j^d(t) \setminus \{j\}$.
  $\mathcal{R}_j^c(t) \leftarrow \mathcal{R}_j^c(t) \bigcup \mathcal{R}_k^c(t), \forall k \in \mathcal{R}_j^d(t)$, $s_{jq}(t) \leftarrow \vee_{k \in \mathcal{R}_j^d(t)} s_{kq}(t),\ \forall q \in \mathcal{T}$.
  for every row $w \in \{1, ..., n\}$ **do**
    if $\mathcal{X}_j(w,:) = (\mathbf{0}_{n+4})^T$ and $\mathcal{X}_k(w,:) \neq (\mathbf{0}_{n+4})^T$ **then**
      $\mathcal{X}_j(w, 1:(n+2)) \leftarrow \mathcal{X}_k(w, 1:(n+2))$.
    end if
  end for
  $\lambda \leftarrow \lambda + 1$.
end while
$\mathcal{R}_i(t) \leftarrow \mathcal{R}_j^c(t)$, and $o_j(t) \leftarrow o_j(t) \setminus \{q\},$ if $s_{jq}(t) = 1, \forall q \in o_j(t)$.
Update the $(n+4)$th position in jth row of $\mathcal{X}_j$ with $d(o_j(t))$ according to $o_j(t)$.
if $\exists k, j \in \mathcal{R}_i(t)$ satisfy $a_j = a_k$, and $u_{jk}(t) \neq \max u_k(t), \forall k \in \mathcal{R}_i(t)$ **then**
  $\lambda \leftarrow 0$.
  while $\lambda < |\mathcal{R}_i(t)|$ **do**
    Send $\mathcal{X}_j(:, (n+3):(n+4))$ to each vehicle $k \in \mathcal{R}_j^d(t) \setminus \{j\}$.
    Receive $\mathcal{X}_k(:, (n+3):(n+4))$ from each vehicle $k \in \mathcal{R}_j^d(t) \setminus \{j\}$.
    for every row $w \in \{1, ..., n\}$ **do**
      if $\mathcal{X}_j(w, (n+3):(n+4)) = (\mathbf{0}_2)^T$ and $\mathcal{X}_k(w, (n+3):(n+4)) \neq (\mathbf{0}_2)^T$ **then**
        $\mathcal{X}_j(w, (n+3):(n+4)) \leftarrow \mathcal{X}_k(w, (n+3):(n+4))$.
      end if
    end for
    $\lambda \leftarrow \lambda + 1$.
  end while
  Use $\mathcal{X}_j$ to calculate the IDs$_j$ of the vehicle set according to (4.7) in order to calculate $\mathcal{T}_{ij}(t)$ for each $\mathcal{R}_{ij}(t) \subseteq \mathcal{R}_i(t)$, and let IDs$\leftarrow \bigcup_{\mathcal{R}_{ij}(t) \subseteq \mathcal{R}_i(t)}IDs_j$.
  if $j \in IDs$ **then**
    $\beta_{jq}(t) \leftarrow j,\ \forall q \in o_j(t)$, and $\beta_{jq}(t) \leftarrow 0,\ \forall q \in \mathcal{T} \setminus o_j(t)$.
  else
    $\beta_{jq}(t) \leftarrow 0,\ \forall q \in \mathcal{T}$.
  end if
  $\lambda \leftarrow 0$.
  while $\lambda < |\mathcal{R}_i(t)|$ **do**
    Send $\beta_j(t)$ to each vehicle $k \in \mathcal{R}_j^d(t) \setminus \{j\}$, and receive $\beta_k(t)$ from k.
    for each $w \in \{1, ..., n\}$ and $k \in \mathcal{R}_j^d(t)$ **do**
      if $\beta_{jw}(t) = 0$ and $\beta_{kw}(t) \neq 0$ **then**
        $\beta_{jw}(t) \leftarrow \beta_{kw}(t)$.
      end if
    end for
    $\lambda \leftarrow \lambda + 1$.
  end while
  Update $\mathcal{T}_i(t)$ in (4.8) with $\mathcal{T}_i(t) \leftarrow \bigcup_{q \in \mathcal{T}} q, \beta_{jq}(t) \neq 0$.
end if

Algorithm 5 DAA Phase 2 for target assignment of vehicle $j, \forall j \in \mathcal{R}_i(t)$, when some of the vehicles in $\mathcal{R}_i(t)$ are the same and $u_{jk}(t) \neq \max u_k(t), \forall k \in \mathcal{R}_i(t)$.

Input: The target set $\mathcal{T}_i(t)$ and vehicle set $\mathcal{R}_i(t)$.
Output: A new target list $o'_j(t)$ is produced.

1: $o'_j(t) \leftarrow \varnothing$.
2: **while** $\mathcal{T}_i(t) \neq \varnothing$ **do**
3: **for** every target $q \in \mathcal{T}_i(t)$ **do**
4: **if** $a^q = a_j$ **then**
5: **if** $o'_j(t) = \varnothing$ **then**
6: $c_{jq}(t) \leftarrow d(p_j(t), q)$.
7: **else**
8: $c_{jq}(t) \leftarrow \min_{l \leqslant |o'_j(t)|+1} \{d(o'_j(t) \oplus_l q) - d(o'_j(t))\}$.
9: **end if**
10: **else**
11: $c_{jq}(t) \leftarrow \infty$.
12: **end if**
13: $\beta_{jq}(t) \leftarrow j$.
14: **end for**
15: $\lambda \leftarrow 0$.
16: **while** $\lambda < |\mathcal{R}_i(t)|$ **do**
17: Send $c_j(t)$ and $\beta_j(t)$ to vehicle $k \in \mathcal{R}_j^d(t) \setminus \{j\}$.
18: Receive $c_k(t)$ and $\beta_k(t)$ from vehicle $k \in \mathcal{R}_j^d(t) \setminus \{j\}$.
19: **for** every target $q \in \mathcal{T}_i(t)$ **do**
20: $k^\star \leftarrow \operatorname{argmin}_{k \in \mathcal{R}_j^d(t)} c_{kq}(t)$.
21: $c_{jq}(t) \leftarrow c_{k^\star q}(t)$.
22: $\beta_{jq}(t) \leftarrow \beta_{k^\star q}(t)$.
23: **end for**
24: $\lambda \leftarrow \lambda + 1$.
25: **end while**
26: $q^\star \leftarrow \operatorname{argmin}_{q \in \mathcal{T}_i(t)} c_{jq}(t)$.
27: **if** $j = \beta_{jq^\star}$ **then**
28: **if** $o'_j(t) = \varnothing$ **then**
29: $o'_j(t) \leftarrow \{q^\star\}$.
30: **else**
31: $l^\star \leftarrow \operatorname{argmin}_{l \leqslant |o'_j(t)|+1} \{d(o'_j(t) \oplus_l q^\star) - d(o'_j(t))\}$.
32: $o'_j(t) \leftarrow o'_j(t) \oplus_{l^\star} q^\star$.
33: **end if**
34: **end if**
35: $\mathcal{T}_i(t) \leftarrow \mathcal{T}_i(t) \setminus \{q^\star\}$.
36: **end while**

$\sum_{k \in \mathcal{R}_{ij}^2(t)} d(o_k(t)) \geqslant d(o_p(t))$ in (4.7), and otherwise $\text{IDs}_j = \mathcal{R}_{ij}^2(t)$. Later on, vehicle j calculates the set IDs_j according to (4.7) through using $\mathcal{X}_j$, and then communicates with the other vehicles in $\mathcal{R}_j^d(t)$ to make the cooperative target set $\mathcal{T}_i(t)$ reach consensus as shown in lines 27 to 43 of Algorithm 4.

Algorithm 6 DAA Phase 3 for assignment check of vehicle $j, \forall j \in \mathcal{R}_i(t)$, when some of the vehicles in $\mathcal{R}_i(t)$ are the same and $u_{jk}(t) \neq \max u_k(t), \forall k \in \mathcal{R}_i(t)$.

Input: $\mathcal{R}_i(t)$ and the assignment $o'_j(t)$, $\lambda \leftarrow 0$.
Output: The total travel distance is nonincreasing.
1: Calculate the travel distance $d(o'_j(t))$ based on $o'_j(t)$.
2: **for** every row $w \in \{1, ..., n\}$ **do**
3: **if** $w = j$ **then**
4: $\mathcal{X}_j(j, n+4) \leftarrow d(o'_j(t))$.
5: **else**
6: $\mathcal{X}_j(j, n+4) \leftarrow 0$.
7: **end if**
8: **end for**
9: **while** $\lambda < |\mathcal{R}_i(t)|$ **do**
10: Send $\mathcal{X}_j(w, n+4)$ to each vehicle $k \in \mathcal{R}_j^d(t) \setminus \{j\}$.
11: Receive $\mathcal{X}_k(w, n+4)$ from each vehicle $k \in \mathcal{R}_j^d(t) \setminus \{j\}$.
12: **for** every row $w \in \{1, ..., n\}$ **do**
13: **if** $\mathcal{X}_j(w, n+4) = 0$ and $\mathcal{X}_k(w, n+4) \neq 0$ **then**
14: $\mathcal{X}_j(w, n+4) \leftarrow \mathcal{X}_k(w, n+4)$.
15: **end if**
16: **end for**
17: $\lambda \leftarrow \lambda + 1$.
18: **end while**
19: $f'_{ij}(t) \leftarrow \sum_{k \in \mathcal{R}_{ij}(t)} d(o'_k(t))$.
20: **if** $f'_{ij}(t) > f_{ij}(t)$ **then**
21: Vehicle j keeps its previous route $o_j(t)$.
22: **else**
23: $o_j(t) \leftarrow o'_j(t)$.
24: **end if**

Then, DAA iteratively assigns the targets in $\mathcal{T}_i(t)$ in Phase 2 shown in Algorithm 5. During each iteration, in lines 3 to 14 of Phase 2, each vehicle j updates its $c_{jq}(t)$ for each target q in the current $\mathcal{T}_i(t)$. In lines 15 to 25 of Phase 2, vehicle j compares the bids coming from its directly CC neighbors for each target q in $\mathcal{T}_i(t)$, and then updates its c_j and β_j according to the bids of the vehicles in $\mathcal{R}_j^d(t)$. Later on, in line 27, vehicle j checks the ID of the vehicle that takes over the target with the smallest inserting cost among those in the current $\mathcal{T}_i(t)$ according to the updated $\beta_j(t)$, and then updates its newly built route $o'_j(t)$ if the target is assigned to j according to lines 28 to 33 of Phase 2. Phase 2 requires more communication among the CC vehicles, but less computation capacity for each vehicle as the computation demand is distributed to each vehicle compared with CAA. A new target list $o'_j(t)$ is constructed from scratch for each CC vehicle j in line 1 of Algorithm 5, which differs from the CBBA [13] and HDA in [83]. The marginal cost $c_{jq}(t)$ is the smallest distance incurred for vehicle j to insert target q into $o'_j(t)$ if vehicle j has the ability to visit target q, i.e., $a_j = a^q$, and otherwise is infinity as

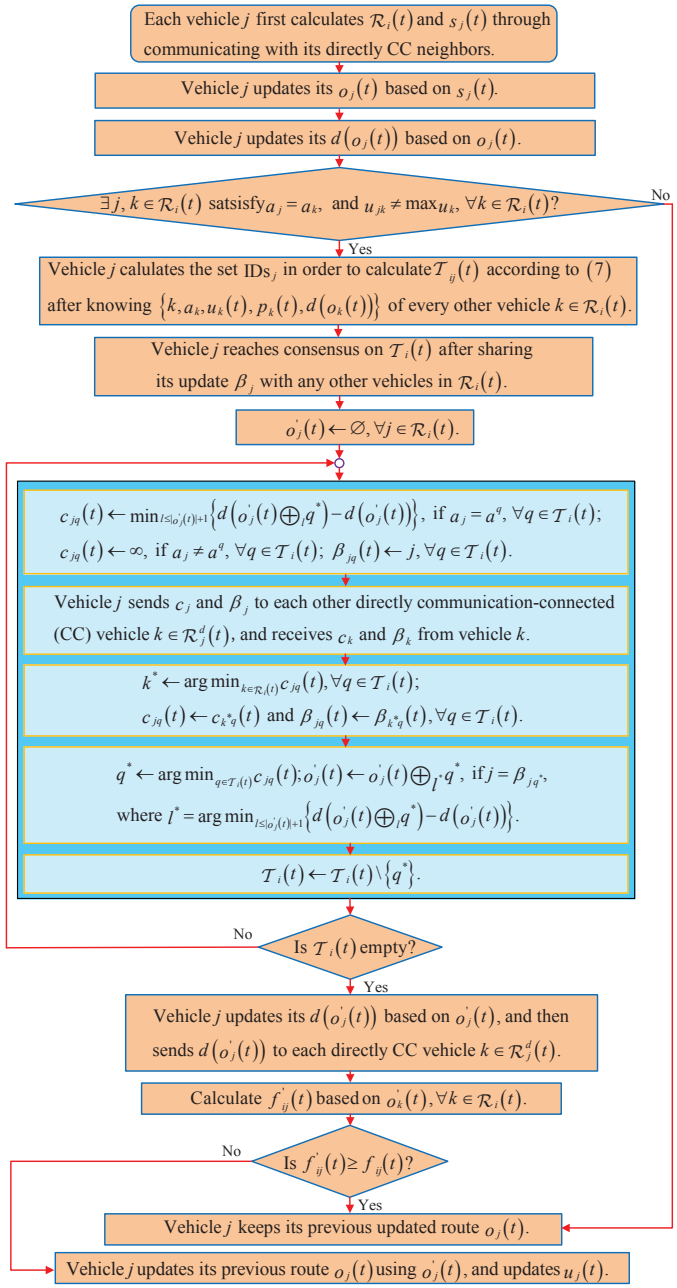

Figure 4.2: The flowchart of the decentralized auction algorithm DAA.

shown in lines 4 to 12 of Algorithm 5. Finally, using Algorithms 6 to check whether the total projected travel distance $f'_{ij}(t)$ resulting from the assignments obtained in Phase 2 is better than the previous $f_{ij}(t)$, each vehicle $j \in \mathcal{R}_i(t)$ updates its route $o_j(t)$. The flowchart of DAA is shown in Fig. 4.2.

There are two reasons for constructing the target list for each CC vehicle from scratch in Phase 2 of DAA. Firstly, the operation in (4.6) might delete some already visited target located at the front of a target bundle, thus leading to the release of all the remaining targets on the bundle according to CBBA. That is equivalent to the construction of the target list from scratch. Secondly, the first target to be inserted into a target list is vital to the following targets' insertion as the marginal cost of inserting a target is greatly affected by those targets that are already in the target list based on (4.9).

Remark 4.2. *Though the targets in each sub-target set containing the targets with the same attribute can only be assigned to the vehicles with the corresponding ability, the multi-vehicle task assignment problem cannot be decoupled completely into several independent task assignment problems for homogeneous vehicles. The reason is that different kinds of vehicles can share information for better coordination if they are CC, and thus they are coupled through communication.*

Now we discuss in more detail why our proposed algorithm DAA works.

4.3.3 Convergence of DAA

We first prove the convergency of DAA.

Lemma 4.3. *During the operation of* DAA*, the following statements hold:*
(i) Each target $q \in \mathcal{T}$ is assigned to at least one vehicle. The assignment may change, but target q remains assigned to at least one vehicle until being visited;
(ii) For vehicle $j \in \mathcal{R}$ and target q, if $s_{jq}(t_0) = 1$ at some time t_0, then $s_{jq}(t) = 1$ for all $t \geqslant t_0$.

Proof. Based on the initialization of the route $o_j(t)$ for each vehicle j, the targets initially assigned to each vehicle are those it can visit. As each vehicle has one ability and the union of all vehicles' abilities is $\mathcal{A}$, every target in $\mathcal{T}$ is initially assigned to at least one vehicle. Thus, if a vehicle has never been CC with any other vehicle with the same ability, an arbitrary target assigned to it is on the vehicle's route unless being visited by the vehicle itself.

When several vehicles are CC, statement (i) follows from the cooperative strategy shown in (4.7). We first consider the case when $\mathcal{T}_{ij}(t) = o_p(t)$, where $p = \operatorname{argmin}_{k \in \mathcal{R}^1_{ij}} d(o_k(t))$. As vehicle $p \in \mathcal{R}^1_{ij}(t)$ has never been CC with any other vehicle with the same ability, an arbitrary target q with attribute $a^q = a_p$ satisfies $q \in o_p(t)$ if q has not been visited by vehicle p.

When $\mathcal{T}_{ij}(t) = \bigcup_{k\in\mathcal{R}^2_{ij}(t)} o_k(t)$, target q is among the cooperative targets if $q \in o_k(t)$ for at least one vehicle $k \in \mathcal{R}^2_{ij}(t)$. Otherwise, q must be on the route of at least one other vehicle, say vehicle g, who has already exchanged information with at least one of the vehicles in $\mathcal{R}^2_{ij}(t)$. As $q \in o_g(t)$, q will be among the cooperative targets of a CC network if vehicle g is in the CC network according to the analysis when $q \in o_k(t)$ and $\mathcal{T}_{ij}(t) = \bigcup_{k\in\mathcal{R}^2_{ij}(t)} o_k(t)$. If vehicle g has not been CC with other vehicles who can visit target q, q is still on the route of vehicle g until being assigned to some other vehicle or being visited by vehicle g.

Based on the above analysis, an arbitrary unvisited target q with attribute $a^q \in \mathcal{A}$ is either among the cooperative targets of a CC vehicle group $\mathcal{R}_{ij}$ or on the route of at least one vehicle g with ability $a_g = a^q$. Once the target is in the cooperative target set $\mathcal{T}_{ij}(t)$, it will be assigned among the CC vehicles until being visited. Otherwise, it will be on the route of at least one of the vehicles who can visit the target. Thus, statement (i) is proved.

Statement (ii) follows directly from the logical "or" operation of $s_j(t)$ in (4.5). It can also be explained by the fact that once a target is visited and its status is known by one vehicle, that vehicle will keep this information.

□

With these properties, we are ready to present the main result on DAA.

Proposition 4.4. *For any given $n, m \in \mathbb{N}$,* DAA *enables all the target locations to be visited in finite time.*

Proof. For vehicle j who is not initially CC with any other vehicle, it inserts all the targets it can visit into its target list $o_j(0)$ using Algorithm 3 where $\mathcal{R}_i(t) = \{j\}$ and $\mathcal{T}_i(t) = \mathcal{T}$ at $t = 0$. If the vehicle cannot communicate with any other vehicle with the same capability during its movement, it will keep moving until all the targets on its route have been visited.

If the vehicle can communicate with other vehicles with the same ability, the cooperative targets determined by (4.7) are assigned among the CC vehicles. Once a target is assigned to a vehicle, it will be kept assigned until being visited based on (ii) of Lemma 4.3. Thus, all the targets will be visited in finite time.

□

Formally, we say that a function $f(x)$ is $O(g(x))$ if there are constants c and x' such that $f(x) \leqslant cg(x)$ for all $x \geqslant x'$.

Theorem 4.5. *For each newly CC vehicle group, the following statements on the assignment obtained by* DAA *hold:*
(i) The assignment converges to that of CAA*;*
(ii) At most $(n + 3m + nm)$ synchronized iterations of communication are needed to

achieve the assignment;
(iii) The computational running time complexity of DAA *on calculating the assignment is* $O(n^2(n+m^2))$*;*
(iv) The space complexity of DAA *is* $O(n^2 \log n)$.

Proof. For statement (i), the proof is carried out by induction. The decentralized DAA and the centralized CAA in essence use the same smallest marginal cost strategy to assign the targets: in CAA a leader vehicle of each CC subnetwork assigns the targets among the CC vehicles after collecting all the necessary information as shown in Algorithm 3; in DAA each of the CC vehicles bids for a target by calculating its smallest marginal cost for inserting the target and then relays the bid to every other CC vehicle. To prove the convergence of the decentralized DAA to the centralized CAA, it suffices to prove that DAA has the same assignment with that of CAA when the initial communication network topologies for the two algorithms are the same. That is to say, if DAA is proven to converge to CAA under the same initial communication network topology, the assignment resulting from DAA will converge to that from CAA for an arbitrary initial communication network. Thus, we only give the proof when all the vehicles are initially CC. After operating Phase 1 shown in Algorithm 4, the vehicles guided by DAA start bidding on the first target to be assigned among all the targets in $\mathcal{T}$. In Phase 2 shown in Algorithm 5, each vehicle updates its information tuple based on the information of those vehicles within its communication range. According to lines 3 to 14 and 19 to 23 of Algorithm 5, each vehicle in $\mathcal{R}$ bids on every target in $\mathcal{T}$ and the final marginal inserting cost of each target is the smallest bid by the vehicles. Then, the target with the smallest marginal inserting cost will be assigned to the corresponding vehicle according to lines 26 to 34 in Algorithm 5. That process is in fact the same as the target election operation and inserting operation in lines 17 to 23 of CAA shown in Algorithm 3; thus, the first targets assigned by CAA and DAA are the same.

Now suppose the first $n-1$ targets have the same assignments by CAA and DAA. The last target, named q, will be assigned to the vehicle with the smallest insertion marginal cost according to lines 17 to 23 of CAA shown in Algorithm 3. On the other hand, in lines 3 to 14 of Phase 2 of DAA shown in Algorithm 5, every vehicle j evaluates the marginal cost of inserting q, and updates its c_{jq} to be the smallest one after communicating with vehicle k satisfying $d(p_j(t), p_k(t)) \leqslant r$ as shown in lines 17 to 23 of Algorithm 5. It will cost at most m rounds of synchronized communication to make c_{jq} reach consensus. At this time, all vehicles in $\mathcal{R}_i(t)$ agree that target q is assigned to the vehicle with the smallest insertion marginal cost based on lines 17 to 23 of Algorithm 5. Thus, the last target is assigned to the same vehicle by CAA and DAA respectively. Herein, the statement is proved.

Then, we prove that DAA produces a stable solution after at most $(n+3m+nm)$ synchronized iterations of communication. There are n targets to be assigned to

m vehicles while each vehicle initially lacks the information on the number of the vehicles employed as they are initially randomly distributed and constrained by limited communication range. In Phase 1 of DAA shown in Algorithm 4, the information on the cooperative target set $\mathcal{T}_i(t)$ and CC vehicle set $\mathcal{R}_i(t)$ for vehicles within each $\mathcal{R}_i(t)$ reaches consensus after at most $3n$ synchronized iterations of communication as $|\mathcal{R}_i(t)| \leqslant n$. In Algorithm 4, after at most n synchronized iterations of communication, every vehicle j among each CC vehicle subgroup i knows $\mathcal{R}_i(t)$ containing the indices of the vehicles that are CC in the subgroup, and has the same $s_j(t)$ that saves the information on which targets have been known to be visited through communicating with the other vehicles within the subgroup as shown in lines 1 to 12 of Algorithm 4. Then, after deleting the targets known to have been visited from its route $o_j(t)$ by each j, at most $2|\mathcal{R}_i(t)|$ synchronized iterations of communication are needed for j to calculate the cooperative target set $\mathcal{T}_i(t)$ as shown in lines 16 to 43 of Algorithm 4. Thus, at most $(n + 2m)$ synchronized iterations of communication are needed to make $\mathcal{T}_i(t)$ and $\mathcal{R}_i(t)$ reach consensus. Then, during each $|\mathcal{R}_i(t)|$ synchronized iterations of communication as shown in lines 15 to 35 of Algorithm 5, DAA assigns one target in $\mathcal{T}_i(t)$ with the smallest marginal inserting cost among the cooperative targets to a vehicle as at most $|\mathcal{R}_i(t)|$ iterations of communication is required for all the vehicles in $\mathcal{R}_i(t)$ to reach consensus on the cost incurred to visit each target. Thus, all the targets in $\mathcal{T}_i(t)$ are assigned to the vehicles in $\mathcal{R}_i(t)$ after at most nm synchronized iterations of communication. Afterwards, to check whether the new assignment for the vehicles in $\mathcal{R}_i(t)$ is better than their previous assignment, it costs at most m synchronized communications to inform the projected travel distance $d(o'_j(t))$ to all the vehicles in $\mathcal{R}_i(t)$ according to lines 9 to 18 in Phase 3 of DAA shown in Algorithm 6. Consequently, the DAA produces a stable solution after at most $(n+3m+nm)$ synchronized iterations of communication for each new CC vehicle group.

For statement (iii), we give the computational running time complexity of DAA as follows. For the first Phase of DAA shown in Algorithm 4, the computational running time complexity is $O(n^2m)$ which is mainly used for updating $s_j(t)$ in line 5 of the algorithm. In Phase 2 of DAA shown in Algorithm 5, updating $c_j(t)$ costs most of the computational running time; in fact in the worst case, the maximum computational running time of $O(n^3)$ is required to update $c_j(t)$ in lines 6 and 8, and the maximum computational running time of $O(n^2m^2)$ is required to update $c_j(t)$ in line 21 of the algorithm. Finally, in the Phase 3 of DAA shown in Algorithm 6, $O(n)$ running time is consumed to calculate $d(o'_j(t))$ for each $j \in \mathcal{R}_i(t)$. Then the total computational running time complexity of DAA is $O(n^2(n + m^2))$ where the running time is dominated by Phase 2. Thus, the proof is complete.

Now we proceed to analyze the space complexity of DAA. For each vehicle $j, j \in \mathcal{R}$, as shown in Algorithm 4, it requires to store a matrix $\mathcal{X}_j \in \mathbb{R}^{n*(n+4)}$ to

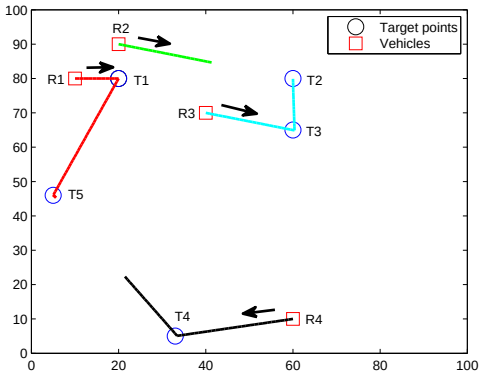

Figure 4.3: The routes for 4 homogeneous vehicles to visit 5 homogeneous targets based on DAA, where the vehicles' communication range is 28m and the arrows are the travel directions of the routes. The total travel distance f is 156m.

record the information of the vehicles CC with j where at most $(n+4)\log n$ bytes of computer space are needed to store the information on each row of $\mathcal{X}_j$ if each variable of $\mathcal{X}_j$ is coded with logarithm to base 10. Then, at most $O(n^2 \log n)$ bytes of computer storage are needed for running DAA. Thus, the proof is complete.

□

We further investigate how the length of the communication range of the vehicles affects the performance of DAA.

Remark 4.6. *For the task assignment problem, a longer communication range of the vehicles does not necessarily lead to a better performance for* DAA.

We now illustrate the main argument for this remark. When several vehicles are CC, they exchange and update their information tuples to adjust their routes. As some targets may have already been visited by vehicles who are not CC, the target information shared by the CC vehicles is local or even outdated, which fails to reflect the targets' current situation. As a result, the incomplete information shared by the CC vehicles can lead to inefficient task assignments; for example, a target already visited is still on the route of some vehicle.

One illustrating case of the task assignment resulting from DAA is shown in Fig. 4.3 and Fig. 4.4 where 4 homogeneous vehicles with unit speed need to visit 5 homogeneous targets $\mathcal{T} = \{1, \cdots, 5\}$ in a 100×100 m^2 area. In Fig. 4.3, the vehicles' communication range is 28m, which makes vehicles 1 and 2 initially CC. The routes of the four vehicles are $p_1(0) \to 1 \to 5 \to 4$, $p_2(0) \to 2 \to 3$, $p_3(0) \to 3 \to 2 \to 1 \to 5 \to 4$, $p_4(0) \to 4 \to 5 \to 1 \to 2 \to 3$, where $p_i(0), i \in \{1, \cdots, 4\}$, is the index of vehicle i' initial position. With the movement of vehicle 3, target

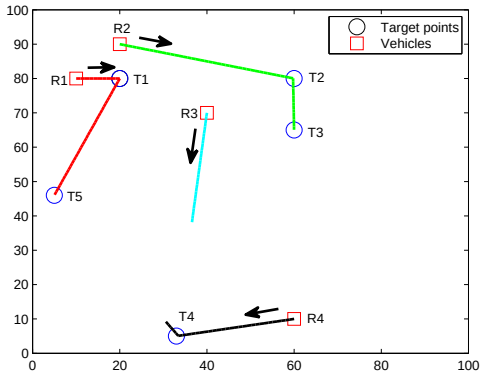

Figure 4.4: The routes for 4 homogeneous vehicles to visit 5 homogeneous targets based on DAA, where the vehicles' communication range is 30m and the arrows are the travel directions of the routes. The total travel distance f is 167m.

3 is first visited, and then vehicle 2 stops moving once it can communicate with vehicle 3 since vehicle 3 is nearer to their cooperative target 2. Reaching target 2, vehicle 3 stops moving since $o_3(t)$ is empty. When vehicle 1 can communicate with vehicle 4, targets 4 and 5 have been visited and then the two vehicles stop moving as their cooperative target set is empty. The total travel distance of the vehicles is 156m, where the travel distances of the four vehicles are 49m, 22m, 36m and 49m respectively.

In Fig. 4.4, the vehicles' communication range is increased to 30m, which makes vehicles 1, 2 and 3 initially CC. After dividing the cooperative targets among the CC vehicles based on DAA, we get the routes of the four vehicles $p_1(0) \to 1 \to 5$, $p_2(0) \to 2 \to 3$, $p_3(0) \to 4$, and $p_4(0) \to 4 \to 5 \to 1 \to 2 \to 3$. With the movement of the vehicles, target 4 is first visited by vehicle 4. Then, vehicle 3 should not move to target 4 as target 4 is already visited. However, vehicle 3 continues moving towards target 4 since it lacks the latest target status until becoming CC with vehicle 4. Afterwards, vehicle 3 and vehicle 4 stop moving since their cooperative target set is empty based on (4.7). Vehicles 1 and 2 stop moving when their assigned targets are visited. The total travel distance to visit all the targets is 167m, where the travel distances of the four vehicles are 47m, 56m, 32m and 32m respectively. The performance of DAA on vehicles under a longer communication range in Fig. 4.4 is worse than that shown in Fig. 4.3, thus illustrating Remark 4.6.

It should be noted that Remark 4.6 holds for some other task assignment algorithms as in [5] due to the vehicles' limited communication range. In the following subsection, we discuss the worst performance guarantee of DAA.

Algorithm 7 The Prim algorithm [62].

Input: Positions of targets in $\mathcal{T}$ and vehicles in $\mathcal{R}$.
Output: A minimum spanning tree (MST) of the graph $\mathcal{G}$.
1: Initialize $MST \leftarrow \mathcal{R}$.
2: **while** $\mathcal{T} \neq \varnothing$ **do**
3: $(j^\star, p^\star) \leftarrow \text{argmin}_{(j,p)\in MST\times\mathcal{T}}\, d(j,p)$.
4: Add $p^\star$ in MST and connect it with $j^\star$ using an edge with length $d(j^\star, p^\star)$ equating to the distance between the two vertices.
5: $\mathcal{T} \leftarrow \mathcal{T} \setminus \{p^\star\}$.
6: **end while**

4.3.4 Worst performance analysis

For vehicles in each CC vehicle set $\mathcal{R}_i(t)$, they update their route information to minimize the total travel distance of the connected vehicles.

Lemma 4.7. *Each information updating for the CC vehicles guided by* DAA *makes the projected total travel distance f in (4.1) non-increasing.*

Proof. Guided by DAA, the CC vehicles in each $\mathcal{R}_{ij}(t)$ re-assign the cooperative targets if the task assignment reduces their total travel distance (4.14) according to lines 20 to 23 of Algorithm 6. Otherwise, each vehicle j in $\mathcal{R}_{ij}(t)$ only updates $s_j(t)$, $o_j(t)$, and the corresponding $d(o_j(t))$ based on their shared information, which makes the projected total travel distance $f_{ij}(t)$ in (4.14) non-increasing. As $f_{ij}(t)$ is a component of f in (4.1), f is non-increasing for each information updating.

□

The analysis of the performance of the proposed DAA, as indicated by Lemmas 4.3, 4.7, Proposition 4.4, Theorem 4.5 and Remark 4.6, still applies when the network is not initially connected.

Let $\mathcal{G}$ be a vehicle-target graph whose vertices contain all the vehicles' initial positions and the target locations. The weight for an edge connecting a vehicle vertex with target vertex is the Euclidean distance between them if the vehicle has the ability to visit the target, and is otherwise positively infinite. We assign zero weight for the edges between each two vertices representing vehicles. We apply the Prim algorithm [62], shown in Algorithm 7, to obtain a minimum spanning tree (MST) of $\mathcal{G}$ that connects all the targets and the vehicles with the minimum total edge weight, and use f_{MST} to be the sum of all the edge weights of the MST. Let f_o be the optimal value for the objective function in (4.1) which is obtained based on all necessary global information, and f_{DAA} be the objective function in (4.1) resulting from DAA.

Lemma 4.8. *The optimal total travel distance f_o is lower bounded by f_{MST}.*

Proof. When $|\mathcal{A}| = 1$, each vehicle can visit every target, thus leading $\mathcal{G}$ to be a complete graph. If all the vehicles are initially CC, the task assignment problem is in fact the one investigated in [67] without considering the vehicles' minimum turning radius. Based on [62, 67], we get $f_{MST} \leqslant f_o$ when $|\mathcal{A}| = 1$.

When $|\mathcal{A}| > 1$, targets with the same attribute can be assigned to vehicles with the corresponding ability. Thus, all the targets are first clustered to subgroups where each subgroup contains the targets with the same attribute. For each target cluster j, the vehicles with the corresponding ability to visit the targets are also included. An MST_j of the graph $\mathcal{G}_j$ containing all targets and the assigned vehicles in each cluster j can be constructed, where the total travel distance of the assigned vehicles is lower bounded by f_{MST_j} [62]. Thus, the total travel distance of all the vehicles is lower bounded by $\sum_j f_{MST_j} = f_{MST}$. Hence, the optimal total travel distance is lower bounded by f_{MST}.

□

From Lemma 4.8, we further guarantee the performance of DAA as follows.

Theorem 4.9. *When all the vehicles are initially CC,* DAA *guarantees the total travel distance* $f_{\text{DAA}} \leqslant 2f_o$.

Proof. Theorem 4.5 shows that the assignment of DAA converges to that of CAA after at most $(n + 3m + nm)$ synchronized iterations of communications. Thus, it suffices to prove the statement through showing the performance of CAA.

The proof is conducted by induction. When all the vehicles are initially CC, the targets are assigned by CAA according to line 18 of Algorithm 3 shown as

$$(j^\star, q^\star) \leftarrow \underset{(j,q)\in\mathcal{R}\times\mathcal{T}^u}{\operatorname{argmin}} c_{jq}(t), \tag{4.16}$$

where the target set $\mathcal{T}^u$ contains the unassigned targets and $c_{jq}(t)$ is determined by (4.9). It is straightforward to find the first target chosen by CAA, which is the same as the first target inserted in the MST according to line 3 of the Prim algorithm shown in Algorithm 7. Thus, $f^1_{\text{CAA}} \leqslant 2f^1_{MST}$ as $f^1_{\text{CAA}} = f^1_{MST}$, where the upper superscript 1 denotes the total projected travel distance after assigning the first target.

Now suppose the first $n-1$ targets assigned by CAA and inserted in the MST are the same and $f^{n-1}_{\text{CAA}} \leqslant 2f^{n-1}_{MST}$. As triangular inequality exists for the Euclidean distances between the targets and from (4.9), (4.10) and (4.16), the marginal

travel distance incurred by assigning the last target q through CAA is

$$\begin{aligned}
\delta f_{\text{CAA}} &= \min_{(j,q)\in\mathcal{R}\times\mathcal{T}^u} c_{jq}(t) \\
&= \min_{j\in\mathcal{R}, l\leqslant|o_j(t)|+1}\{d(o_j(t)\oplus_l q) - d(o_j(t))\} \\
&= \min_{j\in\mathcal{R}}\{\min_{l\leqslant|o_j|-1}\{d(o_j^l,q)+d(q,o_j^{l+1})-d(o_j^l,o_j^{l+1})\}, \qquad (4.17)\\
&\quad d(p_j(0),q)+d(q,o_j^1)-d(p_j(0),o_j^1), d(o_j^{|o_j(t)|},q)\} \\
&\leqslant \min_{j\in\mathcal{R}, l\leqslant|o_j|-1}\{2d(o_j^l,q), 2d(q,o_j^{l+1})\}, \qquad (4.18)
\end{aligned}$$

where o_j^l is the lth target on the ordered target list $o_j(t)$ and $t=0$. On the other hand, according to line 3 of Algorithm 7, the marginal edge weight incurred by inserting the last target q into the MST is

$$\begin{aligned}
\delta f_{MST} &= \min_{(j,q)\in MST\times\{q\}} d(j,q) \\
&= \min_{j\in\mathcal{R}, l\leqslant|o_j|-1}\{d(o_j^l,q), d(q,o_j^{l+1})\}. \qquad (4.19)
\end{aligned}$$

Combining (4.17) and (4.19) and in view of the fact $f_{\text{CAA}}^{n-1}\leqslant 2f_{MST}^{n-1}$, we get

$$\begin{aligned}
f_{\text{CAA}}^n &= f_{\text{CAA}}^{n-1}+\delta f_{\text{CAA}} \\
&\leqslant 2f_{MST}^{n-1}+2\delta f_{MST}. \qquad (4.20)
\end{aligned}$$

Since $f_{MST}^n = f_{MST}^{n-1}+\delta f_{MST}$, it holds that $f_{\text{CAA}}^n\leqslant 2f_{MST}^n$. Thus, combining $f_{\text{CAA}}^n\leqslant 2f_{MST}^n$ and $f_{MST}^n\leqslant f_o$ in view of Lemma 4.8, we arrive at the conclusion. □

In addition, we investigate the worst performance of DAA with an arbitrary initial communication network of the vehicles. After initializing the vehicles' routes at time $t=0$, for each vehicle $j\in\mathcal{R}$, let $d(o_j(0))$ be the projected total distance for vehicle j to visit all the targets on its route $o_j(0)$ if at least one target is on the route, and otherwise $d(o_j(0))=0$.

Proposition 4.10. *With an arbitrary initial communication network of the vehicles,* DAA *guarantees that all the target locations will be visited with the vehicles' total travel distance* $f_{\text{DAA}}\leq\sum_{j\in\mathcal{R}} d(o_j(0))$.

Proof. According to Lemma 4.7, the total projected total travel distance of the vehicles guided by DAA in each CC subnetwork is non-increasing after the information updating for the CC vehicles. Thus, the worst total travel distance of the vehicles is $\sum_{j\in\mathcal{R}} d(o_j(0))$. One of the scenarios leading to the worst total travel distance is the case when each of the vehicles visits all the target locations initially

assigned to it without being able to communicate with any other vehicles during its movement. Thus, the proof is complete.

□

Remark 4.11. *The proposed* DAA *can also be used to optimize other objectives such as the time when the last target is visited or the total waiting time until all the targets are visited by the vehicles. However, the performance of the algorithm needs to be further investigated for optimizing other objectives.*

Now we have presented all the theoretical results, and are ready to show the simulation findings.

4.4 Simulations

In this section, Monte Carlo simulations are carried out to test the decentralized algorithm DAA. We first employ DAA on the task assignment problem for homogeneous vehicles, followed by various simulations on the scenarios with increasing targets' heterogeneity $|\mathcal{A}|$, where $|\mathcal{A}|$ is the cardinality of $\mathcal{A}$. All the experiments have been performed on an Intel Core $i5-$ 4590 CPU 3.30 GHz with 8 GB RAM, with algorithms compiled by Matlab under Windows 7.

The solution quality of each algorithm is defined by

$$q = \frac{f}{f_{MST}}, \tag{4.21}$$

where f is the total travel distance for the vehicles to visit all the targets resulting from minimizing (4.1) and f_{MST} is a lower bound of the optimal total travel distance as shown in Lemma 4.8. Then, the closer the value of q becomes to 1 the better performance the solution obtains.

4.4.1 Task assignment for homogeneous vehicles

The algorithm DAA is first tested on the task assignment for homogeneous vehicles under different communication ranges. Two sets of simulations have been performed in a 1000×1000 m^2 area where the numbers of homogeneous target points and vehicles are 15 and 4, 30 and 6 respectively. For each instance, Monte Carlo simulations have been carried out on 500 scenarios where the positions of the targets and the vehicles are randomly generated. The mean solution results of DAA are shown in Fig. 4.5 and Fig. 4.6 with the comparison with the rendezvous-based algorithm (RBA), the decentralized single-traveling-salesman tour cooperative method (STSTC) and the Greedy Algorithm in [5]. For each instance, the mean q of DAA shown in the two figures is the smallest compared with that of the

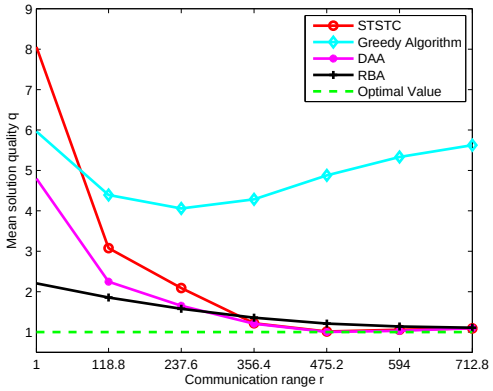

Figure 4.5: The mean solution quality of different algorithms for 4 homogeneous vehicles to visit 15 targets under different communication ranges where $r_c = 712.8m$.

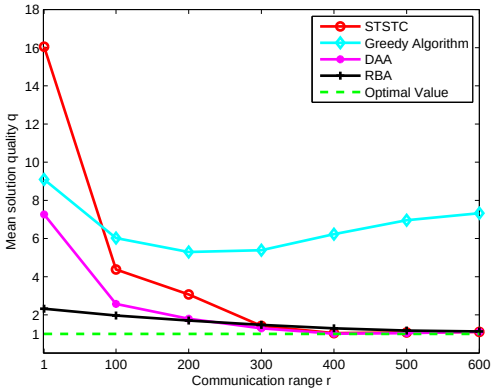

Figure 4.6: The mean solution quality of different algorithms for 6 homogeneous vehicles to visit 30 targets under different communication ranges where $r_c = 600$m.

STSTC and the Greedy Algorithm, which shows the better performance of DAA. Furthermore, DAA outperforms the centralized RBA when the vehicles' communication range is greater than $r_c/2$ where $r_c = 712.8$m leads to the connection of the communication network with probability of 0.99 [65]. This is because the vehicles guided by the RBA need to first rendezvous at the center-of-gravity of the target points to make them CC [5], and thus leading to a waste of travel distance. Finally, the total travel distance of the assignment obtained by DAA is below twice of the optimal one when the vehicles' communication range is greater than $r_c/3$.

Table 4.1: The mean solution quality q of HDA and DAA for $m = 6$ vehicles with different heterogeneities $|\mathcal{A}|$ to visit $n = 30$ target locations under increasing communication range r, where $r_c = 600$m.

q	Algorithm	$\|A\| = 2$	$\|A\| = 3$	$\|A\| = 4$	$\|A\| = 5$	$\|A\| = 6$
$r = 1$m	HDA	3.4717	2.3407	1.7132	1.3638	1.1351
	DAA	3.4717	2.3407	1.7132	1.3638	1.1351
$r = 1 * r_c/6$	HDA	1.9504	1.6221	1.4022	1.2495	1.1351
	DAA	1.9274	1.6009	1.3919	1.2474	1.1351
$r = 2 * r_c/6$	HDA	1.5582	1.4016	1.2855	1.2048	1.1351
	DAA	1.5170	1.3682	1.2578	1.1942	1.1351
$r = 3 * r_c/6$	HDA	1.3126	1.2393	1.1990	1.1658	1.1351
	DAA	1.2624	1.2111	1.1744	1.1555	1.1351
$r = 4 * r_c/6$	HDA	1.1878	1.1727	1.1565	1.1453	1.1351
	DAA	1.1631	1.1601	1.1495	1.1435	1.1351
$r = 5 * r_c/6$	HDA	1.1507	1.1529	1.1442	1.1406	1.1351
	DAA	1.1467	1.1493	1.1427	1.1404	1.1351
$r = 6 * r_c/6$	HDA	1.1453	1.1492	1.1417	1.1398	1.1351
	DAA	1.1437	1.1483	1.1415	1.1399	1.1351

That also verifies the performance bound guarantee of DAA shown in Theorem 2 when the communication network of the vehicles is connected, i.e. $r = r_c$.

4.4.2 Task assignment for heterogeneous vehicles

In this section, we first run DAA on the task assignment problem where $m = 6$ dispersed vehicles under different communication ranges r need to visit $n = 30$ target locations with increasing heterogeneity $|\mathcal{A}|$. Due to the satisfying performance on guiding vehicles for search and rescue task assignment compared with that of CBBA, the heuristic distributed algorithm (HDA) in [83] is compared with DAA for heterogeneous task assignment of multiple vehicles. The HDA uses the marginal significance incurred by inserting each target at all possible positions of the vehicles' routes to assign the targets, and defines the significance of a target by removing it from its corresponding position on a vehicle route containing the target. DAA makes task assignment based on the minimal incurred cost of assigning each unassigned target, which differs from HDA as follows. DAA constructs the routes of the CC vehicles from scratch by assigning each target to the vehicle with the smallest incurred cost, while HDA makes target reassignment directly based on the current routes of the CC vehicles where a target is removed from its current position on a vehicle's route, and is assigned to another vehicle if the marginal significance of inserting the target into the route of the latter vehicle is smaller than

Table 4.2: The mean amount of data (10^5 bytes) relayed for $m = 6$ vehicles with different heterogeneities $|\mathcal{A}|$ guided by HDA, DAA and CAA to visit $n = 30$ target locations under increasing communication range r, where $r_c = 600$m.

Data relayed	Algorithm	$\|A\| = 2$	$\|A\| = 3$	$\|A\| = 4$	$\|A\| = 5$	$\|A\| = 6$
$r = 1 * r_c/6$	HDA	2.6525	1.7352	1.0261	0.4747	4.3152
	DAA	4.7297	4.8322	4.5819	4.4393	4.3152
	CAA	12.8744	13.2027	12.5462	12.1706	11.9303
$r = 2 * r_c/6$	HDA	2.8770	1.9481	1.1728	0.5635	6.4860
	DAA	5.9341	6.5702	6.9101	7.1652	6.4860
	CAA	16.0655	17.8692	18.8337	19.5528	17.9318
$r = 3 * r_c/6$	HDA	3.2648	2.1329	1.3397	0.7086	4.7002
	DAA	6.1134	6.5526	6.5935	6.7829	4.7002
	CAA	16.3616	17.6188	17.7728	18.3142	12.9946
$r = 4 * r_c/6$	HDA	3.4039	2.2532	1.5286	1.0182	2.3446
	DAA	4.6646	4.6149	4.5518	4.4915	2.3446
	CAA	12.2805	12.1993	12.0843	11.9489	6.4821
$r = 5 * r_c/6$	HDA	2.8228	2.1607	1.6917	1.3769	0.9829
	DAA	3.0691	3.0310	2.9442	2.9699	0.9829
	CAA	7.9659	7.8915	7.6851	7.7764	2.7174
$r = 6 * r_c/6$	HDA	2.2226	1.9920	1.8188	1.7053	0.2772
	DAA	2.2593	2.2576	2.2699	2.2590	0.2772
	CAA	5.8137	5.8171	5.8611	5.8391	0.7665

the target's significance when located on the route of the former vehicle. Because of the assumption that vehicles move with the unit speed, the minimization of the total travel distance is equivalent to the the minimization of the total travel time. Thus, it is reasonable to compare DAA with HDA.

For vehicles under each communication range in $\{1, r_c/6, 2r_c/6, ..., 6r_c/6\}$ and each $|\mathcal{A}|$ in $\{2, 3, ..., m\}$, Monte Carlo simulations of DAA and HDA are carried out on 500 scenarios where the locations of the targets and vehicles, the attribute of each target and the ability of each vehicle are randomly generated. The mean solution quality q of DAA and HDA and the mean amount of data (bytes) exchanged through the communication among all the vehicles guided by HDA, DAA and CAA are shown in Table 4.1 and Table 4.2, respectively. Firstly, the most of q shown in Table 4.1 are within twice of the optimal, which shows the satisfying performance of DAA and HDA. Secondly, the table shows that for each $|\mathcal{A}|$ a longer communication range of the vehicles generally leads to a better performance of both DAA and HDA except for the case when $|\mathcal{A}| = m$. The reason is that the vehicles are too heterogeneous to have any cooperation regardless of their communication ranges

as every two vehicles are different when $|\mathcal{A}| = m$. Furthermore, for the vehicles under each communication range r a larger $|\mathcal{A}|$ usually results in a smaller q for DAA and HDA. This is because a large $|\mathcal{A}|$ means that a small number of vehicles can cooperate as vehicles with different capabilities are independent, where less cooperation leads to less dependency on the communication. As a consequence, DAA and HDA are less sensitive to the vehicles' communication range when increasing $|\mathcal{A}|$. However, in Table 4.1 DAA has the smaller average q compared with that of HDA for the majority of the instances while they perform equally well when $r = 1$m and $|\mathcal{A}| = m$ (the cells colored in light gray in Table 4.1). When $r = 1$m or $|\mathcal{A}| = m$, the vehicles rarely cooperate as they either have short communication range or are too heterogeneous to have any cooperation. As DAA and HDA assign the targets both relying on the minimal incurred cost/significance of each target, they have the same initial target assignments when $r = 1$m or $|\mathcal{A}| = m$. If the vehicles have no cooperation during their later movement as when $r = 1$m or $|\mathcal{A}| = m$, the total travel distances of the vehicles guided by DAA and HDA are the same. The only case where HDA has a slightly better performance than DAA is when $r = 6 * r_c/6$ and $|\mathcal{A}| = 5$ (the cell colored in saturate gray). The better performance of DAA over HDA is due to the fact that the CC vehicles guided by DAA exchange and store all the available bids on every target instead of only on the targets that can be visited by the vehicles, where the latter might lead to local optimal as noted in [83].

Firstly, Table 4.2 shows that the mean amount of data relayed among the vehicles guided by the centralized CAA is the largest compared with those of decentralized HDA and DAA where DAA has a higher communication overhead than HDA. The reason is that the centralized CAA requires each CC vehicle to transmit almost all information on its information tuple to a leader vehicle to assign the unvisited targets to the CC vehicles, while each CC vehicle guided by the decentralized DAA updates its route through relaying its updated information to the directly CC vehicles and thus exchanging a smaller amount of information each time as shown in Algorithm 3 to Algorithm 6. The vehicles guided by DAA communicate more than those under HDA because each of the CC vehicles exchanges and storages the necessary information of each other CC vehicles to make the heterogeneous vehicles cooperative, which leads to the better performance of DAA as shown in Table 4.1. Secondly, the communication cost of the vehicles under each heterogeneity $|\mathcal{A}|$ for the algorithms generally first increases and then decreases as the vehicles' communication range r increases. The initial increase of the communication cost is due to the fact that a longer r enables each vehicle to communicate with the others with a higher probability, thus leading to the delivery of more data. However, when r increases above a certain range, a larger number of vehicles might be CC initially and the CC vehicles might not further communicate during their following movement as they have already cooperatively divided the unvisited tar-

gets through the initial communication. Furthermore, the communication cost of the vehicles under each r for the decentralized HDA and DAA generally decreases with the increase of the vehicles' heterogeneity $|\mathcal{A}|$ except for when $|\mathcal{A}| = m$. That is because part of the massages will be relayed between the CC vehicles for task reassignment only when some of the vehicles in $\mathcal{R}_i(t)$ are homogeneous as noted in line 15 of Algorithm 4 where a larger $|\mathcal{A}|$ implies a higher heterogeneity of the vehicles. However, when $|\mathcal{A}| = m$, still a certain amount of massages are relayed between the vehicles in each CC vehicle subgroup to check the personal information of each vehicle such as the vehicle's attribute and ID as shown in lines 1 to 12 in Algorithm 4. As the vehicles are completely heterogeneous when $|\mathcal{A}| = m$, each vehicle will move along its initial route to visit all the target locations on its route, which leads to a long travel time of the vehicle. During the movement of the vehicles under the long travel time, the vehicles in each newly CC vehicle subgroup communicate to check their personal information, which results in a larger communication cost compared with other $|\mathcal{A}|$.

Table 4.3: The mean solution quality of HDA and DAA for $m = 10$ vehicles with different heterogeneities $|\mathcal{A}|$ to visit $n = 100$ target locations under increasing communication range r, where $r_c = 482$m.

q	Algorithm	$\|A\| = 2$	$\|A\| = 4$	$\|A\| = 6$	$\|A\| = 8$	$\|A\| = 10$
$r = 1$m	HDA	5.8170	2.9868	1.9501	1.4592	1.1807
	DAA	5.8170	2.9868	1.9501	1.4592	1.1807
$r = 1 * r_c/6$	HDA	2.3779	1.8156	1.5330	1.3436	1.1807
	DAA	2.2680	1.7458	1.4723	1.3012	1.1807
$r = 2 * r_c/6$	HDA	1.7714	1.5280	1.4125	1.3049	1.1807
	DAA	1.6698	1.4533	1.3264	1.2453	1.1807
$r = 3 * r_c/6$	HDA	1.4318	1.3296	1.3124	1.2669	1.1807
	DAA	1.3292	1.2577	1.2254	1.1979	1.1807
$r = 4 * r_c/6$	HDA	1.2753	1.2289	1.2564	1.2349	1.1807
	DAA	1.1915	1.1953	1.1949	1.1876	1.1807
$r = 5 * r_c/6$	HDA	1.2071	1.1953	1.2203	1.2109	1.1807
	DAA	1.1808	1.1874	1.1912	1.1857	1.1807
$r = 6 * r_c/6$	HDA	1.1842	1.1869	1.1987	1.1937	1.1807
	DAA	1.1791	1.1866	1.1903	1.1852	1.1807

We also apply DAA and HDA for $m = 10$ vehicles under different communication ranges to visit $n = 100$ targets with different attributes $|\mathcal{A}|$. For each instance, Monte Carlo simulations of DAA and HDA are carried out on 500 scenarios where the locations of the targets and vehicles, the attribute of each target and the ability of each vehicle are randomly generated. The mean solution quality q of DAA

Table 4.4: The mean amount of data (10^7 bytes) relayed for $m = 10$ vehicles with different heterogeneities $|\mathcal{A}|$ guided by HDA, DAA and CAA to visit $n = 100$ target locations under increasing communication range r, where $r_c = 482$m.

Data relayed	Algorithm	$\|A\| = 2$	$\|A\| = 4$	$\|A\| = 6$	$\|A\| = 8$	$\|A\| = 10$
$r = 1 * r_c/6$	HDA	2.2190	1.2865	0.6521	0.2486	4.5028
	DAA	4.0723	4.5413	4.6833	4.6830	4.5028
	CAA	11.8651	13.2538	13.6751	13.6770	13.1620
$r = 2 * r_c/6$	HDA	2.5132	1.4259	0.7438	0.2891	7.8799
	DAA	5.0384	6.4732	7.4261	7.8358	7.8799
	CAA	14.6571	18.8778	21.6728	22.8754	23.0335
$r = 3 * r_c/6$	HDA	2.9765	1.6363	0.8263	0.3299	6.2388
	DAA	5.9080	6.9251	7.3767	7.2563	6.2388
	CAA	17.1243	20.1394	21.4819	21.1377	18.2364
$r = 4 * r_c/6$	HDA	2.9632	1.6985	0.9149	0.5127	2.6697
	DAA	4.1055	4.1039	3.9735	3.7931	2.6697
	CAA	11.8341	11.8657	11.5125	11.0012	7.8037
$r = 5 * r_c/6$	HDA	2.3378	1.4852	0.9968	0.7475	1.0360
	DAA	2.2477	2.1291	2.1393	2.1852	1.0360
	CAA	6.4515	6.1238	6.1663	6.3086	3.0284
$r = 6 * r_c/6$	HDA	1.6291	1.2293	1.0247	0.9181	0.3432
	DAA	1.3883	1.3859	1.3708	1.3735	0.3432
	CAA	3.9745	3.9715	3.9325	3.9435	1.0032

and HDA and the mean amount of data (bytes) exchanged through the communication among all the vehicles guided by HDA, DAA and CAA are shown in Table 4.3 and Table 4.4, respectively, where the cells colored in light gray in Table 4.3 are the cases when DAA and HDA perform equally well while DAA outperform HDA in the other instances. Table 4.3 shows the same trend on the performance of DAA and HDA as displayed in Table 4.1, which shows the robustness of DAA. However, the q in Table 4.3 are relatively larger than that in Table 4.1 especially when $r = 1$m. The reason is that a short communication range leads to a less cooperation between the CC vehicles where the vehicles travel with more inefficient routes if the numbers of the target locations and vehicles are big. Table 4.4 shows the same changing trend on the communication cost of HDA, DAA and CAA as that displayed in Table 4.2.

To further investigate the scalability of DAA, we test it on a setup where 10 vehicles need to visit a large number of targets ranging from 100 to 600. Experiments on 500 scenarios have been carried on each instance where the locations of the targets and vehicles, the attribute of each target and the ability of each vehicle

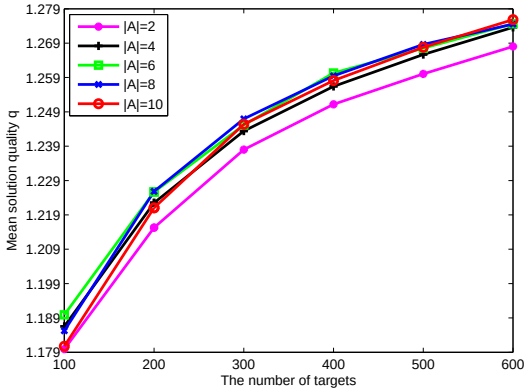

Figure 4.7: The mean solution quality q of DAA for 10 vehicles with different heterogeneities $|\mathcal{A}|$ to visit different numbers of target locations, where the vehicles are initially CC.

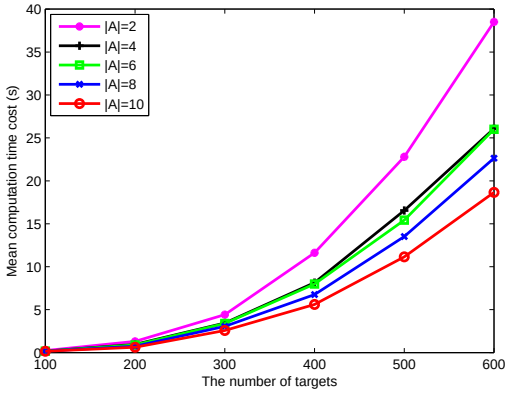

Figure 4.8: The corresponding mean computation time for DAA to calculate the solutions for 10 vehicles visiting different numbers of target locations under different heterogeneities $|\mathcal{A}|$, where the vehicles are initially CC.

are randomly generated, and all the vehicles are assumed initially CC. The mean solution quality q and the corresponding mean computation time are shown in Fig. 4.7 and Fig. 4.8 respectively. For each $|\mathcal{A}|$, the q shown in Fig. 4.7 increases as increasing the number of the target locations. For a given number of target locations, Fig. 4.7 first shows that the mean q generally increases when increasing the targets' heterogeneity $|\mathcal{A}|$. The reason is that a small number of vehicles work together when they have a large $|\mathcal{A}|$ as the vehicles with different capabilities are independent. However, the means q of each instance are within the interval

$[1.169\ 1.279]$, which verifies the satisfying performance of DAA shown in Theorem 4.9. For each $|\mathcal{A}|$, the mean computation time shown in Fig. 4.8 increases as increasing the number of the target locations. The reason is that a large number of the same kind of targets are under consideration when increasing the number of targets. Furthermore, Fig. 4.8 shows that for each number of target locations a larger $|\mathcal{A}|$ incurs a shorter running time. It is because a small $|\mathcal{A}|$ means that large numbers of vehicles and tasks are included in one search for the assignment. The figure also indicates that the computation time is affected by $|\mathcal{A}|$, where a smaller $|\mathcal{A}|$ leads to a larger computation time. The reason is that the marginal inserting cost of an unassigned target q is calculated for a vehicle j only when j has the ability to visit q, i.e. $a^q = a_j$, as shown in line 4 of Algorithm 5, where a smaller $|\mathcal{A}|$ means more vehicles generally have the same ability to visit each q and each vehicle has the ability to visit more targets. Finally, the computation time shown in Fig. 4.8 a little bit downgrades the time complexity of $O(n^2(n+m^2))$. The reason is that the computational running time of $O(n^3)$ is obtained assuming that line 4 of Algorithm 5 applies for all $q \in \mathcal{T}_i(t)$ without considering the attribute of each target. In the simulation experiments, the condition shown in line 4 of Algorithm 5 is not always satisfied as the vehicles are heterogeneous and one vehicle can generally visit a subset of the targets. And for this reason, the computation time in the simulation example is much lower than the worst-case maximum.

4.5 Conclusion

This chapter has studied the task assignment problem where several initially dispersed heterogeneous vehicles under limited communication range need to visit a set of target locations. The decentralized auction algorithm DAA proposed in the chapter has guaranteed that all the target locations are visited in a finite time irrespective of the vehicles' communication range. Furthermore, we have illustrated that a longer communication range of the vehicles does not necessarily lead to a better performance of DAA, which can also be the case for other decentralized algorithms due to the lack of overall environmental information. Finally, extensive simulation results have verified the satisfying performance of DAA, such as the total travel distance being at most twice of the optimal when the vehicles are initially CC, and scalability with growing numbers of vehicles, targets and heterogeneity.

Chapter 5

Precedence-constrained multi-vehicle task assignment

THIS chapter studies the precedence-constrained task assignment problem for a team of heterogeneous vehicles to deliver packages to a set of dispersed customers subject to precedence constraints that specify the sequence priorities on visiting the customers. A truck and a micro drone with complementary capabilities are employed where the truck is restricted to travel in a street network and the micro drone, restricted by its loading capacity and operation range, can fly from the truck to perform the last mile package deliveries. The objective is to minimize the time to serve all the customers respecting every precedence constraint. The problem is shown to be NP-hard, and a lower bound on the optimal time to serve all the customers is constructed by using tools from graph theory. Then, integrating with a topological sorting technique, several heuristic task assignment algorithms are proposed to solve the task assignment problem. Numerical simulations show the superior performances of the proposed algorithms compared with popular genetic algorithms.

5.1 Introduction

In recent years, parcel delivery to customers is facing new challenges as e-commerce has grown vastly [58] where the benefit of using micro drones as additional support for package delivery has been identified [27]. Consequently, some leading retailers or distributors such as Amazon and DHL have planned to employ micro drones for small package deliveries. However, micro drones are subject to short operation range and small payload capacity which greatly restrict their efficiency to function in an autonomous delivery network [53]. To overcome the limitations, some investigation has been done to consider a heterogeneous team consisting of one carrier truck and one micro drone with complementary capabilities [53, 60, 69, 77]. In [53], the package delivery problem for the truck and the drone has been formulated as an optimal path planning problem on a graph, and then the problem is reduced to the generalized traveling salesman problem. Marray and Chu [60] have formulated the heterogeneous parcel delivery problem as a mixed integer linear programming problem and further investigated two

cases where one considers the release and recovery of the drone by using the truck while the other just uses the depot to release and recover the drone. Considering the drone's operation range and capacity constraint, Savuran and Karakaya [69] have designed a genetic algorithm (GA) to plan the route for the drone deployed on a mobile platform to visit a set of fixed targets. In [77], several worst-case analysis theorems have been investigated revealing the maximum amount of time that could be saved as a result of using trucks and drones in combination rather than employing trucks alone for delivering packages. In [60, 77], the truck itself is also allowed to deliver parcels to customers, which is different from [53].

For logistic scheduling, another challenge is that some customers can have priority over the others to be served due to their urgency or importance. In these cases, the precedence constraints on the visiting sequence of customers have to be respected, and the positioning of one customer in the sequence is directly affected by the customers which are required to be served earlier. Precedence constraints have been studied earlier in the so called sequence ordering problem [25], which is also referred to as the sequence problem with precedence constraints [74]. Constructing TSP tours while respecting some precedence constraints yields the precedence-constrained TSP, which was called PCTSP [42]. Considering the loading constraint of unmanned aerial vehicles and the precedence constraints on multiple visits at one target, a GA was proposed for the multi-vehicle task assignment [71]. For the TSP where a given subset of targets is required to be visited in some prescribed linear order, an algorithm guaranteeing quantifiable performances was designed [9]. Each subset of targets with the linear visiting constraints can be treated as one single vertex, thus leading to the transformation of the TSP subject to the precedence constraints in [9] into the standard TSP. A topological sorting technique was integrated with a GA to solve the TSP with precedence constraints in [56]. The topological sorting technique guarantees that the planned path is feasible while the GA uses a crossover operator, mimicking the changes of the moon, to adjust the sequence for visiting the target locations. Later on, an improved GA based on topological sorting techniques was proposed in [82] to solve precedence-constrained sequencing problems. Only one chromosome is needed by the crossover operator to undergo the crossover evolution where each chromosome constructs a feasible solution to the problem.

Motivated by the existing literature just mentioned, we investigate the precedence-constrained heterogeneous delivery problem (PCHDP) for which one drone coordinates with one truck to efficiently deliver packages to a set of dispersed customers subject to precedence constraints that specify which customers need to be visited before which other customers. We first investigate the feasible deployment patterns for the drone to travel from one customer to another in coordination with the truck, and then obtain the travel cost matrix specifying the feasible minimal time for the drone to fly between each pair of customers. Finally, integrating the

topological sorting technique [56, 82], we design several heuristic task assignment algorithms to iteratively put the customers in an ordered manner respecting the precedence constraints. Our main contributions are as follows. Firstly, we prove that the precedence-constrained task assignment problem is NP-hard, and construct a lower bound on the optimal solution by using tools from graph theory. This lower bound can be used to approximately measure the quality of a solution compared with the optimal. Secondly, inspired by [53], we have extended the feasible deployment patterns for the drone to travel between two customers with the coordination of the truck. Lastly, two heuristic algorithms designed in the chapter can obtain satisfying solutions within short computation time even when the number of customer locations is large.

The rest of this chapter is organized as follows. In Section 5.2, the formulation of the precedence-constrained package delivery for heterogeneous vehicles is given. Section 5.3 presents the problem analysis, and in Section 5.4 several task assignment algorithms are presented. We show the simulation results in Section 5.5 and conclude the chapter in Section 5.6.

5.2 Problem formulation

5.2.1 Problem setup

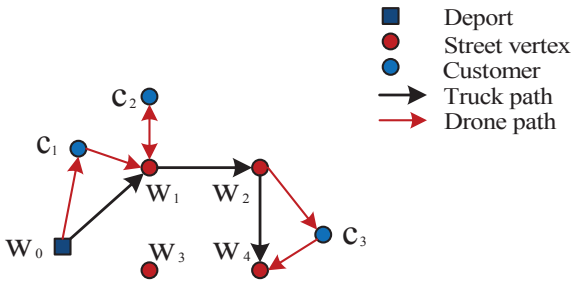

Figure 5.1: One illustration on the heterogeneous package delivery problem with one drone coordinating with one truck to deliver parcels to three dispersed customers.

Now we are ready to define the research problem PCHDP. A drone in coordination with a truck is deployed to deliver packages to a set of n dispersed customers subject to precedence constraints that specify the sequence priorities on visiting the target locations. Each customer receives one package to be delivered by the drone, and the truck is restricted to travel between a set of stopping points as vertices on a graph describing the topology of a street network. Each customer can be visited by the drone released from the truck from at least one stopping point

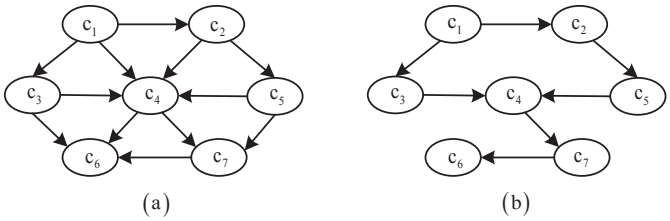

Figure 5.2: The digraph $\mathcal{G}^p = (V^p, E^p)$ shows precedence constraints on serving the customer (target) vertices (a) Digraph in [56] and (b) Simplified digraph.

vertex to ensure that all the customers can be served. The objective is to minimize the time when the last customer is served while satisfying every precedence constraint, and the constraints that the drone can carry only one package each time and has limited operation range. One illustration on the package delivery problem without any precedence constraint is shown in Fig. 5.1.

5.2.2 Formulation as an optimization problem

Let $\mathcal{C} = \{c_1, \ldots, c_n\}$ be a set of vertices representing n customer locations, and the indices of the stopping point vertices are denoted by $\mathcal{W} = \{w_1, \cdots, w_m\}$. Let w_0 denote the depot, a special stopping point vertex, where the heterogeneous vehicle team is initially located. For each $i, k \in \mathcal{I}$ where $\mathcal{I} = \mathcal{W} \cup \{w_0\}$ and $j \in \mathcal{C}$, let the binary decision variable $\sigma_{ijk} = 1$ if and only if it is planned that the drone serves customer j by directly flying from stopping point vertex i and then flying to stopping point vertex k, and the binary variable $y_{ik} = 1$ if and only if it is planned that the truck directly travels from stopping point vertex i to stopping point vertex k. Let $d(i, j)$ denote the Euclidean distance between vertices i and j, and the binary value $p_j^r = 1$ if one requires customer r to be visited before customer j, and $p_j^r = 0$ if there is no such a requirement. As shown in Fig. 5.2 (a), the digraph $\mathcal{G}^p = (V^p, E^p)$ consists of a subset of customer vertices in $\mathcal{C}$ and a set of directed edges E^p showing the precedence constraints among the vertices. It can be easily checked that the problem has feasible solutions only if no direct cycles exist in $\mathcal{G}^p$, i.e. $\mathcal{G}^p$ is acyclic. It is assumed that the drone flies with a constant speed v_d under the maximum fly distance L, and the truck travels with a constant speed v_t under no travel range constraint.

The variable t_j is employed to represent the time when customer $j, j \in \mathcal{C}$, is served, and $P(t)$ is the vertex where the truck is located at time t. Then, the problem is to minimize the time for visiting all the customer locations

$$f = \max_{j \in \mathcal{C}} t_j, \tag{5.1}$$

subject to

$$\sum_{i,k\in\mathcal{I}} \sigma_{ijk} = 1, \quad \forall j \in \mathcal{C}; \tag{5.2}$$

$$\sigma_{ijk} = y_{ik}, \quad \forall i,k \in \mathcal{I}, k \neq i, \forall j \in \mathcal{C}; \tag{5.3}$$

$$y_{ii} = 0, \quad \forall i \in \mathcal{I}; \tag{5.4}$$

$$P(t_j - \frac{d(i,j)}{v_d})\sigma_{ijk} = \sigma_{ijk}i, \quad \forall i,k \in \mathcal{I}, \forall j \in \mathcal{C}; \tag{5.5}$$

$$(d(i,j) + d(j,k))\sigma_{ijk} \leqslant L, \quad \forall i,k \in \mathcal{I}, \forall j \in \mathcal{C}; \tag{5.6}$$

$$(t_r - t_j)p_j^r \leqslant 0, \quad \forall r,j \in \mathcal{C}. \tag{5.7}$$

Constraint (5.2) ensures that each customer is served; (5.3) guarantees the drone to be recharged by the truck after serving each customer; (5.5) makes sure the drone's path is feasible through coordinating with the truck; (5.6) ensures the drone's fly distance is within its capability; (5.7) guarantees the precedence constraints on visiting the customers are satisfied.

After formulating the task assignment problem as a constrained minimization problem, we present in the following section the analysis of the optimization problem.

5.3 Problem analysis

5.3.1 Proof of NP-hardness

We can simplify the digraph $\mathcal{G}^p = (V^p, E^p)$, specifying the precedence constraints on visiting the customer locations, whenever the following two conditions hold at the same time: (i) one customer vertex c_i has one edge directly pointing at another customer vertex c_j and (ii) c_i has multiple directed paths to c_j. An example is shown in Fig. 5.2 (a), where customer c_1 has three independent directed paths to c_4 as $c_1 \to c_3 \to c_4$, $c_1 \to c_2 \to c_4$ and $c_1 \to c_4$. Since c_2 is required to be visited before c_4 and c_1 is required to be visited before c_2, the precedence constraint from c_1 to c_4 becomes redundant. Then, after deleting some redundant precedence constraints, the digraph shown in Fig. 5.2 (a) can be simplified to Fig. 5.2 (b).

Now consider the undirected graph $\mathcal{G} = (V, E, D)$ consisting of a vertex set $V = \mathcal{C} \cup \mathcal{I}$, an edge set $E = E_{\mathcal{C}} \cup E_{\mathcal{I}}$, and a cost matrix D that saves the weight of each edge in E. $E_{\mathcal{I}}$ is a fully connected edge set containing the edge (w_i, w_k) for every pair of stopping point vertices $w_i, w_k \in \mathcal{I}$. $E_{\mathcal{C}}$ contains pairs of flight edges (w_i, c_j) and (c_j, w_k) for all $c_j \in \mathcal{C}$ and $w_i, w_k \in \mathcal{I}$ where the drone can fly from vertex w_i to serve customer c_j and then return to w_k. These w_i are called the *viable deployment vertices* for customer vertex c_j from which the drone

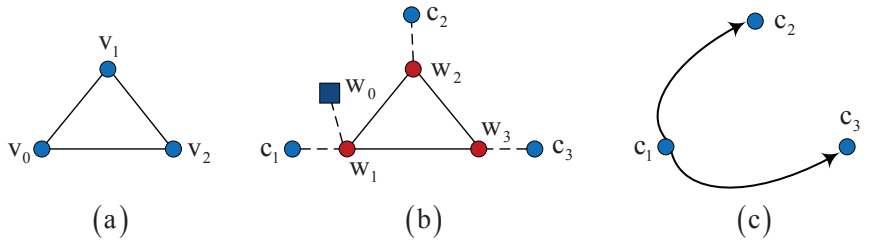

Figure 5.3: A transformation from the OTSP on graph $\mathcal{G}'$ to the PCHDP on graph $\mathcal{G}$ and $\mathcal{G}^p$ where (a) $\mathcal{G}' = (V', E', D')$, (b) the $\mathcal{G} = (V, E, D)$ with $D(w_{i+1}, w_{j+1}) = D'(v_i, v_j), \forall v_i, v_j \in V'$, $D(w_0, w_1) = 0$, $D(c_i, w_i) = 0$ and $D(w_i, c_i) = 0, \forall w_i \in V$, and (c) $\mathcal{G}^p$.

can be released to reach c_j, and then the drone can return at lease one stopping point vertex w_k. The set V_j is employed to contain all such w_i for customer c_j. Let $D = (d(i,j))_{(m+n+1)\times(m+n+1)}, i, j \in V$, where $d(i,j)$ specifies the distance between the two vertices i and j associated with the edge (i,j).

In graph theory, the open TSP (OTSP) involves determining a Hamiltonian path with the minimal length connecting in sequence each vertex exactly once in a directed or undirected graph, and the TSP determines a Hamiltonian cycle with the minimal length that is a cycle. Determining whether such cycles and paths exist in graphs is the NP-complete Hamiltonian path problem [30, 199-200]. The requirement for the traveling salesman to return to the starting city does not change the computational complexity of the problem. So the OTSP is NP-hard as well [28]. Now we show that the optimization problem PCHDP is also NP-hard.

Theorem 5.1. *The precedence-constrained task assignment problem PCHDP is NP-hard.*

Proof. To prove the NP-hardness of PCHDP, it suffices to show that: (i) every instance of the OTSP can be reduced to an instance of PCHDP in polynomial time and (ii) an optimal solution to PCHDP leads to an optimal OTSP solution. Let $\mathcal{G}' = (V', E', D')$ with $|V'| = n$ be an input to the OTSP, where $v_i, i \in \{1, \ldots, n-1\}$, represent the $n-1$ dispersed cities to be visited and v_0 is the depot where the traveling salesman is initially located. To prove (i), as shown in Fig. 5.3, we give a polynomial-time transformation of $\mathcal{G}'$ into $\mathcal{G} = (V, E, D)$ and $\mathcal{G}^p$ which are the inputs to PCHDP.

The PCHDP is constructed such that each customer c_i, $i \in \{1, \ldots, n\}$, corresponds to the vertex v_{i-1} in V', and c_i has only one unique viable deployment stopping point vertex w_i. A starting vertex (depot) w_0, where the truck and the drone start the delivery task, is added to V. Now for each edge (v_i, v_j) in E' with the weight $d'(v_i, v_j)$, add a sequence of directed edges from customer ver-

tex c_{i+1} to c_{j+1} as (c_{i+1}, w_{i+1}), (w_{i+1}, w_{j+1}) and (w_{j+1}, c_{j+1}) with the weight $d(w_{i+1}, w_{j+1}) = d'(v_i, v_j)$ while $d(c_{i+1}, w_{i+1})$ and $d(w_{j+1}, c_{j+1})$ are both zero. In addition, add a bidirectional edge from w_0 to w_1 with $d(w_0, w_1) = 0$. Let the truck in PCHDP have the same travel speed as that of the traveling salesman in OTSP. Finally, to construct $\mathcal{G}^p$, let customer vertex c_1 have precedence constraints on the other customers as the directed edges with arrows shown in Fig. 5.3 (c), and there are no other precedence constraints among the customers. Thus, the transformation from $\mathcal{G}'$ to $\mathcal{G}$ and $\mathcal{G}^p$ is constructed, and we have obtained the inputs to PCHDP.

To prove (ii), after the transformation of $\mathcal{G}'$ to $\mathcal{G}$ and $\mathcal{G}^p$, it is straightforward to see that an optimal solution to PCHDP is also optimal to the OTSP based on the edge weights of the graph $\mathcal{G}$ and the precedence constraints among the customers, shown in Fig. 5.3 (b) and Fig. 5.3 (c). From Fig. 5.3 (b) and Fig. 5.3 (c), a PCHDP solution is of the form $P_d = (w_0, w_1, c_1, w_1, \cdots, w_n, c_n, w_n)$ and $P_t = (w_0, w_1, \cdots, w_n)$ where P_d is the path of the drone and P_t is the path of the truck. The PCHDP solution can be employed to generate an OTSP path of the form $P' = (v_0, \cdots, v_{n-1})$ by either extracting the order of the stopping point vertices $(w_0, w_1, \cdots, w_n)$ from P_d as the truck needs to visit every w_i to serve each c_i or directly from P_t. If $E_{P'}$ contains the optimal sequence of edges in P', then $E_{P_t} = E_{P'}$. Since $d(w_0, w_1) = 0$, $d(c_i, w_i) = 0$ and $d(w_i, c_i) = 0$, it is straightforward to check that the shortest time for the drone to serve all the customers satisfies $\sum_{e \in E_{P_d}} d(e) = \sum_{e \in E_{P'}} d'(e)$. Thus, the proof is complete.

□

Remark 5.2. *If every customer location only has one precedence constraint requiring it to be visited either before or after another customer location, the resulting problem is a variant of the single-vehicle Dial-a-Ride Problem to design one vehicle route to serve a set of customers at the required destinations [14].*

5.3.2 A lower bound on the optimal solution

It can be costly to solve PCHDP optimally due to the NP-hardness of the problem. As a consequence, it is natural to design heuristic algorithms to find sub-optimal solutions. Then, one issue arises on how to evaluate the level of optimality of a sub-optimal solution as the optimal is typically unknown. In this section, a lower bound on the minimal time for the vehicles to serve all the customers while satisfying every precedence constraint is constructed through obtaining a min-cost arborescence (MCA) of a weighted digraph $\mathcal{G}^d = (V^d, E^d, D^d)$ by the Edmonds' algorithm [22]. The sum of all the edge weights of the MCA is minimal among all the arborescences of $\mathcal{G}^d$. The vertex set $V^d = \mathcal{C} \cup \{w_0\}$ consists of the indices of the n customer vertices and the depot. The edge set E^d, deduced from the digraph

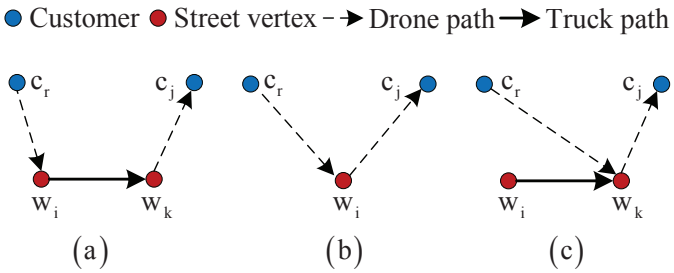

Figure 5.4: Three potential micro deployment patterns for the drone to serve two customers c_r and c_j successively by the usage of stopping point vertices w_i and w_k: (a) Pattern α, (b) Pattern β and (c) Pattern γ.

$\mathcal{G}^p$, contains every directed edge orienting from vertex c_r to c_j for all $c_r, c_j \in V^d$ if vertex c_r can be visited before vertex c_j. The matrix D^d contains the feasible minimal time for the drone to fly from an arbitrary vertex c_r to every c_j, which is obtained as follows.

Inspired by [53], for every customer vertex c_r, let the set V_r contain the indices of the viable deployment stopping point vertices from which the drone is viable to be released to serve c_r. Extending the results in [53], in Fig. 5.4 we show three potential micro time deployment patterns for the drone located at customer vertex c_r with the remaining fly distance L_{c_r} to serve customers c_j when the truck is located at stopping point vertex w_i. The corresponding travel times of the drone in the three micro deployment patterns are computed respectively as:

$$t_\alpha(c_r, c_j) = \min_{\substack{L_{c_r} \geqslant d(c_r, w_i), \\ \forall w_i \in \mathcal{W}, \forall w_k \in V_j}} \{\frac{d(c_r, w_i)}{v_d} + \frac{d(w_i, w_k)}{v_t} + \frac{d(w_k, c_j)}{v_d}\}; \tag{5.8}$$

$$t_\beta(c_r, c_j) = \min_{\substack{L_{c_r} \geqslant d(c_r, w_i), \\ \forall w_i \in V_j}} \frac{d(c_r, w_i) + d(w_i, c_j)}{v_d}; \tag{5.9}$$

$$t_\gamma(c_r, c_j) = \min_{\substack{L_{c_r} \geqslant d(c_r, w_k), \\ \forall w_k \in V_j}} \{\max\{\frac{d(c_r, w_k)}{v_d}, \frac{d(w_i, w_k)}{v_t}\} + \frac{d(w_k, c_j)}{v_d}\}. \tag{5.10}$$

Let $L_{c_r} = L - \min_{\forall w_h \in V_r} d(w_h, c_r)$, then the shortest time for the drone located at the customer vertex c_r to serve c_j is

$$t^\star(c_r, c_j) = \min\{t_\alpha(c_r, c_j), t_\beta(c_r, c_j), t_\gamma(c_r, c_j)\}, \tag{5.11}$$

which is the weight $D^d(c_r, c_j)$ for the directed edge $(c_r, c_j) \in E^d$.

We give an example on formulating the directed E^d and the associated D^d based on the directed $\mathcal{G}^p$ shown in Fig. 5.2 (b) and the corresponding weighted

undirected $\mathcal{G}$. A vertex set $S_r, \forall c_r \in \mathcal{C}$, is used to save the indices of the customer vertices before which c_r can be served. S_r is achieved in a backward manner. Firstly, let $S_r = \mathcal{C} \setminus \{c_r\}, \forall c_r \notin V^p$. Then, in Fig. 5.2 (b), S_6 is first calculated as $S_6 = \cup_{c_r \in \mathcal{C} \setminus V^p} c_r$. Afterwards, $S_r = \cup_{p_j'^r=1}(S_j \cup \{c_j\})$ where $p_j'^r = 1$ if the customer vertex c_r has an edge directly pointing at customer vertex c_j in the simplified $\mathcal{G}^p$ as c_1 and c_2 in Fig. 5.2 (b). Thus, $S_7 = S_6 \cup \{c_6\}$. Then, S_i can be obtained iteratively for every $c_i \in V^d$. Finally, an edge $E^d(c_r, c_j)$ exists connecting vertex c_r and every $c_j \in S_r$, and the corresponding $D^d(c_r, c_j) = t^\star(c_r, c_j)$ saves the shortest feasible time for the drone to travel from c_r to c_j as shown in (5.11); for the other cases $D^d(c_r, c_j) = \infty$. Let f_a be the sum of all the edge weights of an MCA of $\mathcal{G}^d$, and f_o be the optimal for the objective function shown in (5.1). Now the property of the optimal solution is investigated.

Proposition 5.3. *It holds that* $f_a \leqslant f_o$.

Proof. According to Definition 1, an optimal path for the drone to serve all the customers while respecting all the precedence constraints on visiting the customers is an arborescence of the weighted digraph $\mathcal{G}^d$. As f_a is the sum of all the edge weights of an MCA of $\mathcal{G}^d$, it is straightforward to check that $f_a \leqslant f_o$.

□

Having done the theoretical analysis, we construct several heuristic algorithms in the next section.

5.4 Task assignment algorithms

In this section, we first introduce one topological sorting technique, based on which we propose three task assignment algorithms.

5.4.1 Topology sorting technique

Through topological sorting, all the feasible paths in a directed acyclic graph can be obtained [79]. In [56, 82], the precedence-constrained TSP is solved by employing a topological sorting technique which iteratively sorts the customer vertices in $\mathcal{G}^{p'}$ without any predecessor in each iteration. Initially, let $\mathcal{G}^{p'} = \mathcal{G}^p$. The customer vertices without any predecessor are called *viable customer vertices*, which can be inserted into the TSP path behind the customer vertices that are the predecessors of them. Once inserting a viable customer vertex, to update $\mathcal{G}^{p'}$, the customer vertex and the precedence constraints corresponding to the edges leaving the customer in the current $\mathcal{G}^{p'}$ are deleted. We give an example to show how to construct a TSP path to visit all the customers from the representation scheme by considering the precedence constraints shown in Fig. 5.2 (b). In the digraph,

the first customer vertex sorted to be inserted into the TSP path is c_1, since c_1 is the only customer in $\mathcal{G}^{p'}$ without any predecessor. Then, c_1 is stored in the TSP path, and at the same time c_1 and the edges (c_1, c_3), (c_1, c_2) originating from c_1 are deleted from $\mathcal{G}^{p'}$. In the next iteration, c_2 and c_3 are viable customer vertices in the resulting $\mathcal{G}^{p'}$, which can then be inserted into the path after c_1. The process continues until all the customer vertices in $\mathcal{G}^p$ are inserted.

5.4.2 Task assignment algorithms

Inspired by [56, 82], the precedence-constrained package delivery task assignment problem can be solved by first iteratively inserting a viable customer vertex into the drone's path based on the travel cost matrix D^d, and then planing the path for the truck to ensure the drone's path is feasible. That is to say the truck needs to release the drone at one stopping point vertex to serve each customer, and then recharges the drone at one stopping point vertex after the customer is served as the drone's capacity of carrying packages is only one.

Let $\mathcal{R}_l$ save the indices of the ordered customers already inserted into the drone's path after iteration l, and the customer set $\mathcal{T}^A_{\mathcal{R}_l}$ contain the indices of those customer vertices that have no predecessor in $\mathcal{G}^{p'}$ and have not been inserted after iteration l. Let $\mathcal{T}^j_{\mathcal{R}_l} = \{c_r \in \mathcal{R}_l : (c_r, c_j) \in E^p\}$ for each $c_j \in \mathcal{T}^A_{\mathcal{R}_l}$, and the set $\mathcal{T}^j_a$ save the ordered locations in $\mathcal{R}_l$ in which the customer c_j is viable to be inserted while satisfying every precedence constraint on visiting c_j. Obviously, c_j can only be inserted into $\mathcal{R}_l$ behind all the customer vertices in $\mathcal{T}^j_{\mathcal{R}_l}$. If $\mathcal{T}^j_{\mathcal{R}_l} = \varnothing$, c_j is viable to be inserted at any position of $\mathcal{R}_l$. For the task assignment algorithms, $\mathcal{R}_l$ is initialized as $\{w_0\}$ where the drone and the truck are initially located.

Nearest inserting algorithm

The first heuristic algorithm is the nearest inserting algorithm (NIA) which puts the customer vertices in a sequence based on the time for the drone to travel from the customer vertices already inserted into the path to the one that can be inserted.

In iteration $l+1$, NIA finds the customer $c_{j^\star} \in \mathcal{T}^A_{\mathcal{R}_l}$ to be inserted and the associated inserting position $q^\star + 1$ as

$$(q^\star, c_{j^\star}) = \underset{q \in \mathcal{T}^j_a, c_j \in \mathcal{T}^A_{\mathcal{R}_l}}{\operatorname{argmin}} t^\star(\mathcal{R}_l(q), c_j), \tag{5.12}$$

where $\mathcal{T}^j_a = \{p, ..., |\mathcal{R}_l|\}$; $p = \max_{c_r \in \mathcal{T}^j_{\mathcal{R}_l}} \text{find}(\mathcal{R}_l, c_r)$ is the farthest position to the end of $\mathcal{R}_l$ after which customer c_j is viable to be inserted; and $\mathcal{R}_l(q)$ is the qth ordered customer on the path $\mathcal{R}_l$. The operator find$(\mathcal{R}_l, c_r)$ finds the location in

$\mathcal{R}_l$ where the customer vertex c_r is located. Then, the path $\mathcal{R}_l$ is updated to

$$\mathcal{R}_{l+1} = \begin{cases} \{\mathcal{R}_l, c_{j^\star}\}, \text{ if } q^\star = |\mathcal{R}_l|, \\ \{\mathcal{R}_l(1:q^\star), c_{j^\star}, \mathcal{R}_l(q^\star+1:|\mathcal{R}_l|)\}, \text{ otherwise}, \end{cases} \tag{5.13}$$

where $|\mathcal{R}_l|$ is the size of $\mathcal{R}_l$ and $\mathcal{R}_l(1:q^\star)$ saves the ordered customer vertices located between the first and the $q^\star$th positions of $\mathcal{R}_l$.

After the insertion of $c_{j^\star}$, we delete all the edges implying the precedence constraints initiating from $c_{j^\star}$ and the vertex $c_{j^\star}$ to update $\mathcal{G}^{p'}$. Then, the topological sorting technique is used to update $\mathcal{T}^A_{R_{l+1}}$ which saves the indices of viable customer vertices after iteration $l+1$. The inserting procedures (5.12) and (5.13) continue until all the customer vertices in $\mathcal{C}$ are inserted into the drone's path.

Now we show an example on how NIA works as follows. Assume that the current drone path for visiting the customers subject to the precedence constraints shown in Fig. 5.2 (b) is $\mathcal{R}_l = \{c_1, c_2, c_5\}$. Then, the viable customer set is $\mathcal{T}^A_{\mathcal{R}_l} = \{c_3\}$ as c_3 is the only customer without any predecessor after deleting the customer vertices already in $\mathcal{R}_l$ and the corresponding edges from $\mathcal{G}^{p'}$. Since c_1 is the only customer that is required to be visited before c_3, c_3 is viable to be inserted at any place behind c_1. Assume that $\mathcal{R}_l = \{c_1, c_3, c_2, c_5\}$ after the insertion of c_3. Then, the next viable customer c_4 can only be inserted after c_5 as c_5 is required to be visited before c_4, where $q = 4$ according to (5.12). One feasible drone path is $\mathcal{R}_l = \{c_1, c_3, c_2, c_5, c_4, c_7, c_6\}$ after iteratively using the topological sorting technique and the inserting procedure.

Minimum marginal-cost algorithm

The second task assignment algorithm is the minimum marginal-cost algorithm (MMA), which determines the next customer to be inserted and the corresponding inserting position based on the marginal travel time incurred by inserting the customer. The marginal travel time incurred by inserting customer c_j at the qth position of $\mathcal{R}_l$ is approximated as

$$t(\mathcal{R}_l \oplus_q c_j) - t(\mathcal{R}_l) = \begin{cases} t^\star(\mathcal{R}_l(q-1), c_j), \text{ if } q = |\mathcal{R}_l|+1, \\ t^\star(\mathcal{R}_l(q-1), c_j) + t^\star(c_j, \mathcal{R}_l(q)) \\ -t^\star(\mathcal{R}_l(q-1), \mathcal{R}_l(q)), \text{ otherwise}, \end{cases} \tag{5.14}$$

where the operation $\mathcal{R}_l \oplus_q c_j$ inserts customer c_j at the qth position of $\mathcal{R}_l$. Target c_j is inserted to the end of $\mathcal{R}_l$ if $q = |\mathcal{R}_l|+1$, and $t(\mathcal{R}_l)$ denotes the total travel time for the drone to deliver packages to all the customers in $\mathcal{R}_l$ based on the travel cost matrix D^d. The incurred marginal time can be approximated by the usage of the minimal travel time $t^\star(c_r, c_j)$ shown in (5.11).

MMA determines the customer $c_{j^\star} \in \mathcal{T}^A_{\mathcal{R}_l}$ to be inserted into the path $\mathcal{R}_l$ and the associated inserting position $q^\star$ in iteration $l+1$ as

$$(q^\star, c_{j^\star}) = \underset{p+1 \leqslant q \leqslant |\mathcal{R}_l|+1, c_j \in \mathcal{T}^A_{\mathcal{R}_l}}{\operatorname{argmin}} \{t(\mathcal{R}_l \oplus_q c_j) - t(\mathcal{R}_l)\}, \tag{5.15}$$

where $p = \max_{c_r \in \mathcal{T}^j_{\mathcal{R}_l}} \text{find}(\mathcal{R}_l, c_r)$. Then, the path $\mathcal{R}_l$ is updated to

$$\mathcal{R}_{l+1} = \mathcal{R}_l \oplus_{q^\star} c_{j^\star}. \tag{5.16}$$

After the insertion of $c_{j^\star}$, all the edges implying the precedence constraints initiating from $c_{j^\star}$ and the vertex $c_{j^\star}$ are deleted to update $\mathcal{G}^{p'}$. The customer inserting process continues until all the customers are inserted into the path.

Second-order minimum marginal-cost algorithm

The customer ordering strategy (5.15) shows that the customers already ordered in the iteration l directly affect the customer to be inserted in the next iteration as well as the inserting position. To accommodate the future customer to be inserted, we propose the second-order minimum marginal-cost algorithm (SMMA) which calculates the cost incurred by inserting a customer both considering the current iteration and the future iteration. For SMMA, the customer $c_{j_1^\star}$ to be inserted and its inserting position $q_1^\star$ in iteration $l+1$ satisfy

$$c_{j_1^\star} = \underset{\substack{p_2+1 \leqslant q_2 \leqslant |\mathcal{R}_l|+2, \\ c_{j_1} \in \mathcal{T}^A_{\mathcal{R}_l}, c_{j_2} \in \mathcal{T}^A_{\mathcal{R}_l} \setminus \{c_{j_1}\}}}{\operatorname{argmin}} \{t((\mathcal{R}_l \oplus_{q_1^\star} c_{j_1}) \oplus_{q_2} c_{j_2}) - t(\mathcal{R}_l)\}, \tag{5.17}$$

where $q_1^\star = \operatorname{argmin}_{p_1+1 \leqslant q_1 \leqslant |\mathcal{R}_l|+1} \{t(\mathcal{R}_l \oplus_{q_1} c_{j_1}) - t(\mathcal{R}_l)\}$, $p_1 = \max_{c_r \in \mathcal{T}^{j_1}_{\mathcal{R}_l}} \text{find}(\mathcal{R}_l, c_r)$, and $p_2 = \max_{c_r \in \mathcal{T}^{j_2}_{\mathcal{R}_l \oplus_{q_1^\star} c_{j_1}}} \text{find}(\mathcal{R}_l \oplus_{q_1^\star} c_{j_1}, c_r)$. Then, in iteration $l+1$ the path $\mathcal{R}_l$ is updated to

$$\mathcal{R}_{l+1} = \mathcal{R}_l \oplus_{q_1^\star} c_{j_1^\star}. \tag{5.18}$$

After the insertion of $c_{j_1^\star}$, the vertex $c_{j_1^\star}$ and all the edges implying the precedence constraints initiating from $c_{j_1^\star}$ are deleted to update $\mathcal{G}^{p'}$. The customer inserting process continues until all the customers are inserted into the drone's path.

It should be noted that the stopping point vertices used for formulating the minimal travel time for the drone to fly between each two customers might not be feasible to construct a truck path coordinating with the drone to serve all the customers due to the truck's limited travel speed. To show more detail, let the stopping point vertex be w_r where the drone is released to serve customer c_r. Then,

the feasible shortest time $t(c_r, c_j)$ for the drone to successively visit customer c_j after visiting c_r is obtained by equations from (5.8) to (5.11) where the drone's remaining fly distance L_{c_r} is $L - d(w_r, c_r)$, thus leading to $t^\star(c_r, c_j) \leqslant t(c_r, c_j)$ in (5.11) as $L - d(w_r, c_r) \leqslant L - \min_{\forall w_h \in V_r} d(w_h, c_r)$. That is because a longer remaining fly distance L_{c_r} would enable the drone to have more choices on choosing stopping point vertices w_i and w_k to serve customers c_r and c_j as shown in Fig. 5.4. Thus, after achieving the drone's path for visiting the customers based on the algorithms NIA, MMA and SMMA, $L_{c_r} = L - d(w_r, c_r)$ is used to calculate the feasible travel time for the drone released from stopping point vertex w_r to fly directly from customer c_r to customer c_j while considering the truck movement.

5.4.3 Computational Complexity

In this section, we analyze the computational complexity of running the NIA, MMA and SMMA. The three algorithms iteratively insert a customer vertex to the drone's path whose length is $|\mathcal{R}_l| = l$ after the lth iteration of the inserting operation. The computational complexity of NIA is determined by (5.12) where finding p requires $|\mathcal{T}^j_{\mathcal{R}_l}||\mathcal{R}_l|$ basic operations in the $(l+1)$th iteration of the assignment. Thus, to find $q^\star$ and $c_{j^\star}$ in (5.12), $|\mathcal{T}^j_a||\mathcal{T}^A_{\mathcal{R}_l}||\mathcal{T}^j_{\mathcal{R}_l}||\mathcal{R}_l|$ basic operations are needed in the $(l+1)$th iteration, where $|\mathcal{T}^j_a| \leqslant |\mathcal{R}_l|$, $|\mathcal{T}^A_{\mathcal{R}_l}| \leqslant n - |\mathcal{R}_l|$, and $|\mathcal{T}^j_{\mathcal{R}_l}| \leqslant |\mathcal{R}_l|$. As a consequence, at most $l^3(n-l)$ basic operations are required in the $(l+1)$th iteration. Taking the sum for l to change from 1 to n, we get the computational complexity of NIA $\sum_{l=1}^{l=n} l^3(n-l)$, resulting in $O(n^5)$. Similar to NIA, the computational complexity of MMA is determined by (5.15) where at most $2|\mathcal{T}^j_a||\mathcal{T}^A_{\mathcal{R}_l}||\mathcal{T}^j_{\mathcal{R}_l}||\mathcal{R}_l|$ basic operations are required in the $(l+1)$th iteration. Thus, the computational complexity of MMA is also $O(n^5)$.

The computational complexity of SMMA is determined by (5.17) where finding p_1 requires $|\mathcal{T}^{j_1}_{\mathcal{R}_l}||\mathcal{R}_l|$ basic operations, finding $q_1^\star$ requires at most $2|\mathcal{R}_l|$ basic operations, and finding p_2 requires $|\mathcal{T}^{j_2}_{\mathcal{R}_l \oplus_{q_1^\star} c_{j_1}}||\mathcal{R}_l \oplus_{q_1^\star} c_{j_1}|$ basic operations in the $(l+1)$th iteration of the assignment. Thus, at most $2|\mathcal{R}_l||\mathcal{T}^A_{\mathcal{R}_l}|(|\mathcal{T}^A_{\mathcal{R}_l}|-1)(|\mathcal{T}^{j_1}_{\mathcal{R}_l}||\mathcal{R}_l|+ 2|\mathcal{R}_l| + |\mathcal{T}^{j_2}_{\mathcal{R}_l \oplus_{q_1^\star} c_{j_1}}||\mathcal{R}_l \oplus_{q_1^\star} c_{j_1}|)$ basic operations are required in the $(l+1)$th iteration, which is at most $2l(2l^2+4l+1)(n-l)^2$. Then, we know the computational complexity of SMMA is $\sum_{l=1}^{l=n} 2l(2l^2+4l+1)(n-l)^2$, which is $O(n^6)$.

Now we have presented all the theoretical results of this chapter. In the following section, we carry out simulation studies.

5.5 Simulations

Numerical simulations are carried out to test the proposed algorithms in comparison with the GA for the precedence-constrained assignment problem [82], in which the GA is an improver version of that in [56]. The genetic parameters for the compared GA are set as follows according to [82]. The maximum generation number is 2000; the population size is 20; the crossover rate is 0.5 and the mutation rate is 0.05 [82]. All the experiments have been performed on an Intel Core $i5-4590$ CPU 3.30 GHz with 8 GB RAM, and the algorithms are compiled by Matlab under Windows 7. In addition to evaluating the travel time f shown in (5.1), the solution quality of each algorithm is also quantified by

$$q = \frac{f}{f_a}, \tag{5.19}$$

where f_a is the sum of all the edge weights of an MCA of the weighted directed customer-vehicle graph $\mathcal{G}^d$. Since $f_a \leqslant f_o$, from Proposition 5.3 where f_o is the maximal travel time of an optimal solution, the value of the ratio q closer to 1 means a better performance of the solution.

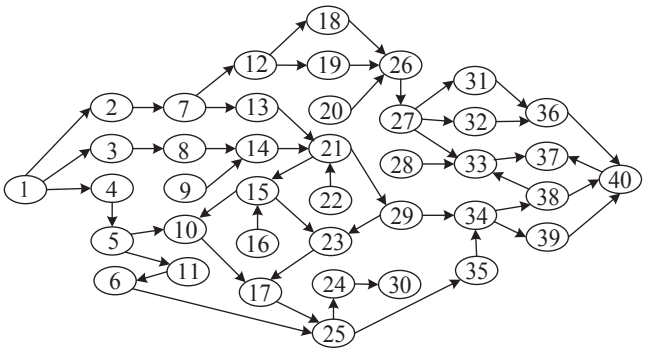

Figure 5.5: The digraph $\mathcal{G}^p$ contains the precedence constraints on serving 40 customers.

The algorithms are first tested on the task assignment problem where 40 customer vertices are subject to the constraints shown in Fig. 5.5 which is simplified from Fig. 11 in [56]. Ten instances of the initial positions of the customers are randomly generated in a square area with edge length $L_0 = 10^3$m, and there are 121 stopping point vertices evenly distributed in the area. The drone and the truck initially located at the original position are first assumed to travel with the unit speed. For each instance, we test the algorithms' performances when increasing the drone's maximum fly distance L. The average travel time to deliver packages to all the customers and the corresponding average q of the algorithms on

Table 5.1: The average travel time (s), resulting from the algorithms (A), for the vehicles to deliver packages to 40 customer locations under different operation ranges L of the drone and $v_d/v_t = 1$.

A \ L/L_0	0.15	0.30	0.45	0.60	0.75	0.90
GA	15134	14628	14490	14192	14258	14243
NIA	16580	15973	15897	15793	15786	15759
MMA	14818	14207	13950	13883	13864	13862
SMMA	14596	14016	13725	13633	13619	13611

Table 5.2: The corresponding average solution quality q of the algorithms (A) for employing the vehicles to deliver packages to 40 customer locations under different operation ranges L of the drone and $v_d/v_t = 1$.

A \ L/L_0	0.15	0.30	0.45	0.60	0.75	0.90
GA	2.4552	2.4462	2.4227	2.3721	2.3821	2.3795
NIA	2.6909	2.6738	2.6617	2.6442	2.6432	2.6385
MMA	2.4049	2.3726	2.3304	2.3195	2.3164	2.3160
SMMA	2.3682	2.3393	2.2911	2.2764	2.2742	2.2728

the instances are shown in Table 5.1 and Table 5.2 respectively. For the drone to deliver packages to all the customers while satisfying the precedence constraints, Table 5.1 first shows that the average travel time resulting from each algorithm decreases with the increase of the drone's operation range L. It is reasonable as the drone generally has more viable deployment stopping point vertices for serving each customer when increasing its operation range, which leads to more efficient paths for the drone to travel between two customers with the cooperation of the truck. The average q of the algorithms shown in Table 5.2 always has the same changing trend as the average travel time when increasing the drone's maximum fly distance. This might be due to the smaller difference between the real travel time $t(c_r, c_j)$ and the $t^\star(c_r, c_j)$ shown in (5.11) when increasing the drone's operation range. For every instance shown in Table 5.2, GA has the smaller q compared with NIA, but it has the largest q compared with MMA and SMMA, which verifies the satisfying performance of MMA and SMMA. Table 5.2 also shows that SMMA has the better performance than MMA. That is because SMMA employs the predictive strategy shown in (5.17) to determine the serving sequence of each customer.

To further test the performance of the algorithms, simulation experiments on the problem with 120 customers where every customer has only one precedence constraint requiring it to be served either before or after another customer as in

Table 5.3: The average travel time (s), resulting from the algorithms (A), for the vehicles to deliver packages to 120 customer locations under different operation ranges L of the drone and $v_d/v_t = 1$.

A \ L/L_0	0.15	0.30	0.45	0.60	0.75	0.90
GA	48059	46052	46074	45089	44822	45048
NIA	22733	20644	20778	20730	20732	20724
MMA	18665	17373	17320	17308	17307	17304
SMMA	18188	17182	17104	17095	17099	17097

Table 5.4: The corresponding average solution quality q of the algorithms (A) for employing the vehicles to deliver packages to 120 customer locations under different operation ranges L of the drone and $v_d/v_t = 1$.

A \ L/L_0	0.15	0.30	0.45	0.60	0.75	0.90
GA	4.1413	3.9769	3.9763	3.8930	3.8714	3.8883
NIA	1.9590	1.7815	1.7938	1.7897	1.7899	1.7893
MMA	1.6068	1.4989	1.4944	1.4934	1.4933	1.4931
SMMA	1.5675	1.4834	1.4767	1.4760	1.4764	1.4762

the Dial-a-Ride Problem [14]. For the simulation, 10 instances of the customers' initial locations and destinations are randomly generated in the same square area with the same number of stopping point vertices. The drone and the truck are also assumed to travel with the unit speed. For each instance, we test the algorithms' performances when increasing the drone's operation range L. The average time to serve all the customers and the corresponding average q of the algorithms on the instances are shown in Table 5.3 and Table 5.4 respectively. For every algorithm, Table 5.3 first shows that the average travel time for the drone to deliver packages to all the customers generally decreases when increasing the drone's operation range, which is the same compared with Table 5.1. Table 5.3 also shows that the three proposed algorithms have a better performance compared with the GA where their average q values are within twice of the optimal as shown in Table 5.4. That again verifies the satisfying performance of MMA and SMMA.

Finally, for the same environment setup for the 120 customers, we investigate the algorithms' performances when increasing both the drone's fly speed v_d and maximum fly distance L. The truck is assumed to travel with the unit speed as $v_t = 1$ while the drone's speed is increase to $v_d/v_t = 1.4$. The average time to deliver packages to all the customers and the corresponding average q of the algorithms on the 10 instances are shown in Table 5.5 and Table 5.6 respectively. For each algorithm, the average travel time shown in Table 5.5 also decreases with the

Table 5.5: The average travel time (s), resulting from the algorithms (A), for the vehicles to deliver packages to 120 customer locations under different operation ranges L of the drone and $v_d/v_t = 1.4$.

A \ L/L_0	0.15	0.30	0.45	0.60	0.75	0.90
GA	47335	41759	38672	36115	35315	34946
NIA	21017	17678	17199	16999	16946	16863
MMA	16497	14763	14464	14381	14359	14344
SMMA	16400	14495	14116	14065	14066	14053

Table 5.6: The corresponding average solution quality q of the algorithms (A) for employing the vehicles to deliver packages to 120 customer locations under different operation ranges L of the drone and $v_d/v_t = 1.4$.

A \ L/L_0	0.15	0.30	0.45	0.60	0.75	0.90
GA	5.5826	4.9611	4.6161	4.3278	4.2482	4.2150
NIA	2.4792	2.0988	2.0518	2.0346	2.0360	2.0320
MMA	1.9444	1.7527	1.7256	1.7210	1.7251	1.7285
SMMA	1.9314	1.7207	1.6840	1.6832	1.6858	1.6934

increase of the drone's operation range L. Table 5.5 also shows that NIA, MMA and SMMA have better performance compared with GA, and Table 5.6 shows that the average q values of MMA and SMMA are still within twice of the optimal, which verifies the superior performance of MMA and SMMA. Table 5.3 and Table 5.5 show that the drone's average travel time to serve all the customers decreases more rapidly when increasing the drone's speed in comparison with increasing its maximum fly distance. That is because the time for the drone to travel between two customers decreases when increasing its speed, which directly leads to the decrease of the total time to serve all the customers. However, the drone's speed cannot be increased too much to ensure safety when delivering the parcels in the city. Meanwhile, increasing the drone's operation range might not necessarily decrease its total travel time as the drone needs to return to the truck to be recharged with parcel after serving each customer. In Table 5.7, we also show the corresponding average computation time for the algorithms to achieve the solutions shown in Table 5.5. Table 5.7 shows that the average computation times of GA are far larger than those of NIA, MMA and SMMA where the SMMA is most time-consuming among the proposed algorithms as indicated in Section 5.4.3. Concluding from the above analysis, MMA and SMMA are more efficient than NIA and GA in every instance while SMMA performs better than MMA. However, SMMA needs more computation time than MMA. Then, it is suggested to use MMA for planning the

Table 5.7: The corresponding average computation time (s) for the algorithms to plan the paths for the vehicles to deliver packages to 120 customer locations under different operation ranges L of the drone and $v_d/v_t = 1.4$.

A \ L/L_0	0.15	0.30	0.45	0.60	0.75	0.90
GA	128.88	128.85	128.81	128.50	130.62	131.75
NIA	0.27	0.31	0.31	0.30	0.31	0.32
MMA	0.85	0.84	0.85	0.86	0.87	0.85
SMMA	37.63	37.55	37.36	37.63	37.42	37.60

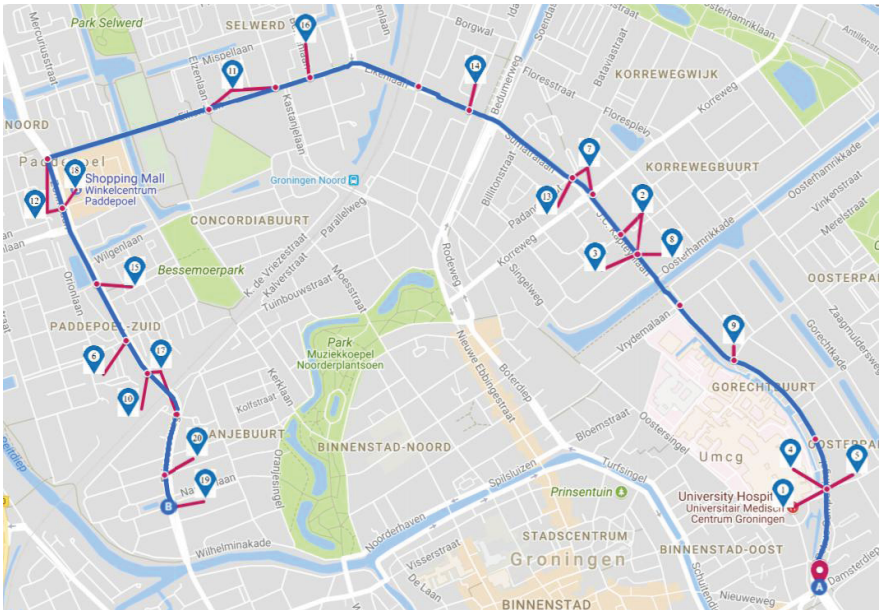

Figure 5.6: The paths resulting from SMMA for the truck and the drone to deliver packages to 20 customers where $L = 250$m and $v_d = 30$ km/hr and $v_t = 40$ km/hr. The total travel time of the drone is 1849.6s.

routes for the vehicles online as it can achieve the satisfying solution under short computation time while to use SMMA if more computation time is allowed as it offers a better solution.

Fig. 5.6 and Fig. 5.7 present a realistic package delivery instance on a Google street map of a residential neighbourhood in Groningen, The Netherlands. The neighborhood considered is a residential area outside the busy city center, but not too far away from the city's high-speed ring road. So it is an ideal test area for the possible coordination between the truck and drone. For each of the drone's landing on a delivery point or each of its loading of one package on the truck,

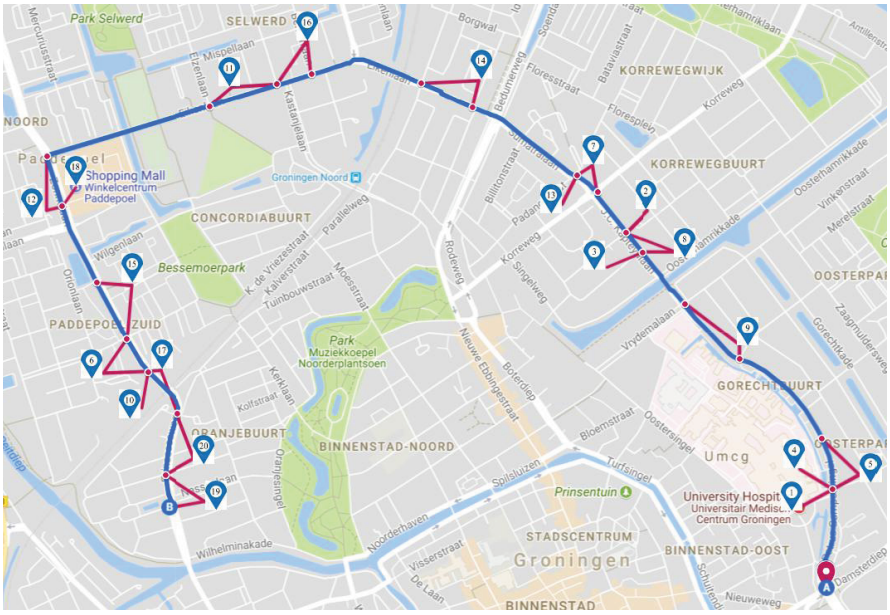

Figure 5.7: The paths resulting from SMMA for the truck and the drone to deliver packages to 20 customers where $L = 300$m and $v_d = 30$ km/hr and $v_t = 40$ km/hr. The total travel time of the drone is 1775.9s.

we assume a landing time of 30s is needed. We assume the speeds of the drone and the truck are $v_d = 30$ km/hr and $v_t = 40$ km/hr, respectively. Fig. 5.6 shows the paths for the truck and the drone to visit 20 delivery points where the visiting of the target locations respects the presence constraints shown in Fig. 5.5, and $L = 250$m. The total time for the drone and the truck to visit all the delivery points is 1849.6s, which is approximately a reduction of 10% in the time to serve all the customers by using a single delivery truck. Fig. 5.7 shows the paths for the truck and the drone to visit the 20 delivery points where the drone's maximum flight range is increased to $L = 300$m. The total time for the drone and the truck to visit all the delivery points is 1775.9s. It is worth mentioning that the time to deliver packages to the 20 customers decreases in the two instances even though the sequences for serving the customers are the same, which is due to the increase of the drone's operation range.

5.6 Conclusion

In this chapter, we have investigated the precedence-constrained package delivery problem where one drone coordinates with one truck to efficiently deliver parcels

to a set of customers while satisfying the precedence constraints on visiting the customers. The problem has been shown to be NP-hard and a lower bound on the minimum time to serve all the customers has been found. Integrated with a topological sorting technique, we have designed several heuristic task assignment algorithms. Numerical experiments have shown that the proposed algorithms can quickly obtain satisfying solutions to the precedence-constrained task assignment problem compared with the existing genetic algorithm. The proposed algorithms will be extended for the precedence-constrained package delivery with one truck coordinating with multiple drones. Another research direction is to investigate the precedence-constrained package delivery with one truck coordinating with one drone where the drone can deliver multiple packages in one run.

Chapter 6
Conclusions

THIS chapter summarizes the main results of this thesis and recommends possible directions for future research.

6.1 Conclusions

This thesis has studied the task assignment problem for multiple vehicles in challenging environments subject to constraints resulting from the environments, the vehicles, and the tasks to be completed, respectively. In particular, the currents in a drift field, the vehicles' limited communication range, and the precedence constraints on visiting the target locations have been considered in the multi-vehicle task assignment problem. We now review the main results in the thesis.

In Chapter 2 we have investigated the task assignment for multiple vehicles in a time-invariant drift field where a vehicle can reach any other locations in the field. The objective is to employ several dispersed vehicles to visit a set of target locations in the drift field while trying to minimize the vehicles' total travel time. Based on optimal control theory, we have first designed a path planning algorithm, which can generate the time-optimal path for a vehicle to travel between two prescribed locations in a drift field. The path planning algorithm provides the cost matrix for the target assignment, and generates routes once the target locations are assigned to a vehicle. Using tools from graph theory, a lower bound on the optimal solution is found, which can be used to measure the proximity of a solution from the optimal. We have proposed several clustering-based task assignment algorithms in which two of them guarantee that all the target locations will be visited within a computable maximal travel time, which is at most twice of the optimal when the cost matrix is symmetric.

Chapter 3 has addressed the multi-vehicle task assignment in a time-invariant drift field with obstacles, which is a follow-up of Chapter 2. The vehicles have different capabilities, and each kind of vehicles needs to visit a certain type of target locations; each target location might have the demand to be visited more than once by different kinds of vehicles. A path planning method has been designed to enable the vehicles to move between two prescribed locations in a drift field with minimal time while avoiding any obstacle. In addition, a distributed task assign-

ment algorithm has been proposed which enables the vehicles to efficiently visit all the target locations in the drift field.

In Chapter 4 we have studied the task assignment problem in which multiple dispersed heterogeneous vehicles with limited communication range need to visit a set of target locations while trying to minimize the vehicles' total travel distance. Each vehicle initially has the position information of all the targets and of those vehicles that are within its limited communication range, and each target demands a vehicle with some specified capability to visit it. We have designed a decentralized auction algorithm which first employs an information consensus procedure to merge the local information carried by each communication-connected vehicle subnetwork. Then, the algorithm constructs conflict-free target assignments for the communication-connected vehicles, and guarantees that the total travel distance of the vehicles is at most twice of the optimal when the communication network is initially connected.

Chapter 5 has presented the precedence-constrained task assignment problem for a truck and a micro drone to deliver packages to a set of dispersed customers subject to precedence constraints that specify which customers need to be visited before which other customers. The truck is restricted to travel in a street network and the micro drone, restricted by its loading capacity and operation range, can fly from the truck to perform the last mile package deliveries. The objective is to minimize the time to serve all the customers respecting every precedence constraint. The problem has been shown to be NP-hard, and a lower bound on the optimal time to serve all the customers has been constructed by using tools from graph theory. Integrating with a topological sorting technique, we have designed several heuristic task assignment algorithms to solve the task assignment problem.

6.2 Recommendations for future research

This thesis has investigated the multi-vehicle task assignment problem in different challenging environments. As follow-ups, several practical issues still need to be studied for the multi-vehicle task assignment problem in the future work.

Capacity-constrained multi-vehicle task assignment problem. In Chapters 2 and 4, we have addressed the multi-vehicle task assignment problems in a drift field and subject to limited communication range, respectively. However, we have not considered the vehicles' capacity constraint under which each vehicle can visit a maximum number of target locations. Neither have we looked at the case where each vehicle has a maximum travel time. The vehicles' maximum travel time constrains the vehicles' operation range, and the vehicles' load capacity restricts how many targets can be maximumly assigned to each vehicle, which greatly affect how the target locations are assigned to the vehicles. Since the practical capacity

constraints of the vehicles might complicate the multi-vehicle task problem, the performance of the task assignment algorithms proposed in this thesis might no longer be guaranteed. Of course, it will bring more practical contribution if one considers the vehicles' capacity constraints in the multi-vehicle task assignment problem.

Multi-drone and multi-truck supported package delivery task assignment problem. In Chapter 5, we have investigated the task assignment problem for a truck and a micro drone to deliver packages to a set of dispersed customers, where the truck is restricted to travel in a street network and the micro drone with package loading capacity of one and limited operation range can fly from the truck to perform the last mile package deliveries. With the increase of the loading capacity of the drone and the loading capacity of the truck, one drone can deliver packages to several customers after each package recharging from the truck, and the truck can carry multiple drones to perform the package delivery. As a result, it is both interesting and urgent to investigate how to plan the routes for one or several trucks, each of which carries several drones in order to increase their delivery efficiency, such as minimizing the maximum delivery time when the last customer is served.

Precedence-constrained multi-vehicle task assignment problem. In Chapter 5, we have investigated the precedence-constrained package delivery problem for a truck and a micro drone to deliver packages to a set of dispersed customers where only the micro drone performs the last mile package deliveries. Generally speaking, more vehicles such as drones and trucks are needed to perform the package deliveries if the number of customers to be served is large. Then, the sequence for visiting the customers assigned to each drone should respect the precedence constraints on serving the customers. Thus, it is of great importance to organize a proper assignment of customers to the drones so that all the customers can be quickly served while respecting every precedence constraint.

Bibliography

[1] Jose J Acevedo, Begoña C Arrue, Jose Miguel Diaz-Bañez, Inmaculada Ventura, Ivan Maza, and Anibal Ollero. One-to-one coordination algorithm for decentralized area partition in surveillance missions with a team of aerial robots. *Journal of Intelligent & Robotic Systems*, 74(1-2):269–285, 2014.

[2] Chang Wook Ahn and Rudrapatna S Ramakrishna. A genetic algorithm for shortest path routing problem and the sizing of populations. *IEEE Transactions on Evolutionary Computation*, 6(6):566–579, 2002.

[3] David L Applegate, Robert E Bixby, Vasek Chvatal, and William J Cook. *The traveling salesman problem: a computational study*. Princeton university press, 2011.

[4] Xiaoshan Bai, Weisheng Yan, and Ming Cao. Clustering-based algorithms for multivehicle task assignment in a time-invariant drift field. *IEEE Robotics and Automation Letters*, 2(4):2166–2173, 2017.

[5] Xiaoshan Bai, Weisheng Yan, Ming Cao, and Jie Huang. Task assignment for robots with limited communication. In *Control Conference (CCC), 2017 36th Chinese*, pages 6934–6939. IEEE, 2017.

[6] Tolga Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219, 2006.

[7] Andrea Bettinelli, Alberto Ceselli, and Giovanni Righini. A branch-and-cut-and-price algorithm for the multi-depot heterogeneous vehicle routing problem with time windows. *Transportation Research Part C: Emerging Technologies*, 19(5):723–740, 2011.

[8] Giulio Binetti, David Naso, and Biagio Turchiano. Decentralized task allocation for surveillance systems with critical tasks. *Robotics and Autonomous Systems*, 61(12):1653–1664, 2013.

[9] Hans-Joachim Böckenhauer, Tobias Mömke, and Monika Steinová. Improved approximations for tsp with simple precedence constraints. *Journal of Discrete Algorithms*, 21:32–40, 2013.

[10] Bela Bollobas. *Modern Graph Theory*, volume 184. Springer Science & Business Media, 1998.

[11] Edoardo Bovio, D Cecchi, and Francesco Baralli. Autonomous underwater vehicles for scientific and naval operations. *Annual Reviews in Control*, 30 (2):117–130, 2006.

[12] Bo Chen and Harry H Cheng. A review of the applications of agent technology in traffic and transportation systems. *IEEE Transactions on Intelligent Transportation Systems*, 11(2):485–497, 2010.

[13] Han-Lim Choi, Luc Brunet, and Jonathan P How. Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics*, 25 (4):912–926, 2009.

[14] Jean-François Cordeau and Gilbert Laporte. The dial-a-ride problem: models and algorithms. *Annals of Operations Research*, 153(1):29–46, 2007.

[15] Torbjørn S Dahl, Maja Matarić, and Gaurav S Sukhatme. Multi-robot task allocation through vacancy chain scheduling. *Robotics and Autonomous Systems*, 57(6):674–687, 2009.

[16] Jeffrey Delmerico, Elias Mueggler, Julia Nitsch, and Davide Scaramuzza. Active autonomous aerial exploration for ground robot path planning. *IEEE Robotics and Automation Letters*, 2(2):664–671, 2017.

[17] Donato Di Paola, Andrea Gasparri, David Naso, and Frank L Lewis. Decentralized dynamic task planning for heterogeneous robotic networks. *Autonomous Robots*, 38(1):31–48, 2015.

[18] M Bernardine Dias, Robert Zlot, Nidhi Kalra, and Anthony Stentz. Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, 94(7):1257–1270, 2006.

[19] Bernhard Dieber, Christian Micheloni, and Bernhard Rinner. Resource-aware coverage and task assignment in visual sensor networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(10):1424–1437, 2011.

[20] Matthew G Earl and Raffaello DAndrea. A decomposition approach to multi-vehicle cooperative control. *Robotics and Autonomous Systems*, 55(4):276–291, 2007.

[21] Eugene Edison and Tal Shima. Integrated task assignment and path optimization for cooperating uninhabited aerial vehicles using genetic algorithms. *Computers & Operations Research*, 38(1):340–356, 2011.

[22] Jack Edmonds. Optimum branchings. *Mathematics and the Decision Sciences, Part*, 1:335–345, 1968.

[23] Mike Eichhorn. Optimal routing strategies for autonomous underwater vehicles in time-varying environment. *Robotics and Autonomous Systems*, 67: 33–43, 2015.

[24] John Willmer Escobar, Rodrigo Linfati, Paolo Toth, and Maria G Baldoquin. A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem. *Journal of Heuristics*, 20(5):483–509, 2014.

[25] Laureano F Escudero. An inexact algorithm for the sequential ordering problem. *European Journal of Operational Research*, 37(2):236–249, 1988.

[26] Krzysztof Fleszar, Ibrahim H Osman, and Khalil S Hindi. A variable neighbourhood search algorithm for the open vehicle routing problem. *European Journal of Operational Research*, 195(3):803–809, 2009.

[27] Dario Floreano and Robert J Wood. Science, technology and the future of small autonomous drones. *Nature*, 521(7553):460, 2015.

[28] Greg N Frederickson, Matthew S Hecht, and Chul E Kim. Approximation algorithms for some routing problems. In *Foundations of Computer Science, 1976., 17th Annual Symposium on*, pages 216–227. IEEE, 1976.

[29] Cheng-Heng Fua and Shuzhi Sam Ge. Cobos: Cooperative backoff adaptive scheme for multirobot task allocation. *IEEE Transactions on Robotics*, 21(6): 1168–1178, 2005.

[30] Michael R Garey and David S Johnson. *Computers and intractability*, volume 29. WH Freeman New York, 2002.

[31] Brian P Gerkey and Maja J Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, 23(9):939–954, 2004.

[32] E Gilbert and Daniel Johnson. Distance functions and their application to robot path planning in the presence of obstacles. *IEEE Journal on Robotics and Automation*, 1(1):21–30, 1985.

[33] Paolo Robuffo Giordano and Marilena Vendittelli. Shortest paths to obstacles for a polygonal dubins car. *IEEE Transactions on Robotics*, 25(5):1184–1191, 2009.

[34] Gary Gordon and Elizabeth McMahon. A greedoid polynomial which distinguishes rooted arborescences. *Proceedings of the American Mathematical Society*, 107(2):287–298, 1989.

[35] Roderich Groß, Michael Bonani, Francesco Mondada, and Marco Dorigo. Autonomous self-assembly in swarm-bots. *IEEE Transactions on Robotics*, 22 (6):1115–1130, 2006.

[36] Gregory Gutin and Abraham P Punnen. *The traveling salesman problem and its variations*, volume 12. Springer Science & Business Media, 2006.

[37] Jonghui Han, Jinsung Ok, and Wan Kyun Chung. An ethology-based hybrid control architecture for an autonomous underwater vehicle for performing multiple tasks. *IEEE Journal of Oceanic Engineering*, 38(3):514–521, 2013.

[38] Wesley H Huang, Brett R Fajen, Jonathan R Fink, and William H Warren. Visual navigation and obstacle avoidance using a steering potential function. *Robotics and Autonomous Systems*, 54(4):288–299, 2006.

[39] Gerald Keller and Nicoleta Gaciu. *Managerial statistics*. South-Western Cengage Learning, 2012.

[40] Donald E Kirk. *Optimal control theory: an introduction*. Courier Corporation, 2012.

[41] G Ayorkor Korsah, Anthony Stentz, and M Bernardine Dias. A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research*, 32(12):1495–1512, 2013.

[42] Mikio Kubo and Hiroshi Kasugai. The precedence constrained traveling salesman problem. *Journal of the Operations Research Society of Japan*, 34(2): 152–172, 1991.

[43] Michail G Lagoudakis, Marc Berhault, Sven Koenig, Pinar Keskinocak, and Anton J Kleywegt. Simple auctions with performance guarantees for multi-robot task allocation. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 1, pages 698–705. IEEE, 2004.

[44] Michail G Lagoudakis, Evangelos Markakis, David Kempe, Pinar Keskinocak, Anton J Kleywegt, Sven Koenig, Craig A Tovey, Adam Meyerson, and Sonal Jain. Auction-based multi-robot routing. In *Robotics: Science and Systems*, volume 5, pages 343–350. Rome, Italy, 2005.

[45] Eduardo Lalla-Ruiz, Christopher Expósito-Izquierdo, Shervin Taheripour, and Stefan Voß. An improved formulation for the multi-depot open vehicle routing problem. *OR spectrum*, 38(1):175–187, 2016.

[46] Gilbert Laporte. Fifty years of vehicle routing. *Transportation Science*, 43(4): 408–416, 2009.

[47] Eugene L Lawler, Jan Karel Lenstra, AHG Rinnooy Kan, David B Shmoys, et al. *The traveling salesman problem: a guided tour of combinatorial optimization*, volume 3. Wiley New York, 1985.

[48] Jan Karel Lenstra and AHG Kan. Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227, 1981.

[49] Xu Li, Inria Lille, Rafael Falcon, Amiya Nayak, and Ivan Stojmenovic. Servicing wireless sensor networks by mobile robots. *IEEE Communications Magazine*, 50(7), 2012.

[50] Lantao Liu and Dylan A Shell. Large-scale multi-robot task allocation via dynamic partitioning and distribution. *Autonomous Robots*, 33(3):291–307, 2012.

[51] Lingzhi Luo, Nilanjan Chakraborty, and Katia Sycara. Distributed algorithms for multirobot task assignment with task deadline constraints. *IEEE Transactions on Automation Science and Engineering*, 12(3):876–888, 2015.

[52] Monirehalsadat Mahmoudi and Xuesong Zhou. Finding optimal solutions for vehicle routing problem with pickup and delivery services with time windows: A dynamic programming approach based on state–space–time network representations. *Transportation Research Part B: Methodological*, 89: 19–42, 2016.

[53] Neil Mathew, Stephen L Smith, and Steven L Waslander. Planning paths for package delivery in heterogeneous multirobot teams. *IEEE Transactions on Automation Science and Engineering*, 12(4):1298–1308, 2015.

[54] Manuel Mazo Jr and Ming Cao. Asynchronous decentralized event-triggered control. *Automatica*, 50(12):3197–3203, 2014.

[55] Kurt Mehlhorn and Peter Sanders. *Algorithms and data structures: The basic toolbox*. Springer Science & Business Media, 2008.

[56] Chiung Moon, Jongsoo Kim, Gyunghyun Choi, and Yoonho Seo. An efficient genetic algorithm for the traveling salesman problem with precedence constraints. *European Journal of Operational Research*, 140(3):606–617, 2002.

[57] Sangwoo Moon, Eunmi Oh, and David Hyunchul Shim. An integral framework of task assignment and path planning for multiple unmanned aerial vehicles in dynamic environments. *Journal of Intelligent & Robotic Systems*, 70(1-4):303–313, 2013.

[58] Eleonora Morganti, Saskia Seidel, Corinne Blanquart, Laetitia Dablanc, and Barbara Lenz. The impact of e-commerce on final deliveries: alternative parcel delivery services in france and germany. *Transportation Research Procedia*, 4:178–190, 2014.

[59] David D Morrison, James D Riley, and John F Zancanaro. Multiple shooting method for two-point boundary value problems. *Communications of the ACM*, 5(12):613–614, 1962.

[60] Chase C Murray and Amanda G Chu. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transportation Research Part C: Emerging Technologies*, 54:86–109, 2015.

[61] Michael Otte and Nikolaus Correll. Any-com multi-robot path-planning with dynamic teams: Multi-robot coordination under communication constraints. In *Experimental Robotics*, pages 743–757. Springer, 2014.

[62] Christos H Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Courier Corporation, 1982.

[63] Daegil Park, Jonghui Han, and Wan Kyun Chung. Optimal mission planning for underwater environment. In *Ubiquitous Robots and Ambient Intelligence (URAI), 2013 10th International Conference on*, pages 554–555. IEEE, 2013.

[64] Jufeng Peng and Srinivas Akella. Coordinating multiple robots with kinodynamic constraints along specified paths. *The International Journal of Robotics Research*, 24(4):295–310, 2005.

[65] Mathew D Penrose. The longest edge of the random minimal spanning tree. *The Annals of Applied Probability*, pages 340–361, 1997.

[66] Christian Prins. A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31(12):1985–2002, 2004.

[67] Sivakumar Rathinam, Raja Sengupta, and Swaroop Darbha. A resource allocation algorithm for multivehicle systems with nonholonomic constraints. *IEEE Transactions on Automation Science and Engineering*, 4(1):98–104, 2007.

[68] Martin Saska, Tomas Baca, Justin Thomas, Jan Chudoba, Libor Preucil, Tomas Krajnik, Jan Faigl, Giuseppe Loianno, and Vijay Kumar. System for deployment of groups of unmanned micro aerial vehicles in gps-denied environments using onboard visual relative localization. *Autonomous Robots*, 41(4):919–944, 2017.

[69] Halil Savuran and Murat Karakaya. Efficient route planning for an unmanned air vehicle deployed on a moving carrier. *Soft Computing*, 20(7): 2905–2920, 2016.

[70] Ke Shi, Hongsheng Chen, and Yao Lin. Probabilistic coverage based sensor scheduling for target tracking sensor networks. *Information Sciences*, 292: 95–110, 2015.

[71] Tal Shima, Steven J Rasmussen, Andrew G Sparks, and Kevin M Passino. Multiple task assignments for cooperating uninhabited aerial vehicles using genetic algorithms. *Computers & Operations Research*, 33(11):3252–3269, 2006.

[72] Stephen L Smith and Francesco Bullo. Monotonic target assignment for robotic networks. *IEEE Transactions on Automatic Control*, 54(9):2042–2057, 2009.

[73] Michaël Soulignac. Feasible and optimal path planning in strong current fields. *IEEE Transactions on Robotics*, 27(1):89–98, 2011.

[74] George Steiner. On the complexity of dynamic programming for sequencing problems with precedence constraints. *Annals of Operations Research*, 26(1): 103–123, 1990.

[75] Paolo Toth and Daniele Vigo. *Vehicle routing: problems, methods, and applications*. Society for Industrial and Applied Mathematics, 2014.

[76] Matthew Turpin, Kartik Mohta, Nathan Michael, and Vijay Kumar. Goal assignment and trajectory planning for large teams of interchangeable robots. *Autonomous Robots*, 37(4):401–415, 2014.

[77] Xingyin Wang, Stefan Poikonen, and Bruce Golden. The vehicle routing problem with drones: Several worst-case results. *Optimization Letters*, 11(4): 679–697, 2017.

[78] You-Chiun Wang. A two-phase dispatch heuristic to schedule the movement of multi-attribute mobile sensors in a hybrid wireless sensor network. *IEEE Transactions on Mobile Computing*, 13(4):709–722, 2014.

[79] Mark Allen Weiss. *Data structures and algorithm analysis in C*. Benjamin/Cummings California, 1993.

[80] Weisheng Yan, Xiaoshan Bai, Xingguang Peng, Lei Zuo, and Jiguo Dai. The routing problem of autonomous underwater vehicles in ocean currents. In *OCEANS 2014-TAIPEI*, pages 1–6. IEEE, 2014.

[81] Jingjin Yu, Soon-Jo Chung, and Petros G Voulgaris. Target assignment in robotic networks: Distance optimality guarantees and hierarchical strategies. *IEEE Transactions on Automatic Control*, 60(2):327–341, 2015.

[82] YoungSu Yun and Chiung Moon. Genetic algorithm approach for precedence-constrained sequencing problems. *Journal of Intelligent Manufacturing*, 22 (3):379–388, 2011.

[83] Wanqing Zhao, Qinggang Meng, and Paul WH Chung. A heuristic distributed task allocation method for multivehicle multitask problems and its application to search and rescue scenario. *IEEE Transactions on Cybernetics*, 46(4): 902–915, 2016.

[84] Daqi Zhu, Huan Huang, and Simon X Yang. Dynamic task assignment and path planning of multi-auv system based on an improved self-organizing map and velocity synthesis method in three-dimensional underwater workspace. *IEEE Transactions on Cybernetics*, 43(2):504–514, 2013.

[85] Lei Zuo, Jicheng Chen, Weisheng Yan, and Yang Shi. Time-optimal coverage control for multiple unicycles in a drift field. *Information Sciences*, 2016.

Summary

Multi-vehicle systems have been increasingly exploited to accomplish difficult and complex missions, where effective and efficient coordinations of the vehicles can greatly improve the team's performance. Motivated by need from practice, we study the multi-vehicle task assignment in various challenging environments.

We first investigate the task assignment for multiple vehicles in a time-invariant drift field. The objective is to employ the vehicles to visit a set of target locations in the drift field while trying to minimize the vehicles' total travel time. Using optimal control theory, a path planning algorithm is designed to generate the time-optimal path for a vehicle to travel between any two prescribed locations in a drift field. The path planning algorithm provides the cost matrix for the target assignment, and generates routes once the target locations are assigned to the vehicles. Using tools from graph theory, a lower bound on the optimal solution is found, which can be used to measure the proximity of a solution from the optimal. We propose several clustering-based task assignment algorithms in which two of them guarantee that all the target locations will be visited within a computable maximal travel time, which is at most twice of the optimal when the cost matrix is symmetric.

In addition, we extend the multi-vehicle task assignment study in a time-invariant drift field with obstacles. The vehicles have different capabilities, and each kind of vehicles needs to visit a certain type of target locations; each target location might have the demand to be visited more than once by different kinds of vehicles. A path planning method has been designed to enable the vehicles to move between two prescribed locations in a drift field with minimal time while avoiding obstacles. This task assignment problem is shown to be NP-hard, and a distributed task assignment algorithm has been designed, which can achieve near-optimal solutions to the task assignment problem.

Furthermore, we study the task assignment problem in which multiple dispersed heterogeneous vehicles with limited communication range need to visit

a set of target locations while trying to minimize the vehicles' total travel distance. Each vehicle initially has the position information of all the targets and of those vehicles that are within its limited communication range, and each target demands a vehicle with some specified capability to visit it. We design a decentralized auction algorithm which first employs an information consensus procedure to merge the local information carried by each communication-connected vehicle subnetwork. Then, the algorithm constructs conflict-free target assignments for the communication-connected vehicles, and guarantees that the total travel distance of the vehicles is at most twice of the optimal when the communication network is initially connected.

In the end we exploit the precedence-constrained task assignment problem for a truck and a micro drone to deliver packages to a set of dispersed customers subject to precedence constraints that specify which customers need to be visited before which other customers. The truck is restricted to travel in a street network and the micro drone, restricted by its loading capacity and operation range, can fly from the truck to perform the last mile package deliveries. The objective is to minimize the time to serve all the customers respecting every precedence constraint. The problem is shown to be NP-hard, and a lower bound on the optimal time to serve all the customers is constructed by using tools from graph theory. Integrating with a topological sorting technique, several heuristic task assignment algorithms are constructed to solve the task assignment problem.

Samenvatting

Multi-voertuig systemen worden in toenemende mate benut voor het volbrengen van moeilijke en ingewikkelde missies, waarbij effectief en efficiënt coördineren van de voertuigen kan leiden tot een aanzienlijke toename van de groepsprestatie. Gemotiveerd door een behoefte uit de praktijk, bestuderen we de taakverdeling van multi-voertuig systemen in verschillende uitdagende scenarios.

Als eerst onderzoeken we de taakverdeling voor meerdere voertuigen in een tijdsinvariant drijfveld. Het doel is om de rijtuigen in te zetten voor het bezoeken van een verzameling van doellocaties in een drijfveld en tegelijkertijd de reistijd van de rijtuigen te minimaliseren. Gebruik makend van optimale regeltheorie, wordt een routeplanning algoritme ontworpen voor het genereren van een tijdsoptimale route voor het voertuig tussen twee willekeurig gedefinieerde locaties in een drijfveld. Het routeplanning algoritme voorziet de kosten-matrix voor de taakverdeling en genereert routes voor de voertuigen nadat deze doellocaties zijn toegekend. Met behulp van hulpmiddelen van graaftheorie wordt een ondergrens voor de optimale oplossing gevonden. Hiermee kan dan de verkregen oplossing worden vergeleken met de optimale oplossing. We dragen verschillende op clustering gebaseerde taakverdeling algoritmes voor, waarvan twee garanderen dat alle doellocaties worden bezocht binnen een maximale reistijd. Deze is hooguit twee keer de optimale reistijd indien de kosten-matrix symmetrisch is.

Daarnaast breiden we het multi-voertuig taakverdeling probleem uit naar een tijds-invariante drijfveld met obstakels. De voertuigen hebben verschillende capaciteiten en elk soort voertuig dient een bepaald type doellocatie te bezoeken; elke doellocatie kan meerdere keren worden bezocht door verschillende types voertuigen. Een routeplanning methode wordt ontworpen opdat de voertuigen in een minimale tijd tussen twee gedefinieerde locaties in het drijfveld kunnen bewegen en obstakels te vermijden. Het is bewezen dat het taakverdeling probleem NP-hard is en een gedistribueerd algoritme is voorgedragen dat vrijwel optimale oplossingen van dit probleem kan behalen.

Verder bestuderen we het taakverdeling probleem waarbij meerdere heterogene voertuigen met een beperkt communicatie bereik verspreid zijn over het drijfveld. Elk voertuig heeft initieel de positie informatie van alle doellocaties en ook van de voertuigen die binnen haar communicatie bereik is. Elke doellocatie vraagt naar een voertuig met een bepaalde capaciteit. We ontwerpen een gedecentraliseerd veiling algoritme die eerst een informatie consensus procedure gebruikt om de locale informatie, van elke door communicatie verbonden deelnetwerken van voertuigen, samen te voegen. Hierna wordt een conflict vrije taakverdeling geconstrueerd voor de communicatie-verbonden voertuigen en wordt gegarandeerd dat de totale reis afstand van de voertuigen hoogstens twee keer de optimale afstand is wanneer het communicatie netwerk inititeel verbonden is.

Als laatste, bekijken het prioriteit gebonden taakverdeling probleem voor een truck en een micro drone voor het bezorgen van pakketen aan een set van klanten. De klanten zijn onderhevig aan prioriteits voorwaarden die aangeven welke klanten eerst bezocht dienen te worden. De truck kan alleen in een weg netwerk voortbewegen en de micro drone, gelimiteerd door de laadcapaciteit en werkbereik, kan vanuit de truck vliegen om de laatste afstand af te leggen voor het bezorgen van de pakketen. Het doel is om de tijd te minimaliseren voor het dienen van alle klanten, rekening houdend met de prioriteit voorwaarden. Het probleem is NP-hard en een ondergrens voor de optimale tijd is gevonden met behulp van hulpmiddelen uit graaftheorie. Verschillende heuristische taakverdeling algoritmes zijn geconstrueerd en samengevoegd met een topologische soorteer techniek om het probleem op te lossen.