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

Bai, Xiaoshan; Yan, Weisheng; Cao, Ming; Xue, Dong

Published in:
IET Control Theory and Applications

DOI:
10.1049/iet-cta.2018.6125

Publication date:
2019

IET Control Theory & Applications

Special Issue: Distributed Optimisation and Learning for Networked Systems

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

ISSN 1751-8644
Received on 30th September 2018
Revised 18th December 2018
Accepted on 29th January 2019
E-First on 11th April 2019
doi: 10.1049/iet-cta.2018.6125
www.ietdl.org

Xiaoshan Bai[1,2] ✉, Weisheng Yan[2], Ming Cao[1], Dong Xue[3]

[1]*Faculty of Science and Engineering, University of Groningen, Groningen 9747 AG, The Netherlands*

[2]*School of Marine Science and Technology, Northwestern Polytechnical University, 127 West Youyi Road, Xi'an 710072, People's Republic of China*

[3]*Department of Electrical and Computer Engineering, Technical University of Munich (TUM), Arcisstrasse 21, D-80209 Munich, Germany*

✉ *E-mail: xiaoshan.bai@rug.nl*

Abstract: This study investigates the task assignment problem where a fleet of dispersed vehicles needs to visit multiple target locations in a time-invariant drift field with obstacles while trying to minimise the vehicles' total travel time. The vehicles have different capabilities, and each kind of vehicles can visit a certain type of the target locations; each target location might require to be visited more than once by different kinds of vehicles. The task assignment problem has been proven to be NP-hard. A path planning algorithm is first designed to minimise the time for a vehicle to travel between two given locations through the drift field while avoiding any obstacle. 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, a distributed algorithm is proposed to assign the target locations to the vehicles using only local communication. The algorithm guarantees that all the visiting demands of every target will be satisfied within a total travel time that is at worst twice of the optimal when the travel cost matrix is symmetric. Numerical simulations show that the algorithm can lead to solutions close to the optimal.

1 Introduction

Multi-vehicle systems have been increasingly employed to perform complex and dangerous missions in logistics, terrain mapping, and environmental monitoring, where coordination of the vehicles is generally necessary to improve the team performance [1]. The group mission for a multi-vehicle system is usually divided into several subtasks such that specialised individual vehicles can deal with each subtask concurrently [2]. A typical multi-vehicle task assignment scenario is the deployment of a fleet of vehicles to visit a group of target locations while minimising some objective, such as the vehicles' maximum travel time [3] and total travel distance [4]. For large-scale multi-robot task allocation, Liu and Shell [5] designed centralised and decentralised algorithms through coarsening and partitioning the utility/cost matrix. Gossip algorithms were proposed in [6] to minimise the vehicles' maximum execution time in heterogeneous multi-vehicle routing problems. However, most of these research works mainly focus on how to effectively and efficiently assign tasks to the vehicles under the assumption that the vehicles can travel freely between two arbitrary given locations where no external disturbances exist.

When winds/currents or obstacles exist in the operation environment, the multi-vehicle task assignment problem consists of two sub-problems, namely how to optimally navigate a vehicle through two given locations and how to assign subtasks as sequences of target locations to individual vehicles [7]. There are some research works investigating the path planning for the employed vehicles to avoid obstacles [8–11]. In [8], distance functions were studied for robot path planning in the presence of obstacles where the main principle is to express obstacle avoidance in terms of distances between the obstacles and the part of the robot that may potentially crash into the obstacles. The path planning of multiple robots with kinodynamic constraints along specified paths was addressed in [9] 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 [10]. The angular acceleration 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 [11]. Some research works studied the path planning for the employed vehicles in a drift filed [12–14]. Considering the ocean currents and obstacles, Han *et al.* [12] planned the path for an autonomous underwater vehicle (AUV) to optimally travel between two prescribed locations by simply following the straight line connecting the two locations. In a similar manner, the velocity synthesis approach was used by Zhu *et al.* [13] to enable multiple AUVs to reach several target locations along the shortest paths in a time-varying (in a discrete time scale) 3D underwater environment. The path planning methods minimising a vehicle's travel distance between two given locations in [12, 13] do not necessarily lead to the minimal time for the vehicle to travel between the locations. Later on, Eichhorn [14] used grid-modelling based graph methods for vehicle path planning in a time-varying environment. However, little attention has been paid to the path planning for vehicles in a drift field with obstacles, not even to mention coordinating multi-vehicles to compete for some tasks in a drift field with obstacles.

In our previous work [15], the sensor delivery task assignment for multiple AUVs has been performed in temporally piece-wise constant ocean currents aiming at optimising the vehicles' total travel time. In addition, for vehicles operating in a time-varying drift field, a co-evolutionary multi-population genetic algorithm (GA) has been designed in [16] for multiple vehicles to deliver products to a set of target locations. In this paper, 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 heterogeneous vehicles while trying to minimise the vehicles' total travel time. The vehicles are heterogeneous in the sense that each kind of vehicles carrying one specified sensor can visit a certain type of targets, and some target locations might require to be visited by vehicles with different capabilities in order to, e.g. measure ocean salinity, temperature and conductivity. To solve the problem, we have designed a path planning method and a distributed task assignment algorithm (DTAA). The path planning

algorithm can generate the time-optimal path for a vehicle to travel between two prescribed locations in a time-invariant drift field while avoiding any obstacle, which provides the travel cost matrix to be used later for the task assignment. Part of the results has been presented in [17]. Our main contributions compared with [17] are as follows. First, we prove that the task assignment problem is NP-hard, which leads to the necessity for constructing some efficient suboptimal task assignment algorithms. The NP-hardness property of the problem implies that possibly unacceptable long computation time is required to achieve the optimal solution when the numbers of vehicles and targets grow, and the tight lower bound on the optimal solution proposed in [17] can be used to measure the performance of a suboptimal task assignment algorithm. Second, the proposed DTAA is fully distributed where every vehicle cooperates with the communication-connected local neighbours while guaranteeing that all the visiting demands of the targets are satisfied within the vehicles' total travel time, which, when the travel cost matrix is symmetric, is at most twice of the optimal one. Last, extensive numerical experiments on the DTAA are conducted in comparison with the popular GAs and a greedy task assignment algorithm (GTAA), which shows that the DTAA can quickly lead to satisfying solutions close to the optimal.

The rest of this paper is organised as follows. In Section 2, the formulation for the multi-vehicle task assignment problem is given. Section 3 presents the path planning algorithm which generates the travel cost matrix for the task assignment, and in Section 4 the task assignment problem is analysed. Section 5 presents the DTAA and Section 6 analyses the performance of the DTAA. We present the simulation results in Section 7 and conclude the paper in Section 8.

2 Problem formulation

To formulate the problem rigorously, we first introduce some notations from graph theory [18]. A graph $\mathbb{G} = (\mathscr{V}, \mathscr{E})$ consists of a set of vertices $\mathscr{V}$, and a set of edges $\mathscr{E}$. A *directed* graph, or a digraph in short, is a graph where each edge in $\mathscr{E}$ is denoted by an ordered pair of vertices. Let $(i, j), i, j \in \mathscr{V}$, denote an edge which starts at vertex i and ends at vertex j. In this paper, we only consider *simple* graphs, i.e. graphs that do not contain self-loops $(i, i), \forall i \in \mathscr{V}$. A graph $\mathbb{G}$ is *undirected* if $(i, j) \in \mathscr{E} \Leftrightarrow (j, i) \in \mathscr{E}$ for each pair of $i, j \in \mathscr{V}$. Undirected graphs can be treated as special directed graphs by replacing each undirected edge be two directed edges in opposite directions. 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 $e_i = (v_{i-1}, v_i), 0 < i \leq n$. A *path* is a walk with distinct vertices. If a walk $W = v_0 v_1 \ldots v_n$ is such that $n \geq 3, v_0 = v_n$, and the vertices $v_i, 0 \leq i < n$, 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 $i, j \in \mathscr{V}$. A graph $\mathbb{G}$ is a tree if it is acyclic and connected.

The definition of the arborescence of a digraph in graph theory is given as below.

Definition 1 (Arborescence): An *arborescence* [19] is a digraph with a single root in which, there is exactly one directed path from the root to every other vertex.

We extend the concept of the arborescence of a digraph with a single root to a general one with multiple roots.

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

Now we are ready to define the research problem.

2.1 Problem description

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 minimise the vehicles' total travel time. Each vehicle has one specified capability to visit a certain kind of targets and several vehicles can 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. Each vehicle initially has the position information of all the targets in a map, and it 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. It is assumed that the communication network of the vehicles is initially connected.

2.2 Formulation as an optimisation problem

We use the vector $\vec{v}_c = [v_{cx}(x, y), v_{cy}(x, y)]^{\mathrm{T}}$ 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 are assumed to move with the constant speed v relative to a static drift field [20]. We assume that the vehicles are free of turning ratio constraints since the dimension of the drift field is significantly larger than the vehicles' size. The kinematics of each vehicle i are

$$\dot{x}_i = v\cos\psi_i + v_{cx}(x_i, y_i), \quad \dot{y}_i = v\sin\psi_i + v_{cy}(x_i, y_i), \tag{1}$$

where $[x_i, y_i]^{\mathrm{T}}$ is vehicle i's position and ψ_i is i's navigation angle.

Let $\mathscr{T} = \{1, \ldots, n\}$ be the labelling of the target locations, and $\mathscr{R} = \{1, \ldots, m\}$ be the labelling of the vehicles. We use $\mathscr{V}$ to denote the set of indices of all the vehicles' initial locations, namely $\mathscr{V} = \{n+1, \ldots, n+m\}$, where each vehicle has one capability $a_j \in \mathscr{A}$ with $|\mathscr{A}| \leq m$. Let $\mathscr{R}_q, q \in \{1, \ldots, |\mathscr{A}|\}$, be the vehicle set that contains the indices of those vehicles with the same capability where $\mathscr{R} = \cup_{q=1}^{|\mathscr{A}|} \mathscr{R}_q$ and $\mathscr{R}_q \cap \mathscr{R}_p = \varnothing$ 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 $\mathscr{R}_q$ and otherwise zero. Let $C = (t(i, j))_{(m+n)\times(m+n)}, i, j \in \mathscr{T} \cup \mathscr{V}$, 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 minimised 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 $i \in \mathscr{T} \cup \mathscr{V}$, $j \in \mathscr{T}$ of the starting and ending locations of a vehicle $k \in \mathscr{R}$ 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 minimisation of the vehicles' total travel time for satisfying all the visiting demands of the target locations is to minimise

$$f = \sum_{i \in \mathscr{T} \cup \mathscr{V}, j \in \mathscr{T}, k \in \mathscr{R}} t(i, j)\sigma_{ijk}, \tag{2}$$

subject to

$$\sum_{i \in \mathscr{T} \cup \mathscr{V}, k \in \mathscr{R}_q} \sigma_{ijk} = y_{jq}, \quad \forall j \in \mathscr{T}, \forall \mathscr{R}_q \subseteq \mathscr{R}; \tag{3}$$

$$\sum_{j \in \mathscr{T}, k \in \mathscr{R}_q} \sigma_{ijk} y_{jq} \leq 1, \quad \forall i \in \mathscr{T} \cup \mathscr{V}, \forall \mathscr{R}_q \subseteq \mathscr{R}. \tag{4}$$

Constraint (3) ensures that the visiting demand of each target is satisfied; (4) means that each target location and each vehicle's initial location are departed at most once for each kind of vehicles $\mathscr{R}_q \subseteq \mathscr{R}$.

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

3 Path planning algorithm

In this section, we conduct a path planning algorithm based on grid modelling 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 labelled by ascending integers from 1, 2, 3, ⋯. Potential moving directions for a vehicle located in a

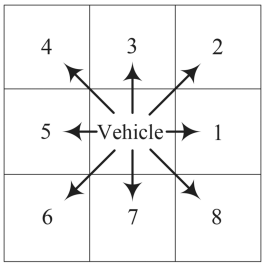

Fig. 1 *Possible moving directions for a vehicle to travel to its neighbour cells*

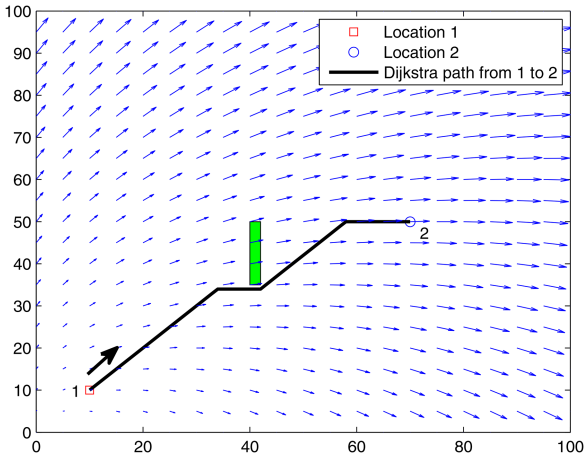

Fig. 2 *Optimal path planning for a vehicle with $v = 1$ to travel between two locations in the drift field $\vec{v}_c = 10^{-2}[0.3x + 0.2y, -0.2x + 0.3y]^{\mathrm{T}}$ with an obstacle coloured in green where the travel time resulting from the Dijkstra algorithm is $t(1,2) = 66.5618\ s$*

cell are shown in Fig. 1, where the angle θ_{net} of the net velocity for a vehicle to move to the neighbouring cell $i \in \{1, \ldots, 8\}$ 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 neighbouring cells with odd number labels as shown in Fig. 1, while the distance is $\sqrt{2}l$ to move to those with even number labels.

The feasibility of moving into a neighbour cell depends not only on whether obstacles occupy the cell, but also on the strength and orientation of currents at the vehicle's current location. The currents within each square cell are assumed to be the same, which is natural when the operation area is discretised with high resolution. We obtained the magnitude of the optimal net velocity of an underwater vehicle in [15], and calculated the rate of change of the vehicle's optimal navigation angle in a drift field in [7] under the assumption that $v > v_c$. Here, we generalise the analysis. Let v_{net} be the magnitude of the net velocity $[\dot{x}_i, \dot{y}_i]^{\mathrm{T}}$ of vehicle i with the motion dynamics of (1), and θ_c be the heading of the currents $\vec{v}_c$. Then, in view of (1), it holds that

$$v_{\text{net}}^2 - 2v_c v_{\text{net}} \cos(\theta_{\text{net}} - \theta_c) - (v^2 - v_c^2) = 0. \tag{5}$$

According to the discriminant of roots of a quadratic equation, the feasible solution v_{net} to (5) exists when

$$|\sin(\theta_{\text{net}} - \theta_c)| \leq v/v_c\,. \tag{6}$$

Based on (6) and the obstacle information, one can determine whether a vehicle can move to a certain neighbour cell. The time for the vehicle to travel from its location to a neighbour cell is the division of the travel distance by v_{net} which can be obtained by (14) in [15]; the time is taken to be infinity if the neighbour cell is impossible for the vehicle to visit due to obstacles or currents.

For each cell, we first use (5) and (6) and the obstacle information to calculate the time for a vehicle to move from the cell to each neighbour 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 [21, pp. 196–206]. 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 (2). 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. 2, where the vehicle avoids the obstacle effectively while adjusting its path to the currents.

Remark 1: The cost matrix C is in general asymmetric, namely $t(i,j) \neq t(j,i)$, and satisfies $t(i,k) \leq t(i,j) + t(j,k)$ as Dijkstra's algorithm finds the path with the minimum travel time between any two locations i and k.

4 Problem analysis

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 travelling 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. It is a relaxation of the vehicle routing problem (VRP) without the capacity constraint [22]. 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 [23]. If the vehicles do not need to return to the starting depots, the VRP is called the open VRP which is also NP-hard [24]. When the number of the vehicles is one, the MTSP is reduced to the travelling salesman problem (TSP) and the open MTSP (OMTSP) 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 [25, pp. 199–200]. The requirement of returning to the starting city does not change the computational complexity of the problem. So the OMTSP is NP-hard as well [26].

Proposition 1: The task assignment problem (2) is NP-hard

Proof: We first prove the NP-hardness of (2) with homogeneous vehicles by showing that: (i) the OMTSP can be reduced to an instance of (2) with homogeneous vehicles in polynomial time and (ii) an optimal solution to (2) with homogeneous vehicles provides an optimal OMTSP solution.

Let $\mathcal{G}' = (V', E', C')$ with $|V'| = n + 1$ being an input to the OMTSP, where n is the number of dispersed target locations. To prove (i), we give a polynomial-time transformation of $\mathcal{G}'$ into an input of $\mathcal{G} = (V, E, C)$ to (2) with homogeneous vehicles, shown in Fig. 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 $t(V_j, V_k) = 0, \forall V_j, V_k \in \mathcal{R}$, 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 (2) 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 (2) with

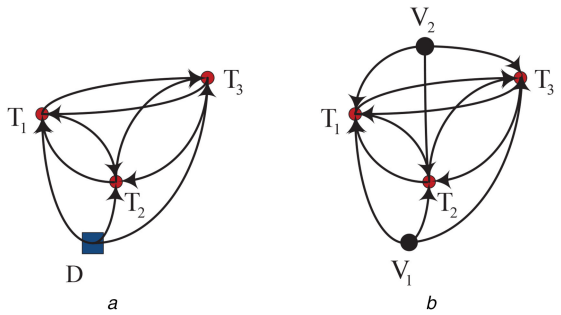

Fig. 3 *Transformation from the OMTSP on graph $\mathcal{G}'$ to the task assignment problem with homogeneous vehicles on graph $\mathcal{G}$ where*
(a) *$\mathcal{G}' = (V', E', C')$,* ***(b)*** *$\mathcal{G} = (V, E, C)$*

Require: Locations of targets in $\mathcal{T}_q$ and vehicles in $\mathcal{R}_q$, the travel cost matrix C for digraph $\mathcal{G}_q$.
Ensure: An MCGA of $\mathcal{G}_q$.
1: Initialize $MCGA \leftarrow \mathcal{R}_q$.
2: **while** $\mathcal{T}_q \neq \emptyset$ **do**
3: $(j^\star, p^\star) \leftarrow \text{argmin}_{(j,p)\in MCGA\times\mathcal{T}_q} t(j,p)$.
4: Add $p^\star$ in $MCGA$ and connect it with $j^\star$ using an edge with weight $t(j^\star, p^\star)$ being the optimal time for a vehicle to travel from $j^\star$ to $p^\star$.
5: $\mathcal{T}_q \leftarrow \mathcal{T}_q \setminus \{p^\star\}$.
6: **end while**

Fig. 4 *Algorithm 1: the extended Prim algorithm for achieving an MCGA of a directed graph*

homogeneous vehicles is also optimal to the OMTSP based on the edge weights of the graph shown in Figs. 3*a* and 3*b*. Thus, (2) 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 (2) can be divided into $|\mathscr{A}|$ sub-problems where subproblem $q \in \{1, \ldots, |\mathscr{A}|\}$ contains the target locations in $\mathcal{T}_q$ needed to be visited by the vehicles in $\mathcal{R}_q$ with the same capability. 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 emphasise why the problem we study is difficult in nature.

Remark 2: A target location might belong to several subproblems when it needs to be visited more than once by vehicles with different capabilities.

4.2 Lower bound on the optimal solution

It can be costly to solve (2) 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 $|\mathscr{A}|$ subgroups $\mathcal{T}_q, q \in \{1, \ldots, |\mathscr{A}|\}$, where $\mathcal{T} = \cup_{q=1}^{|\mathscr{A}|} \mathcal{T}_q$ and the targets within $\mathcal{T}_q$ 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 $\mathcal{T}_q, \forall q \in \mathscr{A}$. 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 [27], used to find a minimum spanning tree for an undirected graph, to obtain a min-cost generalised arborescence (MCGA) for the digraph $\mathcal{G}_q$. The procedure to achieve an MCGA is shown in Algorithm 1 (see Fig. 4).

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 $f^a = \sum_{q=1}^{|\mathscr{A}|} f_q^a$, and employ f^o as the minimum total travel time for (2). Then, we first present a lower bound on f^o.

Theorem 1: The optimal total travel time has a lower bound $f^a \leq f^o$.

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. $f_q^a \leq f_q^o$. 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 $f_q^a \leq f_q^o$.

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

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

Having done the problem analysis, we construct a DTAA in the next section.

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 $\mathcal{T}_q, \forall q \in \mathscr{A}$, to the vehicles in $\mathcal{R}_q$ through using the following distributed auction mechanism and the target ordering principle.

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, $a_j \in \mathscr{A}$ 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 initialised 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 initialised 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 initialises the β_{jr} as

$$c_{jr} = \min_{k \in o_j} t(k, r), \quad \beta_{jr} = j, \tag{7}$$

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

Require: The directly communication-connected vehicle set $\mathscr{R}_N^j$, cost matrix C, target set $\mathscr{T}_q$ and vehicle j' position index p_j.
Ensure: All the targets in $\mathscr{T}_q$ are assigned.
1: $\mathscr{T}_q^u \leftarrow \mathscr{T}_q$.
2: $o_j \leftarrow \{p_j\}$.
3: **while** $\mathscr{T}_q^u \neq \emptyset$ **do**
4: **for** $\forall r \in \mathscr{T}_q^u$ **do**
5: $c_{jr} \leftarrow \min_{k \in o_j} t(k, r)$,
6: $\beta_{jr} \leftarrow j$.
7: **end for**
8: $s \leftarrow 0$.
9: **while** $s < |\mathscr{R}|$ **do**
10: Send c_j and β_j to every vehicle $i \in \mathscr{R}_N^j \setminus \{j\}$.
11: Receive c_i and β_i from every vehicle $i \in \mathscr{R}_N^j \setminus \{j\}$.
12: **for** $\forall r \in \mathscr{T}_q^u$ **do**
13: $i^\star \leftarrow \operatorname{argmin}_{i \in \mathscr{R}_N^j} c_{ir}$,
14: $c_{jr} \leftarrow c_{i^\star r}$,
15: $\beta_{jr} \leftarrow \beta_{i^\star r}$.
16: **end for**
17: $s \leftarrow s + 1$.
18: **end while**
19: $r^\star \leftarrow \operatorname{argmin}_{r \in \mathscr{T}_q^u} c_{jr}$.
20: **if** $j = \beta_{jr^\star}$ **then**
21: $o_{\beta_{jr^\star}} \leftarrow o_{\beta_{jr^\star}} \cup \{r^\star\}$.
22: **end if**
23: $\mathscr{T}_q^u \leftarrow \mathscr{T}_q^u \setminus \{r^\star\}$.
24: **end while**

Fig. 5 *Algorithm 2: distributed auction mechanism for vehicle $j \in \mathscr{R}$*

vehicles outside of $\mathscr{R}_q$, the vehicles in $\mathscr{R}_q$ know the minimum time incurred to visit each target in $\mathscr{T}_q^u$.

Then, for $\forall r \in \mathscr{T}_q^u$, c_{jr} and β_{jr} are updated by vehicle j through local communication with directly connected neighbour vehicles in $\mathscr{R}_N^j = \{i \in \mathscr{R} : dis(i, j) \leq l\}$ as

$$c_{jr} = c_{i^\star r}, \quad \beta_{jr} = \beta_{i^\star r}, \tag{8}$$

where $i^\star = \operatorname{argmin}_{i \in \mathscr{R}_N^j} c_{ir}$, and dis$(i, j)$ is the distance between vehicles i and j, and l is the vehicles' communication range. After at most $|\mathscr{R}|$ synchronised iterative communications, c_j and β_j reach consensus for every vehicle in $\mathscr{R}$. Then, the first target $r^\star$ in $\mathscr{T}_q^u$ to be assigned is

$$r^\star = \operatorname*{argmin}_{r \in \mathscr{T}_q^u} c_{jr}, \tag{9}$$

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

$$o_{\beta_{jr^\star}} = o_{\beta_{jr^\star}} \cup \{r^\star\}, \quad \mathscr{T}_q^u = \mathscr{T}_q^u \setminus \{r^\star\}. \tag{10}$$

The auction process continues until the unassigned target set $\mathscr{T}_q^u$ is empty. The distributed auction mechanism applied to each vehicle j for bidding the targets in $\mathscr{T}_q$ is shown in Algorithm 2 (see Fig. 5).

5.2 Target locations' ordering principle

After the auction operation, all the target locations in $\mathscr{T}$ have been assigned where the target locations assigned to vehicle $j \in \mathscr{R}$ are kept in o_j. Putting the target locations assigned to each vehicle into a sequence to minimise the vehicle's total travel time is in fact the open TSP problem. Let λ_j, initialised 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 initialised 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

$$(r^\star, k^\star) = \operatorname*{argmin}_{1 < k \leq |\lambda_j| + 1, r \in o_j^u} \{t(\lambda_j \oplus_k r) - t(\lambda_j)\}, \tag{11}$$

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

$$\lambda_j = \lambda_j \oplus_{k^\star} r^\star, \quad o_j^u = o_j^u \setminus \{r^\star\}. \tag{12}$$

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.

6 Performance analysis of the DTAA

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

6.1 Convergence performance

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

Theorem 2: The targets in $\mathscr{T}_q, \forall q \in \mathscr{A}$, assigned to each vehicle in $\mathscr{R}_q$ by the DTAA converge to the targets in the vehicle's arborescence resulting from the MCGA of $\mathscr{G}_q$ after at most $|\mathscr{T}_q \parallel \mathscr{R}|$ synchronised communication iterations.

Proof: We first prove that the DTAA produces a stable assignment after at most $|\mathscr{T}_q \parallel \mathscr{R}|$ synchronised iterations for $\mathscr{G}_q$. There are $|\mathscr{T}_q|$ targets to be assigned to $|\mathscr{R}_q|$ vehicles. The DTAA uses the distributed auction mechanism shown in Algorithm 2 (Fig. 5) to assign the targets in $\mathscr{T}_q$ to the vehicles in $\mathscr{R}_q$. After at most $|\mathscr{R}|$ iterations of synchronised communications as shown in lines 10 to 17 of Algorithm 2 (Fig. 5), all the vehicles in $\mathscr{R}$ reach consensus on the bid list c_j keeping the minimum travel cost for the vehicles to visit each unassigned target in $\mathscr{T}_q^u$ 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 $\mathscr{T}_q^u$ 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 $\mathscr{T}_q^u$ being updated as line 23 of Algorithm 2 (Fig. 5). Thus, all the targets in $\mathscr{T}_q$ are assigned to the vehicles in $\mathscr{R}_q$ after at most $|\mathscr{T}_q \parallel \mathscr{R}|$ synchronised 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 $\mathscr{T}_q^u$ is assigned to the vehicle with the smallest travel cost after at most $|\mathscr{R}|$ synchronised communications. That process is in fact the same as the target election operation in lines 3 and 4 of Algorithm 1 (Fig. 4). The arborescence for each vehicle in $\mathscr{R}_q$ is obtained by breaking the zero-weighted edges connecting arbitrary two vehicles

in the MCGA of $\mathcal{G}_q$ obtained by Algorithm 1 (Fig. 4). Herein, the statement is proved. □

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 [28]). 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 1: It holds that $1 \le f^{da}/f^{o}$, *and* $f^{da}/f^{o} \le 2$ *when the travel cost matrix is symmetric.*

Proof: We first prove that $1 \le f_q^{da}/f_q^{o}$. To prove the statement, similar to the multi-vehicle algorithm operating on undirected graphs [28], 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 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 [28]. 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 generalised 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 $f_q^{o} \le f_q^{da}$. 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 $f^{o} \le f^{da}$.

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 generalised arborescence. In view of $f_q^{a} \le f_q^{o}$ in the proof of Theorem 1, it follows that $\sum_{q=1}^{|\mathcal{A}|} f_q^{da} \le 2\sum_{q=1}^{|\mathcal{A}|} f_q^{o}$. 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 Section 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: The task assignment algorithm DTAA guarantees that $f^{\mathrm{DTAA}} \le f^{da}$.

Proof: We first prove that $f_q^{\mathrm{DTAA}} \le f_q^{da}, \forall q \in \mathcal{A}$. The proof is conducted by induction. In Theorem 2, the assignment of the targets in each $\mathcal{T}_q$ resulting from the DTAA is shown to converge to that of the generalised 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 (9). 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 (Fig. 4). Thus, $f_{qj1}^{\mathrm{DTAA}} \le f_{qj1}^{da}$ as $f_{qj1}^{\mathrm{DTAA}} = f_{qj1}^{da}$, 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 $f_{qj|o_j|-1}^{\mathrm{DTAA}} \le f_{qj|o_j|-1}^{da}$, where o_j contains all the targets in the end assigned to vehicle j. As the inequality specified in Remark 1 holds for the optimal travel times between the vertices in $\mathcal{G}$ and according to (11), for the DTAA the marginal travel time incurred by inserting the last target r of o_j into λ_j is

$$\begin{aligned}
\delta f_{qj}^{\mathrm{DTAA}} &= \min_{k \le |\lambda_j|+1} \{t(\lambda_j \oplus_k r) - t(\lambda_j)\} \\
&= \min \{ \min_{k \le |\lambda_j|-1} (t(\lambda_j^k, r) + t(r, \lambda_j^{k+1}) \\
&\quad - t(\lambda_j^k, \lambda_j^{k+1})), t(p_j, r) + t(r, \lambda_j^1) - t(p_j, \lambda_j^1), \\
&\quad t(\lambda_j^{|\lambda_j|}, r)\} \\
&\le \min_{k \le |\lambda_j|-1} \{t(\lambda_j^k, r) + t(r, \lambda_j^k), t(r, \lambda_j^{k+1}) \\
&\quad + t(\lambda_j^{k+1}, r)\},
\end{aligned} \tag{13}$$

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

$$\delta f_{qj}^{da} = t(r, \lambda_j^{k^{\star}}) + \min_{k \le |\lambda_j|} t(\lambda_j^k, r), \tag{14}$$

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

Combining (13) and (14) and in view of $f_{qj|o_j|-1}^{\mathrm{DTAA}} \le f_{qj|o_j|-1}^{da}$, we get

$$\begin{aligned}
f_{qj|o_j|}^{\mathrm{DTAA}} &= f_{qj|o_j|-1}^{\mathrm{DTAA}} + \delta f_{qj}^{\mathrm{DTAA}} \\
&\le f_{qj|o_j|-1}^{da} + \delta f_{qj}^{da}.
\end{aligned} \tag{15}$$

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

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

Remark 3: Theorem 3 gives the worst case performance of the DTAA compared with the optimal solution as f^{da}/f^{o}, *which generalises the upper bound for the task assignment problem with a symmetric cost matrix in [29]. Furthermore, based on Lemma 1 and Theorem 3, the upper bound of* 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.

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 [30] and a GTAA where vehicles always move towards the nearest target. For each pair of target subset $\mathcal{T}_q$ and the corresponding vehicle subset $\mathcal{R}_q$, $q \in \{1, \ldots, |\mathcal{A}|\}$, 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 [31]. 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 each task assignment algorithm is quantified by

$$q = \frac{f}{f^a}, \tag{16}$$

where f is the objective value in (2) and f^a is a lower bound on the optimal solution of the problem as shown in Theorem 1. Thus, a value of the ratio q closer to 1 means a better performance of the solution.

The task assignment algorithms DTAA, GA, and the GTAA are first integrated with the designed path planning algorithm to assign 10 vehicles under different heterogeneities $|\mathscr{A}|$ with $v = 1$ to satisfy the visiting demands of 50 target locations in a square drift field with edge length 10^3 m and

$$\vec{v}_c = 10^{-3}[0.3x + 0.2y,\ -0.2x + 0.3y]^{\mathrm{T}}.$$

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 $|\mathscr{A}| \in \{1, 3, 5, 7, 10\}$, 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 $|\mathscr{A}|$ are shown in Fig. 6, and the average variance of q and the average computation time of the algorithms are shown in Tables 1 and 2, respectively. First, the average q of the DTAA and GTAA displayed in Fig. 6 are within 1.3 times of the optimal under different $|\mathscr{A}|$ while the average q of the GA decreases and the average q of the GTAA increases when increasing $|\mathscr{A}|$, which shows stable performance of the DTAA. When $|\mathscr{A}|$ increases, the number of targets within each target subset $\mathscr{T}_q, q \in \{1, \ldots, |\mathscr{A}|\}$, and the number of vehicles within each vehicle subset $\mathscr{R}_q$ generally decrease, thus leading to a smaller problem size for assigning the targets in each $\mathscr{T}_q$ to the vehicles in $\mathscr{R}_q$. The reason that GA generally performs better when $|\mathscr{A}|$ increases might be due to the resulting smaller problem size. Second, the DTAA far betters than the GA when $|\mathscr{A}| = 1$ while the GA is competitive with the DTAA with the increase of $|\mathscr{A}|$ while that is totally opposite when comparing the DTAA with the GTAA. However, Table 2 shows that the average computation time of the GA increases with the increase of $|\mathscr{A}|$, 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 $|\mathscr{A}|$ as some target locations need to be visited more than once by vehicles with different capabilities and more visiting demands might occur when increasing $|\mathscr{A}|$. Thus, more computation time is needed to find solutions when increasing $|\mathscr{A}|$. The average variances of the solution qualities q of the DTAA is also generally relatively smaller than those of the GA and the GTAA under different $|\mathscr{A}|$ for $n50m10$ as shown in Table 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, GTAA, and the GA on the instances with larger problem sizes where 10 vehicles under different $|\mathscr{A}|$ 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 $|\mathscr{A}| \in \{1, 3, 5, 7, 10\}$. The average q of the solutions are shown in Figs. 7 and 8, and the variances of q and the average computation times of the algorithms are shown in Tables 1 and 2, respectively. The average q of the DTAA under each $|\mathscr{A}|$ shown in Figs. 7 and 8 are within 1.5 times of the optimal as those in Fig. 6, which displays the satisfying and stable performance of the DTAA. The changing trends of the average q of the GA and the GTAA shown in Figs. 7 and 8 are generally the same, which decreases with the increase of $|\mathscr{A}|$. However, the GA and the GTAA become worser with an increase of problem size from $n50m10$ to $n70m10$ and $n90m10$, where the DTAA far betters the GA and the GTAA under each $|\mathscr{A}|$ as shown in Fig. 8. In addition, Table 2 shows that the mean computation times of the DTAA on

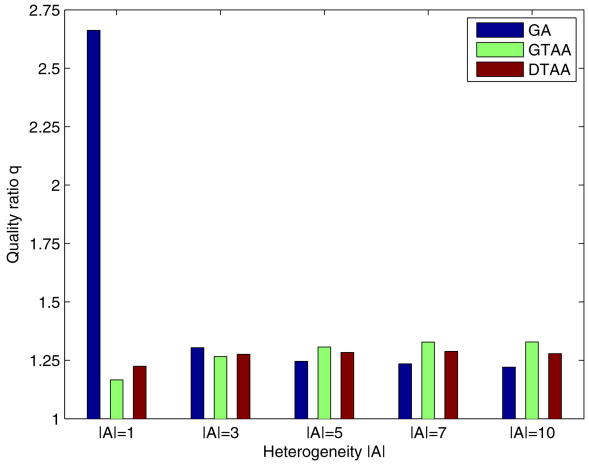

Fig. 6 *Average performance of the algorithms for 10 vehicles under different heterogeneities $|\mathscr{A}|$ to satisfy the visiting demands of 50 target locations in the drift field*

Table 1 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 $|\mathscr{A}|$ where $n50m10$ means 10 vehicles need to satisfy the visiting demands of 50 target locations

I	$\|\mathscr{A}\|$	1	3	5	7	10
$n50m10$	GA	0.4975	0.0210	0.0036	0.0025	0.0015
	GTAA	0.0029	0.0041	0.0025	0.0026	0.0020
	DTAA	0.0035	0.0030	0.0024	0.0024	0.0014
$n70m10$	GA	0.0515	0.1015	0.0453	0.0257	0.0018
	GTAA	0.0299	0.0054	0.0044	0.0022	0.0027
	DTAA	0.0026	0.0017	0.0012	0.0013	0.0009
$n90m10$	GA	0.0297	0.1086	0.0700	0.0570	0.0262
	GTAA	0.0246	0.0045	0.0037	0.0024	0.0017
	DTAA	0.0025	0.0013	0.0011	0.0009	0.0013

Table 2 Average corresponding computation time (s) of the algorithms to get the solution to the task assignment problem under different heterogeneities $|\mathscr{A}|$ and different instances (I)

I	$\|\mathscr{A}\|$	1	3	5	7	10
$n50m10$	GA	37.04	41.12	49.66	58.27	72.74
	GTAA	0.08	0.12	0.17	0.21	0.29
	DTAA	0.19	0.32	0.46	0.60	0.82
$n70m10$	GA	65.06	81.63	105.94	128.53	156.09
	GTAA	0.12	0.17	0.24	0.30	0.40
	DTAA	0.29	0.48	0.71	0.93	1.25
$n90m10$	GA	104.55	134.19	172.45	213.34	273.92
	GTAA	0.15	0.22	0.31	0.40	0.53
	DTAA	0.38	1.19	2.29	3.20	4.40

$n70m10$ 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 1 the variances of q of the algorithms decrease when $|\mathscr{A}|$ increases, which might be due to less cooperation among the vehicles when increasing their heterogeneities.

8 Conclusion

This paper has studied 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 constructed to enable the vehicles to travel between two prescribed locations in a drift fields with the minimal time while avoiding any obstacle.

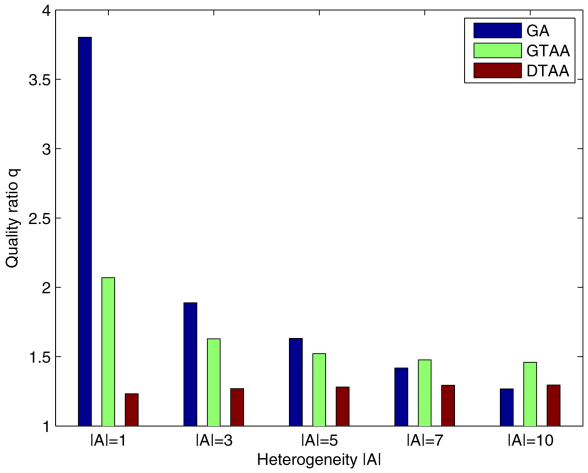

Fig. 7 *Average performance of the algorithms for 10 vehicles under different heterogeneities $|\mathcal{A}|$ to satisfy the visiting demands of 70 target locations in the drift field*

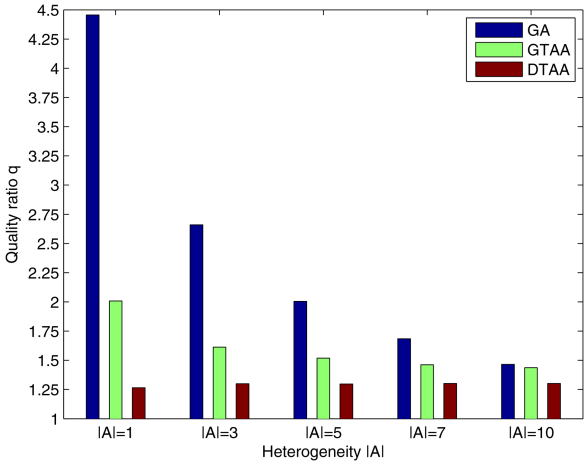

Fig. 8 *Average performance of the algorithms for 10 vehicles under different heterogeneities $|\mathcal{A}|$ to satisfy the visiting demands of 90 target locations in the drift field*

The travel cost matrix resulting from the path planning method provides the route information for the target location assignment. In addition, a DTAA 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 vehicles' total travel time, which is at most twice of the optimal when the travel cost matrix is symmetric. We are also planning to test the path planning algorithm and the task assignment algorithm using a water surface robot testbed. The path planning algorithm can be adjusted for the vehicles' path planning in a 3D operation environment. One challenge for the extension is to properly model the dynamics of the currents in the presence of obstacles.

9 Acknowledgments

Xiaoshan Bai is indebted to the China Scholarship Council (CSC) for its financial support as a PhD student at the University of Groningen, The Netherlands. This work was supported in part by the National Natural Science Foundation of China (grant nos. 61633002 and 51579210).

10 References

[1] Binetti, G., Naso, D., Turchiano, B.: 'Decentralized task allocation for surveillance systems with critical tasks', *Rob. Auton. Syst*, 2013, **61**, (12), pp. 1653–1664

[2] Dahl, T.S., Matarić, M., Sukhatme, G.S.: 'Multi-robot task allocation through vacancy chain scheduling', *Robot. Auton. Syst*, 2009, **57**, (6-7), pp. 674–687

[3] Smith, S.L., Bullo, F.: 'Monotonic target assignment for robotic networks', *IEEE Trans. Autom. Control*, 2009, **54**, (9), pp. 2042–2057

[4] Yu, J., Chung, S.J., Voulgaris, P.G.: 'Target assignment in robotic networks: distance optimality guarantees and hierarchical strategies', *IEEE Trans. Autom. Control*, 2015, **60**, (2), pp. 327–341

[5] Liu, L., Shell, D.A.: 'Large-scale multi-robot task allocation via dynamic partitioning and distribution', *Auton. Robots*, 2012, **33**, (3), pp. 291–307

[6] Franceschelli, M., Rosa, D., Seatzu, C., *et al.*: 'Gossip algorithms for heterogeneous multi-vehicle routing problems', *Nonlinear Anal. Hybrid*, 2013, **10**, pp. 156–174

[7] Bai, X., Yan, W., Cao, M.: 'Clustering-based algorithms for multivehicle task assignment in a time-invariant drift field', *IEEE Robot. Autom. Lett.*, 2017, **2**, (4), pp. 2166–2173

[8] Gilbert, E., Johnson, D.: 'Distance functions and their application to robot path planning in the presence of obstacles', *IEEE J. Robot. Autom.*, 1985, **1**, (1), pp. 21–30

[9] Peng, J., Akella, S.: 'Coordinating multiple robots with kinodynamic constraints along specified paths', *Int. J. Rob. Res.*, 2005, **24**, (4), pp. 295–310

[10] Huang, W.H., Fajen, B.R., Fink, J.R., *et al.*: 'Visual navigation and obstacle avoidance using a steering potential function', *Rob. Auton. Syst.*, 2006, **54**, (4), pp. 288–299

[11] Giordano, P.R., Vendittelli, M.: 'Shortest paths to obstacles for a polygonal dubins car', *IEEE Trans. Robot.*, 2009, **25**, (5), pp. 1184–1191

[12] Han, J., Ok, J., Chung, W.K.: 'An ethology-based hybrid control architecture for an autonomous underwater vehicle for performing multiple tasks', *IEEE J. Ocean. Eng.*, 2013, **38**, (3), pp. 514–521

[13] Zhu, D., Huang, H., Yang, S.X.: '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. Trans. Cybern.*, 2013, **43**, (2), pp. 504–514

[14] Eichhorn, M.: 'Optimal routing strategies for autonomous underwater vehicles in time-varying environment', *Rob. Auton. Syst.*, 2015, **67**, pp. 33–43

[15] Yan, W., Bai, X., Peng, X., *et al.*: 'The routing problem of autonomous underwater vehicles in ocean currents'. MTS/IEEE Conf. OCEANS'14, Taipei, 2014, pp. 1–6

[16] Bai, X., Yan, W., Ge, S.S., *et al.*: 'An integrated multi-population genetic algorithm for multi-vehicle task assignment in a drift field', *Inf. Sci.*, 2018, **453**, pp. 227–238

[17] Bai, X., Yan, W., Cao, M., *et al.*: 'Heterogeneous multi-vehicle task assignment in a time-invariant drift field with obstacles'. 2017 56th IEEE Conf. Decision and Control, Melbourne, Australia, 2017, pp. 307–312

[18] Bollobás, B.: '*Modern graph theory*' (SSBM, New York, 2013)

[19] Gordon, G., McMahon, E.: 'A greedoid polynomial which distinguishes rooted arborescences', *Proc. Am. Math. Soc*, 1989, **107**, (2), pp. 287–298

[20] Soulignac, M.: 'Feasible and optimal path planning in strong current fields', *IEEE Trans. Robot.*, 2011, **27**, (1), pp. 89–98

[21] Mehlhorn, K., Sanders, P: '*Algorithms and data structures: the basic toolbox*' (SSBM, Berlin, 2008)

[22] Bektas, T.: 'The multiple traveling salesman problem: an overview of formulations and solution procedures', *Omega*, 2006, **34**, (3), pp. 209–219

[23] Toth, P., Vigo, D.: '*Vehicle routing: problems, methods, and applications*' (SIAM, Philadelphia, 2014)

[24] Fleszar, K., Osman, I.H., Hindi, K.S.: 'A variable neighbourhood search algorithm for the open vehicle routing problem', *Eur. J. Oper. Res*, 2009, **195**, (3), pp. 803–809

[25] Garey, M.R., Johnson, D.S.: '*Computers and intractability*' (WH Freeman, New York, 2002)

[26] Frederickson, G.N., Hecht, M.S., Kim, C.E.: 'Approximation algorithms for some routing problems'. 17th IEEE Symp. on Foundations of Computer Science, Houston, USA, 1976, pp. 216–227

[27] Papadimitriou, C.H., Steiglitz, K.: '*Combinatorial optimization: algorithms and complexity*' (Courier Corporation, North Chelmsford, 1982)

[28] Rathinam, S., Sengupta, R., Darbha, S.: 'A resource allocation algorithm for multivehicle systems with nonholonomic constraints', *IEEE Trans. Autom. Sci. Eng*, 2007, **4**, (1), pp. 98–104

[29] Lagoudakis, M.G., Berhault, M., Koenig, S., *et al.*: 'Simple auctions with performance guarantees for multi-robot task allocation'. 17th IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Sendai, Japan, 2004, vol. 1, pp. 698–705

[30] Laporte, G.: 'Fifty years of vehicle routing', *Transp. Sci.*, 2009, **43**, (4), pp. 408–416

[31] Ahn, C.W., Ramakrishna, R.S.: 'A genetic algorithm for shortest path routing problem and the sizing of populations', *IEEE Trans. Evol. Comput.*, 2002, **6**, (6), pp. 566–579