Hybrid Multi-Objective Optimization Approach With Pareto Local Search for Collaborative Truck-Drone Routing Problems Considering Flexible Time Windows

Qizhang Luo, Guohua Wu, *Member, IEEE*, Bin Ji, Ling Wang, and Ponnuthurai Nagaratnam Suganthan, *Fellow, IEEE*

***Abstract*— The collaboration of drones and trucks for last-mile delivery has attracted much attention. In this paper, we address a collaborative routing problem of the truck-drone system, in which a truck collaborates with multiple drones to perform parcel deliveries and each customer can be served earlier and later than the required time with a given tolerance. To meet the practical demands of logistics companies, we build a multi-objective optimization model that minimizes total distribution cost and maximizes overall customer satisfaction simultaneously. We propose a hybrid multi-objective genetic optimization approach incorporated with a Pareto local search algorithm to solve the problem. Particularly, we develop a greedy-based heuristic method to create initial solutions and introduce a problem-specific solution representation, genetic operations, as well as six heuristic neighborhood strategies for the hybrid algorithm. Besides, an adaptive strategy is adopted to further balance the convergence and the diversity of the hybrid algorithm. The performance of the proposed algorithm is evaluated by using a set of benchmark instances. The experimental results show that the proposed algorithm outperforms three competitors. Furthermore, we investigate the sensitivity of the proposed model and hybrid algorithm based on a real-world case in Changsha city, China.**

***Index Terms*— Last-mile delivery, truck-drone collaborative routing problem, flexible time windows, multi-objective optimization, hybrid optimization algorithm.**

I. Introduction

SMALL unmanned aerial vehicles (UAV), also known as drones, have been deployed to perform rapid and accurate deliveries in recent years. Drones can travel faster than trucks and are not limited by road conditions, resulting in significant reductions in the cost and time for delivering parcels [1], [2]. However, drones face shortages such as the relatively short travel range and limited capacity [3]. In this case, the truck-drone delivery system in which trucks and drones work collaboratively has attached attention from both industry and academia [1], [4].

The truck-drone delivery system combines the advantages of drones and trucks. In this system, while a truck travels between different customer nodes, drones can be launched from the truck to serve nearby customer nodes simultaneously. Some customer nodes that are unable to reach by truck due to road conditions (e.g., traffic congestions) can be served by drones efficiently. Meanwhile, the truck can supply new parcels and batteries to drones. Although some industrial companies such as UPS and Amazon have tested their truck-drone delivery system since 2017 [1], [4], truck-drone collaborative routing problems still call for further investigations.

Since Murray and Chu [2] first introduced the collaborative routing problem of drones and trucks in 2015, many relevant studies have been carried out [3], [5]–[8]. Previous studies have proved that typical collaborative routing problems of drones and trucks belong to the class of NP-hard problems, thereby the difficulty for solving a collaborative routing problem increases with the number of drones and trucks [4], [8], [9]. In this paper, we address the collaborative routing problem with multiple drones and a truck, which is more challenging than the classical problem that involves one drone and one truck. Meanwhile, our problem is more realistic since it has been proved that multiple drones show more benefits than one drone [4]. As the delivery problem appears in real life, some non-emergency parcel deliveries can violate the time windows to a certain extent when serving customers. Therefore, we allow each vehicle to be capable of serving a customer before and after the earliest and latest times with a given tolerance (i.e., flexible time windows [10]). Since the violation of time window constraints may affect customer satisfaction [11], we consider two objectives (i.e., minimizing the routing costs of drones and trucks and maximizing customer satisfaction). Overall, we term this problem the multi-objective

Manuscript received 6 May 2021; revised 3 October 2021; accepted 5 October 2021. Date of publication 19 October 2021; date of current version 9 August 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62073341 and Grant 61873328, in part by the Natural Science Fund for Distinguished Young Scholars of Hunan Province under Grant 2019JJ20026, and in part by the China Scholarship Council under Grant 202006370285. The Associate Editor for this article was A. Nunez. *(Corresponding authors: Guohua Wu; Ling Wang.)*

Qizhang Luo, Guohua Wu, and Bin Ji are with the School of Traffic and Transportation Engineering, Central South University, Changsha 410075, China (e-mail: qz_luo@csu.edu.cn; guohuawu@csu.edu.cn; cumtjibin@126.com).

Ling Wang is with the Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: wangling@mail.tsinghua.edu.cn).

Ponnuthurai Nagaratnam Suganthan is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, and also with KINDI Center for Computing Research, College of Engineering, Qatar University, Qatar (e-mail: epnsugan@ntu.edu.sg).

This article has supplementary downloadable material available at https://doi.org/10.1109/TITS.2021.3119080, provided by the authors.

Digital Object Identifier 10.1109/TITS.2021.3119080

truck-drone collaborative routing problem considering flexible time windows and multiple drones (Mo-CRPTW-mD).

However, challenges are arisen for solving Mo-CRPTW-mD. In the optimization process, two objectives frequently conflict, which presents a particularly intricate multi-objective optimization problem (MOP). To address the MOP, a typical idea is to model the objective functions as a weighted combination (i.e., weighted-sum method). However, how to quantify the relative weights of costs and customer satisfaction is still an open research topic [11], especially when these two objectives exhibit different dimensions. In this context, population-based multi-objective optimization algorithms would be promising since they can provide a range of solutions that represent the trade-off between objectives, which is important for decision-makers to make an appropriate scheme according to practical situations. We propose a hybrid multi-objective optimization approach, which combines a population-based multi-objective algorithm and a Pareto local search algorithm (PLS) [12]. The motivation is multifold. First, population-based algorithms can generate several Pareto solutions per run and exhibit superior performance on MOPs [13]. Second, the population-based algorithm is a general algorithmic framework for global search, while PLS can effectively explore the local solution space. A hybrid algorithm with an appropriate combination mechanism can provide a balance between global exploration and local exploitation, which is very important in designing a multi-objective algorithm [14]. Third, different hybrid multi-objective algorithms have already been applied to various variants of routing problems and provided good results [11], [15], [16]. All these factors strongly motivate us to employ a hybrid population-based algorithm to solve Mo-CRPTW-mD.

The main contributions of this paper are as follows. First, we introduce a multi-objective truck-drone collaborative routing problem, namely, Mo-CRPTW-mD. In Mo-CRPTW-mD, a truck collaborated with a heterogeneous fleet of drones performs parcel deliveries, and each customer can be visited exactly once by either a truck or a drone with permitting to deviate from customer time windows by a given tolerance. Second, we develop a hybrid multi-objective optimization approach to solve the Mo-CRPTW-mD, which is termed HMOA. A population-based algorithm framework inspired by the non-dominated sorting genetic algorithm (NSGA-II) [17] is developed to conduct the global search, while an improved Pareto local search (PLS) is hybridized with the global search framework. Meanwhile, problem-specific neighborhood search strategies and an adaptive strategy are designed to improve the efficiency of HMOA. Third, we conducted extensive experimental studies based on benchmark instances and a real-world case. Three population-based algorithms were compared with the proposed algorithm to verify its superiority.

The remainder of this paper is organized as follows. Section II contains a brief overview of the relevant literature. Section III formally describes Mo-CRPTW-mD. Section IV presents our HMOA for solving the problem. Section V describes computational experiments. Section VI presents concluding remarks.

II. Related Work

Previous studies can be divided into three categories based on the number of drones and trucks: single-drone and single-truck problem, multi-drone and single-truck problem, and multi-drone and multi-truck problem.

The single-drone and single-truck problem is generally considered as either flying sidekick traveling salesman problem (FSTSP) or traveling salesman problem with drone (TSP-D), which was first introduced by Murray and Chu [2] and Agatz *et al.* [3], respectively. In FSTSP, each drone must visit any node at most once. In contrast to the FSTSP, the TSP-D allows a drone to join the truck at the node where it was released. These two works serve as a basis for several extensions [5], [18]–[21]. For example, Ha *et al.* [5] proposed an extended version of FSTSP, which minimizes operational costs including total transportation cost and one created by waste time a vehicle has to wait for the other. Luo *et al.* [22] considered a variant of the classic two-echelon TSP, in which a truck travels on the road network and a drone travels in areas beyond the road for visiting nodes unreachable by the truck. A similar study was conducted by Liu *et al.* [23], which also considered an energy consumption model of the drone. In addition, several efficient algorithms have been proposed for solving TSP-D, FSTSP, and their variants [19], [24]–[27], such as the branch-and-cut algorithm [24], decomposition-based iterative optimization algorithm [27], and variable neighborhood search algorithm [25].

The multi-drone and single-truck problem is an extension of the single-drone and single-truck problem. Murray and Chu [2] presented a version called parallel drone scheduling TSP (PDSTSP). In PDSTSP, multiple drones are launched from a depot to serve nearby customers, and a truck serves other customers independently. Later, many studies have been carried out to extend this study [28]–[31]. For example, Mbiadou Saleu *et al.* [31] proposed an iterative two-step heuristic algorithm for PDSTSP. Dayarian *et al.* [28] focused on the same-day delivery problem, in which drones perform supplying for a truck instead of delivering parcels. To overcome the flight-range limitation, Kim and Moon [30] considered a drone station that stores drones and charging devices and positioned it near customers. Several studies also formulated the multi-drone and single truck problem by allowing multiple drones to be launched from a single truck [4], [32]–[34]. For example, Murray and Raj [4] introduced an extended version of the FSTSP, which allows multiple drones to be launched by a single truck and retrieved at a different location. Tu *et al.* [34] proposed an adaptive large neighborhood search heuristic (ALNS) for solving a similar problem. Moshref-Javadi *et al.* [32] developed an improved ALNS to solve a multi-drone and single-truck problem derived from real-world instances. Furthermore, several studies assume that the truck must wait for the return of all the drones launched before continuing its route at the same location [33], [35], [36].

In the case of multi-drone and multi-truck problems, Wang and Sheu [8] proposed a branch-and-price algorithm to solve this kind of problem with 12-customer instances. Li *et al.* [37] investigated a two-echelon vehicle routing problem with time windows and mobile satellites, in which the first-echelon vehicles park at customer nodes and wait for second-echelon drones. Ham [38] extended the PDSTSP by introducing multiple drones and multiple trucks and solved the problem by using constraint programming. Ulmer and Thomas [39] proposed a dynamic collaborative vehicle routing problem, in which drones and trucks work separately and may serve a set of customers with a delivery deadline. Sacramento *et al.* [7] and Das *et al.* [40] extended the FSTSP, in which multiple trucks and each carrying a single drone perform the delivery.

Multi-objective optimization has been considered in few studies. Wang *et al.* [26] extended the FSTSP to a multi-objective version and proposed an improved NSGA-II to optimize the cost and time of the routing simultaneously. Das *et al.* [40] considered the travel cost and the customer service level and proposed a modified ant colony optimization algorithm to solve the proposed problem. Note that this study is totally different from our study. The customer service level is calculated in a different manner compared with the customer satisfaction in our study. Moreover, in their study the truck only takes one drone, while we consider multiple drones, which yields more complex constraints. Salama and Srinivas [41] used the ε-constraint method to simultaneously optimize the customer node clustering and routing decisions. Nevertheless, due to the nature of NP-hard of collaborative vehicle routing problems [2], considering multiple objectives and multiple drones significantly increase the complexity for solving already hard-to-solve problems. It is still challenging to solve the multi-objective collaborative routing problem with multiple drones and complex constraints.

Although many above-mentioned studies are important and near equivalent to our paper, we deviate from them and fill gaps in the following manner. First, to the best of our knowledge, we first investigate a multi-objective truck-drone collaborative routing problem where the truck carries multiple drones and the routing costs and customer satisfaction are optimized simultaneously. Second, we adopt flexible time windows to our problem, while most existing collaborative routing problems with time windows use hard time window constraint. Third, we optimize multiple objectives simultaneously based on a population-based algorithm. Sophisticated mechanisms such as the adaptive control of local search, duplicated solution elimination, specialized genetic operators, and specialized neighborhood search strategies are designed to improve the performance of the proposed algorithm.

III. Problem Definition and Formulation

In this section, we propose a multi-objective collaborative routing problem (i.e., Mo-CRPTW-mD) that aims to simultaneously minimize the transportation cost and maximize customer satisfaction. A summary of the notations is described in Table I.

A. Problem Definition

In Mo-CRPTW-mD, a truck delivers parcels assisted by multiple heterogeneous drones carried by the truck. Each drone can be launched from the beginning depot or the truck, and retrieved by the ending depot or the truck at another customer node. Similar to Amazon Prime Air [42], when a drone arrives at a customer node, the drone unloads the parcel autonomously and leaves immediately. After finishing a drone delivery, the drone can be launched again from the truck with a new parcel and a fresh battery. Although the drones can be launched multiple times, they are allowed to be launched at the same node exactly once. Each drone has a unique endurance that must be complied with during the drone delivery. Different from drones, the truck requires a period of service time at a customer node since the driver may need time to unload parcels manually and handle reached drones. Due to uncertain factors such as traffic jams, it is difficult to ensure that the truck and the drones arrive at a node simultaneously. Hence, the truck and drones that first arrive at a rendezvous node need to wait for the others. Furthermore, each customer node is allowed to be served exactly once and associated with a time

TABLE I
NOTATION DESCRIPTION

Notation	Description
	Set and parameter notations
V	Set of all nodes; $V = \{0, 1, \ldots, c, c+1\}$.
V_L	Set of nodes from which a vehicle may depart; $V_L = \{0, 1, \ldots, c\}$.
V_R	Set of nodes to which a vehicle may visit; $V_R = \{1, 2, \ldots, c+1\}$.
D	Set of drones.
C	Set of customer nodes; $C = \{1, 2, \ldots, c\}$.
δ_{ij}	Distances from nodes $i \in V_L$ to $j \in V_R$ traveled by truck.
δ'_{dij}	Distances from nodes $i \in V_L$ to $j \in V_R$ traveled by drone $d \in D$.
τ_{ij}	Travel time from nodes $i \in V_L$ to $j \in V_R$ of truck.
τ'_{dij}	Travel time from nodes $i \in V_L$ to $j \in V_R$ of drone $d \in D$.
s_i	Service time of the truck at node $i \in C$.
c_t	Transportation costs per unit of distance of truck.
c'_d	Transportation costs per unit of distance of drone $d \in D$.
ε_d	Flight endurance of drone $d \in D$.
P	Set of all possible tuples that may be followed by drone $d \in D$, $P = \{< d, i, j, k >: d \in D, i \in V_L, j \in C, k \in V_R, i \neq j, j \neq k, k \neq i\}$.
M	A sufficiently large number.
$[a_i, b_i]$	The desired time window of the delivery at node $i \in C$.
$[e_i, l_i]$	The tolerable time window of the delivery at node $i \in C$.
$\mu_i(t)$	Customer satisfaction when a vehicle arrives at node $i \in C$ and time instance t.
	Decision variables
x_{ij}	$x_{ij} = 1$ if the truck travels from nodes $i \in V_L$ to $j \in \{V_R : i \neq j\}$; otherwise $x_{ij} = 0$.
y_{dijk}	$y_{dijk} = 1$ if drone $d \in D$ travels from nodes $i \in V_L$ to $j \in C$, and then rejoins the truck at $k \in \{V_R :< d, i, j, k >\in P\}$; otherwise $y_{dijk} = 0$.
u_i	Position of the node $i \in V_R$ in the path of the truck.
p_{ij}	$p_{ij} = 1$ if the node $i \in V_L$ is visited by the truck before node $j \in V_R$; otherwise $p_{ij} = 0$.
t_i	Arrival time of truck to node $i \in V_R$.
t'_{di}	Arrival time of drone $d \in D$ to node $i \in V_R$.
r'_{di}	Leaving time of drone $d \in D$ at node $i \in V$.
r_i	Leaving time of truck at node $i \in V$.
w_i^{td}	Length of time that the truck waits for drone $d \in D$ at node $i \in V_R$.
w_i^{ttw}	Length of time that the truck waits for the time window at node $i \in V_R$.
w_{id}^{dt}	Length of time that drone $d \in D$ waits for the truck at node $i \in V_R$.
w_{id}^{dtw}	Length of time that drone $d \in D$ waits for the time window at node $i \in C$.

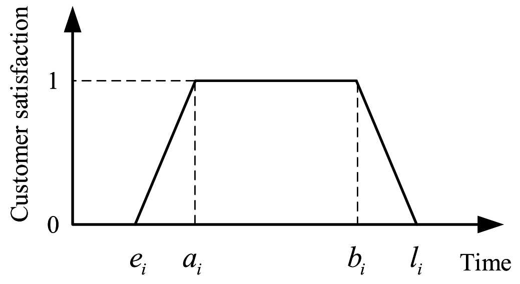


Fig. 1. An illustration of the customer satisfaction function and flexible time window.

window that should be satisfied by the delivery. An illustration of the collaborative routing problem consisting of one truck and two drones is shown in Fig. 2(b).

Besides, we make some other assumptions: (i) Although the truck cannot launch or retrieve multiple drones at a time, launch and recovery activities of drones are scheduled efficiently such that mid-air collisions of drones can be avoided, and the collaborative problem between any two drones at the same location is ignored. (ii) The truck has a sufficiently large capacity to carry both parcels and drones through the entire delivery. (iii) The time used to launch and retrieve drones, as well as supply batteries and parcels to drones is slight and covered by the service time of the truck. (iv) If a drone arrives at a customer node before the time window, it can land and wait for the time window.

B. Mathematical Formulation

The Mo-CRPTW-mD aims to find collaborative routes of the truck and drones with the minimum transportation cost and the maximum overall customer satisfaction. Particularly, the transportation cost is related to traveled distances of drones and the truck, which can be formulated as

$$f_1 = \sum_{i\in V_L}\sum_{j\in V_R} c_t \delta_{ij} x_{ij} + \sum_{d\in D}\sum_{i\in V_L}\sum_{j\in C}\sum_{\substack{k\in V_R,\\ <d,i,j,k>\in P}} c'_d(\delta'_{dij}+\delta'_{djk}) y_{dijk}, \tag{1}$$

where c_t and c'_d are transportation costs of the truck and drone $d \in D$ per unit of distance, respectively.

Customer satisfaction is defined as the fulfillment of customer expectations and related to the arrival time of a vehicle. We assume that customer satisfaction $\mu_i(t)$ equals 1 if a vehicle arrives at customer node $i \in C$ within the appointed time window $[a_i, b_i]$, while it is decreased linearly to 0 as the vehicle serves the customer too early or too late, as shown in Fig. 1. Hence, customer satisfaction $\mu_i(t)$ at node i delivered by a vehicle can be expressed by

$$\mu_i(t) = \begin{cases} \dfrac{t-b_i}{l_i-b_i}, & b_i < t \le l_i \\ 1, & a_i \le t \le b_i \\ \dfrac{t-e_i}{a_i-e_i}, & e_i \le t < a_i \\ 0, & otherwise, \end{cases} \tag{2}$$

The overall customer satisfaction can be calculated as

$$f_2 = \sum_{i\in V_L}\sum_{j\in V_R} x_{ij}\cdot \mu_j(t_j) + \sum_{d\in D}\sum_{i\in V_L}\sum_{j\in C}\sum_{\substack{k\in V_R,\\ <d,i,j,k>\in P}} y_{dijk}\cdot \mu_j(t'_{dj}), \tag{3}$$

Inspired by the models in [5] and [4], the constraints of Mo-CRPTW-mD are introduced in the following.

$$\sum_{i\in V_L} x_{ij} + \sum_{d\in D}\sum_{i\in V_L}\sum_{\substack{k\in V_R,\\ <d,i,j,k>\in P}} y_{dijk} = 1, \quad \forall j \in \{C : j \ne i\}, \tag{4}$$

$$\sum_{j\in V_R} x_{0j} = 1, \tag{5}$$

$$\sum_{i\in V_L} x_{i,c+1} = 1, \tag{6}$$

$$\sum_{j\in C}\sum_{\substack{k\in V_R,\\ <d,i,j,k>\in P}} y_{dijk} \le 1, \quad \forall i \in \{V_L : i \ne j\},\ \forall d \in D, \tag{7}$$

$$\sum_{j\in V_L}\sum_{\substack{j\in C,\\ <d,i,j,k>\in P}} y_{dijk} \le 1, \quad \forall k \in \{V_R : k \ne i\},\quad \forall d \in D, \tag{8}$$

$$\sum_{i\in V_L} x_{ij} = \sum_{k\in V_R} x_{jk}, \quad \forall j \in \{C : j \ne i, k \ne j\}, \tag{9}$$

$$u_i - u_j \ge 1-(c+2)\,p_{ij}, \quad \forall i, j \in \{C : j \ne i\}, \tag{10}$$

$$u_i - u_j \le -1+(c+2)\left(1-p_{ij}\right), \quad \forall i, j \in \{C : j \ne i\}, \tag{11}$$

$$p_{ij} + p_{ji} = 1, \quad \forall i, j \in \{C : j \ne i\}, \tag{12}$$

$$u_i - u_j + 1 \le (c+2)\left(1-x_{ij}\right), \quad \forall i \in C,\quad \forall j \in \{V_R : j \ne i\}, \tag{13}$$

$$2y_{dijk} \le \sum_{h\in V_L} x_{hi} + \sum_{l\in C} x_{lk}, \quad \forall d \in D,\ \forall i, j \in \{C : i \ne h, j \ne i\}, \quad \forall k \in \{V_R :< d, i, j, k >\in P, k \ne l\}, \tag{14}$$

$$u_k - u_i \ge 1-(c+2)\left(1-\sum_{\substack{j\in C,\\ <d,i,j,k>\in P}} y_{dijk}\right), \quad \forall d \in D,\ \forall i \in C,\ \forall k \in \{V_R : k \ne i\}, \tag{15}$$

$$y_{d0jk} \le \sum_{h\in V_L} x_{hk}, \quad \forall d \in D,\ \forall i \in C, \quad \forall k \in \{V_R :< d, 0, j, k >\in P, k \ne h\}, \tag{16}$$

$$r'_{dl} \ge t'_{dk} - M\left(3-\sum_{\substack{j\in C,\\ <d,i,j,k>\in P}} y_{dijk} - \sum_{m\in C n\in V_R,}\sum_{<\upsilon,l,m,n>\in P} y_{\upsilon lmn} - p_{il}\right), \quad \forall i \in \{V_L : i \ne m, i \ne n\}, \quad \forall k \in \{V_R : k \ne i, k \ne m, k \ne n\}, \quad \forall l \in \{C : l \ne i, l \ne k, l \ne j, l \ne m\},\ \forall d \in D, \tag{17}$$

$$r'_{di} \geq t'_{di} - M\left(1 - \sum_{j \in C} \sum_{\substack{k \in V_R, \\ <d,i,j,k> \in P}} y_{dijk}\right), \quad \forall d \in D,\ \forall i \in \{V_L : i \neq j\}, \tag{18}$$

$$t'_{dj} \geq r'_{di} + \tau'_{dij} - M\left(1 - \sum_{\substack{k \in V_R, \\ <d,i,j,k> \in P}} y_{dijk}\right), \quad \forall d \in D,\ \forall j \in C,\ \forall i \in \{V_L : i \neq j\}, \tag{19}$$

$$r'_{dj} \geq t'_{dj} + w^{dtw}_{jd} - M\left(1 - \sum_{i \in V_R} \sum_{\substack{k \in V_R, \\ <d,i,j,k> \in P}} y_{dijk}\right), \quad \forall d \in D,\ \forall j \in \{C : j \neq i\}, \tag{20}$$

$$t'_{dk} \geq r'_{dj} + \tau'_{dij} - M\left(1 - \sum_{\substack{i \in V_L, \\ <d,i,j,k> \in P}} y_{dijk}\right), \quad \forall d \in D,\ \forall k \in V_R,\ \forall j \in \{C : j \neq k\}, \tag{21}$$

$$t'_{dk} - r'_{di} \leq \varepsilon_d + M\left(1 - y_{dijk}\right), \quad \forall d \in D,\ \forall <d,i,j,k> \in P, \tag{22}$$

$$t_j \geq r_i + \tau_{ij} - M\left(1 - x_{ij}\right), \quad \forall i \in V_L,\ \forall j \in \{V_R : j \neq i\}, \tag{23}$$

$$r_j = \max\left(t'_{dj}, t_j + w^{ttw}_j + s_j \sum_{j \in V_L} x_{jk}\right), \quad \forall k \in \{V_R : k \neq j\},\ \forall d \in D, \tag{24}$$

$$r_k \geq t'_{dk} - M\left(1 - \sum_{j \in C} \sum_{\substack{i \in V_R, \\ <d,i,j,k> \in P}} y_{dijk}\right), \quad \forall d \in D,\ \forall k \in \{V_R : k \neq i, k \neq j\}, \tag{25}$$

$$r_k \geq r'_{dk} - M\left(1 - \sum_{l \in C} \sum_{\substack{m \in V_R, \\ <d,k,l,m> \in P}} y_{dklm}\right), \quad \forall d \in D,\ \forall k \in \{V_R : k \neq l, k \neq m\}, \tag{26}$$

$$w^{td}_i = \max\{0, t'_{di} - (t_i + w^{ttw}_i + s_i)\}, \quad \forall i \in C,\ \forall d \in D, \tag{27}$$

$$w^{dt}_{id} = \max\{0, t_i - t'_{di}\}, \quad \forall i \in C,\ \forall d \in D, \tag{28}$$

$$w^{ttw}_i = \max\{0, e_i - t_i\}, \quad \forall i \in C, \tag{29}$$

$$w^{dtw}_{id} = \max\left\{0, e_{di} - t'_{di}\right\}, \quad \forall i \in C,\ \forall d \in D, \tag{30}$$

$$x_{ij}, y_{dijk}, p_{ij} \in \{0,1\}, \quad \forall d \in D,\ \forall i \in V_L,\ \forall j,k \in \{V_R : j \neq k, j \neq i, i \neq k\}, \tag{31}$$

$$0 \leq u_i \leq c + 1, \quad \forall i \in V, \tag{32}$$

$$p_{0j} = 1, \quad \forall j \in C, \tag{33}$$

Constraint (4) ensures that each node can only be visited exactly once. Constraints (5) and (6) indicate that the truck can depart from and return to the depot exactly once. Constraints (7) and (8) state that each drone can be launched from and retrieved at any particular node at most once.

Constraint (9) ensures that if a truck visits a node, then it must depart from the same node. Constraints (10)-(12) determine the order of nodes visited by the truck. Constraint (13) is the truck subtour elimination constraint.

Constraints (14)-(16) describe the spatial coordination of drones and the truck. If there exists a drone delivery $< d, i, j, k >$, constraint (14) ensures that the truck must visit nodes i and k, while constraint (15) indicates that the truck must travel from node i to node k. If a drone is launched from the depot and retrieved at node k, constraint (16) requires that the truck must rendezvous the drone at node k.

Constraints (17)-(22) define the drone delivery. Constraint (17) prevents the sorties of the same drones from overlapping. Constraint (18) ensures that drone d cannot depart from node i to node j and node k until it has arrived at node i. Constraint (19) indicates that before drone d arrives at node j, it should depart from node i and then travel from node i to node j. Once the drone d arrivals at node j, it leaves node j immediately after completing service, which is described in constraint (20). When the drone d arrivals at k, constraint (21) requires that the drone should first depart from j, and then travel from j to k. Furthermore, each drone delivery should be within the flight endurance, which is described in constraint (22).

As to the truck delivery, constraint (23) indicates that before the truck arrives at node j, it should leave node i and then travel from node i to node j. Constraint (24) identifies the leaving time of the truck at node k.

Constraints (25) and (26) guarantee the temporal collaboration of drones and the truck. If there exists a drone d that will land at node k, the truck cannot leave node k until the drone has been retrieved, which is described in constraint (25). Similarly, constraint (26) states that if there exists a drone d that will be launched at node k, the truck cannot leave node k until the drone has been launched.

Finally, constraints (27)-(33) specify the decision variables definitions.

Note that Mo-CRPTW-mD is different from the pickup and delivery problem with transfers (PDPT) [43]. First, in PDPT multiple vehicles perform pickup and delivery tasks independently, while in our problem two kinds of vehicles (i.e., drones and the truck) perform delivery tasks collaboratively, indicating that the drones only can serve nodes nearby the truck and need back to the truck after accomplishing delivery tasks. Second, in PDPT a vehicle may visit a transfer node after another vehicle has visited the same node to transfer parcels, while in our problem two kinds of vehicles can rendezvous at the same node without specified orders.

IV. Solution Approach

This section presents a hybrid multi-objective optimization approach (HMOA) for solving the Mo-CRPTW-mD. The specific features like specialized chromosome representation, genetic operators, neighborhood structures, local search method, adaptive strategy, as well as general process of HMOA are described below.

Algorithm 1 Framework of HMOA

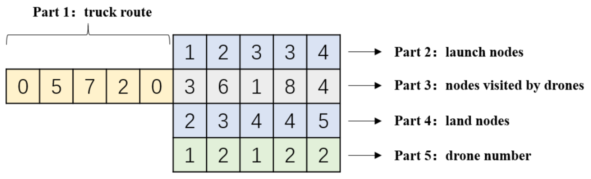
```
Input: All nodes $C$, drone number $m$, population size $n$,
       maximum iterations $iter$
Output: Final Pareto front $PF$
1  Initialization: $prob \leftarrow 0$, $PF \leftarrow \emptyset$,
   $P_1 \leftarrow AssignNodes(C, m, n)$
2  $t \leftarrow 1$
3  while $t \leq iter$ do
4      $Q_t \leftarrow GeneticOperation(P_t)$
5      $R_t \leftarrow Q_t \cup P_t$
6      $R_t \leftarrow RemoveDuplication(R_t, PF)$
7      $NonDominatedSort(R_t)$
8      $P_{t+1} \leftarrow \emptyset$, $i \leftarrow 1$
9      while $|P_{t+1}| + |F_i| \leq n$ do
10         if $i = 1$ then
11             $PF \leftarrow Update1(PF, F_1)$
12             if $rand < prob$ then
13                 $PF \leftarrow PLS(PF)$
14             $F_1 \leftarrow PE$
15         $CrowdingDistance(F_i)$
16         $P_{t+1} \leftarrow P_{t+1} \cup F_i$
17         $i \leftarrow i + 1$
18     $Sort(F_i, \prec_n)$
19     $P_{t+1} \leftarrow P_{t+1} \cup F_i[1 : n - |P_{t+1}|]$
20     $t \leftarrow t + 1$, $prob \leftarrow prob + 1/iter$
```

A. Framework of HMOA

The framework of HMOA is shown in Algorithm 1. HMOA begins by initializing an initial population P_1, an external population PF for storing all nondominated solutions found so far, and a parameter $prob$ for triggering PLS (line 1). n initial solutions in P_1 are created based on a greedy method called $AssignNodes$ detailed in Algorithm 2.

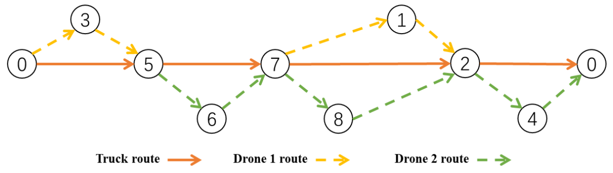
Once the initial population P_1 is formed, all solutions in this population will be optimized iteratively (lines 3-20). In each iteration, an offspring population Q_t will be generated based on problem-specific genetic operations (in Section IV-E) and be combined with paternal population P_t (lines 4-6). Then, we design a simple perturbation strategy $RemoveDuplication$ (in Section IV-F) to remove duplicated solutions from the combined population R_t (line 11). Afterward, the operations such as non-dominated sort, crowding distance calculation, and elite strategy introduced in NSGA-II [17] are adopted to select appropriate solutions from R_t to the next-generation population P_{t+1} (lines 7-19).

A featured place of HMOA is that the Pareto Front F_1 will be used to update the external population PF based on $Update1$ and PF would be explored adaptively based on PLS at each generation (lines 10-13). Then, F_1 will be replaced by PF while PF will be preserved to the next generation (line 14). The purpose of these steps is to balance the computational cost and algorithmic efficiency. Specifically, since a major computational overhead of HMOA comes from PLS, we propose two strategies. The first strategy adaptively determines when to trigger PLS by using $prob$ at each generation. PLS (Algorithm 3) will be triggered if a randomly generated value $rand \in [0, 1]$ is smaller than $prob$. Instead of using a fixed value, $prob$ starts from 0 and increases at each generation based on the current number of generations. The underlying reason is that the population in initial generations is far from the Pareto front and more attentive to maintaining a diverse population, while shifts to ensuring more solutions converge towards the Pareto front in later generations. Another strategy is to select appropriate solutions to be further explored assisted by the external population PF, which is elaborated in Section IV-G. Particularly, $Update1$ (PF, F_1) compares each solution p_i in F_1 with solutions in PF. If all the solutions in PF cannot dominate p_i and p_i is new to PF, p_i will be added to PF. All the solutions in PF dominated by p_i will be removed from PF.



					1	2	3	3	4
0	5	7	2	0	3	6	1	8	4
					2	3	4	4	5
					1	2	1	2	2

(a) Chromosome



(b) Truck-drone collaborative routes

Fig. 2. An illustration of a solution representation.

The iteration repeats until a predefined number of generations $iter$ is reached and returns the final Pareto front PF.

B. Solution Representation

Inspired by [32], a solution of the Mo-CRPTW-mD is represented by 5 parts. As Figure 2 shows, Part 1 of the chromosome is the truck route, which is represented by a permutation of node numbers visited by the truck. Two zero values in the truck route represent the depot. Part 2 and Part 4 are launch and land nodes of drones, which are presented by cell numbers in Part 1. Part 3 includes customer nodes visited by drones, which are presented by customer numbers. Part 5 shows drone numbers.

For example, Figure 2(b) describes Figure 2(a). The column (i.e., an assignment of a customer node served by a drone) $< 1, 3, 2, 1 >$ of the chromosome means drone 1 is launched from the first node (i.e., depot) of the truck route, then hosted by customer node 3, and finally retrieved at the second node (i.e., customer node 5) of the truck route.

C. Initial Solution

Initial solutions are created based on a greedy method called $AssignNodes$ shown in Algorithm 2. Algorithm 2 begins by initializing the initial population P and a threshold value named lower truck limit LTL (line 1). LTL is first defined

Algorithm 2 AssignNodes

```
Input: all nodes $C$, number of drones $m$, number of
       initial solutions $n$
Output: Initial population $P$
1  Initialization: Initial population $P_1 \leftarrow \emptyset$, lower truck
   limit $LTL$
2  while $|P| < n$ do
3      $Pa \leftarrow \emptyset$
4      $[C_t, C_d] \leftarrow$ Randomly divide all nodes in $C$ and let
       $|C_t| = LTL$
5      $C_t \leftarrow NearestNeighborTW(C_t)$
6      while all nodes in $C_d$ are not assigned do
7          $C'_d \leftarrow C_d$
8          for $d = 1$ to $m$ do
9              $i \leftarrow 0, k \leftarrow 0$
10             while $C'_d \neq \emptyset$ or $k \neq |C_t|$ do
11                 $P_d \leftarrow \emptyset$
12                 for $k = i + 1$ to $|C_t|$ do
13                     for $j = 0$ to $|C'_d|$ do
14                         if $d'_{ij} + d'_{ik} \leq \varepsilon$ then
15                             $P_d \leftarrow P_d \cup < i, j, k, d >$
16                 if $P_d \neq \emptyset$ then
17                     $< i', j', k', d > \leftarrow$ find the cheapest pair
                       from $P_d$
18                     $Pa \leftarrow Pa \cup < i', j', k', d >$
19                     Delete $j$ from $C'_d$
20                     $i \leftarrow k'$
21         if $C'_d \neq \emptyset$ then
22             Update $C_t$ and $C_d$ based on $C'_d$
23             $C_t \leftarrow NearestNeighborTW(C_t)$
24     Create a solution $p$ from $Pa$ and $C_t$
25     $P \leftarrow P \cup p$
```

by [4], which indicates the minimum number of customer nodes in the truck route for a feasible solution. The value of LTL can be calculated by

$$LTL = \left\lceil \frac{|C| - m}{m + 1} \right\rceil, \tag{34}$$

The population size n is the maximal number of iterations in the while loop (lines 2-25). In each loop, a set Pa is defined to store assignments of nodes served by drones (line 3). All the nodes are randomly divided into two groups C_t and C_d served by the truck and drones, respectively (line 4). Particularly, the size of C_t equals to LTL and all the nodes in C_d must be drone-eligible nodes.

For the nodes in C_t, we use a variant of the well-known nearest-neighbor heuristic, named $NearestNeighborTW$, to generate a permutation of customer nodes (line 5). Specifically, from the depot, this heuristic chooses the next node that is the closet node from unvisited nodes and constrained by its time window, finally stops when all nodes are visited.

The nodes in C_d are assigned via a greedy method (lines 6-23). In each iteration, all the possible and feasible assignments of nodes served by drone d are stored in P_d. (lines 8-15). If P_d is not empty, the cheapest assignment $< i', j', k', d >$ of P_d would be extracted and stored in Pa (lines 16-18). The cheapest assignment is determined based on its cost since it is very easy to be obtained via the prepared distance matrix. Afterward, j' is considered as an assigned node and will be deleted from C'_d, which is a copy of C_d (line 19). The iteration stops if all nodes in C_d have been assigned (i.e., $C'_d = \emptyset$). Otherwise, C_d is updated by removing all nodes that remain in C'_d, while C_t is updated by adding all the removed nodes to it. All the nodes in C_t are reassigned again based on $NearestNeighborTW$ (lines 21-23).

Finally, a solution p is created based on an assignment result of all nodes and added to P (lines 24-25).

D. Specialized Neighborhood Operators

To improve the quality of solutions, we introduce six heuristic operators into the proposed algorithm.

1) Truck-to-Drone Operator N_1: This heuristic operator aims to change a customer node served by the truck to a customer node served by drones. Specifically, we delete the most expensive and drone-eligible customer node from the truck route and assign a cheap drone route to it. The most expensive customer node is identified by calculating the distance of both two sides of each drone-eligible customer node in the truck route. The customer node with the longest distance of two sides would be treated as the most expensive node and be deleted from the truck route. Meanwhile, the deletion operation would result in changing the length of Part 1. Hence, values in Part 2 and Part 4 would be changed according to Part 1. Afterward, we use the *Repair* heuristic (Algorithm 4) to find a cheap and feasible assignment of the deleted customer node to a drone. If N_1 does not find a feasible solution, it returns the input solution unchanged.

2) Drone-to-Truck Operator N_2: Opposite to N_1, this heuristic operator aims to move a customer node served by a drone to the truck route. Specifically, we delete the most expensive drone assignment and insert the deleted customer node into the cheapest position of the truck route. First, the most expensive assignment from the right side (i.e., Parts 2-5) of the chromosome is identified in terms of the distance of each flight. It is very convenient to calculate the distance since each column of Parts 2-5 can be treated as a flight and includes the information of the launch node, customer node, and land node. Second, the most expensive assignment would be deleted from the current solution and the deleted customer node would be inserted into the position with the shortest distance of two sides of the truck route. Finally, since the length of Part 1 is changed, values in Part 2 and Part 4 should be changed accordingly.

3) Swap N_3: This heuristic operator targets Part 1 and Part 3 of the chromosome, by exchanging positions of two customer nodes served by the truck and a drone, respectively. Note that the two customer nodes are selected randomly and the customer node selected from the truck route should be a drone-eligible node. After changing positions, the solution may be infeasible. In case of infeasibility, the infeasible flight assignments would be deleted from the current solution and the proposed $Repair$ heuristic would be triggered to reassign deleted customer nodes. If the heuristic cannot result in a feasible solution, N_3 returns the input solution unchanged.

4) 2-Opt N_4: The 2-opt is a well-known and efficient heuristic operator used to solve traditional TSP [32]. This heuristic is performed on Part 1 to improve the truck route, by exchanging two of the edges in the route with two other edges. The customer nodes for the 2-opt heuristic operator are randomly selected. Meanwhile, values in Part 2 and Part 4 are also changed according to the permutation in Part 1 and the $Repair$ heuristic is adopted to repair the potential infeasibility of the solution. If no feasible solution can be generated, N_4 returns the input solution unchanged.

5) Greedy-Deletion-Reinsertion N_5: This heuristic focuses on assignments of customer nodes served by drones. First, a customer node associated with the most expensive assignment in terms of the flight distance is identified and deleted from Part 3 of the chromosome, by using the same method as N_2. Second, the $Repair$ heuristic is used to reassign the deleted customer node to a drone. If N_5 cannot find a feasible solution after deletion, it returns the input solution unchanged.

6) Random-Deletion-Reinsertion N_6: Similar with N_5, this heuristic improves drone deliveries by deleting a randomly selected assignment and then reassign a random feasible assignment of the deleted customer node to a drone. If N_6 cannot find a new and feasible assignment after deletion, it returns the input solution unchanged.

We use N_1, N_2, and N_3 to construct a multi-mode mutation operator (in Section IV-E). The underlying reason is that N_1, N_2, and N_3 can change the customer nodes assigned to the truck and drones, resulting in a change of the length of Part 1 or Part 3 of the chromosome. These heuristics are suitable to be occasionally called to improve the diversity of the population. The other heuristic operators (i.e., N_4, N_5, and N_6) focus on improving a solution based on a basis that customer nodes assigned to the truck and drones are fixed. They are incorporated into a Pareto local search method (Algorithm 3) to improve the convergence of the population.

E. Specialized Genetic Operations

1) Crossover: We adopt the one-point crossover in HMOA to allow a good position of customer nodes in a chromosome to be shared with other chromosomes. There exist twofold benefits to use this operation. First, only one customer node is changed in each chromosome, such that at most two assignments may be infeasible. It would consume less computational time to repair an infeasible chromosome. Second, by exchanging a customer node between two good chromosomes, the diversity of the population can be improved to a certain extent while keeping the main structure of chromosomes.

The crossover operation is performed according to a predefined crossover rate and the binary tournament selection [17]. Once two good chromosomes are selected from the population, a random and common crossover position will be selected from Part 1 or Part 2 in two chromosomes. Then, two chromosomes exchange corresponding customer nodes. To ensure the feasibility of chromosomes after a crossover, the $Repair$ heuristic is used to reassign infeasible customer nodes. If the $Repair$ heuristic cannot address the infeasibility of a chromosome, this operator returns the chromosome unchanged.

2) Multi-Mode Mutation: To further improve the diversity of the population, we use a multi-mode mutation introduced by [15]. This operator consists of three above-mentioned heuristics (i.e., N_1, N_2, N_3) and only one heuristic is chosen in each mutation operation. The mutation rate is considerably small since it would destructive to the structure of a chromosome. Meanwhile, we assign three operators (i.e., N_1, N_2, and N_3) the same rates to be triggered.

Furthermore, we propose a simple rule to ensure the quality of a solution. If the length of Part 1 is smaller than LTL, either N_2 or N_3 is considered. If the length of Part 3 is smaller than the number of drones, either N_1 or N_3 is chosen to perform the mutation. Otherwise, one of N_1, N_2, and N_3 is selected.

F. Remove Duplication

Due to the complex constraints of the problem, as well as the nature of HMOA that the proposed heuristics often return solutions unchanged during the optimization, a number of identical solutions may be preserved in the population after genetic operations. We develop a strategy named $RemoveDuplication$ based on the multi-mode mutation operator and a restart strategy to eliminate duplications. Concretely, in the combined population R_t, either the multi-mode mutation is triggered to perturb a solution randomly selected from the external population PF or $AssignNode$ is triggered to create a new initial solution once a duplicated solution is found. We define a parameter named restart rate α to determine the weights of the two operations. If a generated random probability is smaller than α, the multi-mode mutation is triggered. Otherwise, $AssignNode$ is called.

This combination is reasonable. Although the multi-mode mutation can search a larger neighborhood space of a non-dominated solution, it may still find a duplicated solution, thereby trap the population in the local optimum. On the other hand, $AssignNode$ can create a new initial solution that is beneficial to the diversity, but the new solution is far from the Pareto front and thus wastes previous search efforts. Furthermore, compared with using a larger mutation rate that may change an already good solution, just changing a duplicated solution can keep robust of the algorithm while further improving the diversity of the population.

G. Pareto Local Search

The Pareto local search (PLS) works with a set of non-dominated solutions and aims to find new solutions for updating this set iteratively [44]. The conventional PLS explores all non-dominated solutions at each iteration, which is not efficient. Particularly, a non-dominated solution at the early generations may be dominated by a new solution at the next generation. Therefore, inspired by [44], we improve the

Algorithm 3 PLS

```
Input: Pareto Front $PF$, $k_{max}$, heuristics $H$
Output: Improved Pareto front $PF$
1  Initialization: $k \leftarrow 1$, $PF' \leftarrow \emptyset$, $PL \leftarrow \emptyset$
2  $PF' \leftarrow$ Find solutions new to $PF$
3  while $k < k_{max}$ and $PL \neq \emptyset$ do
4      $PL \leftarrow \emptyset$
5      for each $p$ in $PF'$ do
6          for each $N_i$ in $H$ do
7              $p' \leftarrow N_i(p)$
8              if $p \prec p'$ then
9                  $PF \leftarrow Update2(p', PF)$
10                 if $p'$ has been added to $PF$ then
11                     $PL \leftarrow Update2(p', PL)$
12     $PF' \leftarrow PL$
13     $k \leftarrow k + 1$
14 Label all the solutions in $PF$
```

conventional PLS to avoid unbearable computational efforts, by providing more computational efforts to the solutions with a high chance to find new non-dominated solutions.

As can be seen in Algorithm 3, k_{max} determines the maximum iterations of PLS and $H = \{N_4, N_5, N_6\}$. The external population PF can help with identifying the solutions that are new to the current Pareto front at each generation. New solutions may be further improved, thereby only new solutions to PF would be explored in the local search.

In each iteration of the local search, PL is initialized to be empty at the beginning and used to reset PF' at the end (lines 4 and 12). For each neighbor p' of every solution p in $PF\prime$, an $Update2$ operator is triggered to update PF if p' dominates p (lines 9-13). In $Update2(p', PF)$, $p\prime$ is compared with each solution in PF, it would be added to PF if no solution in PF can dominate it. Meanwhile, all the solutions in PF dominated by p' will be deleted. If p' has been added to PF, line 15 would trigger $Update2(p', PL)$ to update PL. The two if conditions in lines 8 and 10 only allow very promising solutions to update PF and PL, such that the computational time can be reduced significantly.

Finally, the iteration repeats until PF' is empty or the maximum number of iterations is reached. In this paper, we set $k_{max} = 5$ to reduce computational time.

H. Repair and Improve Assignment

To address the infeasibility of chromosomes, we propose a heuristic operator called $Repair$ to reassign customer nodes while improving the solution in terms of cost, as Algorithm 4 shows.

In Algorithm 4, the infeasible solution p is firstly updated by removing all assignments associated with infeasible customer nodes in C_{in} (line 2). Then, all possible and feasible assignments for current solution p' are identified and stored in P (lines 3-8). The feasibility of an assignment indicates that this assignment is strictly constrained by flight endurance and not in conflict with other assignments in the current solution p'. If every infeasible customer node in C_{un} can be associated with at least one assignment in P, the infeasible customer nodes are assigned iteratively. Otherwise, this heuristic returns solution p unchanged (lines 9-20).

In each iteration, every promising assignment of all infeasible customer nodes is selected from P based on two criteria. First, the customer node j that has the least potential assignments are preferred (line 11). Second, the cheapest assignment $< i, j, k, d >$ in terms of cost is preferred (line 12). All the promising assignments are stored in P_{in} to update solution p' (line 13). Afterward, P is updated by removing all assignments that are associated with j and in conflict with $< i, j, k, d >$ (line 14). Finally, if all the nodes in C_{in} are arranged, this heuristic output a new solution p' by adding all assignments in P_{in} to p'. Otherwise, the unchanged solution p is treated as output.

Algorithm 4 Repair

```
Input: infeasible customer nodes $C_{in}$, truck nodes $C_t$,
       drone number $m$, solution $p$
Output: Solution $p'$
1  Initialization: Possible pairs set $P \leftarrow \emptyset$, pairs set for
   updating solution $P_{in} \leftarrow \emptyset$
2  $p' \leftarrow$ update $p$ by $C_{in}$
3  for $d = 1$ to $m$ do
4      for $i = 1$ to $|C_t| - 1$ do
5          for $k = i + 1$ to $|C_t|$ do
6              for each $j$ in $C_{in}$ do
7                  if $< i, j, k, d >$ is feasible and not in
                   conflict with assignments in $p$ then
8                      $P \leftarrow P \cup < i, j, k, d >$
9  if Every node in $C_{un}$ has at least one assignment in $P$
   then
10     while $P \neq \emptyset$ do
11         $P_j \leftarrow$ find assignments with the least chances for
           assigning $j$ from $P$
12         $< i, j, k, d > \leftarrow$ find the cheapest assignments
           from $P_j$
13         $P_{in} \leftarrow P_{in} \cup < i, j, k, d >$
14         $P \leftarrow$ update $P$ by $< i, j, k, d >$
15     if all nodes in $C_{in}$ are assigned then
16         $p' \leftarrow$ update $p'$ by $P_{in}$
17     else
18         $p' \leftarrow p$
19 else
20     $p' \leftarrow p$
```

V. Computational Experiments

To assess the performance of the proposed HMOA, we apply it to a set of benchmark instances and a real-world case. All the algorithms are coded in Python and all computations are performed on a 64-bit Windows OS with Intel Core(TM) i7-10700, 2.9 GHz, and 32 GB RAM.

A. Performance Metrics

Since the true PF of each instance is unknown, we use two performance metrics, i.e., hypervolume (HV) and coverage (C-metric) [44] for measuring performance. HV represents the volume among a solution set and a reference point. We select the reference point by the worst objective values from all solutions obtained by all algorithms in all runs. Meanwhile, two objectives are normalized to eliminate the efficiency of different scales. The larger the HV of the solution set, the better its diversity and convergence efficiency. C-metric can be denoted as $C(A, B)$, which represents the percentage of the solutions in B that are dominated by any solution in A. If C(A,B) equals 1, it means each solution in B is dominated by or equal to at least one solution in A. On the other hand, if $C(A, B)$ equals 0, it indicates that no solutions in B can be dominated by any solution in A. A is better than B if $C(A, B)$ is larger than $C(B, A)$. It should be noted that $C(A, B)$ is not necessarily equal to $1 - C(B, A)$.

B. Experiments on Benchmark Instances

1) Benchmark Instances: Since there are no available benchmark instances for the proposed Mo-CRPTW-mD, we extend the well-known TSPTW instances proposed by Dumas *et al.* [45]. These instances are categorized according to the number of customer nodes and the length of time windows, from which we extract 20 instances with the number of customer nodes varies from 20 to 80. Each size of instances consists of 5 datasets. The information of the original Dumas' instances used are customer index, coordinates, ready time a_i, and due time b_i of the time window associated with customer node i. Since original instances require that all customer nodes must be served between the ready time and the due time, we generate a flexible time window $[e_i, l_i]$ for each node i to obtain instances relevant to the Mo-CRPTW-mD model, where $e_i = a_i - wb_i^l(b_i - a_i)$ and $l_i = a_i + wb_i^u(b_i - a_i)$. Particularly, wb_i^l and wb_i^u are two fractions used to set the maximum allowed violations of time windows.

Furthermore, for all instances, the distance between two nodes traveled by truck is calculated with the Manhattan distance, while the distance traveled by a drone is calculated with the Euclidean distance. As the settings in [5], the speeds of drones and the truck are identical, while the cost of the truck is set to 25 times the cost of a drone. We assume that 85% of customer nodes in each instance are drone-eligible. To set the flight range of drones, we use the same method introduced in [32] and assume that 35% of feasible flights can be satisfied by drones in each instance.

2) Experiment Settings: Since there is no multi-objective optimization algorithm that can be directly used to address Mo-CRPTW-mD, we modify two competitors to compare with HMOA, including MOEAD [46] and MOMAD [44]. MOEAD is a well-known decomposition-based algorithm, which is modified by incorporating all the proposed genetic operators into it. MOMAD is a state-of-the-art multi-objective combinatorial optimization algorithm that can be considered as a hybridization of the MOEAD framework and PLS. Six neighborhood strategies proposed in this study and a conventional PLS are appropriately incorporated into MOMAD. Both MOEAD and MOMAD decompose the multi-objective problem based on Tchebycheff Approach. Additionally, we consider the HMOA without PLS (HMOA-noLS) as the third competitor. Note that since our algorithm is inspired by the framework of NSGA-II, HMOA-noLS can also be considered as an improved and problem-specific version of NSGA-II.

The parameter configurations are shown as follows. For each instance, the flexible time window-related parameters $wb_i^l = wb_i^u = 0.2$ and the truck takes 3 drones. The number of populations in each algorithm is set to 200. The crossover rate and mutation rate are set to 0.8 and 0.3, respectively. The restart rate $\alpha = 0.3$.

3) Experiment Results: Four algorithms are performed on the above-described 20 instances 15 times independently. The average and standard deviation of HV, as well as the average CPU time, are represented in Table I. As Dumas' instances, the name of each instance indicates the number of nodes and the width of times windows (e.g., n20w80 means the number of customer nodes is 20 and the width of time windows is 80). For each instance, the best results are in **boldface**. Wilcoxon rank-sum tests with significance level 0.05 are used for the significance tests. The symbols $+$, $\approx$, and $-$ represent that the corresponding algorithm is significantly better than, similar to, and significantly worse than HMOA, respectively.

In Table II, the results show that the Pareto fronts generated by HMOA are significantly better than the other three algorithms. Specifically, HMOA generates significantly higher HV values than HMOA-noLS, MOEAD, and MOMAD on all the instances except instances n20w80_001, n4080w_004, n60w80_002, and n80w80_001, on which HMOA generates higher average HV values but significantly similar with HMOA-noLS. The results can be explained in multi-fold. First, although MOEAD and MOMAD are excellent algorithm frameworks, they cannot address our proposed problem very well, which calls for further problem-specific modifications to improve their performances. Second, compared with MOEAD, HMOA-noLS uses the same genetic operations but significantly outperforms MOEAD since the proposed duplication elimination strategy in HMOA-noLS can further improve the diversity of the population. Third, since the PLS has a strong local search capability, HMOA can achieve better performance than HMOA-noLS on most instances.

Moreover, two hybrid algorithms (i.e., HMOA and MOMAD) take more computational time than the other two algorithms (i.e., HMOA-noLS and MOEAD) without local search. MOMAD takes the most computational time since it adopts two local searches at each generation. Compared with HMOA-noLS, the extra time required by HMOA to perform PLS is negligible for smaller instances, while it shows a trade-off between the execution time and the quality of the fronts for larger instances.

The average C-metric values of all the 15×15 comparisons on each instance are shown in Table III. The larger

TABLE II
HYPERVOLUME AND COMPUTATIONAL TIME OBTAINED FROM BENCHMARK INSTANCES

Instance	HMOA			HMOA-noLS			MOEAD			MOMAD		
	HV		Time (s)	HV		Time (s)	HV		Time (s)	HV		Time (s)
	Avg.	Std. dev.		Avg.	Std. dev.		Avg.	Std. dev.		Avg.	Std. dev.	
n20w80_01	**0.730**	**0.059**	75.899	**0.732**	**0.050**	65.235	0.640	0.061	40.690	0.588	0.072	93.831
n20w80_02	**0.780**	**0.044**	70.987	0.729	0.031	58.910	0.662	0.047	40.822	0.650	0.058	90.093
n20w80_03	**0.688**	**0.037**	70.184	0.647	0.038	59.546	0.498	0.057	44.678	0.565	0.046	99.783
n20w80_04	**0.695**	**0.019**	70.926	0.667	0.040	63.459	0.536	0.041	48.073	0.512	0.065	83.895
n20w80_05	**0.772**	**0.054**	70.820	0.655	0.041	59.532	0.484	0.073	37.978	0.574	0.077	86.444
n40w80_01	**0.740**	**0.058**	274.024	0.677	0.038	201.422	0.617	0.079	109.495	0.607	0.076	331.483
n40w80_02	**0.808**	**0.060**	265.519	0.763	0.061	190.227	0.628	0.068	132.548	0.507	0.107	304.138
n40w80_03	**0.702**	**0.053**	249.584	0.670	0.046	198.016	0.499	0.083	124.696	0.445	0.072	291.576
n40w80_04	**0.755**	**0.044**	209.959	**0.729**	**0.058**	167.231	0.559	0.103	140.842	0.359	0.049	378.955
n40w80_05	**0.657**	**0.051**	273.528	0.618	0.052	188.625	0.482	0.101	129.872	0.448	0.120	437.812
n60w80_01	**0.665**	**0.072**	613.367	0.591	0.091	438.709	0.431	0.127	403.311	0.315	0.043	1197.633
n60w80_02	**0.616**	**0.114**	618.819	**0.531**	**0.148**	447.967	0.288	0.064	391.864	0.335	0.059	887.242
n60w80_03	**0.780**	**0.066**	617.891	0.651	0.152	408.496	0.449	0.143	331.957	0.319	0.078	937.727
n60w80_04	**0.704**	**0.060**	638.221	0.660	0.044	462.132	0.518	0.12	356.855	0.317	0.055	969.164
n60w80_05	**0.693**	**0.116**	631.870	0.612	0.101	468.321	0.479	0.115	341.295	0.394	0.034	909.746
n80w80_01	**0.787**	**0.076**	1530.958	**0.760**	**0.045**	1073.439	0.608	0.063	695.259	0.536	0.071	1918.779
n80w80_02	**0.793**	**0.100**	1378.611	0.725	0.073	986.426	0.582	0.117	681.978	0.444	0.073	1765.777
n80w80_03	**0.720**	**0.102**	1338.237	0.633	0.074	947.103	0.485	0.098	675.878	0.357	0.043	1768.112
n80w80_04	**0.648**	**0.133**	1217.382	0.543	0.090	899.980	0.369	0.088	640.369	0.362	0.126	1851.753
n80w80_05	**0.729**	**0.102**	1481.462	0.666	0.094	1049.871	0.477	0.136	623.440	0.380	0.067	1688.972
$+/-/\approx$					0/16/4			0/20/0			0/20/0	

TABLE III
COMPARISON OF HMOA WITH OTHER ALGORITHMS ON COVERAGE

	C(HMOA,-)			C(-,HMOA)		
	HMOA-noLS	MOEAD	MOMAD	HMOA-noLS	MOEAD	MOMAD
n20w80_01	**0.527**	**0.847**	**0.809**	0.409	0.1	0.053
n20w80_02	**0.675**	**0.897**	**0.859**	0.261	0.044	0.045
n20w80_03	**0.496**	**0.934**	**0.704**	0.364	0.019	0.101
n20w80_04	**0.482**	**0.982**	**0.864**	0.38	0.002	0.012
n20w80_05	**0.817**	**0.97**	**0.872**	0.105	0.013	0.046
n40w80_01	**0.543**	**0.758**	**0.822**	0.328	0.065	0.050
n40w80_02	**0.527**	**0.854**	**0.936**	0.321	0.041	0.011
n40w80_03	**0.489**	**0.903**	**0.982**	0.29	0.015	0.005
n40w80_04	**0.515**	**0.906**	**0.963**	0.33	0.028	0.001
n40w80_05	**0.596**	**0.894**	**0.896**	0.315	0.044	0.034
n60w80_01	**0.583**	**0.931**	**0.998**	0.223	0.021	0.001
n60w80_02	**0.625**	**0.995**	**0.994**	0.216	0.000	0.000
n60w80_03	**0.548**	**0.952**	**0.993**	0.224	0.014	0.000
n60w80_04	**0.677**	**0.816**	**0.997**	0.186	0.056	0.000
n60w80_05	**0.546**	**0.850**	**0.983**	0.331	0.062	0.000
n80w80_01	**0.582**	**0.794**	**0.927**	0.227	0.059	0.002
n80w80_02	**0.635**	**0.877**	**0.989**	0.196	0.048	0.001
n80w80_03	**0.599**	**0.911**	**1.000**	0.248	0.039	0.000
n80w80_04	**0.618**	**0.949**	**0.975**	0.228	0.022	0.006
n80w80_05	**0.569**	**0.881**	**0.990**	0.288	0.046	0.002

values between C(HMOA,-) and C(-,HMOA) are in **boldface**. We can find that C(HMOA,-) values are significantly larger than C(-,HMOA) values on all the instances compared with the three competitors. It further indicates that our proposed HMOA outperforms the other three comparison algorithms. Particularly, most C(HMOA,-) values compared with MOEAD and MOMAD are very close to 1, which indicates that most solutions obtained by these algorithms are dominated by at least one solution of HMOA. On the other hand, most C(-,HMOA) values compare with MOEAD and MOMAD approximate 0, which means very few solutions of HMOA are dominated by those found by MOEAD and MOMAD. Although all the C(HMOA, HMOA-noLS) values are still larger than C(HMOA-noLS, HMOA) values on all instances, on few instances (e.g., n20w80_03 and n20w80_04) less than a half solutions can be dominated by at least one solution of HMOA, which indicates that the coverages of their Pareto fronts are similar.

To visually compare the performance of the four algorithms, instances n20w80_01, n40w80_01, n60w80_01, and n80w80_01 are selected to illustrate the output solutions of the four algorithms in terms of objective space, respectively. The best solutions produced by each algorithm over 15 runs are compared in Figure 3. For better observation, the values of customer satisfaction are converted to negative. The results show that HMOA has better overall convergence and diversity than other algorithms, which confirms the advantage of HMOA.

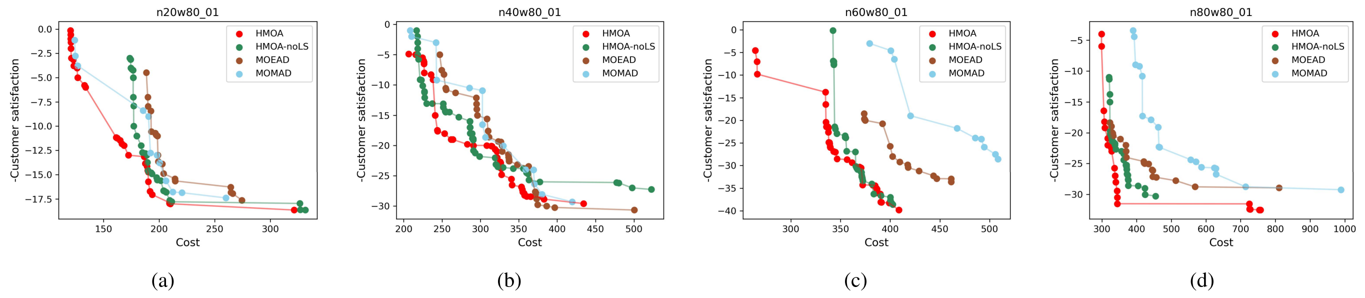


Fig. 3. Comparison of the Pareto fronts obtained by HMOA, HMOA-noLS, MOEAD, and MOMAD on n20w80_01, n40w80_01, n60w80_01, and n80w80_01.

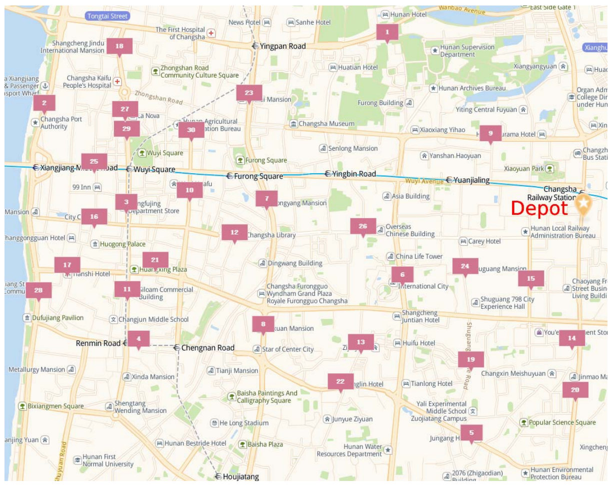


Fig. 4. Geographical positions of 30 customer nodes.

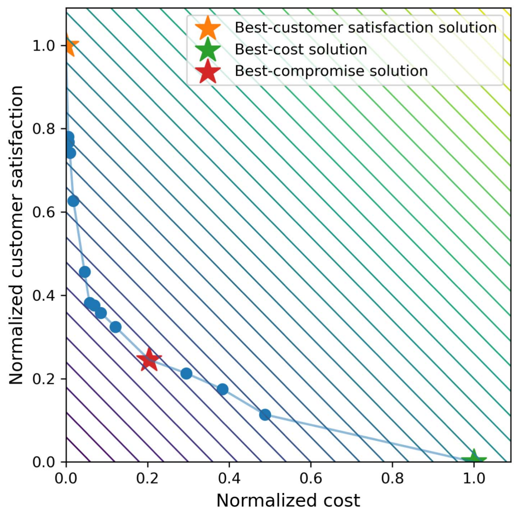


Fig. 5. Normalized representation with isolines for the Pareto front.

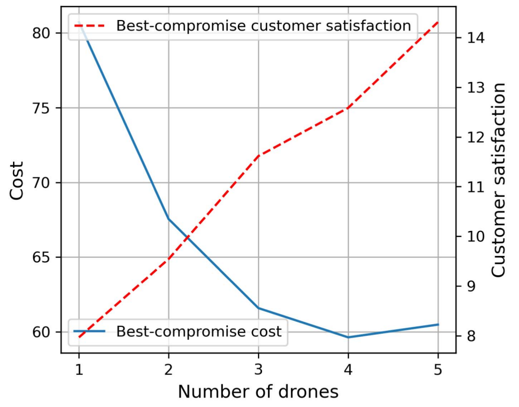


Fig. 6. The average values of the best-compromise solutions by varying the number of drones.

C. Experiments on Real-World Case

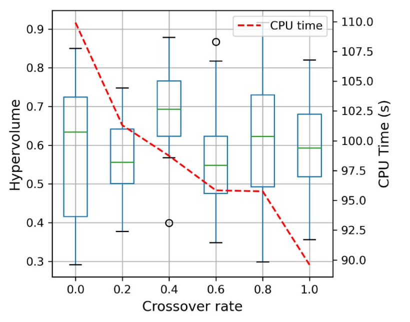
1) Case Description: We develop a real-world case that comes from Changsha, China, to validate the Mo-CRPTW-mD model and analyze the sensitivity of HMOA. As Figure 4 shows, the problem consists of 30 customer nodes in the eastern urban area of Changsha, which are served from a depot near the Changsha Train Station. The distance traveled by truck between each pair of nodes is obtained via Gaode Map API, while the distance traveled by drones between each pair of nodes is calculated according to Euclidean norm. As in [4], [11], the speed of the truck and drones are both set to 50 kilometers per hour, and the flight limitation of each drone is set to 8 kilometers. The transportation costs of using a truck and a drone are 3 Yuan and 0.2 Yuan per kilometer, respectively. The time windows of all the customer nodes are randomly distributed between 8:00 am and 11:00 am. The length of each time window is uniformly distributed within $[20, 40]$ minutes. The service time of a customer node served by the truck is uniformly distributed within $[10, 15]$ minutes.

2) Trade-off Analysis Between Cost and Customer Satisfaction: Since there exists a set of solutions on the Pareto front, the decision-maker needs to select an appropriate scheme according to actual situations. In most situations, the best-compromise solution is desirable, which can be obtained based on its distance to the ideal point [47]. Specifically, every objective value is normalized with respect to the ideal point. Then, a Euclidean norm is applied to evaluate the distance to the ideal point. The solution with the shortest distance is the best-compromise solution.

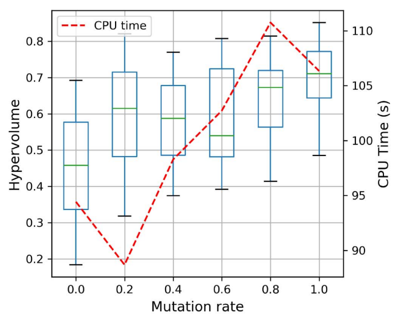
In this section, we select the best Pareto front in terms of average HV values from 15 runs to analyze the trade-off between cost and customer satisfaction. In each run the number of drones $|D| = 2$ and $wb_i^l = wb_i^u = 0.5$ $(i \in \{1, 2, \ldots, c\})$. The other parameters are the same as the experiments on benchmark instances. The normalized representation with isolines for the Pareto front is shown in Figure 5. As we can see, the best-compromise solution, the best-customer

TABLE IV
COMPARISON OF SOLUTIONS CONSTRAINED BY FLEXIBLE TIME WINDOWS AND HARD TIME WINDOWS

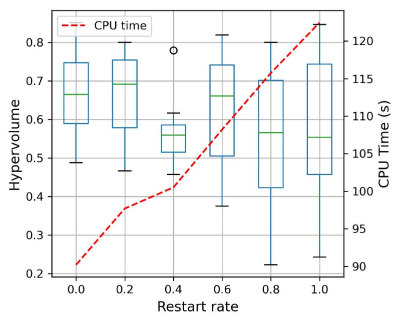
Solutions	Flexible time window		Hard time window		Compared with hard time window	
	Cost (Yuan)	Satisfaction	Cost (Yuan)	Satisfaction	Increase of cost (%)	Increase of satisfaction (%)
Best-customer satisfaction	80.968	11.311	77.567	10.230	4.385	10.567
Best-compromise	68.440	9.416	67.478	6.700	1.426	40.537
Best-cost	64.105	3.413	64.465	2.681	-0.558	27.303



(a) Impact of the crossover rate on HMOA



(b) Impact of the mutation rate on HMOA



(c) Impact of the restart rate on HMOA

Fig. 7. Impact analysis of algorithm parameters.

satisfaction solution, and the best-cost solution are marked by red, orange, and green stars, respectively. Accordingly, the transportation cost and customer satisfaction of the best-compromise solution are 58.217 Yuan and 9.415. The transportation cost and customer satisfaction of the best-customer satisfaction solution are 69.768 Yuan and 11.823. For the best-cost solution, the transportation cost and customer satisfaction of the best-customer satisfaction solution are 55.256 Yuan and 2.

To further analyze the trade-off between transportation cost and customer satisfaction, we identify the best-compromise solution, the best-customer satisfaction solution, and the best-cost solution for all 15 runs. The average values are represented in the left of Table IV. It indicates that if a company uses the best-compromise solution, the transportation cost will decrease 15.473% compared with the best-customer satisfaction solution, and the customer satisfaction will increase 175.886% compared with the best-cost solution.

3) Comparison Between Flexible Time Window and Hard Time Window: We compare the Mo-CRPTW-mD model with $wb_i^l = wb_i^u = 0$ and with $wb_i^l = wb_i^u = 0.5$ $(i \in 1, 2, \ldots, c)$ to analyze the impact of flexible time windows. The results are presented in Table IV. The results show that if the flexible time windows are adopted, both the transportation cost (except the average cost value of best-cost solutions) and overall customer satisfaction are increased. Nevertheless, the increased degree of transportation cost is negligible, while the increased overall customer satisfaction is huge. Specifically, if a company adopts the best-compromise solution, the transportation cost is increased by 1.426% while the customer satisfaction is increased by 40.537% after considering flexible time windows. The results can be explained that flexible time windows can provide more chances for vehicles to serve more customers, such that demands of more customer nodes can be satisfied. Based on the above observations, the decision makers can choose to improve the overall customer satisfaction at the expense of few transportation cost by considering the flexible time windows.

4) Impact Analysis of the Number of Drones: The above experiments on the real-world case consider 2 drones. To investigate the impact of the number of drones on the results, we perform HMOA on the real-world case by varying the number of drones from 1 to 5. The average values of the best-compromise solutions over 15 runs are presented in Figure 6. It indicates that as the increase of the number of drones, lower transportation costs and higher customer satisfaction can be achieved.

5) Impact Analysis of Algorithm Parameters: HMOA involves three major parameters: crossover rate, mutation rate, and restart rate α. We conduct experiments by varying the three parameters from 0 to 1 respectively over 15 runs on the real-world case to study the sensitivity of HMOA. The experimental results are displayed in Figure 7. It can be found that both of the three parameters affect the computational time of HMOA significantly. The higher the crossover rate, the less computational time. On the contrary, the higher the mutation rate and the restart rate, the more computational time. Nevertheless, the HV values do not change linearly with the increase of the crossover rate and the restart rate. The higher mutation rate may obtain higher HV values, but there exists a trade-off between the computational time and the performance of HMOA.

VI. CONCLUSION

In this paper, we consider a multi-objective collaborative routing problem with a truck and multiple drones, as well as flexible time windows constraints, in which customer nodes are able to be served by either a truck or a drone with a certain tolerance. The transportation cost and customer satisfaction are optimized simultaneously by developing a population-based hybrid multi-objective optimization approach (HMOA). The proposed algorithm involves a problem-specific solution representation, genetic operations, as well as an improved PLS incorporated with problem-specific heuristic neighborhood strategies. Meanwhile, an adaptive strategy is implemented to balance the diversity and convergence of the population.

Extensive experimental studies are carried out to evaluate the proposed model and algorithm. The experimental results on benchmark instances show the superiority of the proposed algorithm compared with competitors, and the experimental studies on a real-world case tested the sensitivity of HMOA. Meanwhile, we provide a method to find the best-compromise solutions for assisting decision makings of logistic companies. The experimental results also verified that by adopting the flexible time windows, a logistic company can achieve higher overall customer satisfaction while paying for few external transportation costs in terms of the best-compromise solutions. Besides, the higher the number of drones, the fewer transportation costs and the higher customer satisfaction can be obtained.

Our proposed problem can be easily extended to a multi-truck and multi-drone version. We can assume a scenario where customers are divided into several groups and each group is served by a truck collaborated with a fleet of drones. A clustering method based on node distances such as k-means would be promising to obtain the customer groups. Furthermore, the effects of parcel weights on the travel time and energy consumption of drones are assumed to be negligible in this study. To get close to reality, we would like to incorporate the effects in problems in future studies.

REFERENCES

[1] S. H. Chung, B. Sah, and J. Lee, "Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions," *Comput. Oper. Res.*, vol. 123, Nov. 2020, Art. no. 105004.

[2] C. C. Murray and A. G. Chu, "The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery," *Transp. Res. C, Emerg. Technol.*, vol. 54, pp. 86–109, May 2015.

[3] N. Agatz, P. Bouman, and M. Schmidt, "Optimization approaches for the traveling salesman problem with drone," *Transp. Sci.*, vol. 52, no. 4, pp. 965–981, 2018.

[4] C. C. Murray and R. Raj, "The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones," *Transp. Res. C, Emerg. Technol.*, vol. 110, pp. 368–398, Jan. 2020.

[5] Q. M. Ha, Y. Deville, Q. D. Pham, and M. H. Hà, "On the min-cost traveling salesman problem with drone," *Transp. Res. C, Emerg. Technol.*, vol. 86, pp. 597–621, Jan. 2018.

[6] S. Poikonen, X. Wang, and B. Golden, "The vehicle routing problem with drones: Extended models and connections," *Networks*, vol. 70, no. 1, pp. 34–43, 2017.

[7] D. Sacramento, D. Pisinger, and S. Ropke, "An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones," *Transp. Res. C, Emerg. Technol.*, vol. 102, pp. 289–315, May 2019.

[8] Z. Wang and J.-B. Sheu, "Vehicle routing problem with drones," *Transp. Res. B, Methodol.*, vol. 122, pp. 350–364, Apr. 2019.

[9] P. Kitjacharoenchai, M. Ventresca, M. Moshref-Javadi, S. Lee, J. M. A. Tanchoco, and P. A. Brunese, "Multiple traveling salesman problem with drones: Mathematical model and heuristic approach," *Comput. Ind. Eng.*, vol. 129, pp. 14–30, Mar. 2019.

[10] D. Tas, O. Jabali, and T. Van Woensel, "A vehicle routing problem with flexible time windows," *Comput. Oper. Res.*, vol. 52, pp. 39–54, Dec. 2014.

[11] H. Zhang, Q. Zhang, L. Ma, Z. Zhang, and Y. Liu, "A hybrid ant colony optimization algorithm for a multi-objective vehicle routing problem with flexible time windows," *Inf. Sci.*, vol. 490, pp. 166–190, Jul. 2019.

[12] T. Lust and J. Teghem, "Two-phase Pareto local search for the biobjective traveling salesman problem," *J. Heuristics*, vol. 16, no. 3, pp. 475–510, 2010.

[13] G. Wu, R. Mallipeddi, and P. N. Suganthan, "Ensemble strategies for population-based optimization algorithms—A survey," *Swarm Evol. Comput.*, vol. 44, pp. 695–711, Feb. 2019.

[14] H. Chen, G. Wu, W. Pedrycz, P. N. Suganthan, L. Xing, and X. Zhu, "An adaptive resource allocation strategy for objective space partition-based multiobjective optimization," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 51, no. 3, pp. 1507–1522, Mar. 2021.

[15] K. C. Tan, Y. H. Chew, and L. H. Lee, "A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems," *Eur. J. Oper. Res.*, vol. 172, no. 3, pp. 855–885, 2006.

[16] K. C. Tan, Y. H. Chew, and L. Lee, "A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows," *Comput. Optim. Appl.*, vol. 34, no. 1, pp. 115–151, 2006.

[17] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in *Proc. Int. Conf. Parallel Problem Solving Nature*. Berlin, Germany: Springer, 2000, pp. 849–858.

[18] M. Dell'Amico, R. Montemanni, and S. Novellani, "Drone-assisted deliveries: New formulations for the flying sidekick traveling salesman problem," *Optim. Lett.*, vol. 15, no. 5, pp. 1617–1648, Jul. 2021.

[19] Q. M. Ha, Y. Deville, Q. D. Pham, and M. H. Hà, "A hybrid genetic algorithm for the traveling salesman problem with drone," *J. Heuristics*, vol. 26, no. 2, pp. 219–247, Apr. 2020.

[20] M. Marinelli, L. Caggiani, M. Ottomanelli, and M. Dell'Orco, "En route truck–drone parcel delivery for optimal vehicle routing strategies," *IET Intell. Transp. Syst.*, vol. 12, no. 4, pp. 253–261, May 2018.

[21] S. Poikonen, B. L. Golden, and E. Wasil, "A branch-and-bound approach to the traveling salesman problem with a drone," *Informs J. Comput.*, vol. 31, no. 2, pp. 335–346, 2019.

[22] Z. Luo, Z. Liu, and J. Shi, "A two-echelon cooperated routing problem for a ground vehicle and its carried unmanned aerial vehicle," *Sensors*, vol. 17, no. 5, p. 1144, May 2017.

[23] Y. Liu, Z. Liu, J. Shi, G. Wu, and W. Pedrycz, "Two-echelon routing problem for parcel delivery by cooperated truck and drone," *IEEE Trans. Syst., Man, Cybern., Syst.*, early access, Feb. 10, 2020, doi: 10.1109/TSMC.2020.2968839.

[24] M. Boccia, A. Masone, A. Sforza, and C. Sterle, "A column-and-row generation approach for the flying sidekick travelling salesman problem," *Transp. Res. C, Emerg. Technol.*, vol. 124, Mar. 2021, Art. no. 102913.

[25] J. C. D. Freitas and P. H. V. Penna, "A variable neighborhood search for flying sidekick traveling salesman problem," *Int. Trans. Oper. Res.*, vol. 27, no. 1, pp. 267–290, Apr. 2019.

[26] K. Wang, B. Yuan, M. Zhao, and Y. Lu, "Cooperative route planning for the drone and truck in delivery services: A bi-objective optimisation approach," *J. Oper. Res. Soc.*, vol. 71, no. 10, pp. 1657–1674, 2020.

[27] E. E. Yurek and H. C. Ozmutlu, "A decomposition-based iterative optimization algorithm for traveling salesman problem with drone," *Transp. Res. C, Emerg. Technol.*, vol. 91, pp. 249–262, Jun. 2018.

[28] I. Dayarian, M. Savelsbergh, and J.-P. Clarke, "Same-day delivery with drone resupply," *Transp. Sci.*, vol. 54, no. 1, pp. 229–249, 2020.

[29] M. Dell'Amico, R. Montemanni, and S. Novellani, "Matheuristic algorithms for the parallel drone scheduling traveling salesman problem," *Ann. Oper. Res.*, vol. 289, no. 2, pp. 211–226, Jun. 2020.

[30] S. Kim and I. Moon, "Traveling salesman problem with a drone station," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 49, no. 1, pp. 42–52, May 2019.

[31] R. G. Mbiadou Saleu, L. Deroussi, D. Feillet, N. Grangeon, and A. Quilliot, "An iterative two-step heuristic for the parallel drone scheduling traveling salesman problem," *Networks*, vol. 72, no. 4, pp. 459–474, Dec. 2018.

[32] M. Moshref-Javadi, A. Hemmati, and M. Winkenbach, "A truck and drones model for last-mile delivery: A mathematical model and heuristic approach," *Appl. Math. Model.*, vol. 80, pp. 290–318, Apr. 2020.

[33] M. Moshref-Javadi, S. Lee, and M. Winkenbach, "Design and evaluation of a multi-trip delivery model with truck and drones," *Transp. Res. E, Logistics Transp. Rev.*, vol. 136, Apr. 2020, Art. no. 101887.

[34] P. A. Tu, N. T. Dat, and P. Q. Dung, "Traveling salesman problem with multiple drones," in *Proc. 9th Int. Symp. Inf. Commun. Technol.* New York, NY, USA: Association for Computing Machinery, 2018, pp. 46–53.

[35] Y. S. Chang and H. J. Lee, "Optimal delivery routing with wider drone-delivery areas along a shorter truck-route," *Expert Syst. Appl.*, vol. 104, pp. 307–317, Aug. 2018.

[36] S. M. Ferrandez, T. Harbison, T. Weber, R. Sturges, and R. Rich, "Optimization of a truck-drone in tandem delivery network using k-means and genetic algorithm," *J. Ind. Eng. Manage.*, vol. 9, no. 2, pp. 374–388, 2016.

[37] H. Li, H. Wang, J. Chen, and M. Bai, "Two-echelon vehicle routing problem with time windows and mobile satellites," *Transp. Res. B, Methodol.*, vol. 138, pp. 179–201, Aug. 2020.

[38] A. M. Ham, "Integrated scheduling of m-truck, m-drone, and m-depot constrained by time-window, drop-pickup, and m-visit using constraint programming," *Transp. Res. C, Emerg. Technol.*, vol. 91, pp. 1–14, Jun. 2018.

[39] M. W. Ulmer and B. W. Thomas, "Same-day delivery with heterogeneous fleets of drones and vehicles," *Networks*, vol. 72, no. 4, pp. 475–505, Dec. 2018.

[40] D. N. Das, R. Sewani, J. Wang, and M. K. Tiwari, "Synchronized truck and drone routing in package delivery logistics," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 9, pp. 5772–5782, Sep. 2021.

[41] M. Salama and S. Srinivas, "Joint optimization of customer location clustering and drone-based routing for last-mile deliveries," *Transp. Res. C, Emerg. Technol.*, vol. 114, pp. 620–642, May 2020.

[42] S. M. Shavarani, M. G. Nejad, F. Rismanchian, and G. Izbirak, "Application of hierarchical facility location problem for optimization of a drone delivery system: A case study of Amazon prime air in the city of San Francisco," *Int. J. Adv. Manuf. Technol.*, vol. 95, nos. 9–12, pp. 3141–3153, Apr. 2018.

[43] C. E. Cortés, M. Matamala, and C. Contardo, "The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method," *Eur. J. Oper. Res.*, vol. 200, no. 3, pp. 711–724, Feb. 2010.

[44] L. Ke, Q. Zhang, and R. Battiti, "Hybridization of decomposition and local search for multiobjective optimization," *IEEE Trans. Cybern.*, vol. 44, no. 10, pp. 1808–1820, Oct. 2014.

[45] Y. Dumas, J. Desrosiers, E. Gelinas, and M. M. Solomon, "An optimal algorithm for the traveling salesman problem with time windows," *Oper. Res.*, vol. 43, no. 2, pp. 367–371, 1995.

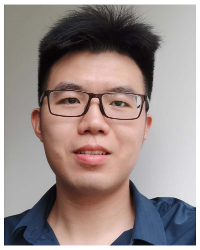
[46] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[47] X. Blasco, J. M. Herrero, J. Sanchis, and M. Martínez, "A new graphical visualization of n-dimensional Pareto front for decision-making in multiobjective optimization," *Inf. Sci.*, vol. 178, no. 20, pp. 3908–3924, Oct. 2008.

Qizhang Luo received the B.S. degree and the M.S. degree in traffic and transportation engineering from Central South University, Changsha, China, in 2015 and 2018, respectively, where he is currently pursuing the Ph.D. degree with the School of Traffic and Transportation Engineering. He is also a Visiting Student with the National University of Singapore. His current research interests include computational intelligence and scheduling, with a focus on their applications in traffic and transportation fields.

Guohua Wu (Member, IEEE) received the B.S. degree in information systems and the Ph.D. degree in operations research from the National University of Defense Technology, China, in 2008 and 2014, respectively.

From 2012 to 2014, he was a Visiting Ph.D. Student with the University of Alberta, Edmonton, Canada. He is currently a Professor with the School of Traffic and Transportation Engineering, Central South University, Changsha, China. He has authored more than 80 refereed articles, including those published in IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS, and IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS. His current research interests include planning and scheduling, computational intelligence, and machine learning. He also serves as an Associate Editor for *Swarm and Evolutionary Computation* journal, an Editorial Board Member for the *International Journal of Bio-Inspired Computation*, and a Guest Editor for *Information Sciences* and *Memetic Computing*. He is also a Regular Reviewer of more than 20 journals, including IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART A: SYSTEMS AND HUMANS and *Information Sciences*.



Bin Ji received the Ph.D. degree from the School of Hydropower and Information Engineering, Huazhong University of Science and Technology, China, in 2018. He is currently a Professor with the School of Traffic and Transportation Engineering, Central South University, China. His research interest includes optimization and its application with transportation.

Ling Wang received the B.Sc. degree in automation and the Ph.D. degree in control theory and control engineering from Tsinghua University, Beijing, China, in 1995 and 1999, respectively.

Since 1999, he has been with the Department of Automation, Tsinghua University, where he became a Full Professor in 2008. He has authored five academic books and more than 300 refereed articles. His current research interests include intelligent optimization and production scheduling.

Prof. Wang was a recipient of the National Natural Science Fund for Distinguished Young Scholars of China, the National Natural Science Award (Second Place) in 2014, the Science and Technology Award of Beijing City in 2008, and the Natural Science Award (First Place in 2003, and Second Place in 2007) nominated by the Ministry of Education of China. He is currently the Editor-in-Chief of the *International Journal of Automation and Control*, and an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, *Swarm and Evolutionary Computation*, and *Expert Systems With Applications*.

Ponnuthurai Nagaratnam Suganthan (Fellow, IEEE) received the B.A. and M.A. degrees from the University of Cambridge, Cambridge, U.K., and the Ph.D. degree from Nanyang Technological University, Singapore. He was a recipient of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award in 2012 and the Highly Cited Researcher Award by the Thomson Reuters in computer science in 2015. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON CYBERNETICS, *Information Sciences*, and *Pattern Recognition*, and the Founding Co-Editor-in-Chief of *Swarm and Evolutionary Computation* journal.