The Multi-visit Drone-assisted Pickup and Delivery Problem with Time Windows

Shanshan Meng [a,b], **Yanru Chen** [a*], **Dong Li** [c]

[a] *School of Economics and Management, Southwest Jiaotong University, Chengdu, China*

[b] *School of Business and Economics, Loughborough University, Loughborough, LE11 3TU, United Kingdom*

[c] *Department of Management Science, Lancaster University, Lancaster, LA1 4YX, United Kingdom*

Abstract

We consider a new combined truck-drone routing problem with time windows in the context of last-mile logistics. A fleet of trucks, each equipped with an identical drone, is scheduled to provide both pickup and delivery services to a set of customers with minimum cost. Some customers are paired, in that the goods picked up from one must be delivered to the other on the same route. Drones are launched from and retrieved by trucks at a pool of designated stations, which can be used multiple times. Each drone can serve multiple customers in one flight. We formulate this problem as a large-scale mixed-integer bilinear program, with the bilinear terms used to calculate the load-time-dependent energy consumption of drones. To accelerate the solution process, multiple valid inequalities are proposed. For large-size problems, we develop a customised adaptive large neighbourhood search (ALNS) algorithm, which includes several preprocessing procedures to quickly identify infeasible solutions and accelerate the search process. Moreover, two feasibility test methods are developed for trucks and drones, along with an efficient algorithm to determine vehicles' optimal waiting time at launch stations, which is important to consider due to the time windows. Extensive numerical experiments demonstrate the effectiveness of the valid inequalities and the strong performance of the proposed ALNS algorithm over two benchmarks in the literature, and highlight the cost-savings of the combined mode over the truck-only mode and the benefits of allowing multiple drone visits.

Keywords: Routing; Multi-visit drone routing problem; Pickup and delivery; Time windows; Adaptive large neighbourhood search

*Corresponding author.

Email addresses: S.Meng@lboro.ac.uk (Shanshan Meng), chenyanru@swjtu. edu.cn (Yanru Chen*), dong.li@lancaster.ac.uk (Dong Li).

1. Introduction

In recent years, the development of technologies makes it possible to use unmanned aerial vehicles, also referred to as drones, in last-mile logistics (Tamke & Buscher, 2021; Masmoudi et al., 2022; Kloster et al., 2023). Drone delivery has been well studied in the literature and implemented by some pioneering companies due to the benefits of lower cost, faster speed, directness of travel, and fewer pollution concerns compared to traditional road vehicle deliveries (Sacramento et al., 2019). Drone pickup, although still in the early stage, has recently been implemented in operations such as healthcare (Poljak & Šterbenc, 2020) and urban retail (Shen, 2022). A third service, which involves picking up goods from one customer and delivering them directly to another without warehousing, has emerged due to the boom in instant retail and food delivery. The main advantages of such a one-to-one (or pair) service are the reduced delivery time and increased customer satisfaction. For example, in Shenzhen, China, drones have been used to provide pickup and delivery services directly between local stores and residents around the business district (Shen, 2022). The pair service is also studied in Luo et al. (2022), where multiple trucks and drones are deployed to provide one-to-one pickup and delivery services in a food delivery setting.

Despite these benefits, drones are limited by their short flight range and low carrying capacity (Tamke & Buscher, 2021), which could be offset by a multi-modal solution that combines drones with trucks. In the combined mode, each truck is equipped with one or more drones, which are dispatched from and retrieved by the truck to perform multiple flights. Before each dispatch, the drones' batteries are replaced with fully charged ones. Dispatching/retrieving can take place at different locations, such as customer locations (Meng et al., 2023). However, drone launch and retrieval usually require a spacious area and specialised devices, e.g., unmanned aprons/hubs, which cannot be easily provided at customer locations (Wang & Sheu, 2019). Therefore it is more practical and safer to designate some dedicated locations (or stations) that trucks can use for drone launch/retrieval. Moreover, the stations may act as resupply centres where batteries can be stored/charged and shared by multiple vehicles.

This paper concerns a truck-drone combined pickup and delivery routing problem with time windows, an important feature to consider due to the growing focus on service quality in e-commerce (Wang et al., 2020b). Each truck is equipped with a single drone (i.e., a vehicle pair) and both can serve customers. A drone can visit multiple customers during each flight (i.e., multi-visit). Both vehicles must rendezvous at designated stations for dispatching, retrieval and battery replacement. We consider three types of services: pickup only, delivery only, and pair service. Our aims are to (i) select the optimal launch and retrieval nodes from a pool of stations and (ii) determine the optimal number of vehicle pairs to deploy and the optimal routes/flights for each vehicle pair so that all customer demands are satisfied with the minimal total cost, which includes the fixed cost for deployed vehicles and the operational cost of both trucks and drones.

An essential problem to consider for drones is their energy consumption, which we model into four components corresponding to different drone manoeuvres: flying, serving, hovering over customer locations

and hovering over retrieval stations. The energy consumption of each manoeuvre is usually modelled as a function of the payload and time duration (Torabbeigi et al., 2020), which could result in a nonlinear relationship when both payload and duration are dependent upon the routing decisions. Note that this is not an issue for delivery only problems, as the payload is always zero while hovering before being retrieved by a truck, while the flying speed and service time are usually assumed to be constant. However, as we simultaneously consider pickup and delivery services, both the payload and duration of hovering are unknown a priori but determined by the route and flight decisions, leading to a more complicated energy consumption calculation. Few studies have taken such detailed and realistic energy consumption modelling into consideration. The only similar work avoided the complication by approximating energy consumption as a weighted sum of payload and duration (Luo et al., 2022), begging the question of its applicability. Moreover, as we consider time windows, the possible energy consumed while hovering at customer locations before their time windows open must be considered and optimised. The dispatching time at the launch stations is therefore an important and complex decision that plays a key role in the total energy consumption, as it directly affects the energy that drones may consume at each customer location and at the retrieval stations. As a result, flights with the same visit sequence may consume different amounts of energy.

Furthermore, due to the precedence constraints imposed by the paired customers, the routes in opposite directions or with different types of customers may result in different energy consumption or become infeasible. We demonstrate the challenges with an example in Figure 1, where customer 1 has a pickup demand of 2 kg while customer 2 has a delivery demand of 2 kg. Let the maximum load capacity of drones be 3 kg. Assume customers 1 and 2 are not paired. If the drone visits customer 1 first, it will consume more energy and the route is even infeasible due to overload (see left graph). However, if customer 2 is visited first, as shown in the middle graph, the drone will consume less energy and the route is load feasible. In contrast, if customers 1 and 2 are paired, customer 1 must be visited first and the route becomes load feasible and consumes the least amount of energy, as an empty drone is launched from the station (see right graph).

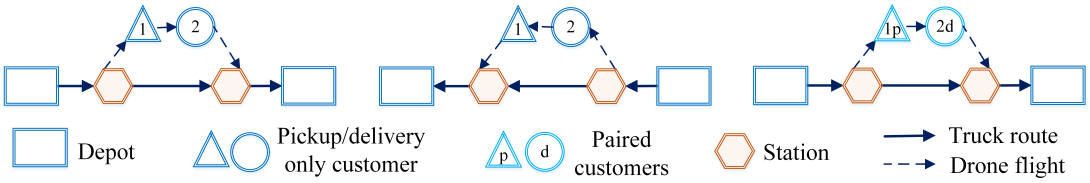


Figure 1. Routes in opposite directions and with different types of customers.

In addition, for pair services, the parcel from the pickup node (1p) must be delivered to its counterpart (2d) on the same route. That is, only when the parcel from node (1p) has been picked up can node (2d) be served. Therefore, as shown in Figure 2, a customer pair cannot be served in parallel by two sub-routes (Figure 2 (a)). At least one station must be visited in-between the customer pair if they are to be served separately by a drone and its truck (Figure 2 (b)). There is no such a requirement if they are both served by either the truck or drone.

Therefore, considering the energy consumption of drones, payload variation at each node, the precedence

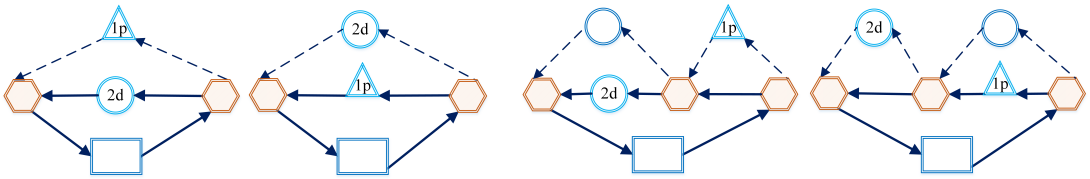


Figure 2. Feasibility of routes for serving customer pairs. For each truck-drone, the sub-routes denote the parallel drone flight and the truck route from the same launch station till their rendezvous at the retrieval station.

constraints of paired customers and the time windows, the problem is significantly more complex than the previous vehicle routing problems with drones (VRP-D). To the best of our knowledge, this is the first work addressing these challenges in the literature. Most previous studies focused on delivery only services (Wang et al., 2017; Kitjacharoenchai et al., 2020), and few investigated both pickup and delivery. One such recent work by Meng et al. (2023) considered a multi-visit drone-routing problem for pickup and delivery services. They formulated the problem as a mixed-integer program and developed a simulated annealing heuristic for large problem instances. However, they assumed that drones can be launched and retrieved at any customer location, and they did not consider either pair service or time windows. Luo et al. (2022) studied pair service with multiple trucks and drones but did not consider the other two types of customers or time windows. The latest work that considered time windows in the context of VRP-D is due to Yin et al. (2023), who explored a combined delivery problem with time windows but did not consider either the pickup or pair service. A comprehensive review of the related literature is provided in Section 2.

In summary, we make the following contributions to the literature. First, to the best of our knowledge, this is the first study to consider three types of customers with time windows in a VRP-D framework. The consideration of both pair service and time windows makes the problem significantly more complicated than those considered in previous works, as mentioned above. Second, we formulate the problem as a large-scale mixed-integer bilinear program (MIBP) with a linear objective function. All the constraints are linear except those concerning the hovering energy consumption for drones, which is the product of the payload and hovering time; both are continuous variables. Even though in principle MIBP can be solved to global optimality by commercial solvers such as Gurobi, we introduce multiple valid inequalities based on the problem structure to accelerate the solution process and allow larger problem instances to be amenable to solvers. Third, for large-size instances, we develop an adaptive large neighbourhood search (ALNS) metaheuristic with problem-tailored operators, pre-processing procedures, and two feasibility test methods for trucks and drones. The results of our extensive numerical experiments demonstrate the effectiveness of the valid inequalities in reducing the solver's run time and the strong performance of the proposed ALNS in obtaining high-quality solutions over two benchmarks in the literature. The results also show the significant cost savings of the combined mode compared against the truck-only mode and the remarkable benefit of allowing multiple visits of drones. We believe the proposed methodologies and resulting managerial insights will help practitioners better manage truck-drone pickup and delivery operations.

The remainder of this paper is organised as follows. Section 2 briefly reviews the related literature.

Section 3 introduces the proposed mathematical model. In Section 4, an ALNS heuristic is developed to handle large-size instances. Section 5 reports a series of numerical experiments and feature analyses. The conclusion is presented in Section 6.

2. Related literature

The combined mode VRP-D has attracted considerable interest in the vehicle routing research community and various applications of drones in last-mile logistics have been investigated. The majority of the literature concerns delivery only operations, while very few studies consider both pickup and delivery services, despite the importance in real-world logistics (Wang et al., 2020a). After a review of these two bodies of literature, we move on to the problems concerning pair service in both the VRP-D and traditional vehicle routing problems (VRP), followed by a summary of VRP-D with time windows. A summary of the most related studies is presented in Supplementary material A.

2.1. VRP-D with delivery and/or pickup services

Wang et al. (2017) pioneered the vehicle routing problem with drones, which considers a fleet of trucks, each carrying a set of drones with the capability of one parcel delivery per flight (i.e., a single visit), for delivering all parcels with the minimal completion time, demonstrating the practical benefits of the VRP-D. Several researchers have studied the VRP-D and its variants. Sacramento et al. (2019) explored the operational cost of the VRP-D and Schermer et al. (2019) extended the VRP-D with en route operations. Both exact methods (Najy et al., 2023; Tamke & Buscher, 2021; Zhen et al., 2023) and heuristics (Rave et al., 2023; Sacramento et al., 2019; Schermer et al., 2019) have been developed to solve the VRP-D. For example, Najy et al. (2023) incorporated the VRP-D into an inventory routing problem and proposed a branch-and-cut algorithm to solve instances with up to 50 customers within 3 h. Rave et al. (2023) extended the VRP-D by assuming that drones can be launched not only from customer locations, but also from microdepots or stations. A specialised ALNS algorithm was developed to solve instances with up to 200 customers.

Some works have extended the VRP-D with multi-visit, allowing drones to serve multiple customers per flight. For example, Kitjacharoenchai et al. (2020) extended the VRP-D with multi-visit consideration and developed a two-staged algorithm to handle instances containing 70 customers. In a follow-up paper, Gu et al. (2022) proposed a hybrid iterative local search heuristic and variable neighbourhood descent procedure to solve the VRP-D with multiple visits, in which a battery endurance model of drones related to payload and time was employed.

Some researches have addressed station launch and retrieval. Dukkanci et al. (2021) studied the energy-minimising and range-constrained drone delivery problem, in which trucks act only as launch pads to dispatch drones at a potential set of sites to serve all customers. Specifically, a nonlinear model of the problem was initially formulated, and then linearised by second-order cone programming and enhanced by perspective

cuts, allowing it to be exactly solved by optimisation software. Later, Kyriakakis et al. (2022) considered the electric vehicle routing problem with drones, in which each EV dispatches drones at pre-designated locations, where it stops and waits for all its drones to return before proceeding to the next location. Moreover, a payload-based energy consumption model for both types of vehicles was considered, and four algorithms based on the ant colony optimisation framework were utilised to tackle the problem. Kloster et al. (2023) introduced the multiple travelling salesman problem with stations, where the autonomous vehicles (drones or robots) are dispatched and retrieved for multiple single visit round trips. The developed metaheuristic algorithms can solve instances with up to 120 customers and 6 stations.

Very few studies have considered combined pickup and delivery services. Karak & Abdelghany (2019) proposed a hybrid vehicle-drone routing problem for pickup and delivery services, in which a single vehicle launches and retrieves multiple drones at a set of pre-specified stations to serve all customer demands. They extended the Clarke and Wright algorithm to solve instances with up to 100 customers and 24 stations. More recently, Meng et al. (2023) proposed the multi-visit drone routing problem for pickup and delivery, in which multiple trucks launch and retrieve drones at customer nodes to provide simultaneous pickup and delivery services. They proposed a two-stage approach to solve instances consisting of 100 customers. Jiang et al. (2023) studied a similar pickup and delivery problem, but assumed that drones can return to any truck nearby to minimise the travelling cost. The proposed metaheuristic can solve instances with 100 requests.

2.2. Consideration of pair service

To the best of our knowledge, the only study concerning drone-assisted one-to-one pair service is Luo et al. (2022). In their work, trucks launch and retrieve drones at customer nodes they serve, providing one-to-one pickup and delivery services to minimise the total cost. However, they did not consider station selection and use, pickup and delivery only services, or time windows. They used a similar but simpler energy consumption model of drones, which does not need to concern the energy consumption of hovering over customer locations as time windows are not considered. They did consider hovering at retrieval nodes, but approximated the energy consumption as a weighted sum of payload and duration. Such a simplification was not justified but allowed them to formulate the problem as a mixed-integer linear program. An iterated local search algorithm was proposed for larger problem instances.

The pair service has been well studied in the traditional VRP literature. For example, Wolfinger (2021) studied a variant of the one-to-one pickup and delivery problem (PDP) with time windows, split loads and transshipments, where a fleet of capacitated vehicles is deployed to serve each customer, whose location can be visited multiple times to minimise the sum of travel and transshipment costs. The problem was modelled via an arc-based mixed-integer formulation and solved by a large neighbourhood search metaheuristic. Dragomir et al. (2022) considered the PDP with alternative locations and overlapping time windows in the C2C setting, in which each request has multiple roaming pickup and delivery locations, and the pickup and delivery of each request are paired. They proposed a multi-start, adaptive, large neighbourhood search

to minimise the travel time. Yu et al. (2022) investigated a two-echelon van-based robot last-mile pickup and delivery system, in which vans, with a no-go area constraint, dispatch and/or collect robots at parking nodes, providing pickup, delivery or pair service to minimise travel costs. The problem was modelled as a mixed-integer program and solved by an ALNS algorithm.

2.3. Consideration of time windows

The pioneering work considering time windows in the VRP-D was by Di Puglia Pugliese & Guerriero (2017). They addressed a truck-drone delivery problem with time windows, in which a set of trucks are equipped with a given number of drones to make deliveries. Drones can perform exactly one delivery per flight. Kuo et al. (2022) studied the travelling costs of the VRP-D with time windows. Each truck carries a drone with the capability of a single visit. They proposed an improved variable neighbourhood search heuristic to solve instances with up to 50 customers. Meanwhile, Wang et al. (2022) studied a truck–drone combined routing problem with time-dependent road travel time and designed an iterated local search heuristic to minimise the total cost. More recently, Yang et al. (2023) considered the uncertainty in ground traffic networks and assumed that the vehicle pair does not have to serve the customers whose time windows cannot be met anymore. They developed a branch-and-price algorithm to maximise the profit of instances with up to 40 requests.

Some works have extended the VRP-D with time windows to allow multiple visits of drones. In particular, Li et al. (2020) proposed a two-echelon vehicle routing problem with time windows, in which trucks act as mobile satellites. They park at customer locations, waiting for drone departures and returns. Masmoudi et al. (2022) introduced the VRP-D with multi-package payload compartments and time windows, in which drones with the capability of multiple visits can return to any truck to maximise total profit. The authors proposed an adaptive multi-start simulated annealing algorithm to solve instances containing 200 customers. Yin et al. (2023) considered a truck-drone routing problem with time windows, which assumes that drones can make multiple visits. They developed an enhanced branch-and-price-and-cut algorithm to solve instances with 45 customers within 3 hours. It is worth mentioning that all of the previous works have considered time windows only with delivery service.

3. Problem description and formulation

We first present the problem description and all the notation used in the model in Section 3.1 and Section 3.2, respectively. The mathematical model is introduced in detail in Section 3.3, followed by the valid inequalities in Section 3.4.

3.1. Problem description

Consider a fleet of identical truck-drone vehicle pairs. If deployed, each pair would start from and return to a single depot, en route serving a set of customers with either pickup or delivery demand within the specified time windows. Each customer must be served exactly once by either a truck or a drone. Some of the customers are paired with each other, where the pickup customer in each pair must be visited before their delivery counterpart, and they must be served by the same vehicle pair. Each drone can serve multiple customers per flight, as long as its load and battery capacities permit. If a drone arrives before the time window of the customer it serves, it spends additional energy hovering before commencing service.

A number of rendezvous stations work as resupply facilities that allow trucks to stop for drone launch and/or retrieval and battery replacement. Each station can be visited multiple times, but at most once by each vehicle pair (Wang & Sheu, 2019). Once a truck launches its drone, it must move to another station for retrieval. When a drone and its truck reconvene at a station, the one that arrives earlier must wait for the other; if the drone arrives earlier, it consumes extra energy hovering and must be retrieved before its energy depletes. Once a drone is retrieved, its battery is replaced with a fully charged one. The objective is to find the optimal number of vehicle pairs to deploy and the optimal routes/flights for each deployed vehicle pair to serve all customers with the minimal total cost, which includes the fixed cost for deployed vehicle pairs and the operational cost. For trucks, the operational cost is proportional to the travelling distance, while for drones, the cost is proportional to their energy consumption. Considering an actual road network, the travel distances for trucks and drones are measured by the Manhattan metric and Euclidean metric, respectively (Zhen et al., 2023). We call this problem the "multi-visit drone-assisted pickup and delivery problem with time windows" (mDPD-T) henceforth. In Figure 3, we depict a feasible solution of a small running example that involves 15 customers served by 2 truck-drone pairs and 5 rendezvous stations. The detailed parameters of this example can be found in Section 5.1.

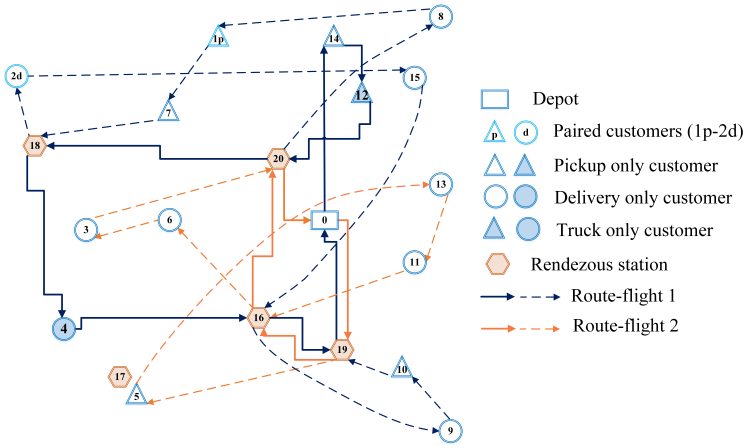


Figure 3: An illustration of the truck-drone combined system of a running example. The filled nodes denote truck-only customers who are not eligible for drone service due to practical requirements such as overweight packages or customer preferences; the non-filled nodes denote drone-eligible customers who can be served by either trucks or drones.

3.2. Notation

The mDPD-T problem alluded to in the previous section can be described by a directed graph $G = (N, A)$, where N is the set of nodes and A is the set of arcs. Node set $N = \{0\} \cup N_c \cup N_s \cup \{n+m+1\} = \{0, 1, .., n+m+1\}$ includes the starting depot (0), the customer set $N_c = \{1, 2, ..., n\}$, the station set $N_s = \{n+1, ..., n+m\}$, and the ending depot $(n+m+1)$. Set N_c contains three types of customers. For paired customers, let the pickup ones be denoted by $N_{cp} = \{1, ..., B\}$ and delivery ones by $N_{cd} = \{B+1, ..., 2B\}$, where B is the number of customer pairs. Note that for each pickup customer $i \in N_{cp}$, its delivery counterpart is indexed by $i + B$. The pickup only customers and delivery only customers are denoted by N_{op} and N_{od}, respectively. Thus, we have $N_c = N_{cp} \cup N_{cd} \cup N_{op} \cup N_{od}$. Let N_t be the set of truck-only customers and N_d be the set of drone-eligible customers. Moreover, let $N_o = N \setminus \{n+m+1\}$ be the set of outbound nodes and $N_e = N \setminus \{0\}$ be the set of inbound nodes; thus, we have the set of all arcs $A = \{(i, j) | i \in N_o, j \in N_e, i \neq j\}$. For trucks, we let $\tilde{N}_t = N_c \cup N_s$ and $A_t = \{(i, j) | i, j \in \tilde{N}_t, i \neq j\}$ be the sets of nodes and arcs that trucks can visit or traverse by themselves, respectively. Similarly, for drones, we let $\tilde{N}_d = N_d \cup N_s$ and $A_d = \{(i, j) | i, j \in \tilde{N}_d, i \neq j\}$ be the sets of nodes and arcs that drones can visit or traverse by themselves, respectively. K denotes the set of available vehicle pairs, each indexed by $k \in K$. For brevity, we list all other parameters in Table 1.

Note that $d_i < 0$ indicates the demand of a pickup customer and $d_i > 0$ the demand of a delivery customer. The time windows of the depot and all the stations are set at $[0, l_{max}]$, where l_{max} is the latest return time for all vehicle pairs back to the depot.

Table 2 lists all the decision variables. Some variables apply to both trucks and drones. For simplicity, we use the same notation but add the *hat* accent to those for drones.

Note that the hovering energy consumption of a drone at a retrieval node (j) is dependent on the last customer it serves (i) before retrieval. This is because the payload while hovering over the retrieval node is given by the departure payload from customer i. The variable $S^k_{i,j}$ is used to avoid infeasible routes and flights for serving customer pairs; a more detailed explanation can be found in Section 3.3.2.

3.3. The model

In what follows, we formulate the problem into an MIBP, in which the two constraints on the hovering energy consumption of drones are bilinear, while all the other constraints and the objective function are linear. The objective function is presented in Section 3.3.1. The routing constraints and timing constraints are introduced in Section 3.3.2 and Section 3.3.3, respectively, followed by the energy consumption constraints of drones in Section 3.3.4. Finally, the demand flow constraints of drones and trucks are introduced in Section 3.3.5 and Section 3.3.6, respectively.

Table 1: Sets and parameters.

Notation	Description
Sets	
N_{cp}	Set of pickup pair customers, $N_{cp} = \{1, ..., B\}$
N_{cd}	Set of delivery pair customers, $N_{cd} = \{B+1, ..., 2B\}$
N_{op}	Set of pickup only customers
N_{od}	Set of delivery only customers
N_c	Set of all customers, $N_c = N_{cp} \cup N_{cd} \cup N_{op} \cup N_{od} = \{1, 2, ..., n\}$
N_s	Set of stations for drone launch/retrieval, $N_s = \{n+1, ..., n+m\}$
N	Set of all nodes, $N = \{0\} \cup N_c \cup N_s \cup \{n+m+1\} = \{0, 1, ..., n+m+1\}$
N_o	Set of outbound nodes for arc starting, $N_o = N \setminus \{n+m+1\}$
N_e	Set of inbound nodes for arc ending, $N_e = N \setminus \{0\}$
N_t	Set of truck-only customers
N_d	Set of drone-eligible customers
$\tilde{N}_t$	Set of nodes (stations and customers) that trucks can visit by themselves, $\tilde{N}_t = N_c \cup N_s$
$\tilde{N}_d$	Set of nodes (stations and customers) that drones can visit by themselves, $\tilde{N}_d = N_d \cup N_s$
A	Set of all arcs, where $A = \{(i,j) \mid i \in N_o, j \in N_e, i \neq j\}$
A_t	Set of arcs that trucks can traverse by themselves, $A_t = \{(i,j) \mid i, j \in \tilde{N}_t, i \neq j\}$
A_d	Set of arcs that drones can traverse by themselves, $A_d = \{(i,j) \mid i, j \in \tilde{N}_d, i \neq j\}$
K	Set of available vehicle pairs; each pair is indexed by $k \in K$
Parameters	
w_0	Curb weight of each drone, i.e., drone's no-load weight, including drone mass and drone battery mass (unit: kg)
η	Energy consumption of drones per kilogram per minute (unit: Wh/(kg·min))
E	Maximum battery capacity of each drone (unit: Wh)
Q^t	Maximum available load capacity of each truck, i.e., total load capacity minus drone supporting material weight (unit: kg)
Q^d	Maximum load capacity of each drone (unit: kg)
v^t	Constant speed of each truck (unit: km/h)
v^d	Constant speed of each drone (unit: km/h)
c^0	Fixed cost of deploying a vehicle pair (unit: $)
c^t	Cost per kilometre for each truck (unit: $/km)
c^d	Cost per watt hour for each drone (unit: $/Wh)
r_{ij}^t	Manhattan distance of arc (i,j) (unit: km)
r_{ij}^d	Euclidean distance of arc (i,j) (unit: km)
d_i	Demand of customer i (unit: kg)
$[u_i, l_i]$	Time window at node i (unit: min)
s_i	Required service time at node i (unit: min)
S_l	Setup time required for drone launch (including parcel loading time) (unit: min)
S_r	Setup time required for drone retrieval (including parcel unloading and battery swapping times) (unit: min)
M	A sufficiently large integer

3.3.1. Objective function

$$min \quad c^t \sum_{k \in K} \left(\sum_{(i,j) \in A} x_{ij}^k + \sum_{(i,j) \in A_t} y_{ij}^k \right) r_{ij}^t +$$

$$c^d \sum_{k \in K} \left(\sum_{(i,j) \in A_d} e_{ij}^{f,k} + \sum_{j \in N_d} (e_j^{s,k} + e_j^{h,k}) + \sum_{i \in N_d} \sum_{j \in N_s} e_{ij}^{h,k} \right) + c^0 \sum_{k \in K} \sum_{j \in \tilde{N}_t} x_{0j}^k \tag{1}$$

The objective function (1) minimises the total costs. The first part is the travelling cost of trucks. The second part is the total energy cost of drones, which includes the flying energy cost, the service and hovering energy cost at the served customers, and the hovering energy cost of drones at retrieval stations. The last term is the fixed cost of deployed vehicle pairs.

Table 2: Decision variables.

Notation	Description
$x_{ij}^k \in \{0,1\}$	Equal to 1 if truck k traverses arc $(i,j) \in A$ with its drone on board and 0 otherwise
$y_{ij}^k \in \{0,1\}$	Equal to 1 if truck k traverses arc $(i,j) \in A_t$ independently and 0 otherwise
$\hat{y}_{ij}^k \in \{0,1\}$	Equal to 1 if drone k traverses arc $(i,j) \in A_d$ independently and 0 otherwise
$w_i^k \in [0, Q^t]$	Payload of truck k leaving from node i
$\hat{w}_i^k \in [0, Q^d]$	Payload of drone k leaving from node i
$\hat{w}_i^{d,k} \in [0, Q^d]$	Delivery weight of drone k leaving from node i
$e_i^k \in [0, E]$	Energy having consumed by drone k upon leaving from node i in a flight
$e_{ij}^{f,k} \in [0, E]$	Energy consumption of drone k flying over arc (i,j)
$e_i^{s,k} \in [0, E]$	Energy consumption of drone k serving customer node i
$e_i^{h,k} \in [0, E]$	Energy consumption of drone k hovering at customer node i
$e_{ij}^{h,k} \in [0, E]$	Energy consumption of drone k hovering at retrieval node j, directly from customer i
$h_i^k \in [0, E/(\eta \cdot w_0)]$	Hovering duration of drone k at node i, which can be either a station or a customer
$\tau_i^k \in [0, l_{max}]$	Arrival time of truck k at node i
$\hat{\tau}_i^k \in [0, l_{max}]$	Arrival time of drone k at node i
$\omega_i \in [u_i, l_i]$	The start time of service at customer i, either by a truck or a drone
$\omega_i^k = \{0, \omega_i\}$	The start time of service at customer i by either the truck or drone in vehicle pair k. It equals ω_i if i is served by k and 0 otherwise
$v_i^k \in [0, l_{max}]$	Departure time of vehicle pair k at station i or depot 0
$S_{i,j}^k \in \{0,1\}$	Equal to 1 if station j is visited by vehicle pair k in-between the service of the pickup pair customer i and its delivery counterpart and 0 otherwise
$V_i^k \in \{0,1\}$	Equal to 1 if a pickup pair customer i and its delivery counterpart are both served in the same flight by drone k and 0 otherwise

3.3.2. Routing constraints

$$\sum_{i \in N_o} x_{ij}^k = \sum_{i \in N_e} x_{ji}^k, j \in N_c, k \in K \tag{2a}$$

$$\sum_{i \in \tilde{N}_t} y_{ij}^k = \sum_{i \in \tilde{N}_t} y_{ji}^k, j \in N_c, k \in K \tag{2b}$$

$$\sum_{i \in \tilde{N}_d} \hat{y}_{ij}^k = \sum_{i \in \tilde{N}_d} \hat{y}_{ji}^k, j \in N_d, k \in K \tag{2c}$$

$$\sum_{i \in N_o} x_{ij}^k + \sum_{i \in \tilde{N}_t} y_{ij}^k = \sum_{i \in N_e} x_{ji}^k + \sum_{i \in \tilde{N}_t} y_{ji}^k \leq 1, j \in N_s, k \in K \tag{3a}$$

$$\sum_{i \in N_o} x_{ij}^k + \sum_{i \in N_d} \hat{y}_{ij}^k = \sum_{i \in N_e} x_{ji}^k + \sum_{i \in N_d} \hat{y}_{ji}^k \leq 1, j \in N_s, k \in K \tag{3b}$$

$$\sum_{i \in N_o} x_{ij}^k + \sum_{i \in \tilde{N}_t} y_{ij}^k + \sum_{i \in \tilde{N}_d} \hat{y}_{ij}^k = \sum_{i \in N_o} x_{i,j+B}^k + \sum_{i \in \tilde{N}_t} y_{i,j+B}^k + \sum_{i \in \tilde{N}_d} \hat{y}_{i,j+B}^k, j \in N_{cp}, k \in K \tag{4}$$

$$\sum_{j \in N_e} x_{0j}^k = \sum_{j \in N_o} x_{j,n+m+1}^k = 1, k \in K \tag{5}$$

$$\sum_{k \in K} \sum_{i \in N_o} x_{ij}^k + \sum_{k \in K} \sum_{i \in \tilde{N}_t} y_{ij}^k + \sum_{k \in K} \sum_{i \in \tilde{N}_d} \hat{y}_{ij}^k = 1, j \in N_c \tag{6a}$$

$$\sum_{k \in K} \sum_{i \in N_o} x_{ij}^k + \sum_{k \in K} \sum_{i \in \tilde{N}_t} y_{ij}^k = 1, j \in N_t \tag{6b}$$

$$\sum_{j \in N_d} \hat{y}_{ij}^k \leq \sum_{j \in \tilde{N}_t} y_{ij}^k, \sum_{j \in N_d} \hat{y}_{ji}^k \leq \sum_{j \in \tilde{N}_t} y_{ji}^k, i \in N_s, k \in K \tag{7}$$

$$\sum_{j \in N_o} x_{ji}^k + \sum_{j \in \tilde{N}_t} y_{ji}^k \leq \sum_{j \in N_d} \hat{y}_{ij}^k + \sum_{j \in N_d} \hat{y}_{ji}^k, i \in N_s, k \in K \tag{8}$$

$$\sum_{j\in N_d} \hat{y}_{ij}^k \le \sum_{j\in N_o} x_{ji}^k + \sum_{j\in N_d} \hat{y}_{ji}^k, i \in N_s, k \in K \tag{9}$$

$$S_{i,j}^k \ge P_{i,j}^k - P_{i+B,j}^k, i \in N_{cp} \cap N_d, j \in N_s, k \in K \tag{10a}$$

$$S_{i,j}^k \le P_{i,j}^k, S_{i,j}^k \le 1 - P_{i+B,j}^k, i \in N_{cp} \cap N_d, j \in N_s, k \in K \tag{10b}$$

$$\tau_j^k \ge \omega_i^k - M(1 - P_{i,j}^k), \omega_i^k \ge \tau_j^k - M \cdot P_{i,j}^k, i \in N_c, j \in N_s, k \in K \tag{10c}$$

$$\sum_{j\in N_s} S_{i,j}^k \ge \sum_{j\in \tilde{N}_d} \hat{y}_{ji}^k + \sum_{j\in \tilde{N}_t} y_{j,i+B}^k - 1, i \in N_{cp} \cap N_d, k \in K \tag{11a}$$

$$\sum_{j\in N_s} S_{i,j}^k \ge \sum_{j\in \tilde{N}_t} y_{ji}^k + \sum_{j\in \tilde{N}_d} \hat{y}_{j,i+B}^k - 1, i \in N_{cp} \cap N_d, k \in K \tag{11b}$$

Constraints (2a)-(2c) ensure vehicle flow conservation at each customer node. Similarly, the flow conservation at each station is imposed by (3a) and (3b). Constraint (4) ensures that each customer pair is served by the same vehicle pair. Constraint (5) ensures that each vehicle pair departs from and returns to the depot once. Constraint (6a) restricts that each customer is served exactly once by either a truck or a drone. Constraint (6b) ensures the truck-only customers are indeed served only by trucks. Moreover, Constraint (7) ensures that if a drone is launched or retrieved at a station, its truck must visit the same station. Similarly, Constraint (8) guarantees that a truck visits a station only if the station is assigned to launch/retrieve its drone. Constraint (9) requires that a drone can only be launched if it is already on its truck or has been retrieved.

Moreover, Constraints (10a) and (10b) ensure that if vehicle pair k visits station j in-between the service of a customer pair $(i, i+B)$, the variable $S_{i,j}^k = 1$, where auxiliary variable $P_{i,j}^k \in \{0,1\}$ is defined in (10c), which is 1 if station j is visited by vehicle pair k after customer i and 0 otherwise. Constraints (11a) and (11b) ensure that if one of a customer pair $(i, i+B)$ is served by truck (drone) k while the other is served by its drone (truck), at least one station must be visited between them, that is, a customer pair cannot be parallelly served by two sub-routes (as described in Figure 2).

3.3.3. Timing constraints

$$\omega_i + s_i \le \omega_{i+B}, i \in N_{cp} \tag{12}$$

$$u_i \le \omega_i \le l_i, i \in N_c \tag{13a}$$

$$0 \le \omega_i^k \le M(\sum_{j\in N_o} x_{ji}^k + \sum_{j\in \tilde{N}_t} y_{ji}^k + \sum_{j\in \tilde{N}_d} \hat{y}_{ji}^k), i \in N_c, k \in K \tag{13b}$$

$$\begin{aligned}&\omega_i - M(1 - \sum_{j\in N_o} x_{ji}^k - \sum_{j\in \tilde{N}_t} y_{ji}^k - \sum_{j\in \tilde{N}_d} \hat{y}_{ji}^k)\\ &\le \omega_i^k \le \omega_i + M(1 - \sum_{j\in N_o} x_{ji}^k - \sum_{j\in \tilde{N}_t} y_{ji}^k - \sum_{j\in \tilde{N}_d} \hat{y}_{ji}^k), i \in N_c, k \in K\end{aligned} \tag{13c}$$

$$\hat{\tau}_i^k - M(1 - \sum_{j\in \tilde{N}_d} \hat{y}_{ji}^k) \le \omega_i^k, i \in N_d, k \in K \tag{14a}$$

$$\tau_i^k - M(1 - \sum_{j \in N_o} x_{ji}^k - \sum_{j \in \tilde{N}_t} y_{ji}^k) \leq \omega_i^k, i \in N_c, k \in K \tag{14b}$$

$$\upsilon_0^k + \frac{r_{0i}^t}{v^t} - M(1 - x_{0i}^k) \leq \tau_i^k \leq \upsilon_0^k + \frac{r_{0i}^t}{v^t} + M(1 - x_{0i}^k), i \in N_e, k \in K \tag{15}$$

$$\omega_i^k + s_i + \frac{r_{ij}^d}{v^d} - M(1 - \hat{y}_{ij}^k) \leq \hat{\tau}_j^k \leq \omega_i^k + s_i + \frac{r_{ij}^d}{v^d} + M(1 - \hat{y}_{ij}^k), i \in N_d, j \in \tilde{N}_d, k \in K \tag{16a}$$

$$\omega_i^k + s_i + \frac{r_{ij}^t}{v^t} - M(1 - x_{ij}^k - y_{ij}^k) \leq \tau_j^k, i \in N_c, j \in N_e, k \in K \tag{16b}$$

$$\upsilon_i^k \geq \tau_i^k + S_l \sum_{p \in N_d} \hat{y}_{ip}^k + S_r \sum_{p \in N_d} \hat{y}_{pi}^k - M(1 - \sum_{j \in N_o} x_{ji}^k - \sum_{j \in \tilde{N}_t} y_{ji}^k), i \in N_s, k \in K \tag{17a}$$

$$\upsilon_i^k \geq \hat{\tau}_i^k + S_l \sum_{p \in N_d} \hat{y}_{ip}^k + S_r - M(2 - \sum_{j \in \tilde{N}_t} y_{ji}^k - \sum_{p \in N_d} \hat{y}_{pi}^k), i \in N_s, k \in K \tag{17b}$$

$$\upsilon_i^k + \frac{r_{ij}^d}{v^d} - M(1 - \hat{y}_{ij}^k) \leq \hat{\tau}_j^k \leq \upsilon_i^k + \frac{r_{ij}^d}{v^d} + M(1 - \hat{y}_{ij}^k), i \in N_s, j \in N_d, k \in K \tag{18a}$$

$$\upsilon_i^k + \frac{r_{ij}^t}{v^t} - M(1 - x_{ij}^k - y_{ij}^k) \leq \tau_j^k, i \in N_s, j \in N_e, k \in K \tag{18b}$$

$$\tau_{n+m+1}^k \leq l_{max} \sum_{i \in N_o} x_{i,n+m+1}^k, k \in K \tag{19}$$

Constraint (12) ensures that each pickup pair customer is served before its delivery counterpart. Constraint (13a) ensures that services must start within time windows. Constraints (13b) and (13c) enforce that if vehicle pair k visits customer i, we have $\omega_i^k = \omega_i$, and 0 otherwise. Constraints (14a) and (14b) ensure that the service does not start until the vehicle arrives. Constraint (15) tracks the arrival time at each node from the depot. If a truck or drone traverses arc (i, j), the arrival time at j is calculated by Constraints (16a) or (16b) if i is a customer node. It is more involved to derive the arrival times if i is a station, as shown by the following constraints. Constraint (17a) ensures that the departure time of vehicle pair k at station i is not earlier than the arrival time of truck k at node i plus the possible launch time (if i is a launch node) and retrieval time (if i is a retrieval node). Moreover, if node i is used for retrieval, Constraint (17b) ensures that the departure time of vehicle pair k is not earlier than the arrival time of drone k at the station plus the retrieval time and possible launch time (if i is also used for launch). Constraints (18a) and (18b) track the arrival time of drones and trucks at each node from a station, respectively. Lastly, Constraint (19) guarantees that all vehicle pairs must return to the depot before l_{max}.

3.3.4. Energy consumption constraints of drones

Given that the energy consumption of drones is sensitive to flight operations, it is essential to develop an appropriate energy consumption estimate for feasible and efficient routing. In view of the important role of payload and duration in the energy consumption of drones (Torabbeigi et al., 2020), we provide the following drone energy consumption model for energy calculation.

- The energy required for drone k to traverse arc (i, j) is calculated as $\eta \cdot (w_0 + \hat{w}_i^k) \cdot r_{ij}^d / v^d$, assuming that drones fly at a constant speed between two locations (Zhen et al., 2023). Some works have considered the energy consumption with varying drone speed; see e.g., Dukkanci et al. (2021).

- The energy required for drone k to serve customer i is calculated as $\eta \cdot (w_0 + \hat{w}_i^k + d_i) \cdot s_i$.
- The energy required for hovering at customer i is calculated as $\eta \cdot (w_0 + \hat{w}_i^k + d_i) \cdot h_i^k$.
- The energy required for hovering at retrieval station i is calculated as $\eta \cdot (w_0 + \hat{w}_{i'}^k) \cdot h_i^k$, where i' is the last customer drone k visits before returning to station i.

The energy consumption constraints of drones are introduced as follows.

$$0 \leq e_{ij}^{f,k} \leq E \cdot \hat{y}_{ij}^k, (i,j) \in A_d, k \in K \tag{20a}$$

$$\eta(w_0 + \hat{w}_i^k)\frac{r_{ij}^d}{v^d} - E(1 - \hat{y}_{ij}^k) \leq e_{ij}^{f,k} \leq \eta(w_0 + \hat{w}_i^k)\frac{r_{ij}^d}{v^d}, (i,j) \in A_d, k \in K \tag{20b}$$

$$0 \leq h_i^k \leq M \cdot H_i^k, i \in N_d, k \in K \tag{21a}$$

$$\omega_i^k - \hat{\tau}_i^k - M(1 - H_i^k) \leq h_i^k \leq \omega_i^k - \hat{\tau}_i^k, i \in N_d, k \in K \tag{21b}$$

$$\omega_i^k \geq \hat{\tau}_i^k - M(1 - H_i^k), \omega_i^k \leq \hat{\tau}_i^k + M \cdot H_i^k, i \in N_d, k \in K \tag{21c}$$

$$0 \leq e_i^{s,k} + e_i^{h,k} \leq E \cdot \sum_{j \in \tilde{N}_d} \hat{y}_{ji}^k, i \in N_d, k \in K \tag{22a}$$

$$\eta(w_0 + \hat{w}_i^k + d_i)(s_i + h_i^k) - E(1 - \sum_{j \in \tilde{N}_d} \hat{y}_{ji}^k) \leq e_i^{s,k} + e_i^{h,k} \leq \eta(w_0 + \hat{w}_i^k + d_i)(s_i + h_i^k), i \in N_d, k \in K \tag{22b}$$

$$0 \leq h_i^k \leq M \cdot \tilde{H}_i^k, i \in N_s, k \in K \tag{23a}$$

$$\tau_i^k - \hat{\tau}_i^k - M(1 - \tilde{H}_i^k) \leq h_i^k \leq \tau_i^k - \hat{\tau}_i^k, i \in N_s, k \in K \tag{23b}$$

$$\tau_i^k \geq \hat{\tau}_i^k - M(1 - \tilde{H}_i^k), \tau_i^k \leq \hat{\tau}_i^k + M \cdot \tilde{H}_i^k, i \in N_s, k \in K \tag{23c}$$

$$0 \leq e_{ij}^{h,k} \leq E \cdot \hat{y}_{ij}^k, i \in N_d, j \in N_s, k \in K \tag{24a}$$

$$\eta(w_0 + \hat{w}_i^k) \cdot h_j^k - E(1 - \hat{y}_{ij}^k) \leq e_{ij}^{h,k} \leq \eta(w_0 + \hat{w}_i^k) \cdot h_j^k, i \in N_d, j \in N_s, k \in K \tag{24b}$$

$$0 \leq e_i^k \leq M(1 - \sum_{j \in N_d} \hat{y}_{ij}^k), i \in N_s, k \in K \tag{25a}$$

$$e_i^k + e_{ij}^{f,k} + e_j^{s,k} + e_j^{h,k} - M(1 - \hat{y}_{ij}^k) \leq e_j^k, i \in \tilde{N}_d, j \in N_d, k \in K \tag{25b}$$

$$e_i^k + e_{ij}^{f,k} + e_{ij}^{h,k} - M(1 - \hat{y}_{ij}^k) \leq E, i \in N_d, j \in N_s, k \in K \tag{25c}$$

Constraints (20a) and (20b) define the energy consumption of drone k flying over arc (i,j). Constraints (21a)-(21c) linearise the hovering time $h_i^k = max\{0, \omega_i^k - \hat{\tau}_i^k\}$ of drone k at each customer it serves, where H_i^k is an auxiliary binary variable and takes 1 if drone k arrives at customer i before the time window opens, and 0 otherwise. Constraints (22a) and (22b) define the energy consumption of drone k at a customer node. Similarly, Constraints (23a)-(23c) linearise the hovering time $h_i^k = max\{0, \tau_i^k - \hat{\tau}_i^k\}$ of drone k at station i, where (23c) defines that if truck k arrives at station i later than its drone, auxiliary binary variable $\tilde{H}_i^k = 1$, and 0 otherwise. Constraints (24a) and (24b) define the energy consumption of drone k hovering at a retrieval station. Last, Constraint (25a) ensures that the battery is full when the drone is launched. Constraint (25b) tracks the energy consumption of drone k after serving a customer in a flight, and Constraint (25c) guarantees that the drone's battery does not run out when it is retrieved. Note that Constraints (22b) and (24b) are bilinear.

3.3.5. Demand flow constraints of drones

$$V_i^k \geq 1 - M(2 - \sum_{j \in \tilde{N}_d} \hat{y}_{ji}^k - \sum_{j \in N_d} \hat{y}_{j,i+B}^k + \sum_{j \in N_s} S_{i,j}^k), i \in N_{cp} \cap N_d, k \in K \tag{26}$$

$$\hat{w}_i^{d,k} - M(1 - \sum_{j \in N_d} \hat{y}_{ij}^k) \leq \hat{w}_i^k \leq \hat{w}_i^{d,k} + M(1 - \sum_{j \in N_d} \hat{y}_{ij}^k), i \in N_s, k \in K \tag{27a}$$

$$\hat{w}_i^{d,k} \leq Q^d(1 - \sum_{j \in N_s} \hat{y}_{ij}^k), i \in N_d, k \in K \tag{27b}$$

$$\hat{w}_i^k - d_j - M(1 - \hat{y}_{ij}^k) \leq \hat{w}_j^k \leq \hat{w}_i^k - d_j + M(1 - \hat{y}_{ij}^k), i \in \tilde{N}_d, j \in N_d, k \in K \tag{28a}$$

$$\hat{w}_i^{d,k} - d_j - M(1 - \hat{y}_{ij}^k) \leq \hat{w}_j^{d,k} \leq \hat{w}_i^{d,k} - d_j + M(1 - \hat{y}_{ij}^k), i \in \tilde{N}_d, j \in N_{od} \cap N_d, k \in K \tag{28b}$$

$$\hat{w}_i^{d,k} - M(2 - V_{j-B}^k - \hat{y}_{ij}^k) \leq \hat{w}_j^{d,k} \leq \hat{w}_i^{d,k} + M(2 - V_{j-B}^k - \hat{y}_{ij}^k), i \in N_d, j \in N_{cd} \cap N_d, k \in K \tag{28c}$$

$$\hat{w}_i^{d,k} - d_j - M(1 + V_{j-B}^k - \hat{y}_{ij}^k) \leq \hat{w}_j^{d,k} \leq \hat{w}_i^{d,k} - d_j + M(1 + V_{j-B}^k - \hat{y}_{ij}^k), i \in \tilde{N}_d, j \in N_{cd} \cap N_d, k \in K \tag{28d}$$

$$\hat{w}_i^{d,k} - M(1 - \hat{y}_{ij}^k) \leq \hat{w}_j^{d,k} \leq \hat{w}_i^{d,k} + M(1 - \hat{y}_{ij}^k), i \in \tilde{N}_d, j \in N_{op} \cup N_{cp} \cap N_d, k \in K \tag{28e}$$

Constraint (26) ensures that $V_i^k = 1$ only if the customer pair $(i, i+B)$ are both served by drone k in a single flight. Constraints (27a) and (27b) ensure that for each drone, the pickup weight from a launch node and the delivery weight to a retrieval node are both zero. Constraint (28a) tracks the payload of drone k after serving each customer. Constraint (28b) tracks the delivery weight of drone k after serving each delivery only customer. The next two constraints track the delivery weight of drone k after serving each delivery pair customer, if its pickup counterpart is served in the same flight (28c) or not (28d). Constraint (28e) tracks the delivery weight of drone k after serving each pickup customer.

3.3.6. Demand flow constraints of trucks

$$\sum_{i \in N_o} \sum_{j \in N_{od}} x_{ij}^k d_j + \sum_{i \in \tilde{N}_t} \sum_{j \in N_{od}} y_{ij}^k d_j + \sum_{i \in \tilde{N}_d} \sum_{j \in N_{od} \cap N_d} \hat{y}_{ij}^k d_j + w_0 \leq w_0^k \leq Q^t, k \in K \tag{29a}$$

$$w_i^k - d_j - M(1 - x_{ij}^k - y_{ij}^k) \leq w_j^k \leq w_i^k - d_j + M(1 - x_{ij}^k - y_{ij}^k), i \in N_o, j \in N_c, k \in K \tag{29b}$$

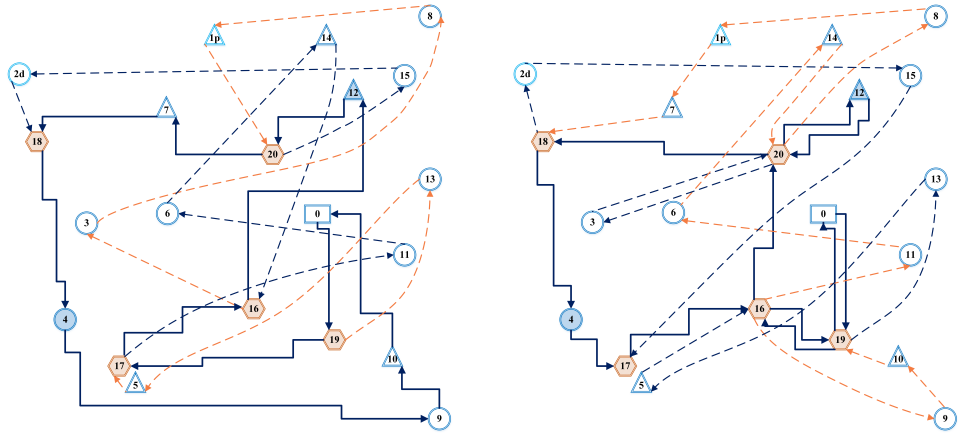
$$\begin{aligned} &w_i^k - w_0 - \hat{w}_j^k - M(2 - x_{ij}^k - \sum_{i \in N_d} \hat{y}_{ji}^k + \sum_{i \in N_d} \hat{y}_{ij}^k) \leq w_j^k \\ &\leq w_i^k - w_0 - \hat{w}_j^k + M(2 - x_{ij}^k - \sum_{i \in N_d} \hat{y}_{ji}^k + \sum_{i \in N_d} \hat{y}_{ij}^k), i \in N_o, j \in N_s, k \in K \end{aligned} \tag{29c}$$

$$\begin{aligned} &w_i^k + w_0 + \hat{w}_{i'}^k - M(2 - y_{ij}^k - \hat{y}_{i'j}^k + \sum_{i \in N_d} \hat{y}_{ji}^k) \leq w_j^k \\ &\leq w_i^k + w_0 + \hat{w}_{i'}^k + M(2 - y_{ij}^k - \hat{y}_{i'j}^k + \sum_{i \in N_d} \hat{y}_{ji}^k), i \in N_o, i' \in N_d, j \in N_s, k \in K \end{aligned} \tag{29d}$$

$$\begin{aligned} &w_i^k + \hat{w}_{i'}^k - \hat{w}_j^k - M(3 - y_{ij}^k - \hat{y}_{i'j}^k - \sum_{i \in N_d} \hat{y}_{ji}^k) \leq w_j^k \\ &\leq w_i^k + \hat{w}_{i'}^k - \hat{w}_j^k + M(3 - y_{ij}^k - \hat{y}_{i'j}^k - \sum_{i \in N_d} \hat{y}_{ji}^k), i \in N_o, i' \in N_d, j \in N_s, k \in K \end{aligned} \tag{29e}$$

Constraint (29a) ensures that the total load carried by each truck from the depot does not exceed its load capacity. Constraint (29b) ensures the demand flow balance of each truck at customer nodes, while (29c)-(29e) ensure the demand flow balance at stations where launch-only, retrieval-only, or both operations are undertaken, respectively.

Remark 1 (Multiple Uses of Stations by Each Vehicle Pair). *The assumption that a station can be used by each vehicle pair at most once can be easily relaxed by replicating the stations into multiple virtual copies (see also Yu et al., 2022). Each station and their virtual copies can still be used at most once but collectively they would allow multiple uses of the same station. Our model and algorithm can then be readily applied to the augmented network. For the running example, we solved it (by the heuristic introduced in Section 4) in two scenarios, where each vehicle pair can use the same station at most once ("single-use") or twice ("double-use"). The best solutions are shown in Figure 4. Overall, one vehicle pair is deployed in both scenarios. In the "single-use" case, as shown in the left figure, the total cost is \$45.25, the truck travels 38.00 km, and drone serves 10 customers in 4 flights. In the "double-use" case, the total cost decreases to \$37.73, the distance travelled by truck reduces to 27.80 km, and all drone-eligible customers (13) are served by drone in 6 flights. Three stations (16,19 and 20) are used twice. The cost reduction is at the expense of longer computational time, which increases quickly from 189.34 s to 835.25 s but is still reasonable. However, our analysis shows that allowing three or more uses of the stations no longer leads to meaningful cost reduction in most cases considered, while the computational time increases significantly and quickly becomes unmanageable. In general cases it could be worth visiting some stations more than twice, but their marginal benefits will gradually diminish.*



(a) Best solution in the "single-use" case

(b) Best solution in the "double-use" case

Figure 4. The best solutions in the"single-use" and "double-use" cases for the running example. For clarity, we use two coloured dotted lines to denote different flights.

3.4. Valid inequalities

In this section, we exploit the problem structure and add additional constraints to tighten the model and accelerate the solution procedure.

Drone flights between unpaired customers

The following proposition restricts the drone flights between any two unpaired customers based on their demand quantities and the capacity of drones.

Proposition 1. *For any two unpaired drone-eligible nodes i and j, if $max\{0,-d_i,d_j\}-2min\{0,-d_i,d_j\}+min\{-d_i,d_j\}>Q^d$, then $\hat{y}_{ij}^k=0$, i.e., no drones can serve customers i and j in succession.*

The linearisation of Proposition 1 is given by the following set of valid inequalities, where D_{ij}, $\overline{D}_{ij}$ and $\widetilde{D}_{ij}$ are auxiliary binary variables. The proof and further details can be found in the Supplementary material G.

$$Q^d \geq d_i + d_j - M(1-D_{ij}), i,j \in N_{od} \cap N_d \tag{30a}$$

$$Q^d \leq d_i + d_j + M \cdot D_{ij}, d_i + d_j \neq Q^d, i,j \in N_{od} \cap N_d \tag{30b}$$

$$\hat{y}_{ij}^k \leq 2 + D_{ij} - z_i^k - z_j^k, i,j \in N_{od} \cap N_d, k \in K \tag{30c}$$

$$Q^d \geq -d_i + d_j - M(1-\overline{D}_{ij}), i \in N_{op} \cap N_d, j \in N_{od} \cap N_d \tag{30d}$$

$$Q^d \leq -d_i + d_j + M \cdot \overline{D}_{ij}, -d_i + d_j \neq Q^d, i \in N_{op} \cap N_d, j \in N_{od} \cap N_d \tag{30e}$$

$$\hat{y}_{ij}^k \leq 2 + \overline{D}_{ij} - z_i^k - z_j^k, i \in N_{op} \cap N_d, j \in N_{od} \cap N_d, k \in K \tag{30f}$$

$$Q^d \geq -d_i - d_j - M(1-\widetilde{D}_{ij}), i,j \in N_{op} \cap N_d \tag{30g}$$

$$Q^d \leq -d_i - d_j + M \cdot \widetilde{D}_{ij}, -d_i - d_j \neq Q^d, i,j \in N_{op} \cap N_d \tag{30h}$$

$$\hat{y}_{ij}^k \leq 2 + \widetilde{D}_{ij} - z_i^k - z_j^k, i,j \in N_{op} \cap N_d, k \in K \tag{30i}$$

These valid constraints on the load capacity of drones play an important role in the efficiency of the solution process because the load capacity of drones is relatively small compared to that of trucks, providing tighter constraints for the whole model.

Visiting sequence for customer pairs

The following cuts tighten the relationship between variables $S_{i,j}^k$ and V_i^k for customer pairs.

$$\sum_{j \in N_s} S_{i,j}^k \leq 1 - V_i^k, i \in N_{cp} \cap N_d, k \in K \tag{31a}$$

$$V_i^k \leq 1 - min\{1, \sum_{j \in N_s} S_{i,j}^k\}, i \in N_{cp} \cap N_d, k \in K \tag{31b}$$

Cut (31a) imposes that if a customer pair are served by the same flight, there should be no station visited between them. Cut (31b) imposes that if a station is visited between a customer pair, they cannot be served by the same flight. Cut (31b) can be linearised as

$$V_i^k \leq 1 - \tilde{S}_i^k, i \in N_{cp} \cap N_d, k \in K \tag{32a}$$

$$\sum_{j \in N_s} S_{i,j}^k \geq 1 - M(1-\tilde{s}_i^k), i \in N_{cp} \cap N_d, k \in K \tag{32b}$$

$$\sum_{j \in N_s} S_{i,j}^k \leq 1 + M \cdot \tilde{s}_i^k, i \in N_{cp} \cap N_d, k \in K \tag{32c}$$

$$1 - M(1 - \tilde{s}_i^k) \leq \tilde{S}_i^k \leq 1, i \in N_{cp} \cap N_d, k \in K \tag{32d}$$

$$\sum_{j \in N_s} S_{i,j}^k - M \cdot \tilde{s}_i^k \leq \tilde{S}_i^k \leq \sum_{j \in N_s} S_{i,j}^k, i \in N_{cp} \cap N_d, k \in K \tag{32e}$$

where auxiliary variable $\tilde{s}_i^k \in \{0, 1\}$ is defined in (32b) and (32c), which takes value of 1 if one or more stations are visited by vehicle pair k in-between the service of the pickup pair customer i and its delivery counterpart, i.e., $\sum_{j \in N_s} S_{i,j}^k \geq 1$, and 0 otherwise. (32$d$) further ensures that if $\tilde{s}_i^k = 1$, the auxiliary binary variable $\tilde{S}_i^k = 1$. Otherwise, as stated in Constraint (32e), if $\tilde{s}_i^k = 0$, $\tilde{S}_i^k = \sum_{j \in N_s} S_{i,j}^k \leq 1$.

Operations at the stations

$$\sum_{j \in N_e} x_{ij}^k \leq \sum_{j \in N_d} \hat{y}_{ji}^k, i \in N_s, k \in K \tag{33a}$$

$$\sum_{i \in N_o} x_{ij}^k \leq \sum_{i \in N_d} \hat{y}_{ji}^k, j \in N_s, k \in K \tag{33b}$$

$$\sum_{i \in N_o} x_{ij}^k + \sum_{i \in N_e} x_{ji}^k \leq 1, j \in N_s, k \in K \tag{33c}$$

Cut (33a) imposes that if truck k leaves station i with its drone on board, the station must be a retrieval node. Cut (33b) imposes that if truck k arrives at station i with its drone on board, the station must be a launch node. Cut (33c) imposes that the inbound and outbound arcs of a station cannot both be travelled by truck k with its drone on board.

4. Adaptive large neighbourhood search

In this section, we propose an adaptive large neighbourhood search (ALNS) for mDPD-T. The ALNS metaheuristic has been used in many VRP variants (Sacramento et al., 2019; Yu et al., 2022) and is adaptable to new problems. Specifically, the initial solution is progressively improved by repeatedly applying a set of destroy (removing some nodes from the current solution) and repair operators (relocating all removed nodes to the destroyed solution) (Ropke & Pisinger, 2006). Further details are presented in Supplementary material B.

4.1. Initial solution construction

Considering the features of the mDPD-T, a greedy heuristic method is designed to obtain a feasible initial solution. The construction process starts by building truck-only routes to serve all pickup only and delivery only customers, followed by an insertion operation to assign all customer pairs to the built routes. Then, a flight construction procedure is applied to reassign some customers to drones. Note that all procedures are performed only if the solution is feasible. The pseudocode for the initial solution construction is presented in Supplementary material C.

Let $\vartheta_i^{d,k}$ be the total delivery only demand truck k has served when leaving customer i, $-d_i$ be the load increase after serving customer i and ϑ_i^k be the sum of all load increases after truck k visits customer

i. Therefore, if $\vartheta_i^{d,k} + max\{0, \vartheta_i^k\} + w_0 \leq Q^t$, truck k is load-feasible after visiting customer i. In addition, $max\{\tau_i^k, u_i\}$ captures the service start time at customer i, and $max\{\tau_i^k, u_i\} + s_i + \frac{r^t_{i,n+m+1}}{v^t}$ captures the earliest arrival time of truck k at the depot after serving customer i. Therefore, if $\tau_i^k \leq l_i$ and $max\{\tau_i^k, u_i\} + s_i + \frac{r^t_{i,n+m+1}}{v^t} \leq l_{max}$, truck k is time-feasible after visiting customer i.

Moreover, for the operation to assign customer pairs, the following rules are applied to determine possible insertion locations and reduce invalid searches. We define a truck route k as an ordered node set

$$R^k := \{0, i_1, i_2, ..., i_g, ..., i_G, n+m+1\},$$

where G is the number of nodes visited by the truck, and define $[I_{i_p}^{lb}, I_{i_p}^{ub}]$ as the possible insertion section of node i_p and $I_{i_p}^{min}$ as the minimum insertion location of node i_p, i.e., if node $i_p \in N_{cd}$, $I_{i_p}^{min}$ refers to the next location of its pickup counterpart. Otherwise, $I_{i_p}^{min} = 1$. Then,

- if $u_{i_p} \geq l_{i_G}$, $I_{i_p}^{lb} = I_{i_p}^{ub} = G + 1$;
- if $l_{i_p} \leq u_{i_1}$ and $I_{i_p}^{min} = 1$, $I_{i_p}^{lb} = I_{i_p}^{ub} = 1$;
- if $l_{i_p}^{min} < l_{i_g} \leq u_{i_p} < l_{i_{g+1}}$, $I_{i_p}^{lb} = g + 1$, else, $I_{i_p}^{lb} = l_{i_p}^{min}$;
- if $u_{i_{g-1}} < l_{i_p} \leq u_{i_g}$, $I_{i_p}^{ub} = g$, else, $I_{i_p}^{ub} = G + 1$.

4.2. Destroy operators

Five destroy operators are developed for the current solution. Note that if all served customers in a flight are removed, this flight and the stations it uses (not used by other flights) will be removed from the solution as well, and if a paired node is removed, its counterpart is also removed.

1. **Node removal (ND)** randomly deletes $\lceil 5\% * n \rceil$ customers, a used station and a customer pair from the solution.

2. **Related removal (RD)** considers removing customers with some similarities. A randomly selected customer (i.e., the seed) is first removed from the solution, and then the node with the smallest relatedness with it becomes the seed and is deleted. Customers are progressively removed and the relatedness with the current seed is updated accordingly until $\lceil 10\% * n \rceil$ nodes are removed. Considering the problem features, the relatedness between two nodes i and j is measured by $\alpha_1(\frac{r^t_{ij}}{v^t} + \frac{r^d_{ij}}{v^d}) + \alpha_2(|u_i - u_j| + |l_i - l_j|) + \alpha_3|d_i - d_j|$, $\alpha_1, \alpha_2, \alpha_3 \in [0, 1]$. Preliminary experiments have obtained the following parameter tuning: $(\alpha_1, \alpha_2, \alpha_3) = (0.50, 0.42, 0.25)$.

3. **String removal (SD)** randomly deletes a string from the current solution. Note that if the removed string also includes stations, the related flights are also deleted. If all nodes on a route are removed, the route is deleted from the solution.

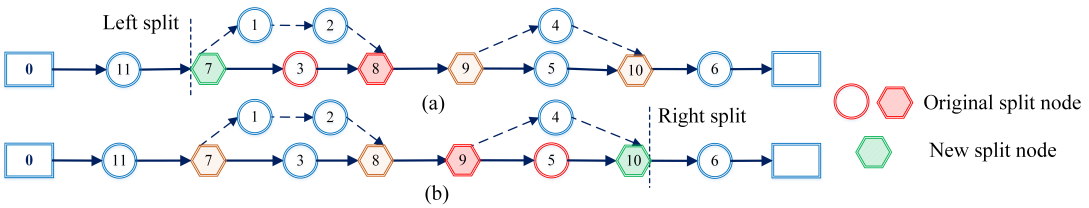


Figure 5: String removal.

Figure 5 presents two situations in which the split nodes must be adjusted. If the randomly selected split node is located between a pair of launch-retrieval stations (i.e., a combination) or a retrieval-only (if it is a left split node)/launch-only station (if it is a right split node) of a flight, the left (right) split node is adjusted to the launch (retrieval) node of this flight. Note that if a split node also serves other flights, it will be kept on the route.

4. **Cluster removal (CD)** occurs in a zone around a random seed customer/used station on the routes. A random seed node is first selected and deleted from the solution, and then one of its two nearest nodes (a customer/used station) becomes the new seed node and is deleted. Note that the next node to be deleted is randomly selected from the two closest nodes to the seed node in the current solution, and if a station is to be deleted, the affected flights will be removed as well. Customers or stations are progressively removed as above procedure until $\lceil 10\% * n \rceil$ nodes have been removed.

5. **Worst removal (WD)** focuses on removing the most-costly customers from the solution. It is driven by a measure of the cost reduction CR_i that gives the difference between the solution cost before and after node i is removed. In particular, we repeat the step that deletes the node with the greatest cost reduction in the current solution until $\lceil 10\% * n \rceil$ nodes are removed.

4.3. Repair operators

Five repair operators are deployed to repair a destroyed solution into a feasible one. For diversity, each node to be relocated is chosen from the removed node list at random. Note that if a removed customer/customer pair cannot be assigned to the existing routes, a new route is greedily constructed.

1. **Random relocation (RR)** assigns all removed customers to random feasible locations. That is, only if the node cannot be randomly relocated to a drone flight, is flight construction or insertion into a truck route performed at random. We repeat the above step for each randomly selected customer until all removed customers are assigned.

2. **Random insertion (RI)** first randomly inserts each randomly selected node into the existing flights until all removed nodes are assigned or attempted. Then for each remaining unassigned node (if any), an insertion into the truck routes with the least cost increase or a new flight construction with the most cost-saving stations is performed, whichever is better. Note that in contrast to the cost reduction, the cost increase CI_i gives the difference between the solution cost after and before node i is relocated. When updating cost increases, we only recalculate CI_i for the affected route.

3. **Greedy insertion (GI)** first inserts each randomly selected node into the existing flights or truck routes with the least cost increase, until all removed customers are assigned or attempted. Then, a new flight is randomly created for each node that is unassigned in the previous step (if any).

4. **Greedy relocation (GR)** considers the best relocation to the existing routes or a new route for each removed node i, i.e., assigning node i to the position with the least cost increase CI_i in the current solution. Specifically we compute all cost increases for node i with all possible relocation positions in the

current solution, either an existing drone flight/truck route or a newly constructed flight/route. Then, node i is relocated to the position with the least cost increase. The cost increases for the next relocated node are then updated, and the procedure is repeated until all removed customers have been assigned.

5. **Greedy relocation with noise (GN)** is a special variant of the GR operator that adds randomness, i.e., noise, to the cost increase calculation. The cost increase of node i is calculated as $CI_i + r_m^t \cdot \mu \cdot \varepsilon$, where r_m^t is the maximum Manhattan distance between the nodes, μ is a noise control parameter set to 0.1, and ε is a random number generated uniformly from the continuous interval [-1, 1].

4.4. Local search

A local search (LS) with three neighbourhood operators is performed to further optimise the current best solution. We apply the SA procedure to perform the local search. In each iteration, the three operators are invoked in sequence to generate neighbourhood solutions, and the best-feasible one is set to the candidate solution.

- Truck node insertion randomly removes a node served by trucks from the current solution and then relocates it randomly to another feasible position on the original or another truck route. If the removed node cannot be inserted into other locations on the truck routes, the old solution is reverted.
- Drone node insertion randomly removes a node served by drones from the current solution and then relocates it randomly to the same flight (other locations) or another flight on the original or another route. Similarly, if all the above attempts are not feasible, the operator reverts to the original one.
- Station exchange invokes a position exchange between two randomly selected stations, at least one of which is used in the solution. Note that if one of the two selected stations is not being used by the current solution, we replace the other station with this one in the current solution.

4.5. Shake procedure

When the total number of iterations reaches a threshold, the shake procedure begins. The shake procedure (SP) randomly removes $\max\{2, \lceil 20\% * F \rceil\}$ flights from the current solution, with F being the number of flights in the solution, and then relocates them to other random-feasible positions, deploying newly selected stations with minimal cost. If no new feasible solution is found in this procedure, the old solution is returned.

4.6. Preprocessing procedure

Based on the characteristics of mDPD-T, some preprocessors are used to accelerate the algorithm by removing some searches that inevitably result in infeasible solutions. Before proceeding, we define virtual time windows for each deployed station.

The truck route of vehicle pair k is redefined as an ordered node set

$$R^k := \{0, i_1, i_2, ..., i_g, ..., i_{\tilde{g}}, ..., i_G, n + m + 1\},$$

and the corresponding drone flights are defined as an ordered arc set

$$F^k := \{(r^1, j_1^1, ..., j_{H_1}^1, \tilde{r}^1), (r^2, j_1^2, ..., j_{H_2}^2, \tilde{r}^2), ..., (i_g, j_1^d, ..., j_{h_d}^d, ..., j_{H_d}^d, i_{\tilde{g}}), ..., (r^D, j_1^D, ..., j_{H_D}^D, \tilde{r}^D)\},$$

where D is the number of flights of vehicle pair k, $(i_g, i_{\tilde{g}})$ is a combination of a flight and $(i_g, i_{\tilde{g}}) \equiv (r^d, \tilde{r}^d)$.

The virtual time window for each deployed station is defined as follows. For launch-only station i_g, the virtual time window $[u_{i_g}, l_{i_g}]$ is $u_{i_g} = u_{i_{g-1}} + s_{i_{g-1}}$ and $l_{i_g} = l_{j_1^d} - s_{i_g}$. For retrieval-only station $i_{\tilde{g}}$, $u_{i_{\tilde{g}}} = u_{j_{H_d}^d} + s_{j_{H_d}^d}$; if its next node $i_{\tilde{g}+1}$ is a customer node/depot, $l_{i_{\tilde{g}}} = l_{i_{\tilde{g}+1}} - s_{i_{\tilde{g}}}$; otherwise, $l_{i_{\tilde{g}}} = l_{j_1^{d+1}} - s_{i_{\tilde{g}}}$. For a launch station i_g that also hosts drone retrieval, $u_{i_g} = u_{j_{H_{d-1}}^{d-1}} + s_{j_{H_{d-1}}^{d-1}}$ and $l_{i_g} = l_{j_1^d} - s_{i_g}$.

4.6.1. Rules for station selection and insertion

For any vehicle pair k, let $o(i)$ be the index of flight using station i. We introduce the following rules.

1. The combinations $(r, \tilde{r})$ that both r and $\tilde{r}$ are on the route and $o(r) + 1 < o(\tilde{r})$ cannot be used to build a new flight for any customer i.

2. For the possible insertion sections (i.e., $[I_{r^d}^{lb}, I_{r^d}^{ub}]$ and $[I_{\tilde{r}^d}^{lb}, I_{\tilde{r}^d}^{ub}]$) of a combination $(r^d, \tilde{r}^d)$ of a new flight $(r^d, j_1^d, ..., j_{h_d}^d, ..., j_{H_d}^d, \tilde{r}^d)$, if $u_{i_{g-1}} \le l_{j_1^d} \le u_{i_g}$, $I_{r^d}^{ub} = g$; if $l_{i_g} \le u_{j_{H_d}^d} < l_{i_{g+1}}$, $I_{\tilde{r}^d}^{lb} = g + 1$.

3. To accelerate the insertion of a new combination $(r, \tilde{r})$, we modify the truck route. As shown in Figure 6, we modify the route by folding the red arcs into their launch stations, as insertions into them must lead to infeasibility. Moreover, once the new launch station r is relocated, the retrieval node $\tilde{r}$ must be inserted into the arcs between station r and the next launch station/the ending depot. For example, the possible insertion arcs for the retrieval station of node 15 are (15, 5) and (5, 13).

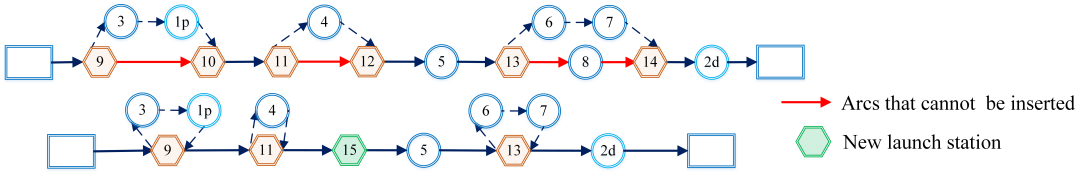


Figure 6: Truck route modification.

4.6.2. Rules for customer insertion

In addition to the rules introduced in Section 4.1, the following rules are added to determine the possible insertion locations of customer i in vehicle pair k.

1. If a pickup pair customer i is in a flight retrieved at station j, then only if $l_{i+B} > v_j^k$, can its delivery counterpart $i + B$ be inserted into the truck route. Otherwise, it may only be inserted into the same flight.

2. If $u_i + s_i + \frac{r_{ij}^d}{v^d} > l_j$ and customer i is served by a flight, then customer j cannot be inserted into this flight after customer i.

3. ϱ_1^s and ϱ_2^s denote the total demand of delivery only and pickup only customers of flight $s = (r^d, j_1^d, ..., j_{h_d}^d, .., j_{H_d}^d, \tilde{r}^d)$, respectively. For customer i to be inserted into s, the following rules apply

- If $l_i \le u_{r^d}$ or $u_i \ge l_{\tilde{r}^d}$, customer i cannot be inserted into flight s.
- If $i \in N_{cd}$, its pickup counterpart is not in s, and $\varrho_1^s + d_i > Q^d$, customer i cannot be inserted into flight s.

- If $i \in N_{cp}$, its delivery counterpart is not in s, and $|\varrho_2^s + d_i| > Q^d$, customer i cannot be inserted into flight s.
- If $i \in N_{od}$ and $\varrho_1^s + d_i > Q^d$, customer i cannot be inserted into flight s.
- If $i \in N_{op}$ and $|\varrho_2^s + d_i| > Q^d$, customer i cannot be inserted into flight s.

4. The following rules apply to determine the possible insertion section of customer i into flight s. Note that if $i \notin N_{cd}$ or $i \in N_{cd}$, its pickup counterpart $i - B$ is not in flight s, $I_i^{min} = 1$. Otherwise, I_i^{min} equals the next location of customer $i - B$ in flight s.

- If $l_{j_{H_d}^d} \leq u_i$ or $I_i^{min} = H_d + 1$, $I_i^{lb} = I_i^{ub} = H_d + 1$.
- If $u_{j_1^d} \geq l_i$ and $I_i^{min} = 1$, $I_i^{lb} = I_i^{ub} = 1$.
- If $I_i^{min} < l_{j_{h_d}^d} \leq u_i < l_{j_{h_d+1}^d}$, $I_i^{lb} = h_d + 1$; otherwise, $I_i^{lb} = I_i^{min}$.
- If $u_{j_{h_d-1}^d} < l_i \leq u_{j_{h_d}^d}$, $I_i^{ub} = h_d$; otherwise, $I_i^{ub} = H_d + 1$.

4.7. Load feasibility test on trucks

In addition to the constraints of time windows, the load capacity of trucks at each node must be respected in each iteration. We adopt the method designed by Meng et al. (2023) and present the details in Supplementary material D.

4.8. Feasibility test on drones

The feasibility of drones involves payload, energy consumption and time window constraints. Since the load feasibility is straightforwards, we focus on the battery capacity feasibility, which is more complex, as it is closely related to time windows and the entire route schedule. The pseudocode for the drone feasibility test for each vehicle pair is presented in Supplementary material E.

Let T_p be the last arrival time at launch station p, which includes two cases: if p is a launch-only station, $T_p = \tau_p$, and if p is a retrieval station, $T_p = max\{\tau_p, \hat{\tau}_p\} + S_r$. T_p^w is the waiting time at station p from time T_p to drone launch.

For each vehicle pair k, we traverse every station p along truck route R^k and calculate the arrival time τ_p of truck k at station p. If p is a launch-only station of flight s, we perform a "launch check": The best waiting time T_p^w at node p is calculated (detailed in Section 4.9), the load feasibility of flight s is verified, and whether the energy e^s consumed when arriving at the retrieval station exceeds the battery capacity is checked. If feasible, we record the energy consumption e^s, drone arrival time $\hat{\tau}^s$, and payload $\hat{w}^s$ of drone k at the retrieval station of flight s. Otherwise, if node p is a retrieval-only station, we perform a "retrieval check": conducting a feasibility test on total energy consumption e'^s by adding the hovering energy consumption at node p to e^s. Otherwise, we first conduct a "retrieval check", update T_p, and then perform a "launch check".

4.9. Waiting time optimisation

The waiting time at each launch station directly affects flight feasibility and energy consumption. This section discusses the waiting time optimisation.

As the number of customers served by a single flight is limited, the waiting time at each launch node can be solved by Gurobi. The detailed models are presented in Supplementary material F. However, as waiting time optimisation must be performed at every launch node in each iteration, the overall computational time of Gurobi increases prohibitively. Therefore, we design the following algorithm to solve this problem.

For each flight $s = (p, p_1, ..., p_z, p_{z+1})$, let $r = (p, \tilde{p}_1, ..., \tilde{p}_{\tilde{z}}, p_{z+1})$ be the truck sub-route of flight s. The definitions of other variables are the same as those in the model but without index k, i.e., for any s and r.

Case 1. The truck travels arc (p, p_{z+1}) directly without serving any customers (i.e., $\tilde{z} = 0$).

Proposition 2. *For $z = 1$, $\tilde{z} = 0$, if $max\{u_{p_1} - T_p - S_l - \frac{r^d_{p,p_1}}{v^d}, 0\} \leq T^w_p \leq l_{p_1} - T_p - S_l - \frac{r^d_{p,p_1}}{v^d}$, the energy consumption of flight s is minimised and independent of the value of T^w_p.*

In view of Proposition 2, if $z = 1$, we set $T^w_p = max\{u_{p_1} - T_p - S_l - \frac{r^d_{p,p_1}}{v^d}, 0\}$.

Proposition 3. *For $z = 2$, $\tilde{z} = 0$, the earliest time that the service starts at customer p_1 is $u_{p_1} + max\{u_{p_2} - \frac{r^d_{p_1,p_2}}{v^d} - s_{p_1} - u_{p_1}, 0\} - max\{u_{p_2} - \frac{r^d_{p_1,p_2}}{v^d} - s_{p_1} - l_{p_1}, 0\}$, allowing minimum waiting time at customers.*

Corollary 1. *For $z = 2$, $\tilde{z} = 0$, if the service start time at customer p_1 is $u_{p_1} + max\{u_{p_2} - \frac{r^d_{p_1,p_2}}{v^d} - s_{p_1} - u_{p_1}, 0\} - max\{u_{p_2} - \frac{r^d_{p_1,p_2}}{v^d} - s_{p_1} - l_{p_1}, 0\}$ and $\frac{r^t_{p,p_3}}{v^t} \leq u_{p_2} + s_{p_2} + \frac{r^d_{p_2,p_3}}{v^d} - (l_{p_1} - \frac{r^d_{p,p_1}}{v^d}) - min\{u_{p_2} - \frac{r^d_{p_1,p_2}}{v^d} - s_{p_1} - l_{p_1}, 0\}$, the energy consumption of flight s is minimised and independent of the value of T^w_p. Additionally, the service start time at customer p_2 is $u_{p_2} - min\{u_{p_2} - \frac{r^d_{p_1,p_2}}{v^d} - s_{p_1} - u_{p_1}, 0\}$.*

In view of Corollary 1, for $z = 2$, if $u_{p_2} \geq \frac{r^d_{p_1,p_2}}{v^d} + s_{p_1} + l_{p_1}$ and $\frac{r^t_{p,p_3}}{v^t} + l_{p_1} - \frac{r^d_{p,p_1}}{v^d} \leq u_{p_2} + s_{p_2} + \frac{r^d_{p_2,p_3}}{v^d}$, we set $T^w_p = l_{p_1} - T_p - S_l - \frac{r^d_{p,p_1}}{v^d}$. Otherwise, if $u_{p_2} < \frac{r^d_{p_1,p_2}}{v^d} + s_{p_1} + l_{p_1}$ and $\frac{r^t_{p,p_3}}{v^t} \leq s_{p_2} + \frac{r^d_{p_2,p_3}}{v^d} + \frac{r^d_{p,p_1}}{v^d} + \frac{r^d_{p_1,p_2}}{v^d} + s_{p_1}$, we set $T^w_p = max\{u_{p_1}, u_{p_2} - \frac{r^d_{p_1,p_2}}{v^d} - s_{p_1}\} - T_p - S_l - \frac{r^d_{p,p_1}}{v^d}$.

For $z \geq 2$, the waiting time T^w_p is bounded by

$$T^w_p \geq max\{max\{u_{p_1}, min\{u_{p_2} - s_{p_1} - \frac{r^d_{p_1,p_2}}{v^d}, l_{p_1}\}\} - T_p - S_l - \frac{r^d_{p,p_1}}{v^d}, 0\} \tag{34a}$$

$$T^w_p \leq min\{l_{p_1}, l_{p_2} - s_{p_1} - \frac{r^d_{p_1,p_2}}{v^d}\} - T_p - S_l - \frac{r^d_{p,p_1}}{v^d} \tag{34b}$$

Case 2. The truck sub-route r serves customers ($\tilde{z} > 0$).

Proposition 4. *For $z = 1$, $\tilde{z} = 1$, if $max\{u_{p_1} - T_p - S_l - \frac{r^d_{p,p_1}}{v^d}, u_{\tilde{p}_1} - T_p - S_l - \frac{r^t_{p,\tilde{p}_1}}{v^t}, 0\} \leq T^w_p \leq min\{l_{p_1} - \frac{r^d_{p,p_1}}{v^d}, l_{\tilde{p}_1} - \frac{r^t_{p,\tilde{p}_1}}{v^t}\} - T_p - S_l$, the energy consumption of flight s is independent of the value of T^w_p.*

In general, for $\tilde{z} \geq 1$, the bounds of waiting time T^w_p can be strengthened by

$$T^w_p \geq max\{min\{u_{p_1} - \frac{r^d_{p,p_1}}{v^d} - T_p - S_l, u_{\tilde{p}_1} - \frac{r^t_{p,\tilde{p}_1}}{v^t} - T_p - S_l\}, 0\} \tag{35a}$$

$$T_p^w \leq min\{l_{p_1} - \frac{r_{p,p_1}^d}{v^d}, l_{\tilde{p}_1} - \frac{r_{p,\tilde{p}_1}^t}{v^t}\} - T_p - S_l \tag{35b}$$

The proofs for the above theoretical results are included in Supplementary material G.

For Case 1, if $z = 1$, we return the best waiting time T_p^w by Proposition 2. Otherwise, if $z = 2$ and the conditions in Corollary 1 are satisfied, the best waiting time T_p^w is returned by Corollary 1. Otherwise, the waiting time T_p^w is optimised by WT opt. _one. For Case 2, we optimise the waiting time T_p^w by WT opt. _two. Pseudocodes for the waiting time optimisation, WT opt. _one and WT opt. _two are shown in Supplementary material F.

5. Numerical experiments

In this section, we study the performance of our proposed model/enhancements and the ALNS heuristic by conducting various computational experiments on a number of randomly generated instances. We begin with a description of test instances in Section 5.1, followed by the detailed computational results in Section 5.2. In Section 5.3, we analyse the impact of three key problem features. The sensitivity analysis around the drone cost is conducted in Section 5.4. The operator analysis and sensitivity study on the effects of three parameters (number of stations, and width and density of time windows) are included in the Supplementary materials I and K. Our algorithm is coded in Python 3.9.12; all experiments are executed on a Windows 10 machine with an Intel(R) Core (TM) i7-10750H processor (2.60 GHz) and 32 GB of RAM. The mathematical model is solved by Gurobi.

5.1. Test instances and parameters

In our experiments, the curb weight (w_0) and load capacity (Q^d) of drones are assumed to be 6 kg and 3 kg, respectively. The unit energy consumption (η) and battery capacity (E) of drones are set to 0.667 Wh/(kg·min) and 180 Wh. The average speed of trucks (v^t) is 30 km/h and that of drones (v^d) is 40 km/h. Considering the features of our problem, we use light trucks (e.g., Wuling Glory S Van) that each has a total payload of 750 kg. The available capacity of each truck (Q^t), excluding the weight of drone supporting materials (i.e., tools, auxiliary equipment, etc.), is assumed to be 650 kg. The unit distance cost for trucks (c^t) is \$0.78/km, which is 8.3 times (i.e., $\alpha = 8.3$) of that of a fully loaded drone (Salama & Srinivas, 2020); thus, the unit distance cost for drones is \$0.094/km, which is equivalent to a unit energy cost (c^d) of \$0.0104/Wh ($= 0.094 * 0.5 * v^d/E$), where 0.5 h is the flying duration of a fully loaded drone. Finally, the required times for launch and retrieval (S_l, S_r) are both set to 1 min and the service time for each customer (s_i) is set to 2 min. The time window of the depot and all stations is [0, 450]. The fixed cost of deploying a vehicle pair is set to \$10. The parameters used in the ALNS are selected by a parameter tuning experiment, which is described in Supplementary material H.

Since there are no benchmark instances available for mDPD-T, we generate 9 groups of instances with 3/5/8 stations and 15/30/50 customers (in a square with 10 km×10 km/12 km×12 km/12 km×12 km

dimension). The locations are uniformly generated by the method in Liu et al. (2020), and the approach of generating time windows can be found in Dellaert et al. (2019). Given drones' load capacity, the customer demand is generated as follows. We first randomly generate demand for 90% of the customers within the range (0,2.3] (i.e., drone-eligible customers) and for the remaining 10% truck-only customers within the range (3,50] (Zhen et al., 2023). Then, we set 10% of all the customers as paired ones and adjust the demand of each pickup pair node to be the same amount as its delivery counterpart, and set 30% as pickup only customers. The remaining ones are all delivery only customers. Each group comprises 6 instances that include three scenarios based on different time windows:

A: for customer i, randomly generate $20 \leq u_i \leq 260$ and $l_i = u_i + 20$;

B: for customer i, randomly generate $60 \leq u_i \leq 360$ and $l_i = u_i + 20$;

C: for customer i, randomly generate $60 \leq u_i \leq 360$ and $l_i = u_i + 90$.

In total 54 instances are generated. We use the short form "A/B/C-#.m.n" to represent each of them, among which the instance A-1.5.15 is the running example that we have used. In addition, to evaluate the effectiveness of our model and the performance of ALNS against the optimum, we also generate 30 small-size instances over a grid of dimensions 10 km ×10 km in scenario A, in which half of the instances have 8 customers and 4 stations while the rest have 10 customers and 5 stations. All instances are available for download from *https://doi.org/10.17632/6x3jsjcc3r.3*.

5.2. Computational results

5.2.1. Effectiveness of the valid inequalities and the performance of ALNS compared to optimality

To validate the mathematical model, study the effectiveness of the valid inequalities, and evaluate the performance of the proposed ALNS, we compare the solutions from Gurobi and ALNS on the 30 small instances. For each instance, the time limit of Gurobi is set to 2 h, and ALNS is repeatedly executed 5 times. Table 3 reports the best solutions from Gurobi without valid inequalities ($s_{best}^{G_1}$) and with them ($s_{best}^{G_2}$), along with their run times cpu^{G_1} and cpu^{G_2}, respectively. For ALNS, the results include the best solution obtained (s_{best}^{A}) and the average run time (cpu^{A}). The last two columns report the percentage gap between the solver solutions with and without valid inequalities and that between the solutions of ALNS and Gurobi with valid inequalities, i.e., $Gap_1\% = 100(s_{best}^{G_2} - s_{best}^{G_1})/s_{best}^{G_1}$ and $Gap_2\% = 100(s_{best}^{A} - s_{best}^{G_2})/s_{best}^{G_2}$. Thus, a negative (positive) gap means better (weaker) performance of the solver solutions with valid inequalities versus without or better (weaker) performance of ALNS versus the solver.

The results indicate the clear effects of the valid inequalities, as the run time of the solver clearly decreases with the inclusion of these inequalities, especially for larger instances, which see a more than 30% reduction on average. As the number of customers increases, the run time of Gurobi increases substantially. Moreover, its performances depends on the specific instances; the run times fluctuate even for instances of the same size. In contrast, ALNS is much more robust.

In terms of solution quality, both Gurobi and ALNS solve all considered instances with 8 customers

and 4 stations to optimality, but ALNS requires significantly less time on average. For instances with 10 customers and 5 stations, with the inclusion of valid inequalities, the number of instances not solved to optimality within the time limit by Gurobi decreases from 5 to 3, and better solutions are obtained for 2 of the remaining 3 instances, confirming the effectiveness of the valid inequalities. In contrast, for all but one of these larger instances, ALNS finds the same or better solutions than Gurobi with much shorter run times, highlighting the remarkable performance of the proposed heuristic.

Table 3: ALNS vs. the solver solution in small instances.

Inst.	Solver solution (without valid inequalities)		Solver solution (with valid inequalities)		ALNS		Gap_1%	Gap_2%
	$s_{best}^{G_1}$(\$)	cpu^{G_1}(s)	$s_{best}^{G_2}$(\$)	cpu^{G_2}(s)	s_{best}^{A}(\$)	cpu^{A}(s)		
A-1.4.8	32.40	111.89	32.40	70.16	32.40	12.50	0.00	0.00
A-2.4.8	37.75	104.56	37.75	72.41	37.75	27.59	0.00	0.00
A-3.4.8	41.32	273.26	41.32	221.32	41.32	19.54	0.00	0.00
A-4.4.8	38.01	376.33	38.01	209.53	38.01	27.61	0.00	0.00
A-5.4.8	35.33	101.42	35.33	79.56	35.33	25.73	0.00	0.00
A-6.4.8	48.12	146.36	48.12	132.03	48.12	32.22	0.00	0.00
A-7.4.8	36.77	141.60	36.77	116.23	36.77	22.26	0.00	0.00
A-8.4.8	34.05	163.89	34.05	138.56	34.05	29.03	0.00	0.00
A-9.4.8	38.55	549.53	38.55	346.48	38.55	38.54	0.00	0.00
A-10.4.8	33.85	185.29	33.85	95.49	33.85	26.58	0.00	0.00
A-11.4.8	40.89	203.20	40.89	201.36	40.89	28.29	0.00	0.00
A-12.4.8	40.25	109.56	40.25	72.36	40.25	36.24	0.00	0.00
A-13.4.8	34.00	389.23	34.00	251.13	34.00	41.73	0.00	0.00
A-14.4.8	34.91	172.39	34.91	138.74	34.91	34.41	0.00	0.00
A-15.4.8	40.27	119.23	40.27	81.46	40.27	36.79	0.00	0.00
Avg.		**209.85**		**148.45**		**29.27**		
A-1.5.10	42.67	6102.03	42.67	2532.16	42.67	125.50	0.00	0.00
A-2.5.10	37.62	5841.23	37.62	3323.59	37.62	48.51	0.00	0.00
A-3.5.10	32.52	3562.19	32.52	1153.26	32.52	48.99	0.00	0.00
A-4.5.10	47.64	7200.00*	46.58	7200.00*	46.39	93.43	-2.23	-0.41
A-5.5.10	32.31	5023.89	32.31	1596.36	32.31	42.91	0.00	0.00
A-6.5.10	52.08	3004.89	52.08	1432.16	52.08	55.86	0.00	0.00
A-7.5.10	45.79	7200.00*	44.93	7200.00*	44.90	61.07	-1.88	-0.07
A-8.5.10	42.00	5123.63	42.00	3853.36	42.00	51.82	0.00	0.00
A-9.5.10	28.40	5231.09	28.40	2855.38	28.40	53.60	0.00	0.00
A-10.5.10	39.71	7200.00*	39.71	7200.00*	39.57	119.11	0.00	-0.35
A-11.5.10	34.10	7200.00*	34.10	5871.23	34.10	46.97	0.00	0.00
A-12.5.10	36.75	3125.65	36.75	1152.36	36.75	227.18	0.00	0.00
A-13.5.10	32.12	7200.00*	31.98	6852.13	31.90	56.59	-0.44	0.25
A-14.5.10	29.38	1526.36	29.38	896.49	29.38	48.68	0.00	0.00
A-15.5.10	34.89	3148.26	34.89	2103.93	34.89	69.06	0.00	0.00
Avg.		**4168.92**		**2801.87**		**76.62**		

*Gurobi does not find an optimal solution in the 2 h time limit.

5.2.2. Performance analysis of ALNS on large-size instances

In this section, we evaluate the performance of ALNS on the 54 large-size instances generated in Section 5.1. Table 4 summarises the complete results. In addition to the best/average costs among 5 runs (c_{best}/c_{avg}) and the truck-only cost (c_t), the initial cost (c_i) of the best solution and the best cost obtained by Gurobi in 2 h (c_g) are also reported for comparison. Additionally, we provide the results regarding the number of deployed vehicle pairs (U), the number of flights (F), the number of customers served by drones (R) and the number of drone-eligible customers (D), showing the characteristics of the best solutions. "Δ_{r-d}%" captures the ratio of the drone-served customers to drone-eligible customers ($\Delta_{r-d}\% = 100R/D$). "Δ_{i-b}%" and "Δ_{t-b}%" reflect the gap between the initial cost and the best cost ($\Delta_{i-b}\% = 100(c_{best} - c_i)/c_i$) and

the gap between the truck-only cost and the best cost ($\Delta_{t-b}\% = 100(c_{best} - c_t)/c_t$), respectively. The last column reports the average run times. For brevity, we use "cost" to denote the total operational cost.

Table 4: Performance of ALNS in scenarios A, B and C.

Inst.	c_t(\$)	c_i(\$)	c_g(\$)	c_{best}(\$)	c_{avg}(\$)	U	F	R	D	$\Delta_{r-d}\%$	$\Delta_{t-b}\%$	$\Delta_{i-b}\%$	cpu(s)
A-1.3.15	67.39	66.41	50.50*	45.78	47.65	2	2	6	13	46.15	-32.07	-31.06	37.89
A-2.3.15	65.21	64.30	45.60*	43.03	44.09	2	2	5	13	38.46	-34.01	-33.08	47.34
B-1.3.15	100.15	87.63	56.13*	52.23	54.65	2	4	9	13	69.23	-47.85	-40.40	39.96
B-2.3.15	89.23	86.54	51.75*	46.85	47.73	2	3	6	13	46.15	-47.50	-45.86	20.61
C-1.3.15	63.34	51.94	52.17*	32.11	35.87	1	2	8	13	61.54	-49.30	-38.18	140.33
C-2.3.15	58.34	58.11	56.02*	33.89	35.03	1	2	8	13	61.54	-41.91	-41.68	141.04
Avg.										**53.85**	**-42.11**	**-38.38**	
A-1.5.15	69.76	60.59	55.57*	35.25	36.50	1	4	10	13	76.92	-49.47	-41.82	191.45
A-2.5.15	59.44	54.56	37.80*	32.29	32.87	2	3	7	13	53.85	-45.68	-40.82	229.97
B-1.5.15	73.01	70.91	45.95*	33.45	36.72	2	5	10	13	76.92	-54.18	-52.83	109.43
B-2.5.15	85.96	75.85	45.65*	35.40	36.64	2	5	9	13	69.23	-58.82	-53.33	113.91
C-1.5.15	73.32	64.21	39.29*	32.53	33.26	1	2	7	13	53.85	-55.63	-49.34	339.53
C-2.5.15	52.10	52.10	45.76*	28.16	31.03	1	4	10	13	76.92	-45.95	-45.95	312.86
Avg.										**67.95**	**-51.62**	**-47.35**	
A-1.8.15	88.80	64.97	35.51*	27.71	28.53	2	4	10	13	76.92	-68.80	-57.35	915.42
A-2.8.15	79.60	54.65	40.43*	23.69	24.60	2	4	10	13	76.92	-70.24	-56.65	718.61
B-1.8.15	63.34	56.05	38.65*	32.06	33.14	2	4	10	13	76.92	-49.38	-42.80	931.25
B-2.8.15	74.20	57.72	34.83*	29.97	32.43	2	4	9	13	69.23	-59.61	-48.08	1231.33
C-1.8.15	79.40	60.33	37.94*	24.02	25.26	1	5	13	13	100.00	-69.75	-60.19	1360.14
C-2.8.15	56.63	48.86	39.81*	23.91	25.97	1	3	11	13	84.62	-57.78	-51.06	1754.32
Avg.										**80.77**	**-62.59**	**-52.69**	
A-1.3.30	186.42	172.37	-	85.71	88.43	3	3	10	27	37.04	-54.02	-50.28	132.09
A-2.3.30	158.96	150.11	-	87.57	90.14	4	2	6	27	22.22	-44.91	-41.66	129.19
B-1.3.30	159.59	146.64	-	78.07	81.16	4	4	10	27	37.04	-51.08	-46.76	149.29
B-2.3.30	136.81	121.40	-	72.84	74.15	3	2	7	27	25.93	-46.76	-40.00	112.86
C-1.3.30	130.88	117.12	-	62.14	67.73	2	4	12	27	44.44	-52.52	-46.94	252.64
C-2.3.30	100.78	93.94	-	61.79	63.85	3	5	17	27	62.96	-38.69	-34.22	371.06
Avg.										**38.27**	**-48.00**	**-43.31**	
A-1.5.30	131.35	108.95	-	52.26	56.13	3	4	15	27	55.56	-60.21	-52.03	421.05
A-2.5.30	167.54	162.36	-	70.57	71.54	3	3	10	27	37.04	-57.88	-56.53	762.03
B-1.5.30	155.84	155.07	-	72.06	75.75	3	7	16	27	59.26	-53.76	-53.53	353.93
B-2.5.30	188.76	172.84	-	81.38	83.61	3	5	11	27	40.74	-56.89	-52.92	629.41
C-1.5.30	152.41	134.24	-	53.03	58.44	2	6	21	27	77.78	-65.21	-60.50	879.06
C-2.5.30	129.32	114.67	-	57.92	62.26	3	4	13	27	48.15	-55.21	-49.49	1429.96
Avg.										**53.09**	**-58.19**	**-54.17**	
A-1.8.30	146.17	130.66	-	63.42	67.36	3	8	18	27	66.67	-56.61	-51.46	2142.38
A-2.8.30	158.96	107.42	-	62.89	65.59	4	8	19	27	70.37	-60.44	-41.45	1853.68
B-1.8.30	154.28	129.79	-	81.25	84.90	3	9	20	27	74.07	-47.34	-37.40	1053.28
B-2.8.30	180.02	156.55	-	70.03	72.42	4	7	16	27	59.26	-61.10	-55.27	2561.43
C-1.8.30	180.02	145.23	-	71.60	73.64	3	9	22	27	81.48	-60.23	-50.70	1532.65
C-2.8.30	160.99	152.97	-	53.73	55.97	2	5	17	27	62.96	-66.63	-64.88	3365.23
Avg.										**69.14**	**-58.73**	**-50.19**	
A-1.3.50	232.28	222.57	-	106.65	111.43	5	4	14	45	31.11	-54.09	-52.08	256.23
A-2.3.50	214.66	197.24	-	84.42	90.64	5	7	24	45	53.33	-60.67	-57.20	168.53
B-1.3.50	232.28	191.54	-	114.16	117.24	5	12	33	45	73.33	-50.85	-40.40	140.92
B-2.3.50	245.39	228.93	-	114.79	121.99	5	3	11	45	24.44	-53.22	-49.86	356.23
C-1.3.50	128.39	128.39	-	84.78	87.23	3	2	7	45	15.56	-33.97	-33.97	2031.46
C-2.3.50	176.59	173.94	-	87.51	94.16	4	3	13	45	28.89	-50.44	-49.69	1036.58
Avg.										**37.78**	**-50.54**	**-47.20**	
A-1.5.50	205.30	198.99	-	86.52	98.31	5	8	25	45	55.56	-57.86	-56.52	1603.16
A-2.5.50	227.92	197.59	-	99.09	103.06	6	7	18	45	40.00	-56.52	-49.85	912.59
B-1.5.50	327.13	286.69	-	131.54	136.76	5	10	25	45	55.56	-59.79	-54.12	1395.23
B-2.5.50	330.88	292.08	-	99.28	104.87	5	12	32	45	71.11	-67.00	-66.01	852.01
C-1.5.50	197.18	184.46	-	75.22	81.31	4	9	30	45	66.67	-61.85	-59.22	2537.29
C-2.5.50	204.98	176.70	-	82.56	89.54	4	7	26	45	57.78	-59.72	-53.28	2953.08
Avg.										**57.78**	**-60.46**	**-56.50**	
A-1.8.50	204.83	184.23	-	91.87	98.08	6	10	29	45	64.44	-55.15	-50.13	2856.03
A-2.8.50	199.21	171.46	-	88.47	89.23	5	8	20	45	44.44	-55.59	-48.40	3956.94
B-1.8.50	281.42	259.44	-	103.92	108.69	6	9	24	45	53.33	-63.07	-59.94	4059.26
B-2.8.50	290.94	263.12	-	109.76	110.83	6	7	21	45	46.67	-62.27	-58.29	2653.86
C-1.8.50	205.14	187.99	-	90.05	97.53	4	10	33	45	73.33	-56.10	-52.10	4751.03
C-2.8.50	185.02	167.32	-	99.68	107.37	4	9	30	45	66.67	-46.12	-40.43	4983.72
Avg.										**58.15**	**-56.38**	**-51.55**	

*Gurobi does not find an optimal solution in the 2 h time limit.

The computational time of ALNS increases with the number of customers and more so with the number of stations. Regardless, most of the instances are solved within 1 h, and all are solved within 2 h. Moreover, for instances with a wider time window (i.e., those in scenario C), the computational times are generally the longest, followed by those in scenarios A and B. In contrast, the costs of instances in scenario C are the smallest, followed by those in scenario A and then in scenario B. Note that the time windows in scenario B are closer to the deadline for returning to the depot than are those in scenario A. In addition, in most instances, the cost in the initial solutions is substantially reduced in the final solutions, typically by 35% to 65%, demonstrating the effectiveness of ALNS in obtaining better solutions.

Regarding the solution characteristics, Gurobi cannot find optimal solutions for any instance within 2 h. Even though it can find the feasible solutions for the first 18 instances with 15 customers, there is a clear gap from the results of ALNS. For all remaining instances Gurobi cannot find any feasible solutions within the time limit. The results demonstrate the strong performance of our ALNS for solving large-size mDPD-T. In addition, the instances in scenario C require the least number of vehicle pairs while providing almost the same or even more flights and drone services as scenarios A and B in most cases. The ratio of drone-served customers to those eligible varies widely from 30% to 80%, but for instances of the same size, the gap between the ratios is relatively small.

Overall, the advantage of adopting drones is obvious. Notably, the costs are reduced by 54.29% on average with respect to the truck-only costs. Even the initial costs in most instances are less than the optimal costs in the truck-only mode, confirming significant cost savings in the combined truck-drone mode.

In addition, to further investigate the performance of ALNS, we compare it against two benchmark heuristics in the literature. The results demonstrate the superior performance of ALNS over the benchmarks in generating high-quality solutions. Please refer to Supplementary material J for the full details.

5.3. Impact of the key problem features

In this section, we apply our model and the solution approach to a real network and explore in-depth the impact of the key problem features on the cost performance and run time. Figure 7 shows the southeast corner of Zhengzhou, the capital city of Henan Province, China. The depot is the Henan Bonded Logistics Center, and three stores (Yonghui Supermarket (Economic Development Wanjincheng store), Dennis All Day Fresh (Seventh Street store) and Huiermei Life Supermarket) are chosen as the pickup pair nodes. Their corresponding delivery nodes and the pickup/delivery only customers are randomly selected from 70 residential districts. The stations are randomly selected from 12 parking lots in the defined area. The demand and time windows (we only consider Scenario A) are set as in Section 5.1. In total ten instances with 5 stations and 30 customers are generated, denoted as D-#.5.30.

For illustration purpose, we also apply the following analyses to the running example A-1.5.15. The detailed solutions are reported in Supplementary material L.

Figure 7: Spatial distributions of the depot, stores, stations and customers.

5.3.1. Waiting time optimisation at the launch nodes for ALNS

In this section, we evaluate the effectiveness of waiting time optimisation on the final solutions. We compare our results with two cases in which no additional waiting time is considered at each launch node and the waiting time at each launch node is determined by the time window(s) of the first served customer(s) of the sub-routes from that node. Specifically, for each pair of sub-routes $s = (p, p_1, ..., p_z, p_{z+1})$ and $r = (p, \tilde{p}_1, ..., \tilde{p}_{\tilde{z}}, p_{z+1})$, if $\tilde{z} = 0$, the waiting time at launch node p is $T_p^w = max\{u_{p_1} - T_p - S_l - \frac{r^d_{p,p_1}}{v^d}, 0\}$. Otherwise, $T_p^w = max\{min\{u_{p_1} - T_p - S_l - \frac{r^d_{p,p_1}}{v^d}, u_{\tilde{p}_1} - T_p - S_l - \frac{r^t_{p,\tilde{p}_1}}{v^t}\}, 0\}$. We solve each case 5 times for each instance in set D and report the best results, average results and average run times in Table 5.

Table 5: Impact of waiting time optimisation.

Inst.	With optimisation				Without optimisation					Optimised by the first node(s)				
	U	c_{best}(\$)	c_{avg}(\$)	cpu(s)	U	c_{best}(\$)	c_{avg}(\$)	cpu(s)	Δ_{best}%	U	c_{best}(\$)	c_{avg}(\$)	cpu(s)	Δ_{best}%
D-1.5.30	2	30.02	32.01	327.32	2	38.71	45.88	430.08	28.95	2	34.69	37.05	346.14	15.56
D-2.5.30	1	29.08	31.17	950.22	1	37.47	42.47	893.10	28.85	2	31.72	35.01	559.16	9.08
D-3.5.30	2	25.66	27.43	423.67	2	35.96	40.86	517.45	40.14	2	28.69	30.41	698.10	11.81
D-4.5.30	2	27.16	28.18	464.34	2	32.69	40.02	623.70	20.36	2	30.56	34.83	416.35	12.52
D-5.5.30	2	28.22	29.17	345.92	2	39.40	45.85	344.49	39.62	2	35.01	42.50	272.33	24.06
D-6.5.30	2	23.60	26.70	582.88	3	32.25	40.17	301.80	36.65	2	29.41	36.17	255.61	24.62
D-7.5.30	2	27.56	28.77	281.95	4	41.27	48.57	192.16	49.75	2	32.34	32.95	177.17	17.34
D-8.5.30	2	32.17	33.12	546.47	2	37.56	44.81	351.23	16.75	2	34.55	39.35	324.84	7.40
D-9.5.30	2	27.10	28.37	667.51	2	37.72	43.71	281.24	39.19	2	29.23	34.58	274.75	7.86
D-10.5.30	2	26.44	28.49	539.15	3	35.20	39.47	379.43	33.13	2	30.56	35.96	375.21	15.58
Avg.									**31.13**					**14.79**

The results indicate that it is necessary and effective to optimise the waiting time at each launch node, evidenced by 31.13% and 14.79% higher cost on average in the other two cases. In addition, our ALNS provides more stable solutions, albeit with longer run times, as the gap between the average cost and best cost is smaller than those of the compared cases. Moreover, more vehicle pairs are needed if the waiting time at each launch node is not optimised.

5.3.2. Pair service

To explore the impact of pair service, we compare the solutions with and without paired customers for all instances in set D. Each instance is solved 5 times and the results are summarised in Table 6, in which the gaps between them are defined as $\Delta_R\% = 100(R^n - R^p)/R^p$, $\Delta_{best}\% = 100(c^n_{best} - c^p_{best})/c^p_{best}$ and $\Delta_{cpu}\% = 100(cpu^n - cpu^p)/cpu^p$.

The pair service significantly increases the computational complexity of our problem, as evidenced by

Table 6: Impact of pair service.

Inst.	With pair service				Without pair service				Δ_R%	Δ_{best}%	Δ_{cpu}%
	R^p	c^p_{best}(\$)	c^p_{avg}(\$)	cpu^p(s)	R^n	c^n_{best}(\$)	c^n_{avg}(\$)	cpu^n(s)			
D-1.5.30	19	30.02	32.01	327.32	17	30.95	34.43	77.91	-10.53	3.10	-76.20
D-2.5.30	19	29.08	31.17	950.22	8	32.53	37.13	120.53	-57.89	11.86	-87.32
D-3.5.30	24	25.66	27.43	423.67	16	25.94	27.30	95.44	-33.33	1.09	-77.47
D-4.5.30	14	27.16	28.18	464.34	13	27.36	28.56	119.13	-7.14	0.74	-74.34
D-5.5.30	14	28.22	29.17	345.92	15	29.77	32.25	83.37	7.14	5.49	-75.90
D-6.5.30	19	23.60	26.70	582.88	11	29.04	34.85	67.28	-42.11	23.05	-88.46
D-7.5.30	16	27.56	28.77	281.95	14	27.60	28.56	78.86	-12.50	0.15	-72.03
D-8.5.30	15	32.17	33.12	546.47	12	30.36	32.87	73.32	-20.00	-5.63	-86.58
D-9.5.30	15	27.10	28.37	667.51	14	25.64	27.05	82.54	-6.67	-5.39	-87.63
D-10.5.30	23	26.44	28.49	539.15	16	27.77	29.91	75.02	-30.43	5.03	-86.09
Avg.									**-19.41**	**3.59**	**-80.19**

the substantially longer CPU times in all instances. In addition, the cost is slightly higher when there are no pair service customers and the number of customers served by drones decreases. This is because pair service does not require any parcels to be carried from or to the depot, which somewhat relaxes the energy and load constraints on service vehicles. On the other hand, each customer pair must be served by the same vehicle pair, which partially offsets some of the above advantages.

5.3.3. Multiple visits of drones

In this section, we investigate the impact of the multi-visit feature of drones on the solutions, in comparison with the case in which drones can serve only a single customer per flight. The best solution for each instance in set D among 5 runs is presented in Table 7, in which R^t corresponds to the travel distances of trucks and the gap between the two cases is calculated as $\Delta_{R^t}\% = 100(R^t_s - R^t_m)/R^t_m$.

Table 7: Impact of multiple visits of drones.

Inst.	Multiple visits				Single visit				Δ_R%	Δ_{R^t}%	Δ_{best}%
	U	R	R^t_m(km)	c_{best}(\$)	U	R	R^t_s(km)	c_{best}(\$)			
D-1.5.30	2	19	31.30	30.02	2	4	46.23	38.77	-78.95	47.70	29.15
D-2.5.30	1	19	29.43	29.08	2	3	42.78	35.00	-84.21	45.36	20.36
D-3.5.30	2	24	23.39	25.66	2	3	44.20	35.86	-87.50	88.97	39.75
D-4.5.30	2	14	28.97	27.16	2	2	39.06	31.19	-85.71	34.83	14.84
D-5.5.30	2	14	31.20	28.22	2	5	44.48	39.06	-64.29	42.56	38.41
D-6.5.30	2	19	21.46	23.60	3	3	45.26	36.70	-84.21	110.90	55.51
D-7.5.30	2	16	28.62	27.56	2	6	45.46	39.11	-62.50	58.84	41.91
D-8.5.30	2	15	34.22	32.17	2	6	46.15	40.91	-60.00	34.86	27.17
D-9.5.30	2	15	29.10	27.10	2	4	42.73	35.83	-73.33	46.84	32.21
D-10.5.30	2	23	24.45	26.44	3	3	41.58	34.27	-86.96	70.06	29.61
Avg.									**-73.42**	**55.49**	**31.67**

The results indicate that allowing multiple visits is preferred, as it substantially reduces the operational cost. The single-visit case has an average cost increase of over 30% compared to the multi-visit case, and the truck travel distance is increased by 55.49% due to the significant reduction in the number of customers served by drones. Additionally, in the single-visit case, more vehicle pairs are required in some instances.

5.4. Sensitivity analysis on drone cost

Given that wide application of drones in last mile logistics is still in the early stage, there remain uncertainties and variabilities in drone cost. In this section we carry out a sensitivity analysis around the unit distance cost ratio of truck-to-drone (denoted by α) on instances A/B/C-#.3.15 and A/B/C-#.5.15

to study the impact of drone cost on the benefit of the combined truck-drone mode over the truck-only mode. In each setting, we change the value of only one parameter while keeping the remaining ones at their original levels (as in Section 5.1). For a more meaningful comparison, the truck-only routes are solved to optimality by Gurobi and others are all solved by ALNS. The average results from 5 runs for each setting are visualised in Figure 8. It is shown that the cost saving over the truck-only mode increases with higher α or lower drone cost, as a lower drone cost leads to more customers being served by drones. Furthermore for the combined mode, the travel distance of trucks decreases quickly with α until it reaches 5, after which the distance does not change much anymore. Figure 8 also highlights that the impact of α is greater for instances with more stations.

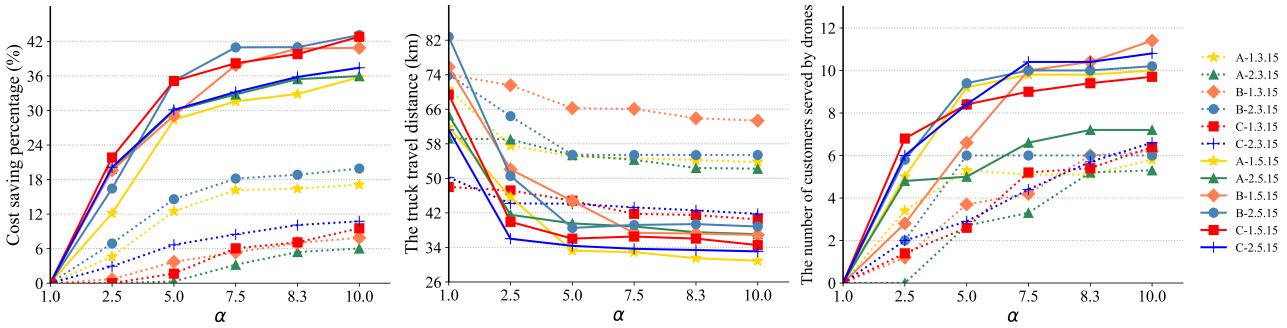


Figure 8: Impact of the cost ratio of truck-to-drone per unit distance.

6. Conclusion

We have studied a new variant of the truck-drone combined routing problem in last-mile logistics with the objective of minimising the total cost. In this model, a fleet of capacitated trucks, each carrying an identical drone with the capability of multiple visits, provides pickup and delivery services within the constraints of time windows. The load-dependent drone energy consumption is explicitly considered. Moreover, three types of customers are being served, and paired customers have strict precedence constraints. Briefly, the key to mDPD-T is to determine (1) the strategic selections of launch and retrieval stations, (2) the number of vehicle pairs needed and (3) a set of combined routes to serve all demands while minimising the total cost. We have formulated the problem as an MIBP model and introduced several valid inequalities to strengthen the model. For large problem instances, we have developed a simple yet effective ALNS-based metaheuristic method in which a greedy heuristic is used to create an initial solution, followed by a set of destroy and repair operators that iteratively improve the initial solution to the best one. Moreover, we have implemented preprocessing procedures to accelerate the solution process, and designed feasibility test methods for trucks and drones. We have also developed an efficient algorithm to optimise the waiting time at launch nodes.

Various experiments have been conducted to validate the model and verify the effectiveness of the valid inequalities and the performance of ALNS. The benefits of drone adoption are clearly demonstrated. A number of comparative studies have been performed to illustrate the applicability of the proposed model and

methodology, and analyse the advantages and characteristics of mDPD-T. Finally, the sensitivity analysis has been used to quantitatively assess the impact of four key factors on problem solutions across various scenarios with valuable managerial insights.

Despite potentially a cost-effective delivery mode, the truck-drone combined service has not yet been widely adopted in last mile logistics. One of the hurdles could be the scheduling and operational challenges in synchronising trucks and their drones to achieve seamless services for customers. If not planned and implemented properly, the benefits of the combined mode might not be materialised. Moreover, due to the uncertainties and variabilities of the drone cost, the cost savings of the combined mode may not always be obvious. The very different drone regulations between regions/countries could be another barrier. Nevertheless, it is definitely worth an effort to properly understand the hurdles for wider adoption of the combined truck-drone mode.

As another future research direction, it would be interesting to study a hybrid mode, in which both the truck-drone combined mode and parallel mode (i.e., trucks and drones work in parallel) are allowed to serve customers. A mathematical program could be formulated to determine the best ways to combine these two modes. It would also be interesting to consider location decisions of the rendezvous stations, as they have profound impact on the overall cost of the combined mode. Finally, a further research topic would be the consideration of uncertainties, such as stochastic travel time or customer demands, to model more realistic applications.

Acknowledgments

This work is supported by the China Scholarship Council [grant no. 202207000043], the National Nature Science Foundation of China [grant no. 72371206] and Service Science and Innovation Key Laboratory of Sichuan Province of China [grant no. KL2106].

References

Dellaert, N., Dashty Saridarq, F., Van Woensel, T., & Crainic, T. G. (2019). Branch-and-price–based algorithms for the two-echelon vehicle routing problem with time windows. *Transportation Science*, 53(2), 463–479.

Di Puglia Pugliese, L. & Guerriero, F. (2017). Last-mile deliveries by using drones and classical vehicles, in *Optimization and Decision Science: Methodologies and Applications: ODS, Sorrento, Italy, September 4-7, 2017 47*, (pp. 557–565), Springer.

Dragomir, A. G., Van Woensel, T., & Doerner, K. F. (2022). The pickup and delivery problem with alternative locations and overlapping time windows. *Computers & Operations Research*, 143, 105758.

Dukkanci, O., Kara, B. Y., & Bektaş, T. (2021). Minimizing energy and cost in range-limited drone deliveries with speed optimization. *Transportation Research Part C: Emerging Technologies*, 125, 102985.

Gu, R., Poon, M., Luo, Z., Liu, Y., & Liu, Z. (2022). A hierarchical solution evaluation method and a hybrid algorithm for the vehicle routing problem with drones and multiple visits. *Transportation Research Part C: Emerging Technologies*, 141, 103733.

Jiang, J., Dai, Y., Yang, F., & Ma, Z. (2023). A multi-visit flexible-docking vehicle routing problem with drones for simultaneous pickup and delivery services. *European Journal of Operational Research*.

Karak, A. & Abdelghany, K. (2019). The hybrid vehicle-drone routing problem for pick-up and delivery services. *Transportation Research Part C: Emerging Technologies*, 102, 427–449.

Kitjacharoenchai, P., Min, B.-C., & Lee, S. (2020). Two echelon vehicle routing problem with drones in last mile delivery. *International Journal of Production Economics*, 225, 107598.

Kloster, K., Moeini, M., Vigo, D., & Wendt, O. (2023). The multiple traveling salesman problem in presence of drone-and robot-supported packet stations. *European Journal of Operational Research*, 305(2), 630–643.

Kuo, R., Lu, S.-H., Lai, P.-Y., & Mara, S. T. W. (2022). Vehicle routing problem with drones considering time windows. *Expert Systems with Applications*, 191, 116264.

Kyriakakis, N. A., Stamadianos, T., Marinaki, M., & Marinakis, Y. (2022). The electric vehicle routing problem with drones: An energy minimization approach for aerial deliveries. *Cleaner Logistics and Supply Chain*, 4, 100041.

Li, H., Wang, H., Chen, J., & Bai, M. (2020). Two-echelon vehicle routing problem with time windows and mobile satellites. *Transportation Research Part B: Methodological*, 138, 179–201.

Liu, Y., Liu, Z., Shi, J., Wu, G., & Pedrycz, W. (2020). Two-echelon routing problem for parcel delivery by cooperated truck and drone. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 51(12), 7450–7465.

Luo, Z., Gu, R., Poon, M., Liu, Z., & Lim, A. (2022). A last-mile drone-assisted one-to-one pickup and delivery problem with multi-visit drone trips. *Computers & Operations Research*, 148, 106015.

Masmoudi, M. A., Mancini, S., Baldacci, R., & Kuo, Y.-H. (2022). Vehicle routing problems with drones equipped with multi-package payload compartments. *Transportation Research Part E: Logistics and Transportation Review*, 164, 102757.

Meng, S., Guo, X., Li, D., & Liu, G. (2023). The multi-visit drone routing problem for pickup and delivery services. *Transportation Research Part E: Logistics and Transportation Review*, 169, 102990.

Najy, W., Archetti, C., & Diabat, A. (2023). Collaborative truck-and-drone delivery for inventory-routing problems. *Transportation Research Part C: Emerging Technologies*, 146, 103791.

Poljak, M. & Šterbenc, A. (2020). Use of drones in clinical microbiology and infectious diseases: current status, challenges and barriers. *Clinical Microbiology and Infection*, 26(4), 425–430.

Rave, A., Fontaine, P., & Kuhn, H. (2023). Drone location and vehicle fleet planning with trucks and aerial drones. *European Journal of Operational Research*, 308(1), 113–130.

Ropke, S. & Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4), 455–472.

Sacramento, D., Pisinger, D., & Ropke, S. (2019). An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones. *Transportation Research Part C: Emerging Technologies*, 102, 289–315.

Salama, M. & Srinivas, S. (2020). Joint optimization of customer location clustering and drone-based routing for last-mile deliveries. *Transportation Research Part C: Emerging Technologies*, 114, 620–642.

Schermer, D., Moeini, M., & Wendt, O. (2019). A hybrid VNS/Tabu search algorithm for solving the vehicle routing problem with drones and en route operations. *Computers & Operations Research*, 109, 134–158.

Shen, T. (2022). Shenzhen's low-altitude economy speeds up, "3 kilometers and 15 minutes" drone service enriches urban retail distribution (in Chinese), URL `https://baijiahao.baidu.com/s?id=1742909245657962155&wfr=spider&for=pc/`.

Tamke, F. & Buscher, U. (2021). A branch-and-cut algorithm for the vehicle routing problem with drones. *Transportation Research Part B: Methodological*, 144, 174–203.

Torabbeigi, M., Lim, G. J., & Kim, S. J. (2020). Drone delivery scheduling optimization considering payload-induced battery consumption rates. *Journal of Intelligent & Robotic Systems*, 97(3), 471–487.

Wang, X., Poikonen, S., & Golden, B. (2017). The vehicle routing problem with drones: several worst-case results. *Optimization Letters*, 11(4), 679–697.

Wang, Y., Peng, S., Zhou, X., Mahmoudi, M., & Zhen, L. (2020a). Green logistics location-routing problem with eco-packages. *Transportation Research Part E: Logistics and Transportation Review*, 143, 102118.

Wang, Y., Wang, Z., Hu, X., Xue, G., & Guan, X. (2022). Truck-drone hybrid routing problem with time-dependent road travel time. *Transportation Research Part C: Emerging Technologies*, 144, 103901.

Wang, Y., Yuan, Y., Guan, X., Xu, M., Wang, L., Wang, H., & Liu, Y. (2020b). Collaborative two-echelon multicenter vehicle routing optimization based on state–space–time network representation. *Journal of Cleaner Production*, 258, 120590.

Wang, Z. & Sheu, J.-B. (2019). Vehicle routing problem with drones. *Transportation Research Part B: Methodological*, 122, 350–364.

Wolfinger, D. (2021). A large neighborhood search for the pickup and delivery problem with time windows, split loads and transshipments. *Computers & Operations Research*, 126, 105110.

Yang, Y., Yan, C., Cao, Y., & Roberti, R. (2023). Planning robust drone-truck delivery routes under road traffic uncertainty. *European Journal of Operational Research*, 309(3), 1145–1160.

Yin, Y., Li, D., Wang, D., Ignatius, J., Cheng, T., & Wang, S. (2023). A branch-and-price-and-cut algorithm for the truck-based drone delivery routing problem with time windows. *European Journal of Operational Research*, 309(3), 1125–1144.

Yu, S., Puchinger, J., & Sun, S. (2022). Van-based robot hybrid pickup and delivery routing problem. *European Journal of Operational Research*, 298(3), 894–914.

Zhen, L., Gao, J., Tan, Z., Wang, S., & Baldacci, R. (2023). Branch-price-and-cut for trucks and drones cooperative delivery. *IISE Transactions*, 55(3), 271–287.