Pickup and delivery problem with multi-visit drones considering soft time windows

Abstract

We extend the pickup and delivery problem with combined truck–drone operation by assuming that a fleet of trucks can stop at customer nodes and launch drones to perform multiple pickup and delivery services with soft time windows. Unlike other studied pickup and delivery problems, one-to-one pickup and delivery services (in which the pickup and delivery requests have a one-to-one relationship), which are typical in instant retail and food delivery, are considered. We mathematically model a mixed-integer nonlinear program and introduce strengthening strategies to capture this scenario, with the objective of minimising the total cost, including the penalty cost of time window violations. We approximate the nonlinear power of hovering drones with a piecewise linear method and propose an efficient metaheuristic approach, along with truck waiting time optimisation, to solve large-size problems. Finally, comprehensive computational experiments are conducted, which demonstrate the applicability of the algorithm and the impacts of different configurations. The numerical results indicate the efficiency of our proposed model and the solution approach, demonstrating the potential operational gain obtained by implementing the combined system. The cost savings rate compared to the truck-only mode is more than 40% on average, and our algorithm outperforms the benchmark algorithms in the literature by more than 10% in terms of solution quality.

Keywords: Vehicle routing problem with drones; Pickup and delivery; Soft time windows; Mixed integer nonlinear program; Metaheuristic

1. Introduction

Drones, generally known as unmanned aerial vehicles, have recently displayed considerable potential in terms of customer service efficiency for last-mile logistics. In recent years, a growing number of companies have begun more widespread drone deliveries. For example, Zipline, an autonomous instant delivery company, has used drones to deliver medical supplies to outposts in Rwanda and Ghana since 2016 and is rapidly expanding its project in the U.S. (Toor, 2016). Wing, a subsidiary of Alphabet, has completed more than 350,000 drone deliveries (Moon, 2021). Drones have several specific advantages over traditional vehicles, such as lower levels of pollutant emissions, faster speeds, and lower costs (Tamke & Buscher, 2021), thereby contributing to easing traffic congestion and developing sustainable transportation systems. However, drones are generally battery-operated and have technical shortcomings. Therefore, the collaborative truck-drone paradigm avoids the above disadvantages and is an attractive solution to the delivery process. Under this paradigm, trucks can still perform delivery/pickup tasks but can also stop at customer locations or other specific points to operate drones to serve other customers in parallel. In this way, a drone can depart from a truck route, make a short detour (flight) to serve customers within its capacity constraints, and then dynamically rendezvous with the truck further down the route. Moreover, multiple flights can be performed along the route by replacing batteries when the drone is retrieved.

Pioneering works on combined truck–drone systems were carried out by Murray & Chu (2015) and Agatz et al. (2018). They introduced the flying-sidekick travelling salesman problem (FSTSP) and travelling salesman problem with drone (TSPD) to minimise the completion time. In these problems, a truck and a drone work collaboratively (as a vehicle pair) to make deliveries, and the drone can only deliver one parcel in a flight (single visit). Additionally, Wang et al. (2017) considered multi-truck usage in a combined system, namely, the vehicle routing problem with drones (VRPD), in which a set of trucks, each with several single-visit drones on board, collaboratively serve customers. Many works have extended the above problems by adding more complicated features, such as multiple drone visits, that is, drones can visit multiple customers in a flight (Meng et al., 2023). However, no studies have explicitly discussed the effect of soft time windows on the pickup and delivery services in the multi-visit VRPD context.

In addition, drone pickups are gradually becoming more common. For example, Wing recently created a new system that enables drones to autonomously pick up packages and perform pickup-delivery jobs without having to immediately return to the origin point (Shakir, 2023). As another example, Zipline uses drones in food delivery, offering error-proof loading and pickup services between customers and restaurants in under 10 minutes without traffic, stoplights, or detours (Zipline, 2025).

Moreover, continuous breakthroughs in drone design and battery technology have promoted drone adoption, thus transcending traditional logistical limits to provide safe, economical and efficient air transport solutions. For example, DJI introduced a long-distance heavy lifter drone for delivery, named FlyCart30, which has a 30 kg maximum payload, a 28 km flight distance without a payload, and a 16 km flight distance with a full payload (DJI, 2025). Technologies that support multiple drone visits have also emerged. For example, the Vulcan UAV Airlift is equipped with a multicompartment payload, allowing for multiple deliveries per flight (Masmoudi et al., 2022).

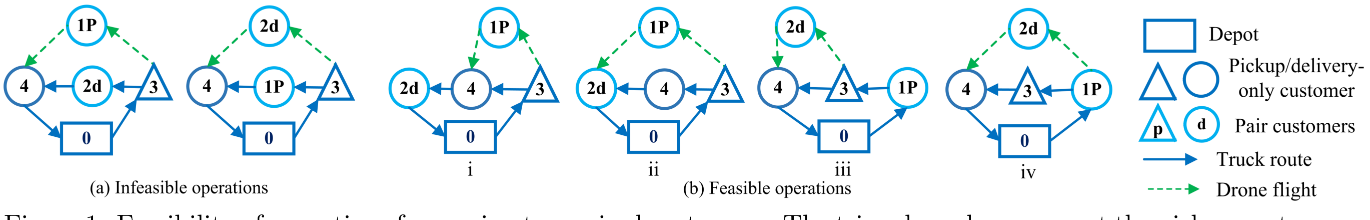
The assumption of soft time windows provides a further extension of the case of no time windows or hard time windows, providing a more universal problem scenario that can effectively balance customer time window requirements and logistics costs. The soft time window allows for early or late delivery of packages, which is favoured in various scenarios with temporal elasticity. For example, general express delivery usually promises to be completed within 3-5 days, without being limited to a specific time so that deliveries can be flexibly scheduled. Community group purchases are usually delivered within a relaxed time slot on a given day, allowing a community to be served centrally and thus improving efficiency. This flexibility simplifies delivery and reduces the high costs associated with hard time window scenarios.

When soft time windows are used, optimising the truck waiting time at each node is important for determining the service start time and performing route selection. In addition, the truck waiting time at each node is very difficult to determine and differs significantly from that in scenarios involving hard time windows. In the latter case, the waiting time is simply the difference between the arrival time and the lower-bound time window if the truck arrives early.

Our study introduces an innovative combined pickup and delivery problem characterised by the multi-visit capability of drones and soft time windows; this is called the pickup and delivery problem with multi-visit drones considering soft time windows (PDmD-sT). The optimisation objective is to minimise the total cost, measured by the sum of the fixed cost, operational cost, and penalty cost due to time window violation.

Our work differs from previous works in the following ways: (i) We investigate a variant of the pickup and delivery problem in which collaborative drones with multi-visit capabilities can be launched from the depot or any customer location; (ii) We introduce a more realistic energy consumption model for drones, in which the power consumption during hovering is nonlinear; and (iii) Each customer has a soft time window, and if the service starts outside this window, a penalty will be incurred.

In addition, three types of services are provided: pickup-only, delivery-only, and one-to-one pickup and delivery (paired service). Each pair of customers contains a pickup pair and a delivery pair, and they have a one-to-one correspondence (Meng et al., 2024a). Therefore, two paired customers cannot be served separately by two parallel sub-routes, as shown in Figure 1(a). That is, at least one retrieval or launch node is visited between them (Figures 1(b)i and 1(b)iii), or one of them is a retrieval or launch node (Figures 1(b)ii and 1(b)iv).



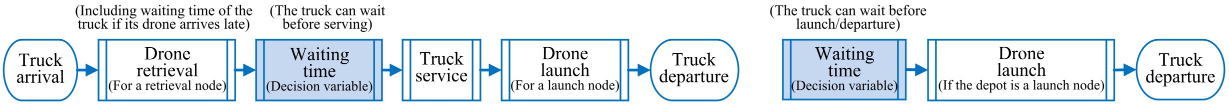
(a) Infeasible operations

(b) Feasible operations

Figure 1: Feasibility of operations for serving two paired customers. The triangle nodes represent the pickup customers, and the circle nodes represent the delivery customers. Nodes with indexes ending with "p" and "d" denote the pickup pair and delivery pair customers, respectively; otherwise, nodes are pickup-only or delivery-only customers. The launch node represents a node from which a truck and its drone depart separately. The retrieval node represents a node at which a truck and its drone arrive separately.

Furthermore, in some cases, a truck cannot serve a customer before retrieving the drone (as shown in Figure 1 (b)ii) and cannot launch the drone before performing a service (as shown in Figure 1 (b)iv)). Therefore, we define the truck service process at each node as shown in Figure 2; this process also contributes to reducing the possible hover time of drones before they are retrieved. Note that we allow trucks to wait at each node to reduce penalties.

In addition, in previous related studies, the duration of drone flying was generally assumed to be fixed and measured by a constant distance or time. However, it is actually influenced by several critical



(a) For a customer node (b) For the depot

Figure 2: The service process of trucks at each node.

factors, such as payload (Dorling et al., 2016), and is sensitive to different operations. For example, power consumption during hovering usually has a nonlinear relationship with the payload and differs from that during flying (Cheng et al., 2020). Moreover, both the payload and hover duration at a retrieval node depend on routing variables in PDmD-sT and are unknown a priori, resulting in rather complicated calculations.

As a result, PDmD-sT routes are asymmetric. An example is presented in Figure 3, where customers 1 and 2 have pickup and delivery demands of 2 kg each and the drone can carry up to 3 kg. For the feasible route shown in Figure 3a, the reverse route (Figure 3b) requires more energy and may even be infeasible due to overload. However, if the two customers are paired, the drone must serve customer 1 first. As shown in Figure 3c, the same route becomes feasible, and the drone is launched without a payload.

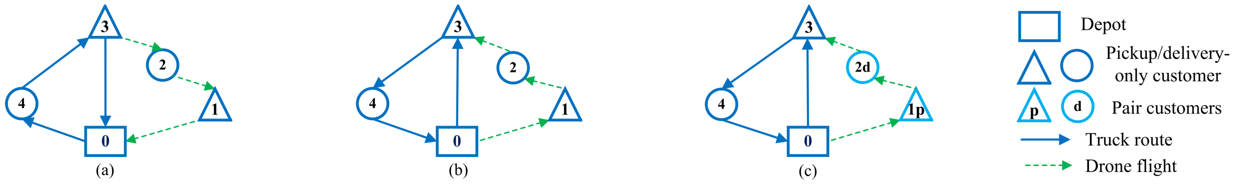


Figure 3: Routes of the same length in different directions and with different types of services.

Recently, Luo et al. (2022) studied a combined one-to-one food pickup and delivery problem. However, delivery-only/pickup-only services and soft time windows were not considered. In response, Meng et al. (2024a) proposed a novel combined pickup and delivery problem with hard time windows. They considered three types of services and provided a scenario in which a set of available stations is dedicated to drone operations. In contrast, we apply a more realistic drone energy consumption model and aim to satisfy time window requirements flexibly. Moreover, we focus on integrating drone operations with customer services, which is rather complicated, as the timing of various operations needs to be coordinated and any adjustments to visits may change drone operations.

The main contributions of our work are threefold. (i) A new VRPD variant, which includes pickup and delivery and encourages multiple drone visits, is derived from practical considerations. We introduce a complex mixed-integer nonlinear program (MINLP). Then, a piecewise linear approach is incorporated to linearise the power calculation, and various valid inequalities are developed to reduce the solution space so that the MINLP can be converted to a mixed-integer bilinear program (MIBLP) and therefore can be solved with software solvers. (ii) As it is more complicated than the already NP-hard vehicle routing problem (VRP), we provide an improved large neighbourhood search (ILNS) method to solve the problem. Moreover, a model with enhanced cuts is developed to determine truck waiting times. (iii) We conduct extensive experiments to assess the efficacy of the introduced ILNS as well as the benefits of drone use, and we analyse the impacts of key factors on the outputs. The results demonstrate the advantageous performance of our algorithm and show the benefits of the combined mode and consideration of truck waiting times.

The remainder of this work is organised as follows. Section 2 reviews related studies. In Section 3, the problem is described, the mathematical model is introduced, and various valid inequalities are provided. The components of the solution approaches are detailed in Section 4, and the computational results and comparisons are presented in Section 5. Finally, Section 6 presents the conclusions and suggests future research.

2. Literature review

In recent years, truck–drone routing and scheduling have received extensive attention in academia because of their promising benefits in terms of cost savings and efficiency improvement. In this section, we briefly review the most closely related works on (1) the basic VRPD and its extensions, (2) the consideration of pickup and delivery, and (3) the TSPD/VRPD with time windows.

2.1. Basic VRPD and its extensions

As mentioned above, Wang et al. (2017) first proposed the VRPD, in which a set of trucks and drones with single-visit capability cooperate to make deliveries in the minimum completion time. To solve this problem, some works introduce exact solution methods (Tamke & Buscher, 2021; Zhen et al., 2023). For example, Zhou et al. (2023) assumed that the trucks must wait at the launch nodes while their paired drones conduct multiple round-trip flights and proposed a branch-and-price (BP) algorithm to minimise travel and waiting times. In addition, many studies have explored heuristic methods. Some researchers, such as Sacramento et al. (2019) and Huang et al. (2022), optimised the basic VRPD with different objectives, and subsequent works included more complicated factors. For example, Rave et al. (2023) considered a more flexible case in which trucks can operate drones at customer locations or microdepots to minimise the total cost. Their proposed metaheuristics can solve problems with 200 customers.

Considering the benefits of multiple drone visits, Wang & Sheu (2019) first introduced a multi-visit VRPD, in which some docks act as transfer stations for drone launch and retrieval. They also developed a BP algorithm for total cost minimisation. Kitjacharoenchai et al. (2020) developed efficient heuristics to solve a multi-visit VRPD, which allows one truck to launch multiple drones to reduce the total arrival time at the depot. Later, Kyriakakis et al. (2022) studied an electric VRPD with the objective of minimising the energy consumed by both types of vehicles. They assumed that drones operated at a set of launch stations and introduced four ant colony-based algorithms to solve problems with 50 customers and 4 launch stations. A recent work by Meng et al. (2024b) investigated a novel combined delivery problem that considers both stochastic truck travel times and soft time windows. They formulated a two-stage stochastic model for the problem and developed a metaheuristic and an implementable method for problem solving and waiting time execution to minimise the total costs of stochastic problems with 100 customers.

2.2. Consideration of pickup and delivery

In recent years, a few works have investigated the pickup and delivery problem in the multi-visit TSPD/VRPD context. Karak & Abdelghany (2019) provided a hybrid pickup and delivery routing problem in which a single truck travels among a set of available stations to launch drones for all deliveries with the minimum travel cost. They improved the Clarke & Wright algorithm for problems with 100 customers. Luo et al. (2022) studied a one-to-one pickup and delivery problem in which multiple vehicle pairs are dedicated to serving customers, with one-to-one correspondence. They proposed an iterated local search method for total cost optimisation. Gu et al. (2023) introduced a new variant of VRPD for on-demand services, called the dynamic combined routing problem, which considers an uncertain set of pickup requests with a final deadline due to a prespecified working time. This problem was defined as a Markov decision process and optimised by a segment-based heuristic for profit maximisation. Meng et al. (2023) considered a combined pickup and delivery scenario with multi-visit drones and proposed a two-stage heuristic that can minimise the total costs of instances with 100 nodes. Jiang et al. (2024) considered a more flexible docking approach in which drones can be retrieved by any trucks to minimise travel costs. Then, Meng et al. (2024a) provided a more complicated pickup and delivery system, in which a set of vehicle pairs assisted by available stations provides three types of services within the time windows. More recently, Yin et al. (2024) investigated a combined pickup and delivery problem in which each node has a set of time windows. Each time window has an availability probability, allowing the service to be carried out within the time window. They assumed that trucks must wait at the launch nodes while their drones perform services and provided a branch-and-price-and-cut (BPC) algorithm to minimise both the operational cost and the expected service failure cost.

In addition, variants of the large neighbourhood search (LNS) metaheuristic have been used in the traditional pickup and delivery literature. For example, Wang et al. (2024) proposed an electric vehicle routing problem with pickup and delivery, which incorporates both multiple trips and vehicle heterogeneity to minimise the total cost. In addition, they introduced and compared two algorithms, the variable neighbourhood search (VNS) and the adaptive large neighbourhood search (ALNS), which can solve tasks involving 1500 customers and 100 charging stations. Similarly, Zhou et al. (2024) considered a variant of the pickup and delivery problem with electric vehicles and time windows. They also developed an efficient ALNS approach with a queuing scheduling program to minimise the total travel and queue time costs. Zhang et al. (2024)

investigated a pickup and delivery problem with time windows for airport baggage transit considering split demand and multiple trips per vehicle. They first used topological sorting to determine flight service times and then developed an ALNS approach to determine the best route. Moreover, Lehmann & Winkenbach (2024) extended the two-echelon pickup and delivery problem with time windows and multiple trips. They deployed both exact formulation and ALNS methods to schedule two-echelon routing. Li et al. (2025) presented a similar problem for agricultural wholesalers and developed a combined ANLS approach that integrates a branch-and-bound algorithm.

2.3. TSPD/VRPD with time windows

Given realistic requirements, the VRPD with time windows is a natural extension of the above problems. Some researchers, such as Coindreau et al. (2021) and Kuo et al. (2022), have examined such problems. These authors provided heuristic methods for basic VRPD cases with hard time windows. Some researchers have since explored more complicated cases. For example, Yang et al. (2023) studied a combined delivery problem with uncertain truck travel times in the context of the TSPD. They developed a BP algorithm to maximise profit while serving customers with hard time windows.

In the above studies, it was assumed that drones can serve only one customer per flight. In response, Li et al. (2020) studied a combined routing problem considering both multi-visit capabilities for drones and hard time windows for customers. They developed an ALNS approach to minimise the operational cost. Later, Yin et al. (2023) provided an arc-based formulation of this problem and introduced a BPC algorithm to minimise the operational costs of problems with 45 customers in 3 h.

Overall, these findings indicate that new VRPD variants should be developed as technology advances. The majority of previous VRPD studies addressed delivery-only issues, and the small number of works involving pickup and delivery services rarely considered customer time windows. To our knowledge, only Meng et al. (2024a) and Yin et al. (2024) introduced VRPD schemes that incorporate time windows and both pickup and delivery. However, as highlighted above, our PDmD-sT expands upon their research by considering soft time windows to support more universal applications and practically considers drone energy consumption. In Table 1, we provide a brief summary of related studies.

Table 1: Summary of the most related studies.

Reference	MV	TW	WT	VE			LL		ST	OB			
				FE	SE	HE	CL	SL	P/D/O	MC	OC	FC	PC
Wang et al. (2017); Tamke & Buscher (2021)							√		D	√			
Sacramento et al. (2019)							√		D		√		
Huang et al. (2022)							√		D		√	√	
Zhen et al. (2023)				√			√		D		√	√	
Zhou et al. (2023)				√			√		D	√			
Kloster et al. (2023)								√	D	√			
Rave et al. (2023)							√	√	D		√	√	
Wang & Sheu (2019)	√							√	D		√	√	
Kitjacharoenchai et al. (2020)	√						√		D	√			
Kyriakakis et al. (2022)	√			√				√	D		√		
Karak & Abdelghany (2019)	√							√	P+D		√		
Luo et al. (2022)	√			√	√	√	√		O	√		√	
Gu et al. (2023)	√						√		P+D		√*		
Meng et al. (2023)	√			√	√	√	√		P+D		√	√	
Jiang et al. (2024)	√						√		P+D		√		
Kuo et al. (2022)		hard					√		D		√		
Coindreau et al. (2021)		hard					√		D		√	√	
Yang et al. (2023)		hard					√		D		√*		
Li et al. (2020)	√	hard					√		D		√		
Yin et al. (2023)	√	hard					√		D		√	√	
Meng et al. (2024a)	√	hard	√	√	√	√		√	P+D+O		√	√	
Yin et al. (2024)	√	hard					√		P+D		√		√
This paper	√	soft	√	√	√	√	√		P+D+O		√	√	√

MV: multiple visits of drones. TW: time window. WT: waiting time optimisation. VE: variable endurance. FE: flying energy consumption. SE: serving energy consumption. HE: hovering energy consumption. LL: launch location. CL: customer location. SL: station location. ST: service types. P: pickup only. D: delivery only. O: one-to-one pickup and delivery. OB: objective. MC: makespan/duration. OC: operational cost. FC: fixed cost. PC: penalty cost. *: profit maximisation.

3. Problem description and model

3.1. *Problem description*

The PDmD-sT concerns a fleet of homogeneous vehicle pairs, which are available to serve all customers who have pickup-only, delivery-only or one-to-one pickup and delivery demands. All vehicle pairs start and return to a single depot at most once. Each customer is served exactly once by a truck or a drone, whereas some customers must be assigned to trucks because their parcels cannot be carried by drones (referred to as heavy parcels). Moreover, each customer has a soft time window that allows for early and late services, but a penalty cost is occurred. The load capacity of the trucks is not bounded, as the truck capacity is rarely a restrictive constraint in the delivery of light parcels (the mass is less than the payload of drones) (Tamke & Buscher, 2021). Considering practical cases, we assume that each route is subject to a maximum working time, within which all vehicles are allowed to return to the depot. Each drone can be dispatched multiple times from different customer locations along a truck route and provides multiple services per flight within its load and battery capacities. When a vehicle pair reconvenes, the vehicles must wait for each other. Moreover, if the drone arrives early, it consumes additional energy waiting and must be retrieved before its battery runs out of power. A drone can be operated at most once at each node, and the battery is replaced once the drone is retrieved. Considering real road networks, the distances travelled by trucks are calculated using the Manhattan metric, whereas the Euclidean metric is used for drones, as drones fly approximately in a straight line (Yang et al., 2023).

The objective is to find the optimal routes/flights to minimise the total cost, which is defined as the sum of the fixed cost of the employed vehicles, the operational cost and the penalty cost due to time window violations. The operational costs for trucks and drones are proportional to the travel distance and energy consumption corresponding to the different operations. An illustrative example is given in Figure 4.

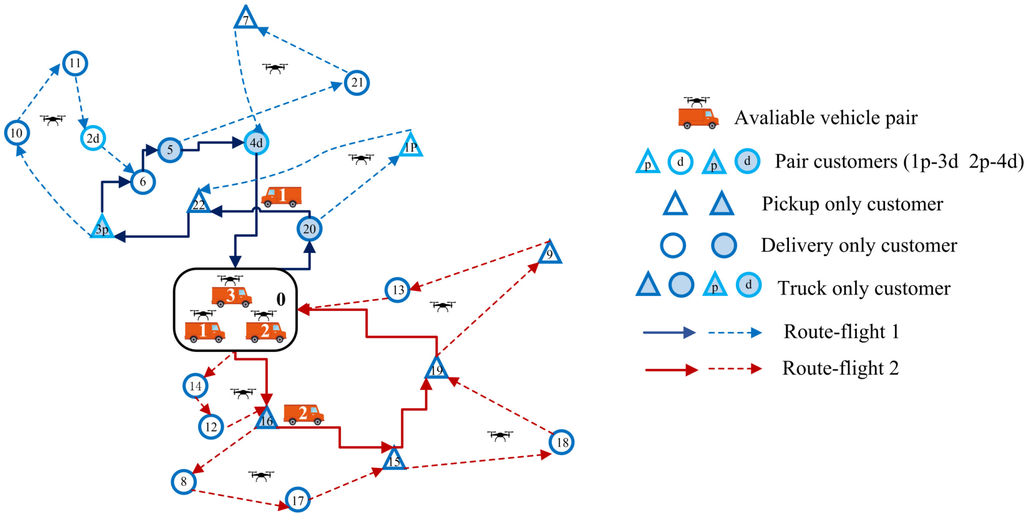


Figure 4: Illustrative example of PDmD-sT.

Overall, the following assumptions are considered for practical applications:

- Multiple homogeneous vehicle pairs are available to serve all customers, each of whom has a pickup-only, delivery-only or one-to-one pickup or delivery request.
- Two paired requests are served by the same vehicle pair, and the pickup request should be served before the counterpart request.
- Some parcels that cannot be transported by drones are heavy parcels and must be transported by trucks.
- Each customer has a soft time window that allows for early and late services but with penalties imposed.
- For each route, the load capacity of trucks is not limited, but vehicles are subject to a maximum working time.
- Each drone can be dispatched multiple times and make multiple visits in a flight within its load and battery capacities.
- Paired vehicles must wait for each other when reconvening. Therefore, a drone must spend extra

energy hovering if it arrives early and must be retrieved before its battery runs out.
- The travel distances of trucks and drones are calculated using the Manhattan and Euclidean metrics, respectively.
- Trucks are allowed to wait before launching drones to minimise possible penalties and hovering energy.

3.2. Notation

Let $G(N, A)$ be a directed graph for the PDmD-sT, where N and A denote the node and arc sets, respectively. $N = \{0\} \cup N_c \cup \{n+1\}$, where $N_c = \{1, 2, ..., n\} = N_p \cup N_d \cup N_{po} \cup N_{do}$ is the set of customers and 0 and $n+1$ correspond to the single depot, denoting the start and end of a tour, respectively. $N_p = \{1, ..., m\}$ and $N_d = \{m+1, ..., 2m\}$ are the sets of pickup-pair and delivery-pair customers, respectively, where m denotes the number of pairs of customers. N_{po} and N_{do} denote the pickup-only and delivery-only customer sets, respectively. Moreover, we define $N_s = \{0\} \cup N_c$ and $N_e = N_c \cup \{n+1\}$ as the sets of outbound and inbound arcs, N_{cd} as the set of customers whose demands can be carried by drones, and K as the vehicle pair set. $[0, l_{max}]$ denotes the depot's time window, which defines the latest return time for all vehicles. The superscript *hat* indicates drones. For clarity, we list the sets and parameters and the variables used in the model in Tables 2 and 3, respectively. The units for distance, weight, cost, time, speed, energy and power are km, kg, \$, min, kg/h, Wh and W, respectively.

Table 2: Sets and parameters used in the model.

Indexes	
i, j, p	Node (customer/depot) index
k	Vehicle pair index
Sets	
N_p	Pickup-pair customer set, $N_p = \{1, ..., m\}$
N_d	Delivery-pair customer set, $N_d = \{m+1, ..., 2m\}$
N_{po}	Pickup-only customer set
N_{do}	Delivery-only customer set
N_{cd}	Set of customers whose parcels can be carried by drones
N_c	Customer set, $N_c = N_p \cup N_d \cup N_{po} \cup N_{do}$
N	All-node set, $N = \{0\} \cup N_c \cup \{n+1\}$
N_s	Outbound set of all arcs, $N_s = \{0\} \cup N_c$
N_e	Inbound set of all arcs, $N_e = N_c \cup \{n+1\}$
A	All-arc set, $A = \{(i, j) \| i \in N_s, j \in N_e, i \neq j\}$
K	Vehicle pair set
Parameters	
c^s	Penalty cost per minute when service is started before the left (lower bound) time window
c^e	Penalty cost per minute when service is started after the right (upper bound) time window
Q	Drone load capacity
E	Drone battery capacity
$P(w)$	Power of drones while hovering with a payload of w
w_0	Cub weight of drones, including the mass of the drone and battery
η	Drone energy consumption per kilogram per minute
d_i	Demand of customer i, where $d_i > 0$ and $d_i < 0$ denote that i is a delivery customer and a pickup customer, respectively
c^0	Fixed cost of using a truck–drone pair
v ($\hat{v}$)	Truck (drone) speed
c	Truck travel cost per kilometer
$\hat{c}$	Drone cost per unit Watt hour
r_{ij} ($\hat{r}_{ij}$)	Travel distance of trucks (drones) from node i to j
τ_i ($\hat{\tau}_i$)	Duration required for trucks (drones) to serve customer i
τ^l (τ^r)	Duration required to launch (retrieve) drone
$[s_i, \tilde{s}_i]$	Soft time window of node i
M	A sufficiently large integer

Note that the variable $Z^k_{ij,p}$ is used to avoid infeasible operations in serving two paired customers (as shown in Figure 1a); the variable $S^k_{ij,p}$ indicates whether two paired customers are served by the same flight, as the payload variations in these two cases are different, as mentioned above.

3.3. Model

We formulate PDmD-sT as a mixed-integer nonlinear program (MINLP). Except for the constraints on hover energy consumed by drones, which are nonlinear, the remaining components of the MINLP are linear. Section 3.3.1 presents the objective function, and Section 3.3.2 provides the routing constraints. Section 3.3.3 introduces the timing constraints, and the constraints on drone capacities related to energy and load are provided in Sections 3.3.4 and 3.3.5, respectively. Finally, we linearise the power of the drones while hovering in Section 3.3.6.

Table 3: Variables used in the model.

Decision Variables	
$x_{ij}^{0k} \in \{0,1\}$	$x_{ij}^{0k}=1$ denotes that vehicle pair k travels from node i to j; $x_{ij}^{0k}=0$ otherwise
$x_{ij}^{k} \in \{0,1\}$	$x_{ij}^{k}=1$ denotes that truck k travels from node i to j by itself; $x_{ij}^{k}=0$ otherwise
$\hat{x}_{ij}^{k} \in \{0,1\}$	$\hat{x}_{ij}^{k}=1$ denotes that drone k flies from node i to j by itself; $\hat{x}_{ij}^{k}=0$ otherwise
$\theta_i^k \in [0, l_{max}]$	Duration required for truck k to wait at node i
Auxiliary Variables	
$z_i^k \in \{0,1\}$	$z_i^k=1$ denotes that truck k serves customer i; $z_i^k=0$ otherwise
$\hat{z}_i^k \in \{0,1\}$	$\hat{z}_i^k=1$ denotes that drone k serves customer i; $\hat{z}_i^k=0$ otherwise
$\epsilon_i \in [0, l_{max}]$	Time at which the service of customer i starts
$a_i^k \in [0, l_{max}]$	Time at which truck k arrives at $i \in N_e$
$\hat{a}_i^k \in [0, l_{max}]$	Time at which drone k arrives at $i \in N_e$
$w_i^k \in [0, Q]$	Load carried by drone k when leaving node i
$w_i^{d,k} \in [0, Q]$	Delivery quantity that drone k carries when leaving node i
$e_i^k \in [0, E]$	Accumulated energy consumed by drone k when leaving node i in a flight
$t_i^{h,k} \in [0, E/P(0)]$	Duration of drone k hovering at node $i \in N_c$
$e_{ij}^{f,k} \in [0, E]$	Energy consumed by drone k in traveling from node i to j
$e_i^{s,k} \in [0, E]$	Energy consumed by drone k in serving node i
$e_i^{h,k} \in [0, E]$	Energy consumed by drone k in hovering at node i
$y_i^k \in \{0,1\}$	$y_i^k=1$ denotes that node i is used for launching drone k; $y_i^k=0$ otherwise
$\tilde{y}_i^k \in \{0,1\}$	$\tilde{y}_i^k=1$ denotes that node i is used for retrieving drone k; $\tilde{y}_i^k=0$ otherwise
$y_{i,j}^k \in \{0,1\}$	$y_{i,j}^k=1$ denotes that a flight of drone k starts and ends at nodes i and j, respectively; $y_{i,j}^k=0$ otherwise
$Z_{ij,p}^k \in \{0,1\}$	$Z_{ij,p}^k=1$ denotes that drone k serves customer p with a flight starting and ending at nodes i and j, respectively; $Z_{ij,p}^k=0$ otherwise
$S_{ij,p}^k \in \{0,1\}$	$S_{ij,p}^k=1$ denotes that two paired customers p and $p+m$ are served by the same flight that starts and ends at nodes i and j; $S_{ij,p}^k=0$ otherwise
$U^k \in \{0,1\}$	$U^k=1$ denotes that vehicle pair k is used; $U^k=0$ otherwise
$b_{i,j}^k \in \{0,1\}$	$b_{i,j}^k=1$ denotes that node i is visited before node j by truck k; $b_{i,j}^k=0$ otherwise
$u_i^k \in \mathbb{Z}^+$	Relative location of node i in the visiting sequence of truck k
$\delta_i^- \in [0, l_{max}]$	Duration of the start time for serving node i before its left time window
$\delta_i^+ \in [0, l_{max}]$	Duration of the start time for serving node i after its right time window
$\bar{a}_i^k \in [0, l_{max}]$	Time at which both truck k and its drone arrive at node i, namely, $max\{a_i^k, \hat{a}_i^k\}$
$\alpha_i \in \{0,1\}$	$\alpha_i=1$ denotes that the service for node i starts before its left time window; $\alpha_i=0$ otherwise
$\beta_i \in \{0,1\}$	$\beta_i=1$ denotes that the service for node i starts after its right time window; $\beta_i=0$ otherwise
$\gamma_i \in \{0,1\}$	$\gamma_i=1$ denotes that drone k arrives at node i earlier than its truck; $\gamma_i=0$ otherwise

3.3.1. Objective function

$$min \quad c\sum_{k\in K}\sum_{(i,j)\in A} r_{ij}\cdot(x_{ij}^{0k}+x_{ij}^{k}) + \hat{c}\sum_{k\in K}(\sum_{(i,j)\in A} e_{ij}^{f,k} + \sum_{i\in N_{cd}} e_i^{s,k} + \sum_{i\in N_c} e_i^{h,k}) + \sum_{i\in N_c}(c^s\delta_i^- + c^e\delta_i^+) + c^0\sum_{k\in K}U^k \tag{1}$$

In objective function (1), the four terms define the truck travel cost, drone energy cost, penalty cost due to time window violations, and fixed cost for vehicle employment, respectively.

3.3.2. Routing constraints

$$\sum_{i\in N_e}(x_{0i}^{0k}+x_{0i}^{k}) = \sum_{i\in N_s}(x_{i,n+1}^{0k}+x_{i,n+1}^{k}) = 1, k\in K \tag{2a}$$

$$\sum_{i\in N_e}(x_{0i}^{0k}+\hat{x}_{0i}^{k}) = \sum_{i\in N_s}(x_{i,n+1}^{0k}+\hat{x}_{i,n+1}^{k}) = 1, k\in K \tag{2b}$$

$$\sum_{i\in N_s}(x_{ij}^{0k}+x_{ij}^{k}) = \sum_{i\in N_e}(x_{ji}^{0k}+x_{ji}^{k}) = z_j^k, j\in N_c, k\in K \tag{3a}$$

$$\sum_{i\in N_s}(x_{ij}^{0k}+\hat{x}_{ij}^{k}) = \sum_{i\in N_e}(x_{ji}^{0k}+\hat{x}_{ji}^{k}), j\in N_c, k\in K \tag{3b}$$

$$2*\hat{z}_j^k \leq \sum_{i\in N_s}\hat{x}_{ij}^k + \sum_{i\in N_e}\hat{x}_{ji}^k, j\in N_{cd}, k\in K \tag{3c}$$

$$\sum_{k\in K} z_i^k + \sum_{k\in K}\hat{z}_i^k = 1, i\in N_c \tag{4a}$$

$$\sum_{k\in K} z_i^k = 1, i\in N_c\setminus N_{cd} \tag{4b}$$

$$z_i^k + \hat{z}_i^k = z_{i+m}^k + \hat{z}_{i+m}^k \leq 1, i \in N_p, k \in K \tag{5}$$

$$\sum_{j \in N_e} y_{i,j}^k = y_i^k \geq \sum_{j \in N_e} x_{ij}^k + \sum_{j \in N_{cd}} \hat{x}_{ij}^k - 1, i \in N_s, k \in K \tag{6a}$$

$$y_i^k \leq \sum_{j \in N_e} x_{ij}^k, y_i^k \leq \sum_{j \in N_{cd}} \hat{x}_{ij}^k, i \in N_s, k \in K \tag{6b}$$

$$\sum_{i \in N_s} y_{i,j}^k = \tilde{y}_j^k \geq \sum_{i \in N_s} x_{ij}^k + \sum_{i \in N_{cd}} \hat{x}_{ij}^k - 1, j \in N_e, k \in K \tag{7a}$$

$$\tilde{y}_j^k \leq \sum_{i \in N_s} x_{ij}^k, \tilde{y}_j^k \leq \sum_{i \in N_{cd}} \hat{x}_{ij}^k, j \in N_e, k \in K \tag{7b}$$

$$\sum_{k \in K} \sum_{j \in N_e} \hat{x}_{ij}^k \leq 1; \sum_{k \in K} \sum_{j \in N_s} \hat{x}_{ji}^k \leq 1, i \in N_c \tag{8a}$$

$$\sum_{j \in N_e} \hat{x}_{ij}^k \leq \sum_{j \in N_s} \hat{x}_{ji}^k + \sum_{j \in N_s} x_{ji}^{0k}, i \in N_c, k \in K \tag{8b}$$

$$\sum_{j \in N_e} x_{ij}^k \geq z_i^k + \sum_{j \in N_{cd}} \hat{x}_{ij}^k - 1, i \in N_c, k \in K \tag{8c}$$

$$\sum_{i \in N_s} x_{ij}^k \geq z_j^k + \sum_{i \in N_{cd}} \hat{x}_{ij}^k - 1, j \in N_c, k \in K \tag{8d}$$

$$\sum_{i \in N_c} z_i^k + \sum_{i \in N_{cd}} \hat{z}_i^k \leq n \cdot U^k, k \in K \tag{9}$$

Constraints (2a) and (2b) stipulate that each pair of vehicles leaves from the depot once. Note that $x_{0,n+1}^{0k} = 1$ indicates that vehicle pair k is unused. In Constraints (3a) and (3b), the vehicle flow balance at each customer node is guaranteed. Constraint (3c) states that drone-served customers must be visited by the drone independently. Constraints (4a) and (4b) stipulate that each customer is served only once and that heavy parcels are assigned exclusively to trucks. Constraint (5) ensures that each pair of customers is assigned to the same route. Constraints (6a) and (6b) determine a launch node, from which the members of a truck–drone pair depart separately. Similarly, (7a) and (7b) determine a retrieval node, which is reached by the members of a truck–drone pair separately. Constraint (8a) ensures that drones can visit each customer location at most once. Constraint (8b) states that only when a drone has been retrieved or is already on its truck can it depart again. Constraints (8c) and (8d) ensure that the trucks leave the launch nodes and arrive at the retrieval nodes independently. Finally, Constraint (9) ensures that if vehicle pair k serves customers, $U^k = 1$.

The following constraints concern visiting sequences that are designed to avoid infeasible services for customer pairs.

$$u_i^k + 1 - (n+2)(1 - x_{ij}^{0k} - x_{ij}^k) \leq u_j^k \leq u_i^k + 1 + n(1 - x_{ij}^{0k} - x_{ij}^k), (i,j) \in A, k \in K \tag{10a}$$

$$b_{0,i}^k = z_i^k = b_{i,n+1}^k, i \in N_c, k \in K \tag{10b}$$

$$u_j^k \geq u_i^k + 1 - (n+2)(1 - b_{i,j}^k), i \in N_s, j \in N_e, k \in K \tag{10c}$$

$$2 * y_{i,j}^k \leq y_i^k + \tilde{y}_j^k, i \in N_s, j \in N_e, k \in K \tag{11a}$$

$$b_{i,j}^k \geq y_{i,j}^k, i \in N_s, j \in N_e, k \in K \tag{11b}$$

$$y_p^k + \tilde{y}_p^k \leq 2 * (3 - y_{i,j}^k - b_{i,p}^k - b_{p,j}^k), i \in N_s, j \in N_e, p \in N_c, k \in K \tag{11c}$$

$$y_{i,j}^k + \hat{z}_p^k \geq 2 * Z_{ij,p}^k, i \in N_s, j \in N_e, p \in N_{cd}, k \in K \tag{12a}$$

$$a_i^k - M(1 - Z_{ij,p}^k) \leq \epsilon_p \leq \hat{a}_j^k + M(1 - Z_{ij,p}^k), i \in N_s, j \in N_e, p \in N_{cd}, k \in K \tag{12b}$$

$$\epsilon_j \leq \epsilon_{p+m} + M(2 - Z_{ij,p}^k - z_{p+m}^k), i \in N_s, j \in N_c, p \in N_p \cap N_{cd}, k \in K \tag{13a}$$

$$\epsilon_p \leq \epsilon_i + M(2 - z_p^k - Z_{ij,p+m}^k), i \in N_c, j \in N_e, p \in N_p \cap N_{cd}, k \in K \tag{13b}$$

Constraint (10a) indicates that if truck k travels from node i to j, $u_i^k + 1 = u_j^k$. Constraint (10b) indicates that if a node is visited by a truck, its visit sequence must be between depots 0 and $n+1$. Constraint (10c)

ensures that if $b^k_{i,j} = 1$, truck k must visit node j after node i. Additionally, Constraints (11) define the variable $y^k_{i,j}$: if $y^k_{i,j} = 1$, nodes i and j must be used for drone launch and retrieval, respectively (Constraint (11a)); node i is visited before node j (Constraint (11b)); and drone k cannot be launched or retrieved again between these two nodes (Constraint (11c)). Constraints (12a) and (12b) ensure that if $Z^k_{ij,p} = 1$, a flight by drone k starts and ends at nodes i and j, respectively, serving customer p. Finally, Constraints (13a) and (13b) require that two paired customers cannot be served in parallel by a pair of subroutes, as shown in Figure 1(a).

3.3.3. *Timing constraints*

$$a^k_i + \theta^k_i - M(1 - z^k_i + \sum_{j \in N_{cd}} \hat{x}^k_{ji}) \leq \epsilon_i \leq a^k_i + \theta^k_i + M(1 - z^k_i + \sum_{j \in N_{cd}} \hat{x}^k_{ji}), i \in N_c, k \in K \tag{14a}$$

$$\begin{aligned} &\bar{a}^k_i + \tau^r + \theta^k_i - M(2 - z^k_i - \sum_{j \in N_{cd}} \hat{x}^k_{ji}) \leq \epsilon_i \\ &\leq \bar{a}^k_i + \tau^r + \theta^k_i + M(2 - z^k_i - \sum_{j \in N_{cd}} \hat{x}^k_{ji}), i \in N_c, k \in K \end{aligned} \tag{14b}$$

$$\hat{a}^k_i - M(1 - \hat{z}^k_i) \leq \epsilon_i \leq \hat{a}^k_i + M(1 - \hat{z}^k_i), i \in N_{cd}, k \in K \tag{15}$$

$$\begin{aligned} &\theta^k_0 + \tau^l \sum_{j \in N_{cd}} \hat{x}^k_{0j} + \frac{r_{0j}}{v} - M(1 - x^{0k}_{0j} - x^k_{0j}) \leq a^k_j \\ &\leq \theta^k_0 + \tau^l \sum_{j \in N_{cd}} \hat{x}^k_{0j} + \frac{r_{0j}}{v} + M(1 - x^{0k}_{0j} - x^k_{0j}), j \in N_e, k \in K \end{aligned} \tag{16a}$$

$$\begin{aligned} &\epsilon_i + \tau_i + \tau^l \sum_{j \in N_{cd}} \hat{x}^k_{ij} + \frac{r_{ij}}{v} - M(1 - x^{0k}_{ij} - x^k_{ij}) \leq a^k_j \\ &\leq \epsilon_i + \tau_i + \tau^l \sum_{j \in N_{cd}} \hat{x}^k_{ij} + \frac{r_{ij}}{v} + M(1 - x^{0k}_{ij} - x^k_{ij}), i \in N_c, j \in N_e, k \in K \end{aligned} \tag{16b}$$

$$\theta^k_0 + \tau^l + \frac{\hat{r}_{0j}}{\hat{v}} - M(1 - \hat{x}^k_{0j}) \leq \hat{a}^k_j \leq \theta^k_0 + \tau^l + \frac{\hat{r}_{0j}}{\hat{v}} + M(1 - \hat{x}^k_{0j}), j \in N_{cd}, k \in K \tag{17a}$$

$$\begin{aligned} &\epsilon_i + \tau_i + \tau^l + \frac{\hat{r}_{ij}}{\hat{v}} - M(2 - \sum_{j \in N_e} x^k_{ij} - \hat{x}^k_{ij}) \leq \hat{a}^k_j \\ &\leq \epsilon_i + \tau_i + \tau^l + \frac{\hat{r}_{ij}}{\hat{v}} + M(2 - \sum_{j \in N_e} x^k_{ij} - \hat{x}^k_{ij}), i \in N_c, j \in N_{cd}, k \in K \end{aligned} \tag{17b}$$

$$\epsilon_i + \hat{\tau}_i + \frac{\hat{\tau}_{ij}}{\hat{v}} - M(2 - \hat{z}^k_i - \hat{x}^k_{ij}) \leq \hat{a}^k_j \leq \epsilon_i + \hat{\tau}_i + \frac{\hat{\tau}_{ij}}{\hat{v}} + M(2 - \hat{z}^k_i - \hat{x}^k_{ij}), i \in N_{cd}, j \in N_e, k \in K \tag{17c}$$

$$\epsilon_i \leq \epsilon_{i+m}, i \in N_p, k \in K \tag{18}$$

$$\hat{a}^k_i \leq \bar{a}^k_i \leq \hat{a}^k_i + M \cdot \gamma^k_i, i \in N_c, k \in K \tag{19a}$$

$$a^k_i \leq \bar{a}^k_i \leq a^k_i + M(1 - \gamma^k_i), i \in N_c, k \in K \tag{19b}$$

Constraints (14a) and (14b) calculate the service start times of trucks at nodes that are not and are retrieval nodes, respectively. Constraint (15) means that drones provide service as soon as they arrive. Constraints (16a) and (16b) give the times at which the trucks arrive at each node from the depot and a customer, respectively. Similarly, Constraints (17a) and (17b) give the times at which drones arrive at each node from the depot and from a customer as the launch node, respectively. Constraint (17c) gives the times at which the drones arrive at each node after serving a customer. Constraint (18) ensures that a pickup-pair node is visited before its counterpart. Constraints (19a) and (19b) are the linearised formulae for the arrival time of a vehicle pair at each node. Note that the variable γ^k_i is defined in Constraint (21c).

The following constraints are specific to soft time windows.

$$\epsilon_i - s_i \leq M(1 - \alpha_i), i \in N_c \tag{20a}$$

$$\delta^-_i \leq M \cdot \alpha_i, i \in N_c \tag{20b}$$

$$s_i - \epsilon_i \le \delta_i^- \le s_i - \epsilon_i + M(1-\alpha_i), i \in N_c \tag{20c}$$

$$\tilde{s}_i - \epsilon_i \le M(1-\beta_i), i \in N_c \tag{20d}$$

$$\delta_i^+ \le M \cdot \beta_i, i \in N_c \tag{20e}$$

$$\epsilon_i - \tilde{s}_i \le \delta_i^+ \le \epsilon_i - \tilde{s}_i + M(1-\beta_i), i \in N_c \tag{20f}$$

$$\alpha_i + \beta_i \le 1, i \in N_c \tag{20g}$$

Constraints (20a)–(20c) calculate $\delta_i^- = max\{s_i - \epsilon_i, 0\}$, and Constraints (20$d$)—(20$f$) calculate $\delta_i^+ = max\{\epsilon_i - \tilde{s}_i, 0\}$, where $\alpha_i = 1$ and $\beta_i = 1$ denote that the service of node i starts before and after its time window, respectively. Finally, Constraint (20g) shows the mutually exclusive relationship between α_i and β_i.

3.3.4. Drone energy constraints

Considering that load and duration play critical roles in drone energy consumption, we introduce a model for drone energy consumption (Luo et al., 2022), which has three components, namely, flying, serving, and hovering, for different operations. For simplicity, we consider a constant drone speed while flying (Zhen et al., 2023) and ignore energy consumption during ascent and descent, as these activities typically consume less energy than long-distance travel and possible hovering.

The energy needed for drone k to travel from node i to j and serve customer i is calculated as $\eta(w_0 + w_i^k)\frac{\hat{r}_{ij}}{\hat{v}}$ and $\eta(w_0 + w_i^k + d_i)\hat{\tau}_i$, respectively, and when drone k hovers at j, the energy is defined as $P(w_i^k) \cdot t_j^{h,k}$, where i is the last customer served by drone k before arriving at j. $P(w_i^k)$ is calculated in Section 3.3.6.

$$0 \le t_i^{h,k} \le M \cdot \gamma_i^k, i \in N_c, k \in K \tag{21a}$$

$$a_i^k - \hat{a}_i^k - M(1-\gamma_i^k) \le t_i^{h,k} \le a_i^k - \hat{a}_i^k + M(1-\gamma_i^k), i \in N_c, k \in K \tag{21b}$$

$$a_i^k \ge \hat{a}_i^k - M(1-\gamma_i^k), a_i^k \le \hat{a}_i^k + M \cdot \gamma_i^k, i \in N_c, k \in K \tag{21c}$$

$$0 \le e_{ij}^{f,k} \le E \cdot \hat{x}_{ij}^k, (i,j) \in A, k \in K \tag{22a}$$

$$\eta(w_0 + w_i^k)\frac{\hat{r}_{ij}}{\hat{v}} - M(1-\hat{x}_{ij}^k) \le e_{ij}^{f,k} \le \eta(w_0 + w_i^k)\frac{\hat{r}_{ij}}{\hat{v}}, (i,j) \in A, k \in K \tag{22b}$$

$$0 \le e_i^{s,k} \le E \cdot \hat{z}_i^k, i \in N_{cd}, k \in K \tag{23a}$$

$$\eta(w_0 + w_i^k + d_i)\hat{\tau}_i - M(1-\hat{z}_i^k) \le e_i^{s,k} \le \eta(w_0 + w_i^k + d_i)\hat{\tau}_i, i \in N_{cd}, k \in K \tag{23b}$$

$$0 \le e_i^{h,k} \le E \cdot \tilde{y}_i^k, i \in N_c, k \in K \tag{24a}$$

$$P(w_i^k) \cdot t_j^{h,k} - M(2 - \sum_{i \in N_s} x_{ij}^k - \hat{x}_{ij}^k) \le e_j^{h,k} \le P(w_i^k) \cdot t_j^{h,k}, i \in N_{cd}, j \in N_c, k \in K \tag{24b}$$

$$0 \le e_i^k \le M(1-y_i^k), i \in N_s, k \in K \tag{25a}$$

$$e_j^k \ge e_i^k + e_{ij}^{f,k} + e_j^{s,k} - M(2 - \hat{x}_{ij}^k - \hat{z}_j^k), i \in N_s, j \in N_{cd}, k \in K \tag{25b}$$

$$E \ge e_i^k + e_{ij}^{f,k} + P(w_i^k) \cdot t_j^{h,k} - M(2 - \hat{x}_{ij}^k - \sum_{i \in N_s} x_{ij}^k), i \in N_{cd}, j \in N_c, k \in K \tag{25c}$$

$$E \ge e_i^k + e_{i,n+1}^{f,k} - M(1 - \hat{x}_{i,n+1}^k), i \in N_{cd}, k \in K \tag{25d}$$

Constraints (21a)–(21c) linearise $t_i^{h,k} = max\{a_i^k - \hat{a}_i^k, 0\}$. Constraints (22)–(24) calculate the energy consumed by drone k in flying over each arc, serving each customer, and hovering at each node, respectively. Constraint (25a) guarantees that the drone has a full battery when leaving each launch node. Constraint (25b) defines the cumulative energy consumed in a flight when drone k serves each node. Constraints (25c) and (25d) ensure that the energy of drone k is not depleted when it is retrieved. Note that if drone k is retrieved at the depot, the energy of hovering is zero.

3.3.5. Drone load constraints

$$S_{ij,p}^k \ge Z_{ij,p}^k + Z_{ij,p+m}^k - 1, i \in N_s, j \in N_e, p \in N_p \cap N_{cd}, k \in K \tag{26a}$$

$$S_{ij,p}^k \leq Z_{ij,p}^k, S_{ij,p}^k \leq Z_{ij,p+m}^k, i \in N_s, j \in N_e, p \in N_p \cap N_{cd}, k \in K \tag{26b}$$

$$w_i^{d,k} - M(1 - y_i^k) \leq w_i^k \leq w_i^{d,k} + M(1 - y_i^k), i \in N_s, k \in K \tag{27a}$$

$$w_i^k - q_j - M(2 - \hat{x}_{ij}^k - \hat{z}_j^k) \leq w_j^k \leq w_i^k - q_j + M(2 - \hat{x}_{ij}^k - \hat{z}_j^k), i \in N_s, j \in N_{cd}, k \in K \tag{27b}$$

$$\begin{aligned} & w_i^{d,k} - q_j - M(2 - \hat{x}_{ij}^k - \hat{z}_j^k + \sum_{(i',j') \in A} S_{i'j',j-m}^k) \leq w_j^{d,k} \\ & \leq w_i^{d,k} - q_j + M(2 - \hat{x}_{ij}^k - \hat{z}_j^k + \sum_{(i',j') \in A} S_{i'j',j-m}^k), i \in N_s, j \in N_d \cap N_{cd}, k \in K \end{aligned} \tag{27c}$$

$$\begin{aligned} & w_i^{d,k} - M(3 - \hat{x}_{ij}^k - \hat{z}_j^k - \sum_{(i'j') \in A} S_{i'j',j-m}^k) \leq w_j^{d,k} \\ & \leq w_i^{d,k} + M(3 - \hat{x}_{ij}^k - \hat{z}_j^k - \sum_{(i'j') \in A} S_{i'j',j-m}^k), i \in N_s, j \in N_d \cap N_{cd}, k \in K \end{aligned} \tag{27d}$$

$$\begin{aligned} & w_i^{d,k} - q_j - M(2 - \hat{x}_{ij}^k - \hat{z}_j^k) \leq w_j^{d,k} \\ & \leq w_i^{d,k} - q_j + M(2 - \hat{x}_{ij}^k - \hat{z}_j^k), i \in N_s, j \in N_{do} \cap N_{cd}, k \in K \end{aligned} \tag{27e}$$

$$w_i^{d,k} - M(2 - \hat{x}_{ij}^k - \hat{z}_j^k) \leq w_j^{d,k} \leq w_i^{d,k} + M(2 - \hat{x}_{ij}^k - \hat{z}_j^k), i \in N_s, j \in N_{po} \cup N_p \cap N_{cd}, k \in K \tag{27f}$$

Constraints ($26a$) and ($26b$) ensure that if $S_{ij,p}^k = 1$, customer pair p and $p + m$ are served by the same flight, starting and ending at nodes i and j, respectively. Constraint ($27a$) requires that the pickup quantity of drones is zero when they are launched. Constraint ($27b$) tracks the load carried on drones after serving customers. Constraints ($27c$) and ($27d$) calculate the drone delivery quantities after each delivery-pair node is served when its pickup pair is not and is served by this flight, respectively. Finally, Constraints ($27e$) and ($27f$) define the drone delivery quantities after serving other types of customers.

3.3.6. Linear approximation for drone power in hover

On the basis of the equation for the power consumption of a multirotor helicopter while hovering (Dorling et al., 2016; Cheng et al., 2020), we compute the power $P(\omega)$ of drones while hovering with a payload of ω as

$$P(\omega) = (w_0 + \omega)^{3/2} \sqrt{\frac{g^3}{2\rho\zeta h}} \tag{28}$$

where g, ρ, ζ and h denote gravity, the fluid density of air, the area of a spinning blade disc and the number of rotors, respectively. Since $\Xi = \sqrt{\frac{g^3}{2\rho\zeta h}}$ can be viewed as a constant parameter for a given drone type and operating condition, Eq. (28) can be written as $P(\omega) = \Xi(w_0 + \omega)^{3/2}$, which is nonlinear but convex. On this basis, we use a piecewise linear (PWL) approach for approximation. This approach is desirable because the original MINLP problem is simplified to an MIBLP problem, which can be solved with software solvers such as Gurobi. The PWL function $\tilde{P}(\omega)$ uses a set of separable linear equations (pieces) to approximate the continuous univariate function $P(\omega)$, as shown in Eq. (29).

$$\tilde{P}(\omega) = \begin{cases} \kappa_1(\omega - \omega_1) + \tilde{\kappa}_1, & \omega_1 < \omega \leq \omega_2 \\ \kappa_2(\omega - \omega_2) + \tilde{\kappa}_2, & \omega_2 < \omega \leq \omega_3 \\ \ldots \quad \ldots & \\ \kappa_{l-1}(\omega - \omega_{l-1}) + \tilde{\kappa}_{l-1}, & \omega_{l-1} < \omega \leq \omega_l \end{cases} \tag{29}$$

where κ_i and $\tilde{\kappa}_i$ are the slope and intercept of each linear piece, respectively, and where ω_i are breakpoints for connecting them. In Supplementary Materials A and B, we present the convex combination model for $\tilde{P}(\omega)$ and the linear fitting process for determining the best locations and number of breakpoints (Algorithm I), respectively. The best breakpoint vector obtained is (0.0, 0.6, 1.21, 1.81, 2.41, 3.0).

3.4. Valid inequalities

Considering the high complexity of the PDmD-sT, a set of valid inequalities is introduced to improve the software that solves the model.

Problem-specific cuts

To reduce the solution space, the following inequalities are used to improve the routing constraints. Given that all arcs except $(0, n+1)$ can be visited at most once, the following inequality holds:

$$\sum_{k\in K}(x_{ij}^{0k} + x_{ij}^{k} + \hat{x}_{ij}^{k}) \leq 1, (i,j) \in A \setminus (0, n+1) \tag{30a}$$

Moreover, the number of customers served by a drone is certainly no smaller than its number of flights:

$$\sum_{i\in N_s} y_i^k \leq \sum_{i\in N_{cd}} \hat{z}_i^k, k \in K \tag{30b}$$

and the number of arcs that a truck traverses must be equal to the number of customers it serves plus 1.

$$\sum_{(i,j)\in A}(x_{ij}^{0k} + x_{ij}^{k}) = \sum_{i\in N_c} z_i^k + 1, k \in K \tag{30c}$$

If drone k flies from node i to node j independently, at least one of the two nodes is served by it; therefore,

$$\hat{z}_i^k + \hat{z}_j^k \geq \hat{x}_{ij}^k, i,j \in N_c, k \in K \tag{30d}$$

Specifically, for a node where a drone only leaves or arrives, the paired truck must serve it.

$$z_i^k \geq \sum_{j\in N_d} \hat{x}_{ij}^k - \sum_{j\in N_s} \hat{x}_{ji}^k, i \in N_c, k \in K \tag{31e}$$

$$z_i^k \geq \sum_{j\in N_d} \hat{x}_{ji}^k - \sum_{j\in N_e} \hat{x}_{ij}^k, i \in N_c, k \in K \tag{30f}$$

Inequalities for drone capacity constraints

The following inequalities are used to remove infeasible visits and arcs.

$$\hat{x}_{ij}^k + \hat{z}_j^k \leq 1 \,\Big|\, \eta(w_0 + d_j)(\frac{\hat{r}_{ij}}{\hat{v}} + \hat{\tau}_j) \geq E, i \in N_s, j \in N_d \cup N_{do} \cap N_{cd}, k \in K \tag{31a}$$

$$\hat{x}_{ij}^k + \hat{z}_i^k \leq 1 \,\Big|\, \eta \cdot w_0 \cdot \hat{\tau}_i + \eta(w_0 + d_i)\frac{\hat{r}_{ij}}{\hat{v}} \geq E, i \in N_p \cup N_{po} \cap N_{cd}, j \in N_e, k \in K \tag{31b}$$

The above constraints require that if the energy consumption of a drone in serving delivery node j directly from i or arriving at node j after picking up a parcel from i is not less than the battery capacity, no drones can traverse arc (i, j) independently. Moreover, given the relatively small capacity of drones, we introduce two cuts to further tighten the solution space. They stipulate that drones cannot serve nodes i and j consecutively under these conditions.

$$\hat{x}_{ij}^k \leq 2 - \hat{z}_i^k - \hat{z}_j^k \,\Big|\, |d_i + d_j| > Q, d_i \cdot d_j > 0, i,j \in N_{cd}, k \in K \tag{32a}$$

$$\hat{x}_{ij}^k \leq 2 - \hat{z}_i^k - \hat{z}_j^k \,\Big|\, d_j - d_i > Q, i \in N_p \cup N_{po} \cap N_{cd}, j \in N_d \cup N_{do} \cap N_{cd}, k \in K \tag{32b}$$

4. Methodology

The PDmD-sT concerns a universal scenario involving time elasticity, which is valuable for practical applications. This new problem cannot be solved directly with an existing solution approach. In addition, given the NP-hard nature of VRP, the addition of multi-visit drones, and the integration of pair services and soft time windows, PDmD-sT is more complex than previous problems. Solving large-size problems within a practical timeframe is relatively challenging. Therefore, a new custom metaheuristic, referred to as the improved large-neighbourhood search (ILNS), is proposed to solve PDmD-sT. The LNS method, which was first introduced by Shaw (1997), has demonstrated good performance for various VRPs and has strong applicability to new problems, especially for pickup and delivery problems (as mentioned in the literature review). Its principal mechanism is to iteratively destroy and repair the current solution to improve the solution quality.

ILNS consists of three sequential procedures of searching for the best solutions. First, a three-phase construction heuristic based on a greedy strategy is applied to produce an initial solution, which is then gradually improved via an LNS with blinks (LNS_B) to find better solutions. Finally, a set of local searches is developed for further optimisation. Moreover, an exact method is designed to determine truck waiting times. The framework is given in Algorithm 1.

Algorithm 1 *Outline of the ILNS*

```
$S^{initial} \leftarrow Construction\_heuristic()$
$S^{best} \leftarrow LNS\_B(S^{initial})$
$S^{best} \leftarrow Local\_search(S^{best})$
return $S^{best}$
```

The proposed ILNS is distinguished from classic LNS as follows: (i) The construction heuristic improves the basic greedy approach by integrating pair service constraints and using the minimum "travel cost + penalty cost" as the search criterion, considering both the travel distance and the impact of soft time windows. (ii) Based on the problem features, we improve the LNS with a set of specially designed removal operators and a repair procedure with blinks. This involves a blink mechanism for adding randomness, a perturbation procedure for further increasing diversity, and a series of local searches that are integrated to achieve comprehensive exploration. Moreover, an adaptive mechanism is applied to obtain the destroyed neighbourhoods with the best performance. In addition, to avoid premature convergence to local optima due to various constraints, simulated annealing (SA) is adopted to guide and control searches by accepting a poor solution with a certain probability (Kirkpatrick et al., 1983). (iii) An exact method is designed to determine truck waiting times so that the penalty cost in each iteration can be well optimised. In addition, we embed this method within the solution feasibility evaluation, which is well suited for PDmD-sT.

4.1. Construction heuristic

The *Construction_heuristic* procedure is initiated by a greedy heuristic to produce a solution S_t that serves all nonpaired customers with trucks alone. Then, each customer pair is relocated to the obtained solution S_t in sequence. Finally, for each truck-only route, we apply *SeqConstFlight* to construct drone flights for some customers. Note that only when the cost is reduced and the solution is feasible can the procedure start. Algorithm 2 gives the pseudocode for *Construction_heuristic.*

Algorithm 2. *Construction_heuristic*

```
1: Initialisation: $k \leftarrow 1$, number of deployed trucks; $S_t \leftarrow \{0\}$, initial truck-only solution; $a_{i_c} \leftarrow 0$, arrival time at
   node $i_c$; $i_c \leftarrow 0$, current visiting node; $U_o \leftarrow N_{po} \cup N_{do}$, set of unassigned pickup-only and delivery-only customers
2: while $U_o \neq \emptyset$ do
3:     $i \leftarrow$ The node in $U_o$ that has the minimum cost to $i_c$
4:     if $a_{i_c} + \tau_{i_c} + \frac{r_{i_c,i}}{v} + \tau_i + \frac{r_{i,n+1}}{v} \leq l_{max}$ then
5:         $S_t \leftarrow S_t \cup \{i\}$; $i_c \leftarrow i$; $U_o \leftarrow U_o \setminus \{i\}$; $a_i \leftarrow a_{i_c} + \tau_{i_c} + \frac{r_{i_c,i}}{v}$
6:     else
7:         $S_t \leftarrow S_t \cup \{0\}$; $i_c \leftarrow 0$; $a_{i_c} \leftarrow 0$; $k \leftarrow k + 1$
8: for $i \in N_p$ do
9:     $S_t \leftarrow$ Greedily relocate $i$ and $i + m$
10: $S^{initial} \leftarrow SeqConstFlight(S_t)$
11: return $S^{initial}$
```

The procedure first initialises the variables by considering the first truck route (k=1), setting the current node i_c as the depot and letting the set of unassigned customers (U_o) include all pickup-only and delivery-only customers (line 1). It then assigns the node $i \in U_o$ that has the minimum cost $c \cdot \frac{r_{i_c,i}}{v} + c^s \cdot max\{s_i - a_i, 0\} + c^e \cdot max\{a_i - \tilde{s}_i, 0\}$ to the current node i_c on the truck route one by one and updates the current node, the set of unassigned customers, and the arrival time, as long as the truck arrives at the depot before l_{max} (lines 3–5). Otherwise, the current truck route ends by returning to the depot, and a new truck route starts from the depot (lines 6–7). The above steps continue until all nodes except for paired nodes are assigned.

Each customer pair is positioned in the obtained truck-only solution with the minimum cost increase, defined by the difference in costs before and after node relocation. If two paired customers cannot be relocated, a new route is created (lines 8–9).

Then, the *SeqConstFlight* procedure is applied to each truck-only route for drone flight construction (line 10). For each truck route $r \in S_t$, given the list U_d of all customers with light parcels consistent with the truck visit sequence, we iterate over each customer $i \in U_d$, attempting to construct a new flight for each, until the first flight is built. Afterwards, we continue to assign each remaining unchecked customer (if any) to the last-performed flight. If this is not feasible or the cost increases, a new flight is built from or for this node. The above procedure is conducted until all remaining customers have been reassigned or reassignment has been attempted.

4.2. Large neighbourhood search with blinks (LNS_B)

The LNS_B method is implemented with a group of removal procedures to destroy the current solution and an iterative repair procedure with blinks to obtain a better one. In addition, a perturbation procedure is applied to shake the current solution when a threshold (ϖ) for the number of iterations is reached. In addition, the solution acceptance criterion of simulated annealing is adopted when a new solution is poor. The control parameter T for the acceptance probability is initialised with T_0 and updated by $\varphi \times T$, where $\varphi \in [0, 1]$ denotes the cooling rate. Supplementary Materials C gives the pseudocode of this process (Algorithm II).

4.2.1. Destroy procedure

The destroy procedure in the PDmD-sT is rather complicated since if a removed node is used for drone launch or retrieval, the procedure must remove the affected flights. Moreover, if a pair node is involved, its counterpart needs to be removed as well. Therefore, if the counterpart is also used for drone launch or retrieval, the procedure must be iteratively carried out until no additional customers need to be deleted, as shown in Algorithm 3.

For each iteration, all the removal operators are adaptively selected according to their performance (weight, ϕ). The weight of the removal operator r after iteration j is updated by $\phi_{r,j+1} = \xi \cdot \phi_{r,j} + (1-\xi)\psi_r/u_r$, where ψ_r and u_r are the score and use times, respectively, and $\xi \in [0, 1]$ is the reaction factor. The values $\psi = \{\psi_1, \psi_2, \psi_3, \psi_4\}$, $\psi_1 > \psi_2 > \psi_3 > \psi_4$, correspond to four scenarios for a new solution: the solution is the best, only improves on the current solution, is poor but accepted and is rejected, respectively.

Algorithm 3. *Destroy_procedure*

```
Input: $S$-current solution
Output: $S_{par}$-destroyed solution; removal operator $r$ $U_{del}$-set of all removal nodes
$U_0 \leftarrow$ set of initial removal nodes by removal operator $r$
$U_{pair} \leftarrow$ Additional pair customers that need to be removed ($U_0$)
$U_0 \leftarrow U_0 \cup U_{pair}$; $U_{del} \leftarrow U_0$
while True do
    $S_{par}$; $U_r \leftarrow Delete(S, U_0)$ #Delete all nodes in $U_0$ as well as the affected flights from $S$
    $U_{dif} \leftarrow$ Additional deleted nodes ($U_r$, $U_0$)
    if $U_{dif} \neq \emptyset$ then
        $U_{del} \leftarrow U_{del} \cup U_{dif}$; $U_{pair} \leftarrow$ Additional pair customers need to be deleted ($U_{dif}$)
        if $U_{pair} \neq \emptyset$ then
            $U_{del} \leftarrow U_{del} \cup U_{pair}$; $S \leftarrow S_{par}$; $U_0 \leftarrow U_{pair}$
        else
            return $S_{par}$; $U_{del}$
    else
        return $S_{par}$; $U_{del}$
```

Eight removal operators are selected by the preliminary experiments to destroy the current solution.

- **Pair removal** randomly removes $\lfloor U(min\{2, m * \varepsilon_1\}, m) \rfloor$ customer pairs from the current solution, where ε_1 is randomly generated in the range $[0, 1]$.
- **Adjacent string removal** randomly deletes one or two adjacent strings from the current solution. Specifically, a node (seed) served by trucks is first randomly selected for a removal string; then, the closest customer to this node on another route becomes the seed of the adjacent removal string. Afterwards, each string (in route s) is removed around its seed, with a length of $\lfloor max\{2, 20\% * |s|\} \rfloor$. Note that "string" denotes a truncated section of a truck-drone route.
- **Sweep removal** first randomly removes a node from the current solution and then removes all nodes whose acute angles (with the depot as the vertex) from it are less than Γ; Γ is taken from $[0, max\{60° * \varepsilon_2, 20°\}]$, where ε_2 is randomly generated within $[0, 1]$.

- **Similarity removal** ($n \geq 3$) first randomly removes a node from the current solution and then removes the first $\lfloor max\{1, 10\%*n\}\rfloor$ nodes that are most similar to it (have the smallest similarity coefficient SC). The SC for two nodes i and j is defined as $\Phi_1|d_i-d_j|+\Phi_2(c^s|s_i-s_j|+c^e|\tilde{s}_i-\tilde{s}_j|)+\Phi_3(r_{ij}+\hat{r}_{ij})+\Phi_4*X_{i,j}$, where $\Phi_1, \Phi_2, \Phi_3, \Phi_4 \in [0,1]$ and $X_{i,j} \in \{0,1\}$. $X_{i,j} = 1$ denotes that i and j are assigned to the same route; $X_{i,j} = 0$ otherwise. The preliminary experiments tuned the values of (Φ_1, Φ_2, Φ_3, Φ_4) to (0.28, 0.49, 0.38, 0.24).
- **Distance removal** ($n \geq 3$) removes nodes with small distances from each other, measured by $r_{ij} + \hat{r}_{ij}$ for two nodes i and j. That is, it first randomly removes a node from the current solution and then removes the first $\lfloor max\{1, 10\% * n\}\rfloor$ nodes closest to it.
- **Time window removal** ($n \geq 3$) is similar to distance removal, with the aim of removing nodes that have similar time windows. The time window similarity for two nodes i and j is defined as $c^s|s_i - s_j| + c^e|\tilde{s}_i - \tilde{s}_j|$.
- **Random removal** ($n \geq 2$) randomly deletes $\lfloor max\{1, 10\% * n\}\rfloor$ customers from the same or different routes.
- **Worst removal** ($n \geq 2$) iteratively removes the node that will result in the greatest cost reduction (the cost difference between not deleting and deleting the node) until $\lfloor max\{1, 10\% * n\}\rfloor$ nodes have been removed.

4.2.2. Repair procedure

In this section, we introduce a greedy relocation with blinks (RB) for solution repair, as shown in Algorithm 4. Specifically, RB relocates each randomly selected node $i_c \in U_{del}$ to the destroyed solution with the minimum cost increase Δ_{min} consecutively until all nodes in U_{del} have been assigned. For each node $i_c \in U_{del}$, RB examines all possible positions, including on a truck route, in an existing drone flight, and in a newly built flight. Moreover, inspired by Christiaens & Vanden Berghe (2020), we apply blinks to add randomness to each feasible relocation. That is, a new feasible solution is evaluated with a probability of $1 - \vartheta$, where ϑ is the blink rate. If a position (or two positions for pair customers) is found for which the relocation cost increase is smaller than the current Δ_{min} and $U(0,1) < 1 - \vartheta$, then Δ_{min} and the current solution will be updated. Otherwise, this solution will be skipped. Notably, for a relocated node $i_c \in N_p \cup N_d$, RB becomes more complicated since nine (3×3) types of relocation must be considered, as shown in detail in Supplementary Materials D.

Algorithm 4. *Greedy relocation with blinks*

```
Input: $S_{par}$, $U_{del}$, $\vartheta$
Output: a new solution $S_{new}$
while $U_{del} \neq \emptyset$ do
    $i_c \leftarrow$ Randomly select a node from $U_{del}$
    Initialisation: $\Delta_{min} \leftarrow +\infty$; $S_{new} \leftarrow \emptyset$
    if $i_c \notin N_p \cup N_d$ then
        for each position (if possible) in each route/flight(including new flight) of $S$ do
            $S' \leftarrow$ Relocate $i_c$ into this location
            if $0 < \Delta_{cost}(S') < \Delta_{min}$ and $U(0,1) < 1 - \vartheta$ then
                $\Delta_{min} \leftarrow \Delta_{cost}(S')$; $S_{new} \leftarrow S'$
        if $S_{new} = \emptyset$ then
            $S_n \leftarrow$ greedily start a new route for $i_c$ from the depot
            $S_{new} \leftarrow S \cup S_n$
        $U_{del} \leftarrow U_{del} \setminus \{i_c\}$
    else   #$i_c \in N_p \cup N_d$, the pickup pair and its counterpart are $i_c^p$ and $i_c^d$
        for any two positions (if possible) in each route/flight(including new flights) of $S$  do
            $S' \leftarrow$ Relocate $i_c^p$ and $i_c^d$ into these two positions
            if $0 < \Delta_{cost}(S') < \Delta_{min}$ and $U(0,1) < 1 - \vartheta$ then
                $\Delta_{min} \leftarrow \Delta_{cost}(S')$; $S_{new} \leftarrow S'$
        if $S_{new} = \emptyset$ then
            $S_n \leftarrow$ greedily start a new route for $i_c^p$ and $i_c^d$ from the depot
            $S_{new} \leftarrow S \cup S_n$
        $U_{del} \leftarrow U_{del} \setminus \{i_c^p, i_c^d\}$
return $S_{new}$
```

4.2.3. Perturbation mechanism

The perturbation method randomly relocates $\lfloor max\{1, \lceil 10\% * n \rceil\}\rfloor$ strings randomly removed from the current solution. If the split points for removing strings are also launch/retrieval nodes for the remaining flights, they will be retained, and the affected flights must select replacements. Note that paired customers need to be removed together.

4.3. Local search

A set of local searches are conducted in sequence to further improve the solution quality of the final output. The local search selects the best feasible solution in each iteration to improve the current solution and is repeated until the number of consecutive unimproved iterations reaches the threshold ($\tilde{\varpi}$).

- **Node insertion** randomly moves a truck-served node (not used for drone launch and retrieval) or a drone-served node to a new position, either on a truck route or in a drone flight, depending on which option is better.
- **Random exchange** randomly exchanges two nodes (exchanges are not conducted between heavy-parcel nodes and drone-served nodes). It uses three types of exchanges based on the selected nodes: (i) drone-served node exchange within one or two flights, (ii) truck-served node exchange, and (iii) exchange between truck-served and drone-served nodes.
- **Mix optimisation** optimises a randomly selected launch/retrieval node i. Specifically, the best of the following three options is executed: (i) Exchange/replace node i with its three closest nodes in terms of Manhattan distance. If node i is replaced, it is greedily relocated to another position. (ii) Use the adjacent node on the route for drone launch/retrieval. (iii) Move node i to its previous or next position in the route.

4.4. Drone feasibility test and truck waiting time optimisation

Our model allows trucks to wait at each node. Obviously, waiting time optimisation plays a key role in cost savings and solution feasibility, as it directly affects the service start times and hover energy consumption of drones. For any vehicle pair, we denote the truck route $S := \{(0, i_1), (i_1, i_2), ..., (i_G, n+1)\}$ and the corresponding drone flights $R := \{(i_0^f, i_1^f, ..., i_j^f, ..., i_{|N_f|+1}^f)\}, f \in F$, where N_f is the set of customers that flight f serves. Then, $N_G = \{i_1, ..., i_G\}$ and $N_F = \bigcup_{f \in F} N_f$ are the sets of customers assigned to the truck and drone, respectively. Moreover, we partition all nodes on the truck route into four subsets, namely, launch-only nodes (N_s^1), retrieval-only nodes (N_s^2), retrieval–launch nodes (N_s^3) and nodes that neither launch nor retrieve (N_s^4). For simplicity, we omit the index k in all the other notation used in the model. With the input values of $e_{i_{|N_f|}^f}$ and $w_{i_{|N_f|}^f}$ when the drone departs from the last-served customer on each flight $f \in F$, the waiting time optimisation process is modelled as

$$min \sum_{i \in N_G \cup N_F} (c^s \delta_i^- + c^e \delta_i^+) + \hat{c} \sum_{i \in N_s^2 \cup N_s^3 \cap N_G} e_i^h \tag{33}$$

subject to:

$$\hat{a}_{i_1^f} = \begin{cases} \theta_{i_0^f} + \tau^l + \frac{\hat{r}_{i_0^f, i_1^f}}{\hat{v}}, i_0^f = 0 \\ \epsilon_{i_0^f} + \tau_{i_0^f} + \tau^l + \frac{\hat{r}_{i_0^f, i_1^f}}{\hat{v}}, else \end{cases}, f \in F \tag{34a}$$

$$\hat{a}_{i_{j+1}^f} = \hat{a}_{i_j^f} + \hat{\tau}_{i_j^f} + \frac{\hat{r}_{i_j^f, i_{j+1}^f}}{\hat{v}}, i_j^f \in N_f, f \in F \tag{34b}$$

$$a_{i'} = \begin{cases} \tau^l + \frac{r_{i,i'}}{v} + \theta_i, 0 \in N_s^1, (i,i') \in \{(0,i_1)\} \\ \frac{r_{i,i'}}{v} + \theta_i, 0 \in N_s^4, (i,i') \in \{(0,i_1)\} \\ a_i + \tau_i + \tau^l + \frac{r_{i,i'}}{v} + \theta_i, i \in N_s^1, (i,i') \in S \setminus (0,i_1) \\ \bar{a}_i + \tau^r + \tau_i + \frac{r_{i,i'}}{v} + \theta_i, i \in N_s^2, (i,i') \in S \\ \bar{a}_i + \tau^r + \tau_i + \tau^l + \frac{r_{i,i'}}{v} + \theta_i, i \in N_s^3, (i,i') \in S \\ a_i + \tau_i + \frac{r_{i,i'}}{v} + \theta_i, i \in N_s^4, (i,i') \in S \setminus (0,i_1) \end{cases} \tag{35}$$

$$\epsilon_i = \begin{cases} a_i + \theta_i, i \in N_s^1 \cup N_s^4 \cap N_G \\ \bar{a}_i + \tau^r + \theta_i, i \in N_s^2 \cup N_s^3 \cap N_G \\ \hat{a}_i, i \in N_F \end{cases} \tag{36}$$

$$E \geq \begin{cases} e_{i^f_{|N_f|}} + e^f_{i^f_{|N_f|}, i^f_{|N_f|+1}} + P(w_{i^f_{|N_f|}}) \cdot t^h_{i^f_{|N_f|+1}}, i^f_{|N_f|+1} \in N_G \\ e_{i^f_{|N_f|}} + e^f_{i^f_{|N_f|}, i^f_{|N_f|+1}}, else \end{cases}, f \in F \tag{37a}$$

$$e^f_{i^f_{|N_f|}, i^f_{|N_f|+1}} = \eta \left(w_0 + w_{i^f_{|N_f|}} \right) \frac{\hat{r}_{i^f_{|N_f|}, i^f_{|N_f|+1}}}{\hat{v}}, f \in F \tag{37b}$$

(19a)-(19b), (21a)-(21c) for every $i \in N_s^2 \cup N_s^3$; (20a)-(20g) for every $i \in N_G \cup N_F$.

The model aims to minimise the sum of the penalty cost and hovering cost of the route. Constraints (34a) and (34b) define the times at which the drone arrives at each node from a launch node and a customer it serves, respectively. Similarly, Constraint (35) defines the time at which the truck arrives at each node. Constraint (36) defines the service start times at each node in different scenarios. Finally, Constraint (37a) ensures that the total energy consumed by each flight does not exceed the battery capacity, and the energy required for the drone to fly across the last arc in each flight is calculated in Constraint (37b).

Gurobi can solve this model easily; however, the model must be solved in every iteration, which is time consuming. To accelerate the computational process, the following observations are made.

Beginning with definitions, let a_i^{start} be the earliest start time for serving node i. That is, if node i is used for drone retrieval, $a_i^{start} = \bar{a}_i + \tau^r$. Otherwise, $a_i^{start} = a_i$. Then, we can obtain the following.

Lemma 1. *If $0 \in N_s^4$, setting θ_0 to zero has no effect on follow-up services or total costs.*

In Supplementary Materials E, we provide all the proofs for Lemma 1, Proposition 1 and the Corollaries.

According to Lemma 1, we have $\theta_0 = 0, 0 \in N_s^4$.

When node i is a nonlaunch node, θ_i can be obtained via Proposition 1.

Proposition 1. *If $a_i^{start} \geq s_i$, $i \in N_s^2 \cup N_s^4$, then $\theta_i = 0$ minimises the penalty cost at node i and yields the earliest service start times for the subsequent nodes, namely, the minimum latency. Otherwise, $\theta_i \in [0, s_i - a_i^{start}]$.*

In view of Proposition 1, if $a_j^{start} \geq s_j$, we set $\theta_j = 0$. Otherwise, we set $\theta_j \in [0, s_j - a_j^{start}]$, $j \in N_s^2 \cup N_s^4$.

For the special case $i_G \in N_s^2 \cup N_s^4$, the optimal waiting time is obtained from Corollary 1.

Corollary 1. *For the last visited node $i_G \in N_s^2 \cup N_s^4$, when $a_{i_G}^{start} < s_{i_G}$, if $s_{i_G} + \tau_{i_G} + \frac{r_{i_G,n+1}}{v} \leq l_{max}$, then $\theta_{i_G} = s_{i_G} - a_{i_G}^{start}$ minimises the penalty cost at node i_G and the latency. Otherwise, $\theta_{i_G} = l_{max} - \tau_{i_G} - \frac{r_{i_G,n+1}}{v} - a_{i_G}^{start}$.*

In view of Corollary 1, when $a_{i_G}^{start} < s_{i_G}$, $i_G \in N_s^2 \cup N_s^4$; if $s_{i_G} + \tau_{i_G} + \frac{r_{i_G,n+1}}{v} \leq l_{max}$, we set $\theta_{i_G} = s_{i_G} - a_{i_G}^{start}$. Otherwise, $\theta_{i_G} = l_{max} - \tau_{i_G} - \frac{r_{i_G,n+1}}{v} - a_{i_G}^{start}$.

When the truck travels from node $i \in N_s^4$ to j, which is a retrieval node, we find θ_i via Corollary 2.

Corollary 2. *For any arc (i,j), with $i \in N_s^4$ and $j \in N_s^2 \cup N_s^3$, when $a_i^{start} < s_i$, if $s_i + \tau_i + \frac{r_{ij}}{v} \leq \hat{a}_j$, then $\theta_i = s_i - a_i^{start}$ minimises the penalty cost at i and the latency. Otherwise, $\theta_i \in [max\{\hat{a}_j - \tau_i - \frac{r_{ij}}{v} - a_i^{start}, 0\}, s_i - a_i^{start}]$.*

In view of Corollary 2, when $a_i^{start} < s_i$, $i \in N_s^4$, and $j \in N_s^2 \cup N_s^3$, if $s_i + \tau_i + \frac{r_{ij}}{v} \leq \hat{a}_j$, we set $\theta_i = s_i - a_i^{start}$. Otherwise, $\theta_i \in [max\{\hat{a}_j - \tau_i - \frac{r_{ij}}{v} - a_i^{start}, 0\}, s_i - a_i^{start}]$.

5. Numerical experiment

To examine our proposed model and algorithm, we generate test instances and perform a series of numerical tests, as described in this section. We code ILNS in Python 3.9.5 and use Gurobi 10.0.2 as the solver. All computations are executed on a laptop with a 2.60 GHz Intel (R) Core (TM) i7-10750H processor with 32 GB of RAM running a Windows 10 operating system. We first present the test instances and parameters in Section 5.1; then, Section 5.2 presents the best results in detail for all generated instances. The benefit of waiting time optimisation is shown in Section 5.3. Finally, Sections 5.4 and 5.5 investigate the problem features and present the impacts of the key parameters, respectively.

5.1. Instance sets and parameters

In the experiment, we randomly generate 48 test instances, which are evenly classified into four sets, each of which includes 12 instances and a size of $n = \{30, 50, 70, 100\}$ nodes, uniformly distributed in a grid network with an area of 10×10, 12×12, 12×12 and 15×15 (units: km^2), respectively. Notably, our proposed model and algorithm are not limited to specific networks (as shown in Section 5.5.2). Each set consists of two types of time window scenarios (A and B; six for each):

A: Each customer i randomly generates s_i in $[20, 260]$, and $\tilde{s}_i = s_i + 20$ or $\tilde{s}_i = s_i + 60$;

B: Each customer i randomly generates s_i in $[60, 360]$, and $\tilde{s}_i = s_i + 20$ or $\tilde{s}_i = s_i + 60$.

The allowable working duration for all vehicles (l_{max}) is set to 450 min. Moreover, for each instance, we randomly set the first 10% of customers as paired nodes and 30% of customers as pickup-only nodes. Then, the demands of 10% of the customers and the remaining customers are randomly selected from the ranges (3, 50] and (0, 2.3] (Meng et al., 2024a), denoting the demands that cannot and can be carried by drones, respectively. In addition, we produce another 30 small instances (based on scenario A), each 1/3 of which includes 10, 11 and 12 customers, respectively, to assess the model, valid inequalities and proposed ILNS method. For convenience, each instance is named in the form "A/B.#.n.wd", where wd denotes the width of the time window. All instances can be downloaded from https://doi.org/10.17632/3yyt8wtkhx.3. Some of the other model parameters are referenced from the relevant literature, as shown in Table 4a, and the values of the ILNS parameters tuned in the preliminary experiments are listed in Table 4b.

Table 4: List of parameters.

Parameter	Value	Reference
Cub weight of drones (w_0)	6 kg	Meng et al. (2023)
Unit energy consumption of drones (η)	2/3 Wh/(kg· min)	–
Drone capacity related to the load (Q)	3 kg	Liu et al. (2020)
Drone capacity related to the battery (E)	180 Wh ($=\eta(Q + w_0) * B$), where the battery endurance is B=0.5 h	Murray & Chu (2015)
Truck speed (v)	30 km/h	Zhen et al. (2023)
Drone speed ($\hat{v}$)	40 km/h	Nguyen et al. (2022)
Travel cost per kilometre for trucks (c)	\$0.78/km	Salama & Srinivas (2020)
Unit energy cost for drones ($\hat{c}$)	\$0.0104 ($=\frac{c}{\varsigma} * \hat{v} * B/E$), where ς =8.3 denotes the truck-to-drone unit cost ratio	Salama & Srinivas (2020)
Durations required to launch and retrieve the drone (τ^l, τ^r)	1 min, 1 min	Murray & Chu (2015)
Durations required to serve i with trucks and drones (τ_i, $\hat{\tau}_i$)	3 min, 2 min	Coindreau et al. (2021)
Unit penalty costs (c^s, c^e)	\$0.5/min, \$1.5/min	Meng et al. (2024b)
Unit fixed cost for using a vehicle pair (c^0)	\$30	–
Parameters used in the power calculation (Ξ)	18 ($=\sqrt{\frac{g^3}{2\rho\zeta h}}$), where g=9.8 N/kg, ρ=1.204 kg/m^3, ζ=0.2m^2, h=6	Dorling et al. (2016)

(a) Parameters used in the model.

Descriptions	Values	Descriptions	Values
T_0- initial temperature	100	ψ_1- score of the new best solution	6
φ- cooling rate	0.92	ψ_2- score of the new improved solution	3
T_f- floor temperature	10	ψ_3- score of an accepted poor solution	1
$iterMax$- number of iterations	10	ψ_4- score of a rejected solution	0
ξ- reaction factor	0.70	ϖ threshold of perturbation	20
ϑ - blink rate	0.15	$\tilde{\varpi}$- threshold of local search	50

(b) Parameters used in the metaheuristic.

5.2. Experiments and results

5.2.1. Evaluation of the model and valid inequalities with small instances

In this section, the solutions obtained by Gurobi and the ILNS for 30 small instances are compared to verify the mathematical model, valid inequalities and ILNS scheme. For each instance, we limit the execution time of Gurobi to 2 h and run the ILNS five times. The comparative results are summarised in Table 5, including the best Gurobi solutions without and with valid inequalities (C^{G_1} and $time^{G_2}$), their run times T^{G_1} and $time^{G_2}$, and the best solution (C) and average run time ($\overline{time}$) obtained for the ILNS. The percentage gaps between the Gurobi solutions (C^{G_1}, C^{G_2}) and between the ILNS solution and the Gurobi solution C^{G_2} are given in columns $\Delta_{G_2,G_1}\%(= 100(C^{G_2} - C^{G_1})/C^{G_1})$ and $\Delta_{i,G_2}\%(= 100(C - C^{G_2})/C^{G_2})$, respectively. Note that all valid inequalities are added before the solver is run.

The runtime results indicate that the valid inequalities are effective, as the time consumed by Gurobi is clearly reduced by more than 30% on average when these inequalities are included. Moreover, the time required for Gurobi increases substantially when more customers are served and fluctuates greatly, whereas the ILNS approach displays strong robustness.

With respect to solution quality, all instances with 10 and 11 customers can be exactly solved by Gurobi and the ILNS, but much less time is required for the ILNS method. For instances with 12 customers, only 3 instances cannot be exactly solved by Gurobi within 2 h when inequalities are included, and 6 instances cannot be solved otherwise. Moreover, better solutions are found for 2 of the remaining 3 instances, verifying the necessity of adding these inequalities. In contrast, the ILNS returns the same or better results in all but one instance, and it consumes much less time, demonstrating its good performance.

Table 5: Comparison of the solutions obtained with the ILNS and Gurobi for small instances.

Instance	Gurobi solution				ILNS		$\Delta_{G_2,G_1}\%$	$\Delta_{i,G_2}\%$
	C^{G_1}($)	$time^{G_1}$(s)	C^{G_2}($)	$time^{G_2}$(s)	C($)	$\overline{time}$(s)		
A.1.10.20	54.33	471.27	54.33	336.28	54.33	101.09	0.00	0.00
A.2.10.20	56.53	254.81	56.53	226.70	56.53	104.45	0.00	0.00
A.3.10.20	60.41	1715.84	60.41	856.16	60.41	113.88	0.00	0.00
A.4.10.20	59.20	1011.66	59.20	766.22	59.20	82.15	0.00	0.00
A.5.10.20	54.39	1440.48	54.39	596.83	54.39	98.38	0.00	0.00
A.6.10.20	62.54	1723.25	62.54	784.12	62.54	109.06	0.00	0.00
A.7.10.20	58.14	634.66	58.14	374.31	58.14	94.31	0.00	0.00
A.8.10.20	65.73	1571.86	65.73	510.56	65.73	130.58	0.00	0.00
A.9.10.20	47.03	196.88	47.03	152.00	47.03	119.15	0.00	0.00
A.10.10.20	60.63	2603.89	60.63	693.80	60.63	102.31	0.00	0.00
Avg.		**1162.46**		**529.70**		**105.54**	**0.00**	**0.00**
A.1.11.20	45.42	610.19	45.42	385.34	45.42	129.32	0.00	0.00
A.2.11.20	54.68	3044.78	54.68	1079.44	54.68	126.54	0.00	0.00
A.3.11.20	59.22	1974.57	59.22	1374.71	59.22	120.86	0.00	0.00
A.4.11.20	58.04	5123.21	58.04	1452.92	58.04	142.03	0.00	0.00
A.5.11.20	54.01	2021.63	54.01	1089.55	54.01	199.33	0.00	0.00
A.6.11.20	53.83	2491.19	53.83	1497.69	53.83	118.08	0.00	0.00
A.7.11.20	46.48	606.55	46.48	623.82	46.48	168.91	0.00	0.00
A.8.11.20	46.71	585.79	46.71	283.10	46.71	97.02	0.00	0.00
A.9.11.20	51.77	1000.23	51.77	664.47	51.77	84.30	0.00	0.00
A.10.11.20	56.06	6028.47	56.06	1748.39	56.06	153.62	0.00	0.00
Avg.		**2348.66**		**1019.94**		**134.00**	**0.00**	**0.00**
A.1.12.20	53.77	2959.38	53.77	2014.70	53.77	167.16	0.00	0.00
A.2.12.20	61.21	7200.00*	61.21	4652.10	61.21	187.43	0.00	0.00
A.3.12.20	63.12	7200.00*	62.57	7200.00*	62.17	156.73	-0.87	-0.64
A.4.12.20	67.34	7200.00*	67.34	7200.00*	67.34	185.14	0.00	0.00
A.5.12.20	57.51	7200.00*	57.51	5702.35	57.51	227.07	0.00	0.00
A.6.12.20	61.24	7200.00*	60.79	7200.00*	59.11	151.67	-0.73	-2.76
A.7.12.20	51.86	7200.00*	51.86	4749.02	51.86	160.13	0.00	0.00
A.8.12.20	60.56	4600.90	60.56	3079.05	60.56	172.10	0.00	0.00
A.9.12.20	59.82	5861.99	59.82	2150.32	59.98	126.52	0.00	0.27
A.10.12.20	50.75	3687.87	50.75	2231.06	50.75	153.14	0.00	0.00
Avg.		**6031.01**		**4617.86**		**168.71**	**-0.16**	**-0.31**

* The optimal solution could not be found by Gurobi within 2 h.

5.2.2. *Metaheuristic performance for large-size instances*

In this section, we first present the detailed results for the 48 large-size instances previously generated, as shown in Table 6; notably, the table lists the best solutions obtained by the ILNS over five runs for each instance. The total cost is presented in column C, which is divided into the operational cost (C_o), penalty cost (C_p) and fixed cost (C_f). The percentages of each in the total cost are defined as $\Delta_{o/c} = 100 * C_o/C$, $\Delta_{p/c} = 100 * C_p/C$ and $\Delta_{f/c} = 100 * C_f/C$, respectively. Moreover, Table 6 shows the total cost of the truck-only mode (C_t) and some quantities related to solution features such as the number of used vehicle pairs ($\#k$), number of conducted flights ($\#f$) and number of drone-served customers ($\#d$). The proportion of drone-served customers to customers whose demands are within the carrying capacity of drones ($D = 90\%n$) is shown in column $\Delta_{d/D}(= 100 * d/D)$, and column $\Delta_{C,C_t}(= 100 * (C - C_t)/C_t)$ presents the percentage gap between PDmD-sT and the truck-only mode. Finally, the average computational time is recorded in column $\overline{time}$.

In general, the total cost and sub-costs in scenarios with tighter time windows are usually greater. Moreover, the average percentage of each sub-cost with respect to the total cost is relatively stable over different instances, and the percentage of penalty costs is rather small and sometimes zero in scenarios with wide time windows, demonstrating the effectiveness of truck waiting time optimisation. With respect to the

Table 6: Solutions produced by ILNS in scenarios A and B.

Instance	C_t(\$)	C_o(\$)	C_p(\$)	C_f(\$)	C(\$)	$\Delta_{o/c}$%	$\Delta_{p/c}$%	$\Delta_{f/c}$%	#k	#f	#d	D	$\Delta_{d/D}$%	Δ_{C,C_t}%	$\overline{time}$(s)
A.1.30.20	158.83	43.96	0.14	30.00	74.10	59.32	0.19	40.49	1	7	16	27	59.26	-53.35	1094.61
A.2.30.20	147.46	45.27	0.91	30.00	76.18	59.43	1.19	39.38	1	7	16	27	59.26	-48.34	1251.63
A.3.30.20	150.73	47.84	5.87	30.00	83.71	57.15	7.01	35.84	1	7	18	27	66.67	-44.46	1135.21
B.1.30.20	156.33	44.58	0.27	30.00	74.85	59.56	0.36	40.08	1	8	19	27	70.37	-52.12	984.26
B.2.30.20	152.17	47.05	5.12	30.00	82.17	57.26	6.23	36.51	1	7	20	27	74.07	-46.00	1053.26
B.3.30.20	161.90	47.57	6.34	30.00	83.91	56.69	7.56	35.75	1	8	18	27	66.67	-48.17	1052.63
A.1.30.60	129.73	32.92	0.00	30.00	62.92	52.32	0.00	47.68	1	6	19	27	70.37	-51.50	756.23
A.2.30.60	104.84	33.57	0.00	30.00	63.57	52.81	0.00	47.19	1	8	19	27	70.37	-39.36	879.53
A.3.30.60	119.12	27.83	0.00	30.00	57.83	48.12	0.00	51.88	1	7	22	27	81.48	-51.45	651.28
B.1.30.60	132.85	31.41	0.00	30.00	61.41	51.15	0.00	48.85	1	7	21	27	77.78	-53.77	953.04
B.2.30.60	118.87	37.56	0.00	30.00	67.56	55.60	0.00	44.40	1	7	21	27	77.78	-43.16	983.78
B.3.30.60	127.34	32.45	0.04	30.00	62.49	51.93	0.06	48.01	1	7	21	27	77.78	-50.93	546.39
Avg.						**55.11**	**1.88**	**43.01**					**70.99**	**-48.55**	
A.1.50.20	257.22	73.05	1.38	60.00	134.43	54.34	1.03	44.63	2	11	32	45	71.11	-47.74	2804.40
A.2.50.20	255.99	80.48	1.97	60.00	142.45	56.50	1.38	42.12	2	15	32	45	71.11	-44.35	2945.08
A.3.50.20	261.62	78.68	0.43	60.00	139.11	56.56	0.31	43.13	2	12	29	45	64.44	-46.83	2706.16
B.1.50.20	259.87	71.60	0.05	60.00	131.65	54.39	0.04	45.58	2	13	29	45	64.44	-49.34	3097.83
B.2.50.20	247.43	76.68	3.94	60.00	140.62	54.53	2.80	42.67	2	13	29	45	64.44	-43.17	2475.54
B.3.50.20	260.09	72.44	0.00	60.00	132.44	54.70	0.00	45.30	2	13	27	45	60.00	-49.08	2985.83
A.1.50.60	203.57	47.63	0.15	60.00	107.78	44.19	0.14	55.67	2	12	33	45	73.33	-47.06	2467.16
A.2.50.60	190.41	54.95	0.01	60.00	114.96	47.80	0.01	52.19	2	12	36	45	80.00	-39.63	2168.26
A.3.50.60	155.45	53.61	0.00	60.00	113.61	47.19	0.00	52.81	2	13	32	45	71.11	-26.92	3322.34
B.1.50.60	195.88	45.56	0.00	60.00	105.56	43.16	0.00	56.84	2	13	38	45	84.44	-46.11	3263.60
B.2.50.60	199.04	50.16	0.00	60.00	110.16	45.53	0.00	54.47	2	15	35	45	77.78	-44.65	2262.57
B.3.50.60	200.83	55.73	0.00	60.00	115.73	48.16	0.00	51.84	2	12	32	45	71.11	-42.37	3052.10
Avg.						**50.59**	**0.48**	**48.94**					**71.11**	**-43.94**	
A.1.70.20	385.25	126.52	0.88	90.00	217.40	58.20	0.40	41.40	3	21	42	63	66.67	-43.57	3416.89
A.2.70.20	395.77	117.47	5.21	90.00	212.68	55.23	2.45	42.32	3	19	40	63	63.49	-46.26	2822.03
A.3.70.20	408.55	135.37	1.63	90.00	227.00	59.63	0.72	39.65	3	20	40	63	63.49	-44.44	4055.58
B.1.70.20	386.76	112.36	0.00	90.00	202.36	55.52	0.00	44.48	3	19	43	63	68.25	-47.68	3699.48
B.2.70.20	371.59	111.05	2.93	90.00	203.98	54.44	1.44	44.12	3	20	43	63	68.25	-45.11	4267.52
B.3.70.20	353.46	112.59	0.61	90.00	203.20	55.41	0.30	44.29	3	22	43	63	68.25	-42.51	3689.17
A.1.70.60	281.05	89.84	0.00	90.00	179.84	49.96	0.00	50.04	3	17	43	63	68.25	-36.01	3959.00
A.2.70.60	296.31	87.68	0.00	90.00	177.68	49.35	0.00	50.65	3	17	48	63	76.19	-40.04	3196.18
A.3.70.60	285.28	111.29	0.38	60.00	171.67	64.83	0.22	34.95	2	15	35	63	55.56	-39.82	4317.89
B.1.70.60	290.82	117.42	0.09	60.00	177.51	66.15	0.05	33.80	2	17	42	63	66.67	-38.96	3539.67
B.2.70.60	314.55	88.36	0.00	90.00	178.36	49.54	0.00	50.46	3	20	45	63	71.43	-43.30	3942.18
B.3.70.60	283.60	89.94	0.00	90.00	179.94	49.98	0.00	50.02	3	19	45	63	71.43	-36.55	3776.79
Avg.						**55.69**	**0.46**	**43.85**					**67.33**	**-42.02**	
A.1.100.20	511.99	161.38	0.98	120.00	282.36	57.15	0.35	42.50	4	26	61	90	67.78	-44.85	6660.61
A.2.100.20	514.35	156.21	0.00	120.00	276.21	56.55	0.00	43.45	4	25	60	90	66.67	-46.30	6623.30
A.3.100.20	476.20	151.70	4.37	150.00	306.07	49.56	1.43	49.01	5	25	57	90	63.33	-35.73	6801.23
B.1.100.20	504.64	182.77	9.70	120.00	312.47	58.49	3.10	38.40	4	26	59	90	65.56	-38.08	7065.58
B.2.100.20	473.89	148.71	0.34	120.00	269.05	55.27	0.13	44.60	4	25	62	90	68.89	-43.23	7573.79
B.3.100.20	453.58	151.83	0.02	150.00	301.85	50.30	0.01	49.69	5	27	66	90	73.33	-33.45	6502.38
A.1.100.60	385.32	130.61	1.23	90.00	221.84	58.88	0.55	40.57	3	23	61	90	67.78	-42.43	6089.26
A.2.100.60	363.07	111.70	0.00	120.00	231.70	48.21	0.00	51.79	4	22	64	90	71.11	-36.18	5671.93
A.3.100.60	372.95	127.39	0.00	120.00	247.39	51.49	0.00	48.51	4	23	63	90	70.00	-33.67	6872.03
B.1.100.60	423.58	140.21	0.00	90.00	230.21	60.91	0.00	39.09	3	22	63	90	70.00	-45.65	6753.47
B.2.100.60	360.39	113.93	0.00	90.00	203.93	55.87	0.00	44.13	3	23	70	90	77.78	-43.41	6850.15
B.3.100.60	345.36	131.67	0.13	90.00	221.80	59.36	0.06	40.58	3	21	61	90	67.78	-35.78	6572.15
Avg.						**55.17**	**0.47**	**44.36**					**69.17**	**-39.90**	

solution characteristics, instances with wide time windows usually require comparatively fewer vehicle pairs but can provide a similar number of drone services with the same number of or fewer flights. Moreover, although more time is required when the instance size increases, most instances can be solved within 2 hours. In addition, the cost advantage of the PDmD-sT over the truck-only mode is obvious, as the cost savings rate is more than 40% on average and even exceeds 45% when customers are distributed within a small area (10 × 10 area). Details regarding the impact of the distribution area can be found in Section 5.5.2.

Furthermore, we compare ILNS with the LNS (Kitjacharoenchai et al., 2020) and ILS (Luo et al., 2022) methods on all instances with 30 and 50 customers. These benchmarks are both designed for the multi-visit VRPD, which shares some similarities with PDmD-sT. To adapt them to our problem, both the construction heuristic and waiting time optimisation are employed for trucks, and the numbers of iterations for LNS and ILS are set to 10 and 15, respectively; these values were tuned in the preliminary experiments. Table 7 presents the best and average results over five runs and the percentage gaps between them ($\Delta_l\% = 100(C^l - C)/C$; $\Delta_i\% = 100(C^i - C)/C$), demonstrating the superior performance of the ILNS for generating high-quality solutions. Specifically, the average gaps of LNS and ILS with respect to ILNS are greater than 10%, and these gaps fluctuate greatly in different instances because of the lack of custom operators associated with problem characteristics such as time windows and paired services. In terms of run time, ILNS is more efficient than the two compared alternatives, requiring the smallest computational time on average, especially for large instances (the smallest ones are shown in bold).

Table 7: ILNS vs. LNS and ILS.

Instance	ILNS			LNS				ILS			
	C(\$)	C_{avg}(\$)	$\overline{time}$(s)	C^l(\$)	C_{avg}(\$)	$\overline{time}$(s)	Δ_l%	C^i(\$)	C_{avg}(\$)	$\overline{time}$(s)	Δ_i%
A.1.30.20	74.10	82.56	1094.61	102.96	116.22	1507.54	38.95	100.29	111.34	**1079.85**	35.34
A.2.30.20	76.18	81.49	1251.63	79.32	88.13	1320.53	4.12	83.16	94.25	**869.72**	9.16
A.3.30.20	83.71	88.56	1135.21	90.27	98.32	1341.06	7.84	94.28	102.72	**1052.03**	12.63
B.1.30.20	74.85	78.83	984.26	101.56	110.36	1263.05	35.68	106.22	115.62	**697.61**	41.91
B.2.30.20	82.17	89.43	**1053.26**	94.62	107.58	1450.19	15.15	97.34	110.45	1157.05	18.46
B.3.30.20	83.91	87.72	**1052.63**	98.26	107.98	1520.37	17.10	102.61	107.92	1250.25	22.29
A.1.30.60	62.92	65.08	**756.23**	64.71	78.16	928.13	2.84	72.53	86.51	818.58	15.27
A.2.30.60	63.57	67.53	**879.53**	68.42	79.47	1142.05	7.63	66.46	67.91	1097.41	4.55
A.3.30.60	57.83	60.42	**651.28**	60.56	68.07	876.35	4.72	63.45	70.22	778.61	9.72
B.1.30.60	61.41	66.52	**953.04**	63.56	71.23	1004.76	3.50	73.34	85.03	1092.58	19.43
B.2.30.60	67.56	70.41	983.78	70.34	77.03	**884.63**	4.11	70.09	80.54	1303.18	3.74
B.3.30.60	62.49	66.81	**546.39**	65.45	79.19	854.23	4.74	68.65	77.91	598.08	9.86
A.1.50.20	134.43	139.53	2804.40	160.68	170.43	**2363.32**	19.53	171.45	187.22	2548.83	27.54
A.2.50.20	142.45	148.17	2945.08	158.66	170.06	**2647.68**	11.38	143.78	151.26	2841.06	0.93
A.3.50.20	139.11	142.39	2706.16	161.64	170.41	2850.71	16.20	145.64	156.09	**2653.26**	4.69
B.1.50.20	131.65	136.27	**3097.83**	161.51	168.26	3506.10	22.68	145.95	163.89	3256.82	10.86
B.2.50.20	140.62	147.04	**2475.54**	148.14	157.23	3554.95	5.35	154.31	165.82	3410.06	9.74
B.3.50.20	132.44	136.11	**2985.83**	145.44	158.01	3250.13	9.82	142.87	154.71	2986.71	7.88
A.1.50.60	107.78	113.09	**2467.16**	111.47	121.84	2874.87	3.42	111.97	122.11	3052.36	3.89
A.2.50.60	114.96	119.46	**2168.26**	124.03	136.13	3121.73	7.89	124.62	133.71	2981.70	8.40
A.3.50.60	113.61	117.39	**3322.34**	132.53	138.60	3952.07	16.65	121.68	133.63	4016.23	7.10
B.1.50.60	105.56	109.76	**3263.60**	117.96	121.38	4697.21	11.75	124.02	130.22	5142.26	17.49
B.2.50.60	110.16	115.28	**2262.57**	112.30	119.71	2935.50	1.94	113.79	124.59	2811.03	3.30
B.3.50.60	115.73	118.75	**3052.10**	124.35	133.35	3352.63	7.45	120.26	126.32	3652.19	3.91
Avg.			**1870.53**			**2216.66**	**11.69**			**2131.14**	**12.84**

5.3. *Effectiveness of truck waiting time optimisation*

In this section, the effectiveness of truck waiting time optimisation is explored; the results of the PDmD-sT with waiting time optimisation and zero waiting times are compared for all instances with 30 and 50 customers. Figure 5 displays the best cost and average computational time among five runs. Employing truck waiting time optimisation is obviously effective, as the penalty cost of PDmD-sT with waiting time optimisation is relatively small and stable for different instances and is even zero for instances with wide time windows. In contrast, the penalty cost of PDmD-sT with zero truck waiting times is high and considerably fluctuates, especially in scenario B, which includes later and broader starts of time windows. In addition, waiting time optimisation is rather time consuming and increases the complexity of PDmD-sT, as a significant increase can be observed in terms of computational time when waiting times are optimised.

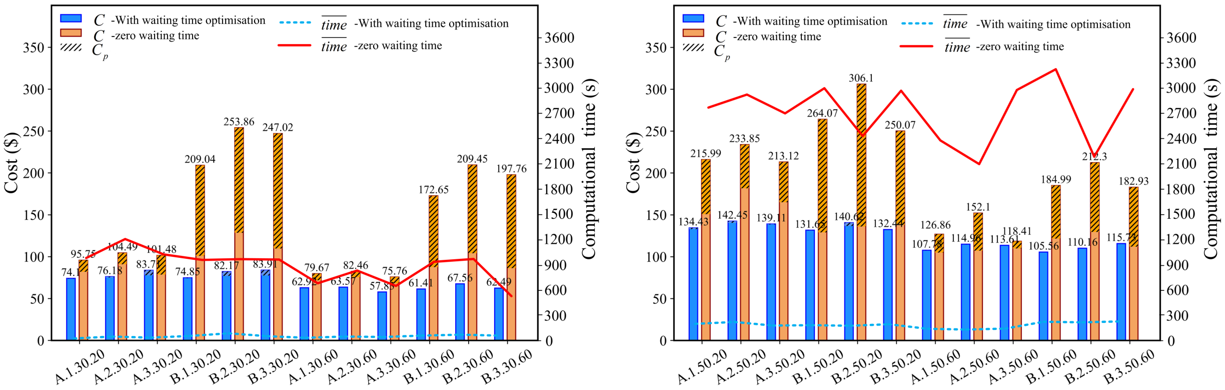


Figure 5: The performance of waiting time optimisation for instances with 30 and 50 customers.

5.4. *Comparisons with different VRPD variants*

To explore the key features of the problem in detail, we perform two additional experiments, namely, comparisons with variants of VRPDs with only single drone visit and hard time windows for all instances A/B.#.50.*wd*.

5.4.1. Multiple visits vs. single visit

In this section, we conduct a series of comparisons with the case in which only single drone visit is allowed to explore the advantages of allowing multiple drone visits. The best results among five runs for all considered instances are summarised in Table 8, where column R_t records the distance travelled by trucks. As shown, allowing multiple drone visits is advantageous, as the total cost is clearly reduced in comparison to that in the single-visit case, in which, although more drone flights are completed (increase by more than 60%), a limited number of customers are still served by drones (decrease by more than 45%); consequently, the travel distance of trucks substantially increases. In addition, it is beneficial to use drones even in the case in which drones can only deliver one parcel per flight, as the total cost is still reduced by an average of 30.02% compared with that for truck-only mode.

Table 8: Impact of allowing multiple drone visits.

Instance	C_t($)	Multiple visits				Single visit				$\Delta_{\#f}$%	$\Delta_{\#d}$%	Δ_{R_t}%	Δ_{C_s,C_m}%	Δ_{C_s,C_t}%
		C_m($)	R_t(km)	$\#f$	$\#d$	C_s($)	R_t(km)	$\#f$	$\#d$					
A.1.50.20	257.22	134.43	74.20	11	32	180.47	111.00	17	17	54.55	-46.88	49.60	34.25	-29.84
A.2.50.20	255.99	142.45	79.60	15	32	168.30	115.00	21	21	40.00	-34.38	44.47	18.15	-34.26
A.3.50.20	261.62	139.11	81.60	12	29	165.13	110.80	18	18	50.00	-37.93	35.78	18.70	-36.88
B.1.50.20	259.87	131.65	68.40	13	29	172.08	112.40	20	20	53.85	-31.03	64.33	30.71	-33.78
B.2.50.20	247.43	140.62	77.00	13	29	181.43	116.00	18	18	38.46	-37.93	50.65	29.02	-26.67
B.3.50.20	260.09	132.44	73.40	13	27	162.87	113.80	20	20	53.85	-25.93	55.04	22.98	-37.38
A.1.50.60	203.57	107.78	43.80	12	33	138.14	82.80	20	20	66.67	-39.39	89.04	28.17	-32.14
A.2.50.60	190.41	114.96	51.00	12	36	139.47	83.20	24	24	100.00	-33.33	63.14	21.32	-26.75
A.3.50.60	155.45	113.61	49.20	13	32	139.66	83.60	22	22	69.23	-31.25	69.92	22.93	-10.16
B.1.50.60	195.88	105.56	35.80	13	38	137.44	78.60	23	23	76.92	-39.47	119.55	30.20	-29.83
B.2.50.60	199.04	110.16	44.00	15	35	138.01	81.20	20	20	33.33	-42.86	84.55	25.28	-30.66
B.3.50.60	200.83	115.73	53.00	12	32	136.75	77.80	22	22	83.33	-31.25	46.79	18.16	-31.91
Avg.										**60.02**	**-46.88**	**64.40**	**24.99**	**-30.02**

Summary. Allowing multiple visits is preferable, as it increases drone utilisation and provides a lower-bound solution in the case of single visit. In addition, it is beneficial to use drones, even in the case in which drones can deliver only one parcel per flight.

5.4.2. Soft time windows vs. hard time windows

This section focuses on exploring the benefits of considering soft time windows. We compare mPDPD-mT with the case in which hard time windows are considered, that is, all services must start within the time windows. Therefore, if a drone arrives before the time window, it will spend extra energy waiting in the latter case. The best results among five runs are reported in Table 9, in which the comparison between the hard time window case and its truck-only mode is also investigated to further confirm the benefits of drone use. Overall, the benefits of soft time windows are obvious, as the cost is usually higher when considering hard time windows for most instances. In particular, although the numbers of flights conducted and drone-served customers are 3. 68% and 3. 97% higher in the case of hard time windows, the average cost still increases by more than 10%, especially for instances with tight time windows. Moreover, the hard time window mode requires more vehicles (marked in bold in Table 9) and does not fully utilise drones. In addition, the results confirm the advantages of drone deployment, as the cost savings over the truck-only mode can still be maintained at 40% in the hard time window case, showing the robustness of our algorithm in solving cases considering different types of time windows. Moreover, considering soft times may increase the complexity of the problem as the CPU time in the mPDPD-mT cases increases.

Table 9: Impact of considering soft time windows.

Instance	Soft time windows					Hard time windows						$\Delta_{\#f}$%	$\Delta_{\#d}$%	$\Delta_{\overline{time}}$%	Δ_{C^h,C^h_t}%	Δ_{C^h,C^s}%
	C^s($)	$\#k$	$\#f$	$\#d$	$\overline{time}$(s)	C^h_t($)	C^h($)	$\#k$	$\#f$	$\#d$	$\overline{time}$(s)					
A.1.50.20	134.43	2	11	32	2804.10	278.54	149.79	**3**	13	33	687.67	18.18	3.13	-75.48	-46.22	11.43
A.2.50.20	142.45	2	15	32	2945.08	262.38	156.42	**3**	14	35	744.39	-6.67	9.38	-74.72	-40.38	9.81
A.3.50.20	139.11	2	12	29	2706.16	274.60	187.66	**4**	13	33	745.05	8.33	13.79	-72.47	-31.66	34.90
B.1.50.20	131.65	2	13	29	3097.83	263.36	190.04	**4**	14	31	833.76	7.69	6.90	-73.09	-27.84	44.35
B.2.50.20	140.62	2	13	29	2475.54	256.97	149.73	**3**	13	34	578.87	0.00	17.24	-76.62	-41.73	6.48
B.3.50.20	132.44	2	13	27	2985.83	261.44	132.44	2	13	27	755.78	0.00	0.00	-74.69	-49.34	0.00
A.1.50.60	107.78	2	12	33	2467.16	203.57	108.64	2	12	33	918.68	0.00	0.00	-62.76	-46.63	0.80
A.2.50.60	114.96	2	12	36	2168.26	209.03	146.35	**3**	14	35	637.28	16.67	-2.78	-70.61	-29.99	27.31
A.3.50.60	113.61	2	13	32	3322.34	211.68	113.61	2	13	32	739.48	0.00	0.00	-77.74	-46.33	0.00
B.1.50.60	105.56	2	13	38	3263.60	195.88	105.56	2	13	38	819.85	0.00	0.00	-74.88	-46.11	0.00
B.2.50.60	110.16	2	15	35	2262.57	202.48	110.16	2	15	35	903.26	0.00	0.00	-60.08	-45.59	0.00
B.3.50.60	115.73	2	12	32	3052.10	218.39	115.73	2	12	32	1392.32	0.00	0.00	-54.38	-47.01	0.00
Avg.												**3.68**	**3.97**	**-70.63**	**-41.57**	**11.26**

Summary. Considering soft time windows is beneficial though may increase problem complexity, as it decreases vehicle usage and provides a lower bound solution in the case of hard time windows. In addition, our algorithm is robust in solving cases considering different types of time windows, and the cost savings over the truck-only mode can still be maintained at 40% even when considering hard time windows, confirming the advantages of deploying drones.

5.5. *Sensitivity analysis*

Considering that the performance of the combined concept depends highly on factors such as time windows, node distributions, percentage of heavy parcels and unit cost, we explore their impacts in this section. Note that we only vary the value of one considered parameter at a time, leaving the rest unvaried to monitor the effect of each parameter on the results.

5.5.1. *Impact of time windows*

A sensitivity analysis is conducted to investigate the impact of time window features related to width (wd) and density (the proportion of customers with time windows, $tw\%$) on the performance of PDmD-sT.

Figure 6 shows the impact of the time window width (wd) on the cost breakdowns in instances with 30 and 50 customers. Instances with wider time windows typically have lower costs, as wider time windows allow vehicles to schedule more flexible routes, thus incurring less cost. Although ILNS aims to reduce the fixed cost at the expense of increasing the operational cost, the sum of these costs shows a similar downwards trend as the total cost. In addition, most services can begin within the time window when $wd \geq 40$, as the penalty cost is reduced to almost zero.

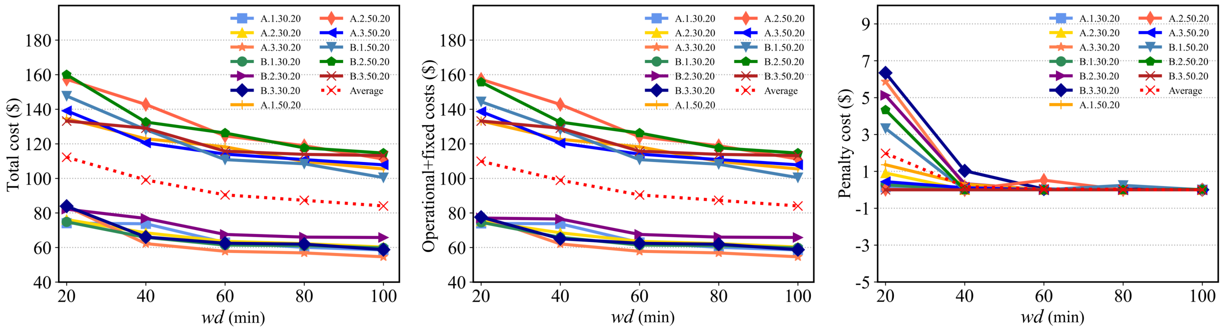


Figure 6: The impact of the time window width on PDmD-sT performance.

Next, we explore the impact of the time window density ($tw\%$) on the cost breakdown in instances with 30 customers, as shown in Figure 7. Although the fixed costs are the same across different scenarios (a vehicle pair is deployed), both the total cost and the operational cost generally increase with $tw\%$. In contrast, the penalty costs are relatively small and are even zero in most instances when $tw\% \leq 60\%$; they begin to increase when $tw\% > 80\%$. Moreover, instances with tighter time windows are generally more strongly affected by $tw\%$. Obviously, a low $tw\%$ helps reduce costs, and more services can begin within or near their time windows.

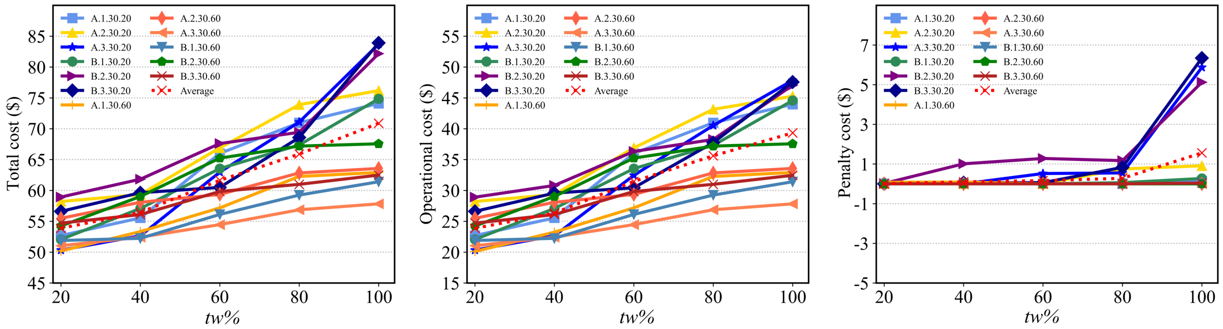


Figure 7: The impact of time window density on PDmD-sT performance.

Summary. Instances with wide or low-density time windows are usually associated with the lowest costs. When the width of the time window is greater than 40 min or the density of the time window is lower than 60%, the penalty costs are relatively small; that is, most customers can be served within their time windows.

5.5.2. *Impact of node distributions*

Given the presumed uniform distribution attributes, we design a set of instances, each with 30 customers, to further investigate the impact of the node distribution on the final results. Inspired by Nguyen et al. (2022), three types of node distributions (*nd*), namely, random (r), clustered (c), and random-clustered (rc), are considered, and the depot position settings (*dp*) follow three patterns: central (c), edge (e) and random (r), located at the centre, edge and a random position of the grid, respectively. The time window type (*tw*) and customer demands generated follow Section 5.1. In addition, three distribution areas are considered, 10×10, 15×15 and 20×20, for comparison.

Table 10: Impact of node distributions on mPDPD-mT performance.

Instance			10×10				15×15				20×20			
tw	*cd*	*dp*	C_t($)	C($)	$\frac{\#d}{\#f}$	Δ_{c,c_t}%	C_t($)	C($)	$\frac{\#d}{\#f}$	Δ_{c,c_t}%	C_t($)	C($)	$\frac{\#d}{\#f}$	Δ_{c,c_t}%
A	c	c	149.53	78.22	2.13	-47.65	196.91	116.13	1.60	-41.02	243.91	182.70	1.50	-25.10
A	c	e	157.39	88.48	2.00	-43.78	213.37	137.06	1.44	-35.76	274.03	221.47	1.38	-19.18
A	c	r	149.59	78.75	2.00	-47.36	197.00	126.39	1.60	-35.84	249.88	196.28	1.40	-21.45
B	c	c	134.91	61.01	2.25	-54.78	179.79	102.25	1.75	-43.13	234.29	142.52	1.56	-39.17
B	c	e	160.79	70.82	2.00	-55.95	197.21	124.43	1.60	-36.90	257.53	170.30	1.56	-33.87
B	c	r	138.72	65.63	2.00	-52.69	185.23	112.52	1.78	-39.25	241.56	148.53	1.56	-38.51
Avg.					**2.06**	**-50.37**			**1.63**	**-38.65**			**1.49**	**-29.55**
A	r	c	156.70	89.08	1.88	-43.15	231.83	132.54	1.60	-42.83	299.43	190.91	1.40	-36.24
A	r	e	159.33	91.07	2.13	-42.84	251.07	143.43	1.67	-42.87	320.25	271.10	1.56	-15.35
A	r	r	157.75	90.06	2.00	-42.91	239.07	137.01	1.45	-42.69	306.18	192.20	1.45	-37.23
B	r	c	171.04	88.01	2.25	-48.54	234.93	139.06	1.60	-40.81	292.49	194.27	1.33	-33.58
B	r	e	172.45	103.41	1.60	-40.03	254.70	157.70	1.55	-38.08	304.55	211.35	1.33	-30.60
B	r	r	171.91	92.46	1.89	-46.22	242.70	142.36	1.55	-41.34	302.05	203.42	1.27	-32.65
Avg.					**1.96**	**-43.95**			**1.57**	**-41.44**			**1.39**	**-30.94**
A	rc	c	139.50	82.81	2.14	-40.64	181.28	107.64	1.80	-40.62	229.62	138.89	1.70	-39.51
A	rc	e	152.71	88.43	2.43	-42.09	207.75	125.52	1.89	-39.58	277.83	192.35	1.45	-30.77
A	rc	r	139.53	86.70	2.00	-37.86	185.92	116.33	1.89	-37.43	258.13	162.06	1.55	-37.22
B	rc	c	145.58	73.66	2.11	-49.40	196.88	119.59	1.89	-39.26	231.77	141.47	1.64	-38.96
B	rc	e	151.77	80.29	2.00	-47.10	210.64	128.94	2.00	-38.79	263.85	169.53	1.50	-35.75
B	rc	r	145.98	76.51	2.11	-47.59	197.64	120.70	1.78	-38.93	260.80	163.06	1.55	-37.48
Avg.					**2.13**	**-44.11**			**1.88**	**-39.10**			**1.57**	**-36.61**

Table 10 shows the lowest total costs for truck-only mode and the PDmD-sT among five runs, as well as the corresponding percentage gap and ratio of the number of drone-served customers to that of flights conducted ($\frac{\#d}{\#f}$). Overall, the total costs for both modes increase as the distribution area expands; thus, fewer customers are served by one flight. Moreover, the changes are most significant when the distribution area expands to 20×20. In addition, the costs among different node distributions are similar. Specifically, the cost of PDmD-sT is lowest when nodes are clustered (c), followed by the costs in the random-clustered (rc) and random distribution (r) cases. In addition, we observe that the location of the depot clearly affects both modes. In particular, when the depot is located along an edge, the cost is typically the highest, whereas when it is at the centre, the cost is usually the lowest. Furthermore, the average cost compared with that in truck-only mode is reduced by approximately 40%, and the cost is reduced by more than 40% when the nodes are distributed in a 10 × 10 area, which is consistent with the results in Section 5.2.2, highlighting the robust performance of the proposed ILNS for different distribution types.

Summary. The proposed ILNS scheme can achieve robust performance for different topologies. The node distribution and distribution density, especially the location of the depot, clearly affect the combined mode. A high-density distribution or centrally located depot leads to the highest cost savings compared with the cost in truck-only mode, and more customers can be served per flight.

5.5.3. *Impact of percentage of heavy parcels*

To assess the performance of the algorithm under different demand patterns, we investigate the impact of changes in the proportions of customers whose parcels cannot be carried by drones (namely, heavy parcels) with instances A/B.#.30.*wd*. Figure 8 displays the best results among five runs in terms of total costs, number of drone-served customers, and distance travelled by trucks for different proportions of heavy parcels (*hp*). Overall, the cost gradually rises with increasing proportion of heavy parcels; accordingly, fewer customers are served by drones, and longer distances are travelled by trucks. However, even though 40% of the customers must be served by trucks, approximately 50% of the customers are still served by drones for most instances, maintaining a relatively high proportion of customers served by drones, indicating that the ILNS approach remains effective when the proportion of heavy parcels changes. Moreover, the combined mode is still advantageous over the truck-only mode (hp=100%), even though only 20% of parcels can be carried by drones (hp=80%). In addition, the total cost and distance travelled by trucks are usually lower in instances with wider time windows, and more customers are served by drones.

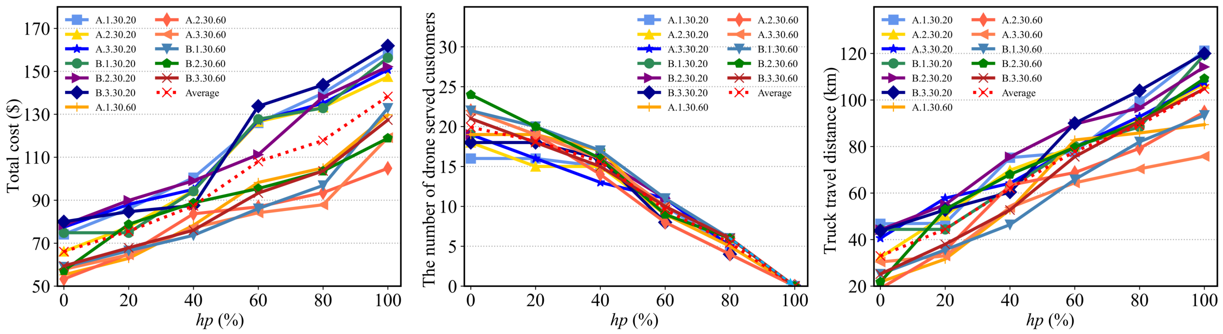


Figure 8: The impact of the percentage of heavy parcels on PDmD-sT performance.

Summary. The proposed ILNS can maintain effectiveness when the proportion of heavy parcels changes. Although the total cost increases and the number of customers served by drones decreases because more parcels cannot be carried by drones, a considerable proportion of drone-served customers can still be maintained.

5.5.4. Impact of cost parameters

In this section, the impacts of cost parameters, such as the truck–drone cost ratio per distance and the unit penalty costs (penalty cost rates), on the PDmD-sT results in cases with 30 customers are assessed. As the unit cost ratio (ς) is important for determining the cost advantage of PDmD-sT over truck-only mode, we set six values for ς (assuming that the unit truck cost is relatively fixed) and run the ILNS five times for each tested instance in each scenario. The best results are displayed in Figure 9 , with reported results similar to those of the previous figure. Although the cost advantage of PDmD-sT decreases with decreasing ς because of higher drone costs, the total cost is still obviously less than that of the truck-only mode, even if $\varsigma = 1$. Moreover, as long as $\varsigma \geq 4$, drones can continue to serve more than 50% of customers, and the distance travelled by trucks is relatively small.

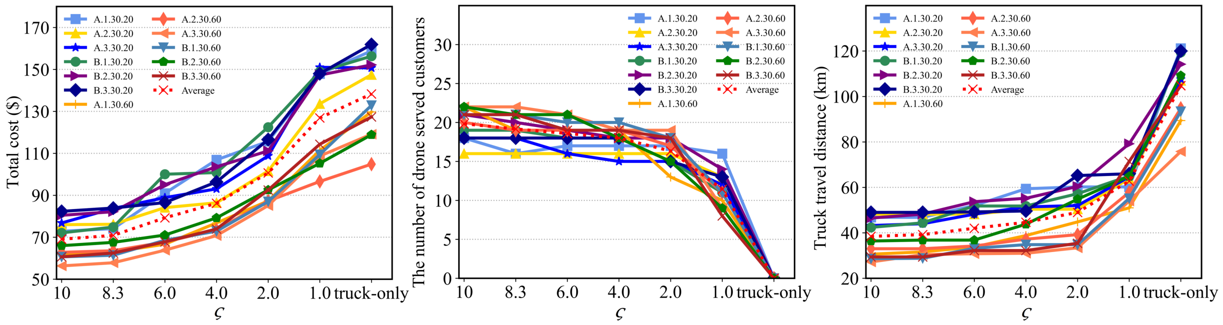


Figure 9: The impact of drone cost on PDmD-sT performance.

In addition, considering that penalty cost rates (c^s, c^e) affect overall system punctuality, as well as the variation in cost breakdowns, we investigate the impact of penalty cost rates on outputs by multiplying the rates by different σ values. The best results from five runs are displayed in Figure 10. In general, the total cost increases with σ, especially for instances with tighter time windows. Moreover, the average penalty costs are relatively small and stable, but the total cost and the other two sub-costs increase slowly with σ on average, as ILNS aims to reduce the cost of penalties with the trade-off of increasing the other types of cost so that more services can begin within or near their time windows. Therefore, service punctuality can be flexibly satisfied by increasing or decreasing penalty cost rates, and a higher penalty cost tends to enable more customers to be served within the specified time windows.

Summary. Although the benefits of PDmD-sT are sensitive to the cost of drones, the total cost is still significantly less than that of the truck-only mode as long as the cost ratio $\varsigma \geq 4$. Even if the cost ratio is

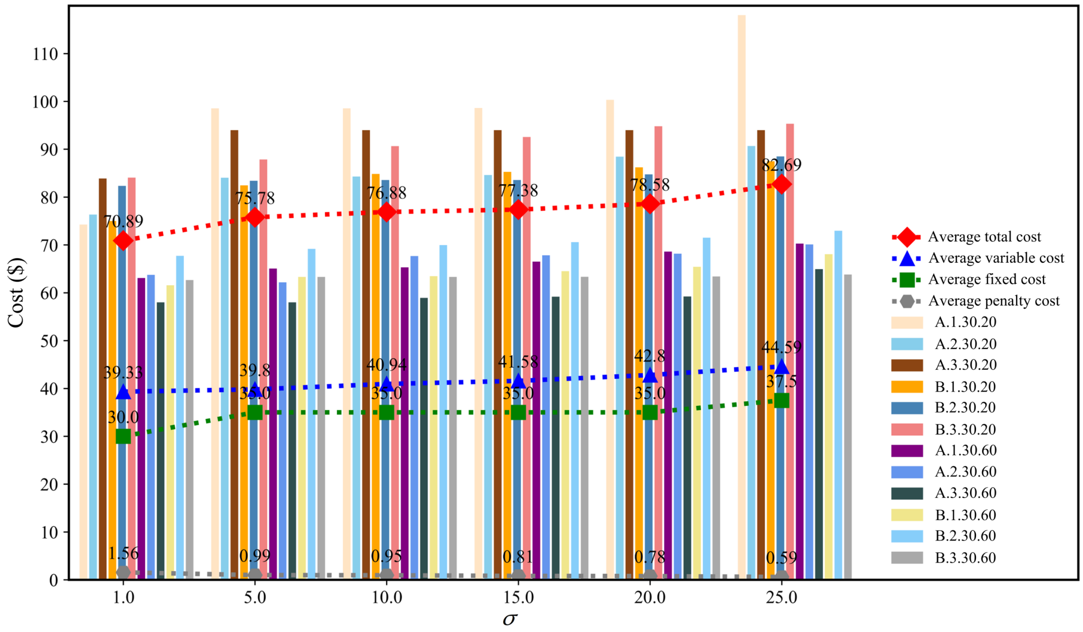


Figure 10: The impacts of penalty cost rates on PDmD-sT performance.

reduced to 1, advantages over the truck-only mode still exist, including in cases with relatively wide time windows. In addition, a higher penalty cost ensures that more customers are served in or near the time window, but at the expense of increasing total costs.

6. Conclusions and future studies

In this study, we assess a new VRPD variant for last-mile logistics, namely, the pickup and delivery problem with multi-visit drones considering soft time windows. In this problem, multiple truck–drone pairs are available to collaboratively perform pickup and delivery services for customers within pre-specified periods. Moreover, some paired customers are considered. Each customer has a soft time window, and a service starting outside this window will incur a penalty cost. Trucks can wait at each node they visit, and drones can be operated multiple times, serving multiple customers at a time. We take minimising total cost, including penalty costs, as the objective, and the problem is modelled as a mixed-integer nonlinear program with several enhanced inequalities, in which the battery power consumption during hovering is nonlinear and approximated via a piecewise linear approach. In addition, a custom improved metaheuristic approach is developed to address large-size problems, and an exact method with theoretical cuts is designed to evaluate drone energy feasibility and determine truck waiting times. Extensive numerical tests were conducted to assess the benefits of the combined mode and verify that the proposed metaheuristic can produce a high-quality solution. The addition of truck waiting time optimisation was then thoroughly evaluated, highlighting its effectiveness in cost reduction. Finally, a set of comparisons was conducted to quantitatively explore problem features and several key parameters, and the results demonstrated their profound impacts on problem solutions and yielded valuable managerial insights.

Due to some critical barriers in last-mile logistics for the development of drone transport, such as technology, infrastructure and regulatory challenges, drone delivery has remained at a small scale. Nevertheless, both the academic and industrial communities are optimistic about the broad adoption of drone deliveries, as mentioned in the Introduction. With continuous development, it is not surprising that in the foreseeable future, drone adoption will become increasingly common.

In future studies, scenarios in which each truck is equipped with multiple drones, trucks can launch and retrieve any drones nearby, or multiple flights can be performed by a drone from the same location can be considered, as these scenarios would be more universal. Moreover, considering uncertainty in drone travel times could be another promising extension of the problem, as drone energy consumption varies according to weather conditions. In this work, the method for waiting time optimisation is effective but may be time consuming, so it would also be interesting to include other heuristic algorithms in future works. Moreover, it is also very interesting to further study the impact of the soft time windows on customer satisfaction from the perspective of service recipients.

References

Agatz, N., Bouman, P., & Schmidt, M. (2018). Optimization approaches for the traveling salesman problem with drone. *Transportation Science*, 52(4), 965–981.

Cheng, C., Adulyasak, Y., & Rousseau, L.-M. (2020). Drone routing with energy function: Formulation and exact algorithm. *Transportation Research Part B: Methodological*, 139, 364–387.

Christiaens, J. & Vanden Berghe, G. (2020). Slack induction by string removals for vehicle routing problems. *Transportation Science*, 54(2), 417–433.

Coindreau, M.-A., Gallay, O., & Zufferey, N. (2021). Parcel delivery cost minimization with time window constraints using trucks and drones. *Networks*, 78(4), 400–420.

Dorling, K., Heinrichs, J., Messier, G. G., & Magierowski, S. (2016). Vehicle routing problems for drone delivery. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(1), 70–85.

Gu, R., Liu, Y., & Poon, M. (2023). Dynamic truck–drone routing problem for scheduled deliveries and on-demand pickups with time-related constraints. *Transportation Research Part C: Emerging Technologies*, 151, 104139.

Huang, S.-H., Huang, Y.-H., Blazquez, C. A., & Chen, C.-Y. (2022). Solving the vehicle routing problem with drone for delivery services using an ant colony optimization algorithm. *Advanced Engineering Informatics*, 51, 101536.

Jiang, J., Dai, Y., Yang, F., & Ma, Z. (2024). A multi-visit flexible-docking vehicle routing problem with drones for simultaneous pickup and delivery services. *European Journal of Operational Research*, 312(1), 125–137.

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.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671–680.

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.

Lehmann, J. & Winkenbach, M. (2024). A matheuristic for the two-echelon multi-trip vehicle routing problem with mixed pickup and delivery demand and time windows. *Transportation Research Part C: Emerging Technologies*, 160, 104522.

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.

Li, J., Cang, L., Wu, Y., & Zhang, Z. (2025). Two-echelon collaborative many-to-many pickup and delivery problem for agricultural wholesale markets with workload balance. *Omega*, 130, 103164.

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., Chen, Y., & Li, D. (2024a). The multi-visit drone-assisted pickup and delivery problem with time windows. *European Journal of Operational Research*, 314(2), 685–702.

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.

Meng, S., Li, D., Liu, J., & Chen, Y. (2024b). The multi-visit drone-assisted routing problem with soft time windows and stochastic truck travel times. *Transportation Research Part B: Methodological*, 190, 103101.

Moon, M. (2021). Alphabet's wing tests drone deliveries from shopping center rooftops in Australia, URL `https://techcrunch.com/2021/10/06/alphabets-wing-tests-drone-deliveries-from-shopping-center-rooftops-in-australia/`.

Murray, C. C. & Chu, A. G. (2015). The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. *Transportation Research Part C: Emerging Technologies*, 54, 86–109.

Nguyen, M. A., Dang, G. T.-H., Hà, M. H., & Pham, M.-T. (2022). The min-cost parallel drone scheduling vehicle routing problem. *European Journal of Operational Research*, 299(3), 910–930.

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.

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.

Shakir, U. (2023). Wing's new delivery network lets its drones deliver more with fewer stops, URL `https://www.theverge.com/2023/3/9/23631068/wing-delivery-network-drone-autoloader`.

Shaw, P. (1997). A new local search algorithm providing high quality solutions to vehicle routing problems. *APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK*, 46.

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.

Toor, A. (2016). This startup is using drones to deliver medicine in Rwanda, URL `https://www.theverge.com/2016/4/5/11367274/zipline-drone-delivery-rwanda-medicine-blood`.

Wang, L., Ding, Y., Chen, Z., Su, Z., & Zhuang, Y. (2024). Heuristic algorithms for heterogeneous and multi-trip electric vehicle routing problem with pickup and delivery. *World Electric Vehicle Journal*, 15(2), 69.

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

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

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.

Yin, Y., Li, D., Wang, D., Yu, Y., & Cheng, T. (2024). Truck-drone pickup and delivery service optimization with availability profiles. *Naval Research Logistics*.

Zhang, Z., Che, Y., & Liang, Z. (2024). Split-demand multi-trip vehicle routing problem with simultaneous pickup and delivery in airport baggage transit. *European Journal of Operational Research*, 312(3), 996–1010.

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.

Zhou, H., Qin, H., Cheng, C., & Rousseau, L.-M. (2023). An exact algorithm for the two-echelon vehicle routing problem with drones. *Transportation Research Part B: Methodological*, 168, 124–150.

Zhou, S., Zhang, D., Yuan, W., Wang, Z., Zhou, L., & Bell, M. G. (2024). Pickup and delivery problem with electric vehicles and time windows considering queues. *Transportation Research Part C: Emerging Technologies*, 167, 104829.

Zipline (2025). Deliver food to your custoners by drone in under 10 minutes, URL `https://www.flyzipline.com/solutions/restaurants`.