The Multi-visit Traveling Salesman Problem with Multi-Drones

Zhihao Luo[**,a,c], Binbin Pan[*,**,c], Zhenzhen Zhang[b], Zhong Liu[a], Andrew Lim[c]

[a] *College of Systems Engineering, National University of Defense Technology, Changsha 410073, China*
[b] *School of Economics and Management, Tongji University, Shanghai 200092, China*
[c] *Department of Industrial Systems Engineering and Management, National University of Singapore, Singapore 117576*

Abstract

The use of drones for parcel delivery has recently attracted wide attentions due to its potential in improving efficiency of the last-mile delivery. Though attempts have been made on combined truck-drone delivery to deploy multiple drones which can deliver multiple packages per trip, many have placed extra assumptions to simplify the problem. This paper investigates the multi-visit traveling salesman problem with multi-drones (MTSP-MD), whose objective is to minimize the time (makespan) required by the truck and the drones to serve all customers together. The energy consumption of the drone depends on the flight time, the self-weight of the drone and the total weight of packages carried by the drone, which declines after each delivery throughout the drone flight. The MTSP-MD problem consists of three complicated sub-problems, namely (1) the drone flight problem with both a payload capacity constraint and an energy endurance constraint, (2) the traveling salesman problem with precedence constraints, and (3) the synchronization problem between the truck route and the drone schedules. The problem is first formulated into a mixed-integer linear program (MILP) model and we propose a multi-start tabu search (MSTS) algorithm with tailored neighborhood structure and a two-level solution evaluation method that incorporates a drone-level segment-based evaluation and a solution-level evaluation based on the critical path method (CPM). The experimental results demonstrate the accuracy and efficiency of our proposed algorithm and show a significant cost reduction when considering multi-visits, multi-drones, and drones with higher payload capacity and higher battery capacity.

Key words: traveling salesman, drone, last-mile delivery, integer programming, heuristics

1. Introduction

The concept of using unmanned aerial vehicle (UAV) or drones for package delivery was first proposed and tested in 2013 by Amazon in the US for the last-mile delivery of small packages . The effort sparked significant interest from various companies across R&D (e.g., Google, Workhorse), logistics (e.g., UPS), e-commerce (e.g., JD), and automobile manufacturers (e.g., Mercedes-Ben, Rinspeed), all of which have invested heavily in related research in recent years. While factors such as high density, tall infrastructure, and strict urban airspace guidelines have made it difficult to implement drone delivery in urban areas (Otto et al., 2018), this mode of delivery can potentially bring cost benefit and better service quality for the last-mile delivery in less densely populated areas with a widely spread populace and infrastructure with sufficient airspace for drone flight.

From an academic perspective, Murray & Chu (2015) proposed the flying sidekick traveling salesman problem (FSTSP), which is the first paper to outline the scenario of deploying both a delivery truck and a drone together for package delivery in logistics distribution. FSTSP allows drones to deliver only one package per flight and studies the synchronization problem between truck route and drone flights. Subsequent research extended this basic FSTSP with more complex synchronization constraints, such as allowing multiple packages per flight, deploying multiple drones per truck, and various other constraints. One important extension introduced in Murray & Raj (2020) is the multiple flying sidekicks traveling salesman problem (mFSTSP) with multiple drones on the truck. However, the mFSTSP assumes: (1) a drone is capable of carrying only a single package at a time, and (2) the truck is only allowed to launch or retrieve one drone at any point in time. Recent technological advancements have made the first assumption less relevant because newer drone models capable of carrying multiple consumer packages simultaneously have been proposed and developed (Wang & Sheu, 2019; Kitjacharoenchai et al., 2020; Liu et al., 2020; Poikonen & Golden, 2020b). Furthermore, as more researches have been devoted to investigate the deployment of multiple drones per truck to increase the efficiency of last-mile deliveries (Kitjacharoenchai et al., 2019; Murray & Raj, 2020; Yoon, 2018; Wang & Sheu, 2019), it is

*Corresponding author
**These authors contributed equally to this work.
Email address: panbinbin11@gmail.com (Binbin Pan)

beneficial to design automatic systems that can handle multiple launching and retrieval of drones simultaneously to increase efficiency of the system and reduce manpower requirement.

This paper aims to exclude the above-mentioned two assumptions for a futuristic problem. We formulate this problem as a multi-visit traveling salesman problem with multi-drones (MTSP-MD), in which each drone is capable of delivering packages to multiple customers per flight. All drones are limited by both energy consumption constraints based on flight-time and payload, as well as the maximum payload capacity constraints. Launch and retrieval operations can be executed at the depot and any customer node, and multiple delivery operations can be carried out concurrently at a node using an automated flight control system on the truck. An example of the solution for problem with 2 drones and 30 customers is illustrated in Figure 1. The objective is to minimize the latest time (i.e. makespan) required by the truck and drones to serve all customers. The MTSP-MD problem can be viewed as a combination of three sub-problems: (1) a drone routing problem with payload capacity and flight endurance constraints, (2) a TSP with precedence constraints, and 3) a synchronization problem between the truck route and multiple drone schedules.

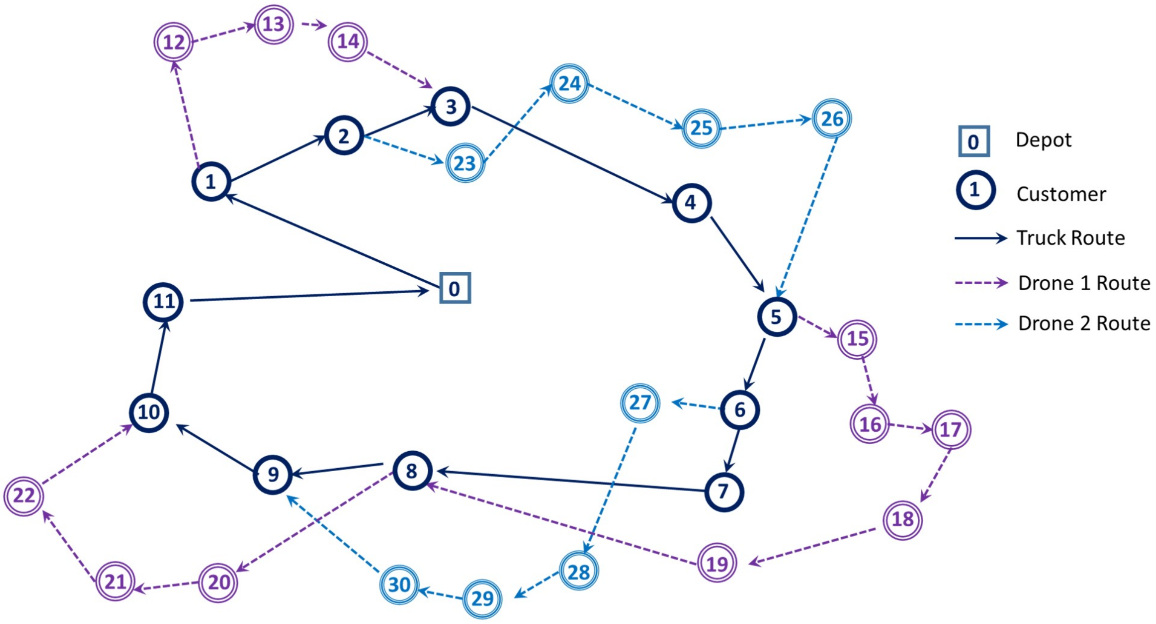

Figure 1: An example of MTSP-MD Solution with 30 customers and 2 drones.

We propose a mixed-integer linear programming (MILP) model for the MTSP-MD problem, which can be used to study the unique features of the problem and is solvable directly using off-theshelf solvers on small-scale instances. A multi-start tabu search (MSTS) heuristic is designed to solve larger instances with tailored relocation and swap neighborhood structures and a two-level feasibility-evaluation method, which consists of a drone-level segment-based evaluation method (Vidal et al., 2014) and a solution-level evaluation based on the critical path method (CPM) (Evans, 1992). The correctness and efficiency of the proposed algorithm is verified on newly generated instances extended from Solomon's instances (Solomon., 1987). Experimental results demonstrate the significant cost reduction when considering multi-visits, multi-drones and drones with more payload capacity and better flight endurance. We see these results as strong motivation for further academic research as well as an effort to commercialize this application for the logistics industry.

This paper is organized into the following sections. Section 2 presents a review on relevant literature. Section 3 introduces the problem in detail with a mathematical model. Section 4 describe the different components of the MSTS algorithm. Detailed results of the experimental studies are presented in Section 5. Finally, concluding remarks on this work and implications for future research are provided in Section 6.

2. Related works

This review focuses on literature where trucks and drones are deployed together for last-mile delivery. Readers may refer to Otto et al. (2018); Chung et al. (2020); Boysen et al. (2020); Macrina et al. (2020) for comprehensive reviews on other drone applications, such as disaster management, remote reconnaissance, data collection, and Intelligence, Surveillance and Reconnaissance (ISR).

2.1. Fundamental issues in coordinated routing problems with trucks and drones

In FSTSP (Murray & Chu, 2015), the drone is allowed to deliver only one package per flight with a maximum flying distance constraint. When not in use, it is carried by the truck to the next location. The drone can be launched and retrieved at any customer location and the depot and is allowed to directly deliver a parcel to a customer from and then return to the depot. This helps to significantly simplify the synchronization problem as these direct drones trip can be trivially scheduled before the departure of the truck. Various papers have extended the FSTSP to study the development of new models (Dell'Amico et al., 2019), formulate extensional constraints (Jeong et al., 2019), and propose different meta-heuristics (de Freitas & Penna, 2018, 2020). The traveling salesman problem with drones (TSP-D) extends the FSTSP but disallows drone trips that start and end at the depot directly. Both exact approaches (Agatz et al., 2018; Poikonen et al., 2019; Bouman et al., 2018) and heuristic approaches (Yurek & Ozmutlu, 2018; Ha et al., 2018, 2020; El-Adle et al., 2019; Wang et al., 2019b) have been applied to solve the TSP-D problem, while other research efforts introduced new and practical route characteristics (Marinelli et al., 2017; Boysen et al., 2018; bin Othman et al., 2017; Poikonen & Golden, 2020a; Carlsson & Song, 2018). The vehicle routing problem with drones (VRP-D) is another extension of the FSTSP that involves deployment of multiple trucks (Poikonen et al., 2017; Wang et al., 2017; Schermer et al., 2018, 2019a,b; Kitjacharoenchai et al., 2019; Chiang et al., 2019; Sacramento et al., 2019; Das et al., 2020). The FSTSP, TSP-D, and VRP-D all have a simplified synchronization problem with one drone per truck. Furthermore, the limitation of one customer per drone trip allows pre-computation of all feasible drone trips as a three-dimensional tuple in $O(n^3)$ time for the drone routing problem.

Currently, more complicated problems are being proposed to extend FSTSP in two directions: (1) multiple drones per truck, and (2) multiple deliveries per drone trip with a more sophisticated model for drone flight endurance.

2.2. *Extension with multi-drones per truck*

Yoon (2018) and Tu et al. (2018) individually proposed the traveling salesman problem with multiple drones (TSP-mD) with a maximum distance constraint. Most literature on this extension enforces simplified constraints to reduce the complexity of multi-drone scheduling, for example, some require the truck to remain stationary before the drones complete delivery and return (Cojocaru et al., 2017; Peng et al., 2019; Moeini & Salewski, 2019; Moshref-Javadi et al., 2020a,b), while others restrict trucks to travel on a pre-determined route (Boysen et al., 2018; Hu et al., 2019). Murray & Raj (2020) compared several different models for the flight endurance and adopted the model based on drone payload for the mFSTSP. The mathematical model in Murray & Raj (2020) employs a large number of binary variables to represent precedence relationships between any two operations at a node, which greatly increases the complexity of solving the synchronization problem. Other research has chosen to simplify this sequencing problem by assigning different priorities to operations (Yoon, 2018), or allowing only a single operation at a node (Kitjacharoenchai et al., 2020). It is noteworthy to mention that Campbell et al. (2018b) evaluated the hybrid truck-drone delivery system with multiple drones using a continuous approximation modeling technique. Its results clearly suggest a great advantage with multiple drones per truck over the truck-only delivery.

2.3. *Extension with multi-visits per drone flight*

Multi-visits per drone trip has mainly been explored for applications in surveillance (Campbell et al., 2018a) or reconnaissance (Luo et al., 2017, 2018; Liu et al., 2019). Applications of this extension in logistics include Liu et al. (2020) and Wang et al. (2019a) which proposed heuristics to solve the multi-visit drone delivery problem with a single truck-drone team using a piecewise linear energy consumption function. In contrast, Gonzalez-R et al. (2020) adopted a linear energy consumption function and developed an iterative greedy search algorithm to solve a similar problem. However, these efforts restrict the problem to a single drone per truck.

The synchronization problem for multi-visits becomes more complex to model and solve with multiple drones per truck. Poikonen & Golden (2020b) proposed the k-multi-visit drone routing problem (k-MVDRP) with multiple drones on a truck, which requires that: (1) all drone trips from the same launch node must end at the same retrieval node, and (2) after launching any drones, the truck must travel directly to the retrieving node without stopping or passing by any other nodes. As such, the feasibility of each drone flight can be evaluated independently of other flights and the synchronization problem is simplified. Wang & Sheu (2019) proposed another variant to restrict the retrieval operations to the docking nodes or the depot. This constraint can be relaxed with automated flight control systems on the truck so that all customer nodes can serve as retrieval nodes. Kitjacharoenchai et al. (2020) formulated the two-echelon vehicle routing problem with drones (2EVRPD) with multi-visits and multiple drones per truck, which enforces a drone flight endurance model based on the maximum length of a drone route and restricts at most one launching or retrieving operation at each customer node. As a result, more customers must be assigned to the truck route as launching or retrieving nodes when more drone trips are created. This contradicts the original motivation to reduce makespan with more parallel drone deliveries.

Table 1: Summary of related papers

	Papers	Problem	NoD	NoT	NoC	Endurance	Retrieving Nodes
Fundamental	Murray & Chu (2015); Dell'Amico et al. (2019) de Freitas & Penna (2018, 2020)	FSTSP	1	1	1	Max Distance	Customers
	Agatz et al. (2018); Poikonen et al. (2019); Bouman et al. (2018) Yurek & Ozmutlu (2018); Ha et al. (2018, 2020) El-Adle et al. (2019); Wang et al. (2019b)	TSP-D	1	1	1	Max Distance	Customers
	Poikonen et al. (2017); Wang et al. (2017); Das et al. (2020) Schermer et al. (2018, 2019b); Sacramento et al. (2019)	VRP-D	1	M	1	Max Distance	Customers
Multi-drones	Yoon (2018); Tu et al. (2018)	TSP-mD	M	1	1	Max Distance	Customers
	Kitjacharoenchai et al. (2019)	MTSP-D	M	M	1	None	Customers
	Murray & Raj (2020)	mFSTSP	M	1	1	Max Energy	Customers
Multi-visit	Luo et al. (2017, 2018); Liu et al. (2019)	2E-GURP	1	1	M	Max Distance	Docking Nodes
	Poikonen & Golden (2020b)	k-MVDRP	M	1	M	Max Energy	Selected Customers
	Kitjacharoenchai et al. (2020)	2EVRPD	M	M	M	Max Distance	Customers
	Wang & Sheu (2019)	VRP-Ds	M	M	M	Max Distance	Docking Nodes
	This paper	MTSP-MD	M	1	M	Max Energy	Customers

2.4. Summary

Table 1 summarizes the literature mentioned in this section into groups, in which NoD represents the number of drones per truck, NoT represents the number of trucks, NoC represents the number of customers that drone can serve in a flight, and M abbreviates "Multiple". The column Endurance indicates the constraints on the flight time of drones. Compared to previous literature, the proposed MTSP-MD is more practical as (1) it employs a drone flight endurance model based on the payload of the multiple parcels and its flight time, (2) it deploys multiple drones from the truck, (3) it allows multiple deliveries per drone trip, and (4) it allows multiple operations at any customer node and the depot(s).

3. Problem statement and mathematical model

This section provides a formal description and an MILP formulation of the MTSP-MD problem. A summary of parameter notations used is presented in Table 2. Notations for the truck and the drones are labelled with superscript "G" and "U" respectively for better clarity, where "G" stands for ground vehicle (truck) and "U" stands for UAV (drone).

3.1. Notations and problem description

Let $C = \{1, \ldots, n\}$ be the set of customers, each of which requires a package of weight w_i. Node $n+1$ is designated as the depot and $V = C \cup \{n+1\}$ represents the set of all nodes. Then the

MTSP-MD problem is defined over the graph (V, E), where the arc set $E = \{(i, j)|i, j \in V,\ i \neq j\}$. A single truck is equipped with a homogeneous fleet of R drones to serve all customers. Each customer $i \in C$ must be served by either the truck or one of the drones exactly once, and customers visited by the truck must be serviced directly by the truck. The delivery to customer i takes a service time of s_i^U or s_i^G if it is served by a drone or by the truck respectively. The travel speeds of the truck and the drones are different, which leads to different travel times. The travel times for arc $(i, j) \in E$ are t_{ij}^G and t_{ij}^U for the truck and the drones respectively.

The truck is capable of carrying all packages and drones on board and has no constraints on its travel distance, while a drone has a self-weight w^U and a maximum weight capacity Q on the carried packages per trip. Therefore, a set of customers $C^G \subseteq C$ must be served by the truck due to the maximum capacity of drones. The set $C^U = C \setminus C^G$ represents customers who can be served by either the truck or a drone.

A drone can be launched and retrieved by the truck at the depot or a customer's node and is capable of carrying multiple packages at the same time. A drone trip is defined as a single drone flight used in the solution, which contains a launch node, a sequence of customers serviced by the drone, and a retrieval node. A drone trip must adhere to the maximum package weight (Q) constraint and the maximum battery capacity (θ) constraint. The drone's endurance model will be discussed in Section 3.2.

If a drone is not used, it will be carried by the truck along the truck route to the next node. We assume that a drone cannot be retrieved at the same node where the drone trip originates from. This is beneficial as the drone flies faster than the truck and concurrent movements of both drones and truck can increase efficiency of delivery. The time for loading, taking off and landing are negligible compared to the flight time from the launch node to the retrieval node and are hence assumed to be zero in this study. When a dispatched drone arrives at the retrieval node earlier than the truck, it is allowed to hover at the location until the truck arrives. During the hovering period, the drone consumes energy and must be retrieved by the truck before it runs out of energy.

Multiple launch and retrieval operations are allowed to happen concurrently at a node, which are assumed to be handled by an automated flight control system as in Poikonen & Golden (2020b). Specifically, the MTSP-MD problem adopts the policy that all dispatched drones from the same node should be launched at the same time when the truck departs from the node.

A drone can perform multiple non-overlapping drone trips in the MTSP-MD. Whenever a drone returns to the truck, its battery will be replaced with a fully-charged one for the next trip and replacement time is assumed to be negligible.

3.2. Drone flight endurance

Zhang et al. (2020) highlights that drone energy consumption is affected by various groups of factors, such as the drone design, environment, drone dynamics and delivery operations, and provides a review and comparisons on the various energy consumption models proposed for drones delivery. In this paper, we extend the energy consumption model in Liu et al. (2020), in which the energy consumption rate is positively correlated with the self-weight, payload, and flight speed of the drone. Assuming that drones travel at a constant speed during flight, we introduce a parameter α to represent energy consumption rate per weight per time. Then the energy consumption P_{ij} for a drone flight along arc $(i, j) \in E$ in the MTSP-MD depends on the flight time as well as the total weight of the drone and parcels:

$$P^F_{ij} = \alpha \times \left(w^U + w^U_i\right) \times t^U_{ij}, \tag{a1}$$

where w^U_i denotes the total payload of the drone when it leaves node i. The drone consumes energy while serving a customer or hovering over the retrieval node, of which the energy consumption parameter is assumed to be α too. Since the drone only hovers at a location during retrieval, this implies that a hovering drone has zero payloads. Meanwhile, when a drone is serving a customer, the total weight on board is equal to the weight of the current customer's package plus the total payload of the drone when it departs. Hence, the actual energy consumption P^S_i for serving customer $i \in C$ and the energy consumption P^T from hovering can be simplified into the following:

$$P^T = \alpha \times w^U \times t^H, \tag{a2}$$

$$P^S_i = \alpha \times \left(w^U + w_i + w^U_i\right) \times s^U_i, \tag{a3}$$

where t^H represents the duration of the drone hovering time.

Table 2: Parameter notation

Notation	***Description***
C	*Set of all customers*
C^U	*Set of customers that can be served either by a drone or by the truck*
C^G	*Set of customers that can only be served by the truck*
V	*Set of all nodes*
R	*The maximum number of drones a truck can carry*
K	*The set of drone trips used in a solution*
t^U_{ij}	*Time cost when drone flies along arc* $(i, j) \in E$
t^G_{ij}	*Time cost when truck travels along arc* $(i, j) \in E$
Q	*The maximum weight capacity of carried packages by a drone*
s^U_i	*The time required for a drone to serve customer i*
s^G_i	*The time required for the truck to serve customer i*
w_i	*The weight of the package required by customer i*
w^U	*The self-weight of a drone*
θ	*The maximum battery capacity of the drone*
α	*The energy consumption rate per weight per time*
M	*A sufficiently large positive number for BIG-M method*

3.3. Objective and decision variables

The objective of this problem is to minimize the time required for the truck and all drones to deliver all assigned parcels and return to the depot (i.e., to minimize makespan). This is achieved by solving for a series of decision variables, of which a summary is provided in Table 3.

We discuss the decision variables related to the truck route first. z^G_i is a binary decision variable that indicates whether the customer $i \in C$ is visited and served by the truck. y_{ij} is a binary decision variable that indicates if the truck travels along arc $(i, j) \in E$. The schedule of the truck is defined based on two continuous decision variables as below. $t^{G,A}_i \geq 0$ represents the truck's arrival time at node $i \in V$, and $t^{G,L}_i \geq 0$ captures the departure time of the truck from node $i \in V$.

Since drones are homogeneous, we do not assign drone trips to drones specifically in the model but just ensure to have enough drones on board of truck for drone trips. Specifically, at most $n-1$ customers can be assigned to drone trips because drones are not allowed to launch from and return to the depot to perform a drone trip directly in our setting. Therefore, at most $n-1$ drone trips exist in a solution. Then, we can define $K = \{1, 2, \ldots, n-1\}$ to represent the set of available drone trips for any particular solution in the mathematical model. A drone trip $k \in K$ will be set to empty if it is not used in the solution. Moreover, an integer decision variable r_i $\forall i \in V$ is defined to indicate the number of drones carried by the truck when it departs from node i, and updated based on the number of drones transported from the previous location, and the numbers of drones launched and retrieved at i.

We define the decision variables for the drone trips as below. x_{ijk} is a binary variable that denotes if the k-th drone trip travels along arc (i, j), $\forall (i, j) \in E,\ k \in K$. z^U_{ik} is a binary decision variable that indicates whether the customer $i \in C$ is served in the drone trip $k \in K$. Similar to $t^{G,A}_i$ and $t^{G,L}_i$, continuous decision variables $t^{U,A}_i \geq 0$ and $t^{U,L}_i \geq 0$ indicate the drone's arrival time at and departure time from customer i respectively. For a launch node, $t^{U,L}_i$ is equal to $t^{G,L}_i$ as both the truck and the drones will depart from the customer at the same time. For a retrieval node, $t^{U,A}_i$ indicates the arrival time of the last drone at node i. A special set of constraints is used to enforce the flight endurance limitations for each drone trip, since the model does not directly capture each drone's arrival time at the retrieval node specifically.

Continuous decision variable $w^U_i \geq 0$ captures the drone's payload at a customer $i \in C$ and continuous decision variable $p^U_i \geq 0$ represents the drone's remaining energy when leaving node $i \in V$. For customers i that are selected as launch nodes, p^U_i will be directly set to θ. Note that w^U_i is not well defined for a launch node, as multiple drones can be launched from the same customer. Instead, the total payload of a specific drone trip at the launch node is calculated at the first customer node visited by the drone trip. As is assigned to exactly one drone trip, no index k is needed to differentiate the drone trips for decision variables $t^{U,A}_i$, $t^{U,L}_i$, w^U_i and p^U_i.

Finally, h^L_{ik} and h^R_{ik} are both binary decision variables that indicate if node $i \in V$ is the launch node or the retrieval node of drone trip $k \in K$ respectively.

Table 3: List of decision variables

Name	***Description***
$z_i^G \in \{0,1\}$	*Indicates whether customer i is served by the truck.*
$y_{ij} \in \{0,1\}$	*Indicates whether the truck travels along arc* $(i, j) \in E$.
$t_i^{G,A} \geq 0$	*The truck's arrival time at node i*
$t_i^{G,L} \geq 0$	*The truck's departure time from node i*
$0 \leq r_i \leq R$	*The number of drones on the truck that are not in flight upon departure from node i*
$z_{ik}^U \in \{0,1\}$	*Indicates whether customer i is served by the drone trip k.*
$x_{ijk} \in \{0,1\}$	*Indicates whether the drone trip k travels along arc* $(i, j) \in E$.
$t_i^{U,A} \geq 0$	*The drone's arrival time at node i*
$t_i^{U,L} \geq 0$	*The drone's departure time from node i*
$w_i^U \geq 0$	*The total payload of the drone at point of departure from node i*
$0 \leq p_i^U \leq \theta$	*The remaining energy of the drone at point of departure from node i*
$h_{ik}^L \in \{0,1\}$	*Indicates whether node i is the launch node of drone trip k*
$h_{ik}^R \in \{0,1\}$	*Indicates whether node i is the retrieval node of drone trip k*

3.4. Routing constraints

The objective function and the general constraints based on the arc-based model for the truck-drone routing problem are as follows:

$$\min \; max\, \{t_{n+1}^{U,A}, t_{n+1}^{G,A}\} \tag{1}$$

$$s.t. \sum_{k \in K} z_{ik}^U + z_i^G = 1, \;\forall i \in C^U \tag{2}$$

$$z_i^G = 1, \;\forall i \in C^G \tag{3}$$

$$\sum_{j \in V} x_{jik} \geq z_{ik}^U, \;\forall i \in C, \;\forall k \in K \tag{4}$$

$$z_i^G = \sum_{j \in V} y_{ji}, \;\forall i \in C \tag{5}$$

$$h_{ik}^R + \sum_{j \in V} x_{ijk} = h_{ik}^L + \sum_{j \in V} x_{jik}, \;\forall i \in C, \;\forall k \in K \tag{6}$$

$$\sum_{j \in V} x_{ijk} \leq 1, \ \forall i \in V, \ \forall k \in K \tag{7}$$

$$\sum_{j \in V} x_{jik} \leq 1, \ \forall i \in V, \ \forall k \in K \tag{8}$$

$$\sum_{j \in V} y_{ij} = \sum_{j \in V} y_{ji} \leq 1, \ \forall i \in V \tag{9}$$

$$\sum_{k \in K} x_{ijk} + \sum_{k \in K} x_{jik} \leq 1, \ \forall (i, j) \in E \tag{10}$$

$$z_{ik}^U + z_{jk}^U \geq 1 - M \times \left(1 - x_{ijk}\right), \ \forall (i, j) \in E, \ \forall k \in K \tag{11}$$

$$h_{ik}^L \geq 1 - M \times \left(2 - x_{ijk} - z_{jk}^U + z_{ik}^U\right), \ \forall i \in V, \forall j \in C, i \neq j, \ \forall k \in K \tag{12}$$

$$h_{jk}^R \geq 1 - M \times \left(2 - x_{ijk} - z_{ik}^U + z_{jk}^U\right), \ \forall i \in C, \forall j \in V, i \neq j, \ \forall k \in K \tag{13}$$

$$\sum_{k \in K} h_{ik}^L \leq M \times \sum_{j \in V} y_{ji}, \ \ \forall i \in V \tag{14}$$

$$\sum_{k \in K} h_{ik}^R \leq M \times \sum_{j \in V} y_{ij}, \ \ \forall i \in V \tag{15}$$

$$h_{ik}^L + h_{ik}^R \leq 1, \ \ \forall i \in V, \ \forall k \in K \tag{16}$$

$$\sum_{i \in V} h_{ik}^L = \sum_{i \in V} h_{ik}^R \leq 1, \ \forall k \in K \tag{17}$$

$$\sum_{i \in C} y_{n+1,i} = \sum_{i \in C} y_{i,n+1} = 1 \tag{18}$$

The objective function (1) seeks to minimize the latest time at which either the truck or a drone returns to the depot. Constraints (2) ensure each customer is served exactly once and Constraints (3) ensure that customers can only be serviced by the truck are served accordingly. Constraints (4) ensure that customers served by a drone must be visited by the drone trip. Constraints (5) enforce that customers visited by the truck must be served by the truck directly. The route of each drone flight is a non-closed loop with a launch node and retrieval node that are different and indicate the start and end of the flight. The launch node only has an out-degree and the retrieval node only has an in-degree. Constraints (6) provide a flow balance equation for all nodes visited by drone trip $k \in K$, which can handle launch and retrieval nodes as well. Constraints (7) and (8) appropriately restrict the total in-degrees for each node and total out-degrees for each drone trip respectively to ensure that each customer is served only once. Constraints (9) both maintain a balanced flow for the truck and ensure that a customer can be visited at most once by the truck. Constraints (10) require each edge to be visited by drones at most once. Constraints (11) require each edge visited by the drone to have at least one customer serviced in this drone trip, and prevents the drone from flying along the route of the truck unnecessarily.

Constraints (12) and (13) determine the launch and retrieval node of the drone trip $k \in K$ respectively, while Constraints (14) and (15) ensure that both the launch and retrieval nodes must be visited and served by the truck. Constraints (16) ensure the retrieval node of a drone trip differs from the launch node of the same drone trip. Constraints (17) ensure an equal number of launch and retrieval nodes in each drone trip, with at most one launch node in each drone trip. Finally, Constraint (18) ensures only a single truck is used.

3.5. Duration constraints

This section outlines the following constraints that represent the schedule of the drone trips and appropriately model drone flight endurance limitations.

$$r_{n+1} + \sum_{j \in C} \sum_{k \in K} x_{(n+1)jk} = R \tag{19}$$

$$r_i + \sum_{l \in V} \sum_{k \in K} x_{ljk} \geq r_j + \sum_{l \in V} \sum_{k} x_{jlk} - M \times \left(1 - y_{ij}\right), \ \forall i \in V, \ \forall j \in C, i \neq j \tag{20}$$

$$r_i + \sum_{j \in V} \sum_{k \in K} x_{j,n+1,k} \geq R - M \times (1 - y_{i,n+1}), \ \forall i \in V \tag{21}$$

$$\sum_{i \in C} w_i \times z_{ik}^U \leq Q, \ \forall k \in K \tag{22}$$

$$w_j^U \geq \sum_{l \in C} \left(w_l \times z_{lk}^U\right) - w_j - M \times \left(3 - x_{ijk} - z_{jk}^U - h_{ik}^L\right), \ \forall i \in V, \forall j \in C, i \neq j, \ \forall k \in K \tag{23}$$

$$w_j^U \geq w_i^U - w_j - M \times \left(3 - \sum_{k \in k} x_{ijk} - \sum_{k \in k} z_{ik}^U - \sum_{k \in k} z_{jk}^U\right), \ \forall i, j \in C, i \neq j \tag{24}$$

$$p_i^U \geq \theta - M \times \sum_{k \in k} z_{ik}^U, \ \forall i \in V \tag{25}$$

$$p_j^U \geq p_i^U - \alpha \times \left(t_{ij}^U + s_j^U\right) \times (w_j^U + w^U + w_j) - M \times \left(3 - x_{ijk} - h_{ik}^L - z_{jk}^U\right), \ \forall (i, j) \in E, \forall k \in K \tag{26}$$

$$p_j^U \geq p_i^U - \alpha \times \left(t_{ij}^U + s_j^U\right) \times (w_i^U + w^U) - M \times \left(3 - x_{ijk} - z_{ik}^U - z_{jk}^U\right), \ \forall (i, j) \in E, \forall k \in K \tag{27}$$

$$p_i^U \geq \alpha \times t_{ij}^U \times w^U + max\left\{\alpha \times \left(t_j^{G,A} - t_i^{U,L} - t_{ij}^U\right) \times w^U, 0\right\} - M \times \left(2 - x_{ijk} - h_{jk}^R\right),$$
$$\forall i \in C, j \in V, i \neq j, \forall k \in K \tag{28}$$

Constraints (19)-(21) track the number of drones on the truck, while Constraints (19) calculate the number of drones on board at the point the truck leaves the depot. Constraints (20) balance the number of drones launched, retrieved and remained. Constraints (21) are the special case of constraints (20) for the depot. Constraints (22)-(24) track changes in payload throughout each drone trip. The formula $\sum_{i \in C} w_i \times z_{ik}^U$ calculates the total payload that the drone must carry in drone trip $k \in K$. Thus, Constraints (23) enforce the drone's maximum payload. When a drone is launched along arc $(i, j) \in E$, node $i \in V$ must be served by the truck and node $j \in C$ must be served by this drone. As mentioned in Section 3.3, the value of w_i^U at the launching node cannot be obtained directly. Therefore, in this constraint, $\sum_{l \in C} \left(w_l \times z_{lk}^U\right)$ is calculated as the weight of all the customers' packages. Constraints (24) calculate the drone's payload when leaving node $j \in C$ in a scenario where the antecedent node is served by a drone.

Constraints (25)-(28) represent the energy consumption of operating drones based on formula (a1), (a2) and (a3). Constraints (25) ensure that the drone is fully charged when it departs from the launch node. Constraints (26) regulate energy consumption and updates p_j^U when the drone travels from launch node $i \in C$ to the first customer $j \in C$ in this drone trip. The total payload when leaving node i for the drone trip is set to $w_j^U + w^U + w_j$. When the drone leaves node $j \in V$, the remaining energy is equal to the energy left at node $i \in V$ minus the energy consumed during the flight from node $i \in V$ to node $j \in C$, as well as during service at node $j \in V$. Constraints (27) follow the same logic as (26) to calculate energy consumption when a drone travels between two nodes (i and j) to be serviced. The only difference is that the total payload is equal to $w_i^U + w^U$ when the drone leaves node i. Constraints (28) pertain to the retrieval progress, where the first node of the arc must be a customer node. When the drone leaves node $i \in C$ and is retrieved at node $j \in V$, the remaining energy level must be greater than the total energy consumption of $\alpha \times t_{ij}^U \times w^U$. If the drone arrives at the retrieval node earlier than the truck, according to (a2), the drone should hover and wait for $t_j^{G,A} - t_i^{U,L} - t_{ij}^U$ time units while consuming $\alpha \times \left(t_j^{G,A} - t_i^{U,L} - t_{ij}^U\right) \times w^U$ units of energy. Constraints (28) ensure that the drone will not run out of energy from hovering at node $j \in V$.

3.6. Time constraints

This section outlines time constraints that synchronise the truck and drone schedules.

$$t_j^{G,A} \geq t_i^{G,L} + t_{ij}^G - M \times \left(1 - y_{ij}\right) \ \forall (i, j) \in E \tag{29}$$

$$t_i^{G,L} \geq t_i^{G,A} + s_i^G - M \times \left(1 - z_i^G\right), \ \forall i \in C \tag{30}$$

$$t_i^{G,L} \geq t_i^{U,A} - M \times \left(1 - z_i^G\right), \ \forall i \in C \tag{31}$$

$$t_j^{U,A} \geq t_i^{U,L} + t_{ij}^U - M \times \left(1 - \sum_{k \in K} x_{ijk}\right), \ \forall (i, j) \in E \tag{32}$$

$$t_i^{U,L} \geq t_i^{U,A} + s_i^U - M \times \left(1 - \sum_{k} z_{ik}^U\right), \ \forall i \in C \tag{33}$$

$$t_i^{U,L} \geq t_i^{G,L} - M \times \left(1 - h_{ik}^L\right), \ \forall i \in C, \ \forall k \in K \tag{34}$$

$$t_i^{U,L} \leq t_i^{G,L} + M \times \left(1 - h_{ik}^L\right), \ \forall i \in C, \ \forall k \in K \tag{35}$$

Constraints (29) update the arrival time of the truck at a node based on its departure time from the previous node and the corresponding travel time. Constraints (30) ensure that the truck cannot depart from the customer node before the service is completed. Constraints (31) restrict the truck from leaving the node before all drones have been retrieved at this node. Constraints (32) are similar to Constraints (29) and calculate the time consumption for the drone. Constraints (33) calculates the time spent to serve customer $i \in C$. Constraints (34) and (35) limit that both the drones and the truck leave the launch node at the same time.

The above model is complete and correct for the MTSP-MD to solve for a minimum makespan. However, additional constraints can be included to speed up the exact solver as well as to ensure proper values for the drone trips. Due to limited space, we outline these constraints in the Appendix.

4. Heuristic algorithm

In this section, we discuss the heuristic approach for solving the MTSP-MD. First, we present the tabu search (TS) procedure and the MSTS algorithm in Section 4.1. Next, we discuss the solution representation in Section 4.2 and the construction algorithm in Section 4.3 before we define the neighborhood structures used in the TS procedure in Section 4.4. Lastly, we discuss how to evaluate the feasibility of a given solution efficiently in Section 4.5 with a two-level evaluation method. Note that the solution evaluation and feasibility checking is used in both the TS procedure and the construction algorithm.

For the sake of consistency, the truck route represents the sequence of customers visited by the truck, and we differentiate a single drone trip from the drone schedule: a single drone trip consists of a launch node, a sequence of customers visited, and a retrieval node; while a drone schedule contains a sequence of non-overlapping single drone trips assigned to the same drone. Note that a drone trip $k \in K$ can be empty in Section 3, but an empty single drone trip will always be removed from the drone schedule in this section (i.e. empty drone trips will never exist on our heuristic solution).

4.1. TS Procedure and MSTS algorithm

TS has been applied in solving various routing related problems successfully (Gendreau et al., 1994; Toth & Vigo, 2003; Qiu et al., 2018; Pan et al., 2020a), which introduces tabu moves to avoid repeated visits to previously visited solutions during the search process. Normally TS starts with an initial solution and searches for the best non-tabu solution in the neighborhood of the current solution at each iteration until the termination criterion are met. In our implementation, the TS procedure uses the customized neighborhood structures for the MTSP-MD as described in Section 4.4 and uses arcs as tabu control. Whenever an arc is removed from the incumbent solution, it is marked as tabu for the next μ iterations, where parameter μ is tuned during the parameter tuning stage. Note that the TS does not differentiate arcs traveled by a drone from the arcs traveled by the truck for the sake of tabu control. In this way, TS prevents the algorithm from revisiting previous solutions and encourages thorough searching of the local region. An aspiration to revoke the tabu status of an arc is allowed when the move leads to a new solution with a lower cost than the best solution found thus far by the algorithm. The TS terminates when it fails to find better solutions in a consecutive number of ϵ_{TS} steps, which is a dynamic measure as explained below.

We embed the TS procedure into a MSTS algorithm (Algorithm 1) to increase the efficiency of the algorithm. In each iteration, the MSTS algorithm either constructs a new initial solution (Line 14) using the construction algorithm (Section 4.3) or restarts from the best found solution (Line 12), and passes the solution to the TS procedure (Line 5) to search for better solutions. The strength of the TS procedure depends on the parameter ϵ_{TS}, which represents the number of consecutive non-improving steps allowed within the TS procedure before the termination of the search. The algorithm employs a dynamic approach to update ϵ_{TS} (Line 4) as follows: (1) When the MSTS iteration fails to find a better solution, the counter C_{cni} is increased, which increases the strength of the stronger TS procedure via ϵ_{TS} by searching a larger solution space; (2) When a better solution is found, C_{cni} is reset to 0 to speed up subsequent TS iterations; and (3) the value of ϵ_{TS} is lower-bounded by the parameter ρ^{LB}. The MSTS algorithm terminates after it has exceeded its maximum run time or when the number of consecutive non-improving TS iterations reaches $\omega_{msts} \times n$, where the parameter ω_{msts} is set to 300 in our computational experiments.

Algorithm 1 Multi-start Tabu Search (MSTS)

1: Construct a new initial solution S
2: $S_{best} \leftarrow S$, $C_{cni} \leftarrow 0$
3: **while** $C_{cni} < \omega_{msts} \times n$ and not exceeding max run time **do**
4:     $\epsilon_{TS} \leftarrow \max\{C_{cni}, \rho^{LB}\}$
5:     $S \leftarrow TS(S, \epsilon_{TS})$
6:     **if** $cost(S) < cost(S_{best})$ **then**
7:         $C_{cni} \leftarrow 0$, $S_{best} \leftarrow S$
8:     **else**
9:         $C_{cni} \leftarrow C_{cni} + 1$
10:     **end if**
11:     **if** A randomly generated boolean value is true **then**
12:         $S \leftarrow S_{best}$
13:     **else**
14:         Construct a new initial solution S
15:     **end if**
16: **end while**
17: **return** S_{best}

4.2. Solution representation

A solution to the problem contains information on customer assignments to the truck and drones, the sequences of customers visits by both the truck and drones, as well as the launch and retrieval nodes of all drone trips. The solution in Figure 1 can be represented as a Directed Acyclic Graph (DAG), where the customers are represented as nodes, and the precedence orders of visits by the drone or the truck are represented using directed arcs (Figure 2).

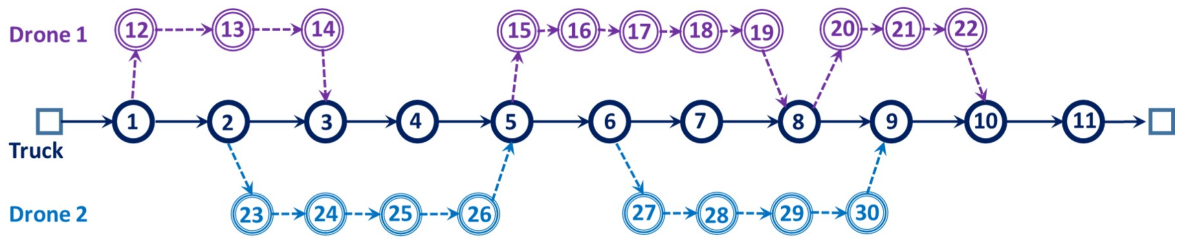

Figure 2: Solution in DAG

4.3. Solution construction

We design a simple and fast route-first-drone-second construction algorithm (Yurek & Ozmutlu, 2018; Schermer et al., 2018) based on the unique features of the MTSP-MD and the two-level solution evaluation method (Algorithm 2). First, a giant TSP tour without any drone assignments is randomly created as the truck route. In the second phase, only customers in C^U who are not used as launching or retrieving nodes in the current solution will be considered for insertion in order of their positions in the truck route. The algorithm initializes each drone schedule with a single drone trip. Afterward, the algorithm attempts to insert one customer from the truck route to an existing single drone trip at a time. If insertion is not possible, the algorithm will attempt to create a new single drone trip for the selected customer at any feasible positions of the truck route. If this attempt also fails, the selected customer will be marked as processed and remain in the truck route during the construction phase. The construction algorithm does not attempt to insert all customers in C^U into drone schedules to avoid generating initial solutions that are over-constrained and difficult to improve by the TS procedure later. Formally, the construction terminates when less than $(1-\phi)\times n$ customers in C^U are yet to be processed, where the parameter ϕ will be tuned during the parameter tuning stage. The detailed construction algorithm is included in the Appendix due to space constraints.

4.4. *Neighbourhood structures for TS*

Relocation and swap operators are widely used by heuristic methods in solving routing problems (Pan et al., 2020b) and are hence extended as neighborhood structures for the MTSP-MD problem based on the unique features of the problem. A total of 9 neighborhood structures from four groups are proposed, as summarized in Table 4.

Table 4: Summary of neighbourhood structures for TS

Group	**S/N**	**Operator**	**Special attention**
Intra-drone schedule	1	Relocate	Allow to create or remove single drone trip
	2	Swap	Not applicable
Inter-drone schedule	3	Relocate	Allow to create or remove single drone trip
	4	Swap	Not applicable
Intra-truck route	5	Relocate	Update of launch and retrieval nodes
	6	Swap	Update of launch and retrieval nodes
Inter operators between the truck route and a drone schedule	7	Relocate from truck route	Update of launch and retrieval nodes, and allow to remove single drone trip
	8	Relocate to truck route	Allow to remove single drone trip
	9	Swap	Update of launch and retrieval nodes

The intra-drone schedule operators work on the schedule of a particular drone and either relocate a customer or swap the positions of two customers within the drone schedule. Note that the operators can involve two different drone trips performed by the same drone because a single drone trip is normally too short for operations. Similarly, the inter-drone schedule operators work on the schedules of two drones with the relocation and swap operators. Note that relocation to a drone schedule allows the algorithm to create a new single drone trip for the customer, while relocation from a drone schedule can remove a single drone trip with only one customer and relocate the affected customer accordingly.

The intra-truck operators resemble the traditional intra-operators in VRP problems with an additional step to update the launch and retrieval nodes of the affected drone trips when necessary. For intra-truck swaps, relevant nodes for two in all affected drone trips (Figure 3 and 4) are swapped.

For intra-vehicle route relocation, the launch node of the affected drone trip will be set to the next customer in the truck route, while the retrieval node of the affected drone trip will be set to the previous customer in the truck route (Figure 5 and 6). This mechanism increases the chance of finding feasible moves since it can potentially reduce travel times and energy consumption by drone trips.

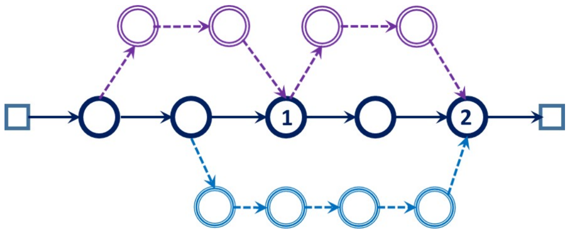

Figure 3: Before intra-truck route swap

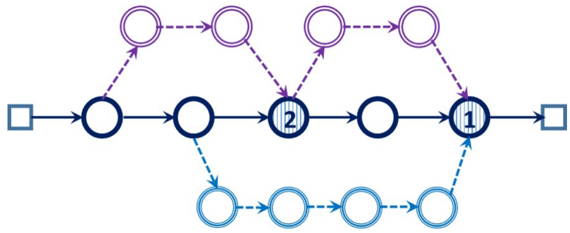

Figure 4: After intra-truck route swap

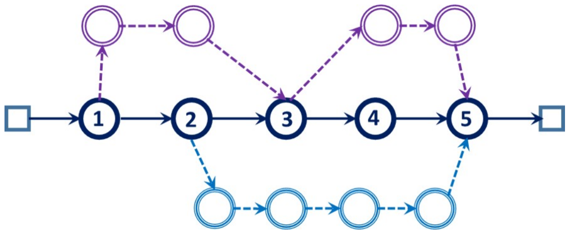

Figure 5: Before intra-truck route relocate

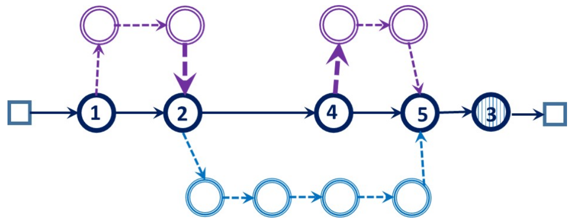

Figure 6: After intra-truck route relocate

The truck-drone operators update the launch and retrieval nodes for affected single drone trips in the same manner as the intra-truck operators. This update rule applies for relocating customers from the truck route to a drone trip (7 and 8), and swapping of customers between the truck route and a drone trip (Figure 9 and 10). For the relocation operator, the affected customer is simply removed from the drone trip and inserted into the truck route directly.

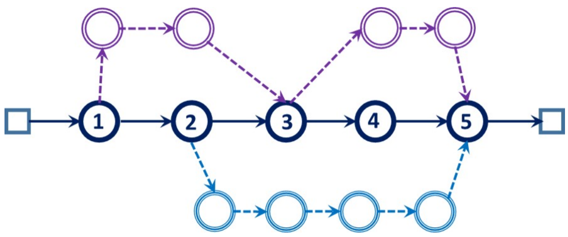

Figure 7: Before inter-truck route & drone trip relocation

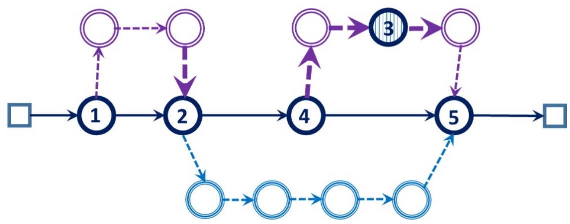

Figure 8: After inter-truck route & drone trip relocation

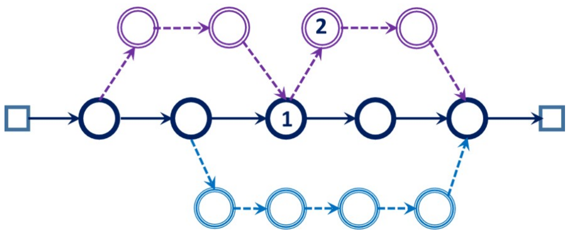

Figure 9: Before inter-truck route & drone trip swap

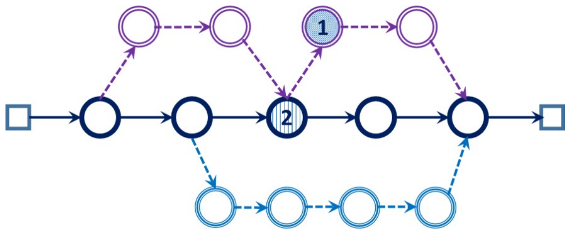

Figure 10: After inter-truck route & drone trip swap

4.5. Solution evaluation and feasibility checking

Given a feasible solution, the makespan is equivalent to the length of the longest path (i.e. critical path) of the DAG, which can be determined with the critical path method (CPM) (Evans, 1992). In essence, CPM finds a topological order for all nodes in a DAG, performs forward propagation and backward propagation to determine the earliest start and end times, latest start and end times, and time float for each node, and then identifies the critical path and its corresponding length.

However, feasibility evaluation is not as straightforward for an MTSP-MD solution, since it depends on the feasibility of all the single drone trips both at the drone-level and solution-level. At the drone-level, a single drone trip is feasible only if: (1) its energy consumption during the flight does not exceed the drone's battery capacity; and (2) its payload is not more than the maximum payload of the drone. This can be evaluated efficiently with the segment-based evaluation method in Section 4.5.1. At the solution-level, as drones are allowed to arrive at the retrieval node earlier than the truck as long as there is sufficient energy, the feasibility of a single drone trip depends on the truck route and the schedules of other drone trips. Hence, we modify the DAG accordingly in Sections 4.5.2 and discuss the solution-level feasibility checking in detail in 4.5.3. The high-level pseudo-code of the solution cost evaluation algorithm is shown in Algorithm 2, which returns ∞ if the solution is infeasible.

4.5.1. Drone-level feasibility evaluation

Vidal et al. (2014) proposed an efficient segment-based evaluation of cumulative cost on a truck route, which can be adopted to evaluate the feasibility of a drone trip at the drone-level. We use the term "segment" in this section to represent a segment of customer visits by a single drone trip. Let

Algorithm 2 Solution cost evaluation

```
if any single drone trip is infeasible at drone-level then
    Return ∞
end if
Construct the modified DAG for the solution
Build topological order for the DAG nodes
Perform forward and backward propagation to determine critical path and arrival time at each customer node
if any single drone trip is infeasible at solution-level then
    Return ∞
else
    Return the length of the critical path
end if
```

a segment be denoted as $\sigma = (\sigma_1,\ \sigma_2, ...\sigma_L)$. From this, we pre-compute and store the following information for σ: the total duration $D(\sigma)$ which includes both travel time and service time incurred along the segment, the total demand $Q(\sigma)$ which includes the no-payload weight of the drone and payload of all parcels to be delivered, and total energy consumption $F(\sigma)$ from the launch node to the retrieval node which is independent of the truck's route. Note that $F(\sigma)$ does not contain any possible hovering energy consumption, which will be handled in the solution-level evaluation instead. Single customer segments are initialized differently for the launch node, the retrieval node as well as the customers served by the drone, as summarized in Table 5. For any two segments $\sigma^1 = (\sigma^1_1,\ \sigma^1_2, ...\sigma^1_{L_1})$ and $\sigma^2 = (\sigma^2_1,\ \sigma^2_2, ...\sigma^2_{L_2})$, we denote the segment by appending σ^2 to the end of σ^1 as $\sigma^1 \bigoplus \sigma^2$. Note that σ^1 cannot contain any retrieval node and σ^2 should not contain any launch node. The associated values for $\sigma^1 \bigoplus \sigma^2$ can be calculated with the following equations:

$$Q(\sigma^1 \oplus \sigma^2) = Q(\sigma^1) + Q(\sigma^2) \tag{36}$$

$$D(\sigma^1 \oplus \sigma^2) = D(\sigma^1) + t^U_{\sigma^1_{L_1},\sigma^2_1} + D(\sigma^2) \tag{37}$$

$$F(\sigma^1 \oplus \sigma^2) = F(\sigma^1) + F(\sigma^2) + \alpha * \left(D(\sigma^1) + t^U_{\sigma^1_{L_1},\sigma^2_1}\right) * Q(\sigma^2) \tag{38}$$

Lastly, σ is feasible at the drone-level if and only if $Q(\sigma) \leq Q$ and $F(\sigma) \leq \theta$. Feasibility evaluation of a single drone trip at the drone-level can be done in $O(1)$ time independent of the length of a single drone trip. This speeds up processing during the TS procedure, especially for test instances with more powerful drones that can serve more customers per single drone trip.

Table 5: Initialization of drone trip segment

	Duration $D(\sigma)$	Demand $Q(\sigma)$	Energy consumption $F(\sigma)$
Launch Node	0	0	0
Visit Node (i)	s_i^U	w_i	$\alpha s_i^U w_i$
Retrieval Node	0	w^U	0

4.5.2. *Modified DAG for CPM*

We modify the DAG (Figure 11) based on the unique features of the MTSP-MD (Figure 12).

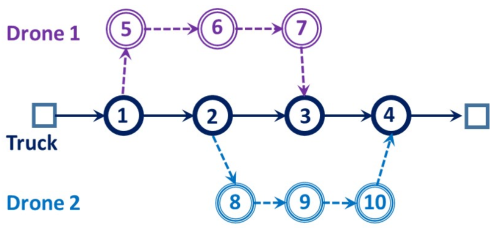

Figure 11: Original DAG of a MTSP-MD solution

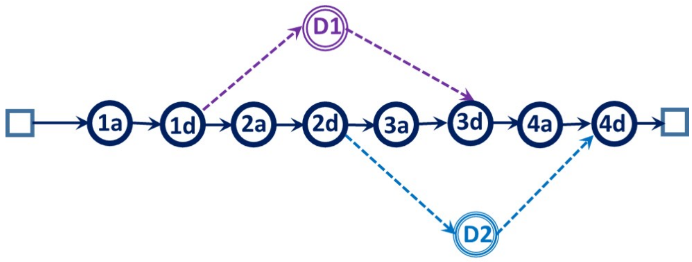

Figure 12: Modified DAG of a MTSP-MD solution

First, each customer which is assigned to the truck and used as either launching node or retrieval node by a drone is split into the arrival node and the departure node. For example, the customer 1 is represented by two nodes, $1a$ for the arrival of the truck and $1d$ for the departure of both the truck and the drones. Note that (1) the duration of $1a$ is equal to s_1 and the duration of $1d$ is simply 0; (2) the distance between $1a$ and $1d$ is 0; and (3) the distance between $1d$ and $2a$ is equal to $t^G_{1,2}$. Second, we aggregate a single drone trip into a single node. For example, node $D1$ represents the single drone trip $(1, 5, 6, 7, 3)$ and its duration is equal to the total duration from the launch node $1d$ to the retrieval node $3d$ without consideration of possible hovering of the drone at the customer 3. The distances between $1d$ and $D1$ and between $D1$ and $3d$ are both 0.

The splitting of customer nodes in the modified DAG is necessary to allow for overlapping operations of customer service by the truck and automatic retrieval of drones. For example, the drone serving $(1, 5, 6, 7, 3)$ can land at the customer 3 any time between the truck's arrival at node $3a$ and the departure of both the truck and the drone at node $3d$. The modified DAG does not enforce the order of service at the customer 3 and the landing of the drones. Furthermore, the split enforces the retrieval of all drones at their respective retrieval nodes before the truck departs. For example, the drone leaves customer 1 together with the truck and other drones at node $1d$, and must arrive at node $3d$ before the departure of the truck from customer 3. This split is also necessary to evaluate the feasibility of a single drone trip at the solution-level when drone hovering and awaiting for truck arrival is required (refer to Section 4.5.3).

4.5.3. Solution-level feasibility evaluation

A CPM algorithm is invoked to calculate relevant information about the critical path based on the modified DAG. Thereafter, we evaluate the solution-level feasibility, i.e, whether any single drone trip has incurred a waiting time that consumes more energy than what is remaining upon arrival at the retrieval node. For example, $D1$ in Figure 12 is feasible at solution-level if and only if the actual truck arrival time at node $3a$ less the actual truck/drone departure time at node $1d$ is earlier than the maximum hovering time allowed for the single drone trip. Mathematically, the maximum hovering time $T^{Hover}_{max}(\sigma)$ can be calculated by $(\theta - F(\sigma))/(\alpha w^U)$, and the difference between the truck arrival and drone arrival is $max\{0, t^a_{\sigma_L} - t^d_{\sigma_1} - D(\sigma)\}$, where σ is the drone trip, $t^a_{\sigma_L}$ is the truck arrival time at retrieval node σ_L and $t^d_{\sigma_1}$ is the departure time of the truck and drones from launch node σ_1. $T^{Hover}_{max}(\sigma)$ is pre-computed and stored for each single drone trip in the solution for faster solution evaluation as explained in Section 4.5.1.

5. Computational experiments

We first describe the test instances in Section 5.1 and provide detailed experimental results in Section 5.2. All programs are coded in Java and run in single-thread mode on a Ubuntu 18.04.3 LTS server with Intel(R) Xeon(R) Silver 4216 CPU at 2.10 GHz. The MSTS program is run 10 times with different random seeds for each test instance. The MILP model is solved with IBM ILOG CPLEX 12.8.0 (IBM CPLEX, 2017). The parameters are tuned using the automatic configuration tool *IRACE* (López-Ibáñez et al., 2016) in a similar way as in Pan et al. (2020b). More specifically, a total of 20 instances are randomly chosen as training instances, and a budget of 200 iterations with each for 300 seconds is given to execute the MSTS algorithm. The list of parameters and their best values returned by IRACE are reported in Table 6.

Table 6: Parameters and tuning results

Parameter	Type	Range	Final Value
μ	Integer	[4,20]	11
ρ^{LB}	Integer	[5,100]	20
ϕ	Real	[0.1,0.8]	0.73

5.1. Test instances

As the MTSP-MD is a new problem, new test instances are derived from the widely used Solomon test instances (Solomon., 1987) with pre-defined drone configurations to evaluate the performance of the proposed algorithm. Table 7 presents details of the three types of drones used: "L" for low capacity, "M" for medium capacity, and "H" for high capacity. With a higher θ and v, L drones can carry the same package for a longer time and travel a further distance compared to the other two drones. On the other hand, with a higher Q, it can carry more payload packages of heavier weights as well. Only three basic test instances, namely C101, R101 and RC101, are adopted for the MTSP-MD problem as the other Solomon's instances share the same customer locations as one of them. The service times of C101 are changed to 10 from 90 to ensure consistency with the other two test instances. The travel speed of the truck is set to be 1 for all instances so that t_{ij}^{G} is equivalent to the Euclidean distance d_{ij}.

Table 7: Drone profiles

	L	M	H
Q	35	55	80
w^{U}	5	5	5
θ	800	1200	1600
α	1	1	1
v	2	2.5	3

The MTSP-MD test instances include various customer sizes (8, 10, 15, 25, 50, 100), drone sizes (1, 2, 3, 4), and three drone profiles. We use the notation "$n-R-T-XXXXX$" to represent a test instance, where n is the set of all customers, R is the maximum number of drones, T represent the drone's capacity profile, and *XXXXX* denotes the original test instance from Solomon which it is derived from (i.e. C101, R101 or RC101).

Test instances with 8, 10, 25, and 50 customers are allocated 300 seconds per run, while instances with 100 customers are allocated 600 seconds per run. The CPLEX solver is given a run time limit of 2 hours. All test instances used in this paper and the detailed routing plans are available online at `http://www.computational-logistics.org/orlib/M-TSPMD`.

5.2. *Experimental results and analysis*

We evaluate the correctness of the MSTS algorithm in Section 5.2.1 and present the results on medium and large test instances in Section 5.2.2. We also analyse the impact of multi-visit on the cost of the solutions in Section 5.2.3 and discuss some observations on the sub-optimal single drone trips found in the solution in 5.2.4.

5.2.1. *Analysis on small scale instances comparison on CPLEX*

As optimal solutions can only be obtained for small scale instances by the exact solver due to the high complexity of the MTSP-MD problem, we limit the customer size of instances to 8 or 10 and apply both the MSTS algorithm and the CPLEX solver on these instances for comparison (Table 8). The test instances are grouped by number of customers and number of drones used. For CPLEX, the table reports number of optimal solutions found ("$\#OPT$") and the average run time required ("Avg RT"). For MSTS, the table presents the number of best known solutions found ("$\#BKS$") and the average run time to the BKS ("Avg T_{best}").

Results show that the CPLEX solver solved all test instances with 8 customers to optimality, but was unable to solve 9 instances with 10 customers within the time limit of 2 hours. The time required to solve the instances increases dramatically when the number of customers increases from 8 to 10. On the other hand, the MSTS algorithm found optimal solutions for all instances with 8 customers. The MSTS performs reasonably well and finds 27 BKS for the instances with 10 customers in significantly shorter times. It is interesting to note that the test instances with 10 customers and 1 or 2 drones are relatively more difficult for the CPLEX to solve than the instances with 3 or 4 drones in terms of Avg RT and #OPT. The detailed results per instance are included in the Appendix.

Table 8: CPLEX vs MSTS on small instances

		$n = 8$				$n = 10$			
		CPLEX		MSTS		CPLEX		MSTS	
	#inst.	#OPT	Avg RT	#BKS	Avg T_{best}	#OPT	Avg RT	#BKS	Avg T_{best}
$R = 1$	9	9	78.2	9	3.5	4	5976.9	7	8.0
$R = 2$	9	9	46.0	9	2.6	5	4638.9	4	4.9
$R = 3$	9	9	30.2	9	0.8	9	2150.8	7	14.2
$R = 4$	9	9	19.6	9	10.9	9	1485.2	9	5.1

5.2.2. Results from the MTSP-MD heuristic algorithm

We conduct further experiments on test instances with 25, 50 and 100 customers, where the results (Table 9) can be used as a benchmark reference for future studies. The cost (C_{best}) of best known solutions (BKS) and the average cost (C_{avg}) of the solutions found are reported for each test instance. The "ratio" under $n = 50$ is defined as (C_{best} for $n = 50/C_{best}$ for $n = 25$) for the same instance. As shown in Figure 13, while this ratio varied for different instances, we observed that the average ratio across all test instances is consistent with the ratio between customer sizes and the variances are in general higher for the instances with 100 customers than the ones for instances with 50 customers. It is interesting that the variances of the ratio for the test instances with low capacity drones are higher. Table 10, which presents the average coefficient of variation (CV) of the best solutions cost over 10 runs for test instances grouped by drone profile type and customer size, shows that the CV is normally lowest for the drones with high capacity. It suggests that it could be more difficult to solve the test instances with drones of lower capacity.

Additionally, results show that for each instance with the same number of customers, (1) the savings margin on solution cost decreases gradually when more and more drones of the same profile are added; and (2) deploying drones with profiles that have better capabilities can significantly reduce total costs. The findings are consistent with similar results reported in the literature (Campbell et al., 2018b). Graphical representations of the solutions are provided in Figures 14 - 17 for four instances with the same number of customers and drone profile. It is worthy to point out that the utilization rates of the drones are very high for all the BKS. Indeed, it never occurs in the four BKS when a truck carries an idling drone on route in the four BKS.

Table 9: Results for n=25/50/100

	n = 25		n = 50			n = 100		
Inst.	C_{best}	C_{avg}	C_{best}	Ratio	C_{avg}	C_{best}	Ratio	C_{avg}
1-L-C101	219.20	221.64	477.94	2.18	497.32	1179.93	5.38	1328.32
1-L-R101	305.65	314.76	553.25	1.81	589.44	1089.03	3.56	1161.18
1-L-RC101	295.05	302.10	627.33	2.13	707.36	1259.73	4.27	1369.31
1-M-C101	207.05	208.80	424.57	2.05	449.82	1042.42	5.03	1186.89
1-M-R101	285.93	288.39	514.75	1.80	533.84	950.62	3.32	1021.62
1-M-RC101	272.46	277.19	608.69	2.23	669.26	1133.37	4.16	1218.44
1-H-C101	199.26	201.30	421.00	2.11	435.08	966.70	4.85	1020.30
1-H-R101	261.76	271.82	484.32	1.85	507.01	919.96	3.51	951.74
1-H-RC101	258.26	262.86	551.32	2.13	584.65	1035.51	4.01	1080.32
2-L-C101	171.74	173.93	359.14	2.09	377.61	995.72	5.80	1124.23
2-L-R101	232.23	239.12	442.26	1.90	468.65	839.59	3.62	910.43
2-L-RC101	248.78	258.81	597.52	2.40	653.42	1018.64	4.09	1157.52
2-M-C101	150.66	153.43	312.80	2.08	328.88	787.57	5.23	894.17
2-M-R101	198.25	206.86	364.50	1.84	384.94	692.18	3.49	736.59
2-M-RC101	207.12	217.07	447.54	2.16	548.36	803.17	3.88	909.35
2-H-C101	141.15	143.51	285.38	2.02	300.97	709.68	5.03	759.72
2-H-R101	187.61	190.83	342.52	1.83	353.12	647.68	3.45	675.72
2-H-RC101	193.33	202.84	411.25	2.13	456.28	748.37	3.87	781.70
3-L-C101	146.45	152.46	319.78	2.18	338.28	1009.48	6.89	1109.91
3-L-R101	194.64	199.63	367.51	1.89	404.18	691.13	3.55	794.26
3-L-RC101	226.93	235.06	604.60	2.66	676.49	975.94	4.30	1123.47
3-M-C101	119.33	126.29	251.55	2.11	263.26	644.98	5.41	744.62
3-M-R101	160.99	164.94	299.47	1.86	311.90	564.37	3.51	612.63
3-M-RC101	187.87	196.14	431.60	2.30	497.16	714.33	3.80	805.97
3-H-C101	113.81	115.86	235.44	2.07	245.12	570.67	5.01	612.41
3-H-R101	147.62	150.76	274.81	1.86	286.11	513.18	3.48	547.30
3-H-RC101	162.90	167.53	336.80	2.07	380.39	628.66	3.86	656.62
4-L-C101	140.61	144.58	299.51	2.13	317.02	914.85	6.51	1019.66
4-L-R101	170.65	176.46	340.49	2.00	366.48	625.21	3.66	686.92
4-L-RC101	214.10	223.69	605.32	2.83	661.08	943.82	4.41	1056.06
4-M-C101	106.24	109.25	223.92	2.11	233.59	507.91	4.78	629.16
4-M-R101	134.44	140.33	246.84	1.84	267.20	466.77	3.47	504.88
4-M-RC101	165.73	176.03	389.88	2.35	447.79	609.71	3.68	687.86
4-H-C101	94.19	96.36	194.00	2.06	208.80	510.09	5.42	535.32
4-H-R101	121.35	128.49	233.69	1.93	240.34	423.81	3.49	456.55
4-H-RC101	140.67	147.55	330.13	2.35	356.73	502.78	3.57	581.72
			Avg	2.09			4.32	

Table 10: Average coefficient of variation of best solution costs by customer size and drone profile over 10 runs

	Drone profile type		
Customer size	L	M	H
n = 25	1.9%	2.0%	1.7%
n = 50	4.3%	4.5%	3.2%
n = 100	6.6%	6.9%	4.4%

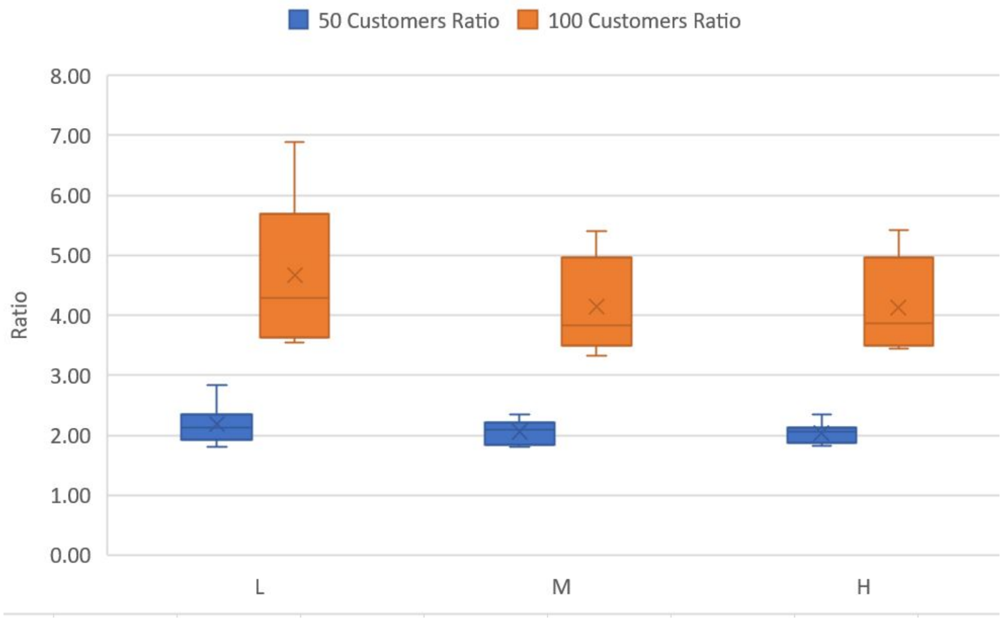

Figure 13: Box plot for ratios

5.2.3. *Multi-visits vs. single visit*

To investigate the impact of allowing multiple visits in the context of the MTSP-MD, we modify the MSTS algorithm to only allow single-visits on instances with 100 customers. A comparison between multi-visits and single visit is done by calcuating a cost ratio of (best cost of multi-visit / best cost of single-visit) *100% for each instance, and results on the average ratio are presented and grouped by drone size and drone profile in Table 11. These results demonstrate that allowing multiple visits can reduce costs when compared to single-visits due to: (1) the greater number of drones that can be deployed at the same time to reduce costs; and (2) drone profiles with better performance help to improve cost savings by carrying more payload and enduring longer flight distances when required. Together with our results from 5.2.2, these findings motivate further research on drone delivery variants that allow for multiple visits and multiple drones per truck, along with higher drone payload capacity. A detailed comparison is presented in the Appendix.

5.2.4. *On sub-optimal single drone trips*

It is noteworthy that the overall best solution might contain single drone trips that are not optimal for the customers served in the trip. For example, the BKS of 25-4-H-R101 (Figure 17)

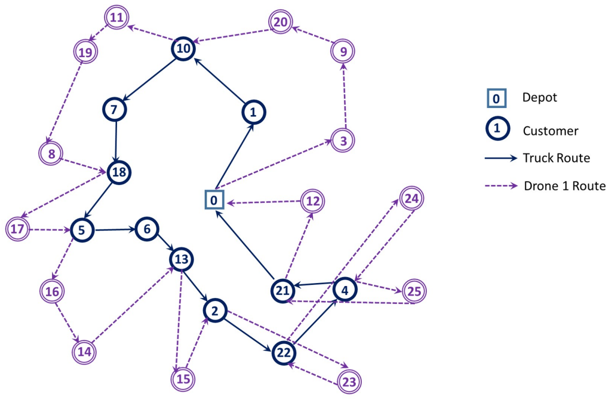

Figure 14: BKS for 25-1-H-R101

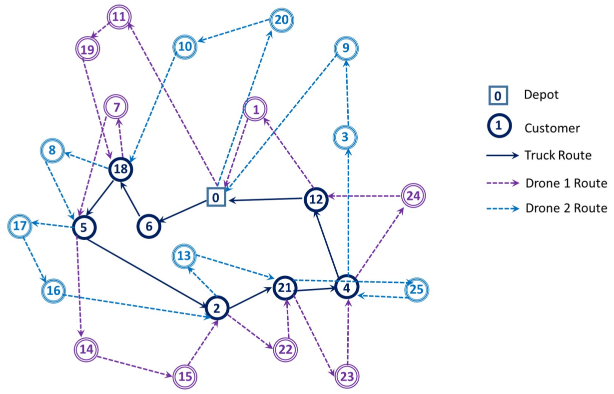

Figure 15: BKS for 25-2-H-R101

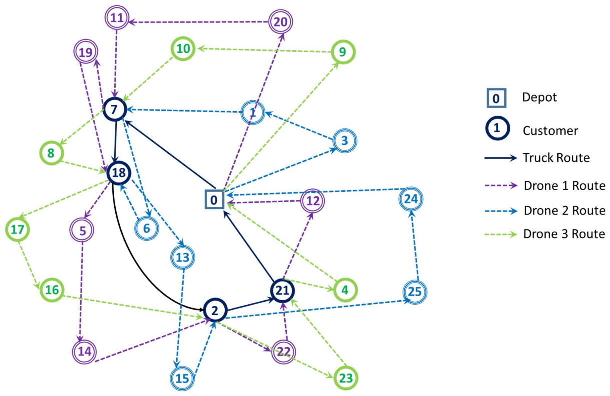

Figure 16: BKS for 25-3-H-R101

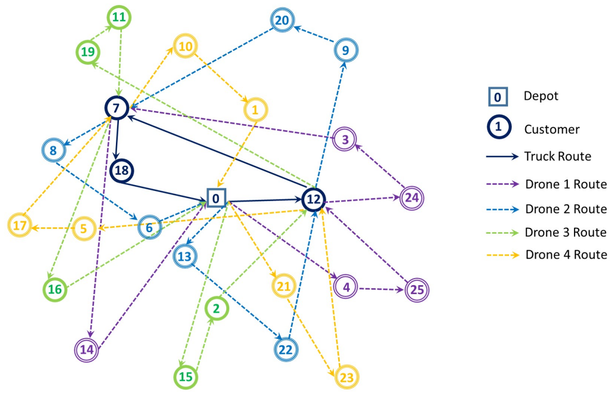

Figure 17: BKS for 25-4-H-R101

Table 11: Cost ratio of multi-vist over single-visit

Type	$R = 1$	$R = 2$	$R = 3$	$R = 4$
L	93.5%	88.3%	87.7%	87.1%
M	85.1%	80.4%	79.7%	73.1%
H	83.3%	77.9%	74.4%	72.0%

contains a sub-optimal single drone trip (12-11-19-7), which takes a longer time compared to the optimal single drone trip (12-19-11-7). However, since another drone trip 12-5-17-7 takes a longer time, (12-11-19-7) is not part of the critical path and the MSTS algorithm does not seek to improve this single drone trip any longer as it will not improve the total makespan.

Furthermore, incorporating flight endurance into the model changes the solution structure substantially, since a given single drone trip with a shorter flight duration will not be feasible if constraints on payload capacity and flight endurance are violated. For example, let Trip1 represent the single drone trip (12-11-19-7) and Trip2 represent (12-19-11-7) in Figure 17. Although Trip2 has a shorter total flight time than Trip1, the package for customer 11 is transported over a longer distance in Trip2. When w_{11} is significantly large enough, Trip2 can consume more energy than θ and become infeasible while Trip1 is still feasible.

Potential mitigating strategies include changing the objective function to minimize the total route length (cost), adopting multi-objective optimization approaches to optimize both makespan and energy consumption, or adopting post-optimization procedures for all single drone trips.

6. Conclusion

In this paper, we proposed the MTSP-MD problem that determines the shortest critical path for a single truck equipped with a homogeneous fleet of drones capable of serving multiple customers in a single flight. A MILP model is formulated for the problem, for which commercial solvers can only solve for small-size instances. To tackle medium and large-size instances which are more practical for real-world scenarios, we design an improved multi-start tabu search (MSTS) algorithm which consists of three main components: (1) a quick feasibility evaluation method applicable at both drone-level and solution-level; (2) a random-sequence based construction algorithm to generate initial solutions; and (3) a TS algorithm with tailored neighborhood structures. A set of 180 test instances is derived from Solomon's data to validate the performance of the proposed algorithm. Our computational results show great potential in cost reduction with multi-visit, multi-drones, and drones with higher capacity.

We see a myriad of opportunities for future research in this area. For example, the MTSP-MD can be extended to allow for parallel operation of multiple trucks, that can possibly allow deployed drones to be retrieved by another truck with spare capacity. Other extensions include the deployment of heterogeneous drone fleets to serve use cases that involve pickups and deliveries, such as perishable food delivery. Other areas for improvement include developing more efficient heuristic approaches and incorporating more real-world considerations into the model. New theoretical bounds on features such as drone multi-visits and critical path scheduling with complex synchronization constraints will be valuable in assessing the performance of heuristic approaches for large-scale problems. Additional studies may explore multi-objective optimization or total cost minimization by removing sub-optimal single drone trips.

Acknowledgements

The authors thank the editor and three anonymous reviewers whose comments helped to improve and clarify this manuscript. This research was partially supported by NRF Singapore [Grant NRFRSS2016-004], and MOE Academic Research Fund of Singapore Tier 2 [Grant MOE2017-T2-2-153]. Zhihao Luo was supported by the China Scholarship Council (CSC) Grant #201903170177.

References

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

Bouman, P., Agatz, N., & Schmidt, M. (2018). Dynamic programming approaches for the traveling salesman problem with drone. *Networks*, *72*, 528–542.

Boysen, N., Briskorn, D., Fedtke, S., & Schwerdfeger, S. (2018). Drone delivery from trucks: Drone scheduling for given truck routes. *Networks*, *72*, 506–527.

Boysen, N., Fedtke, S., & Schwerdfeger, S. (2020). Last-mile delivery concepts: a survey from an operational research perspective. *OR Spectrum*, (pp. 1–58).

Campbell, J. F., Corberán, Á., Plana, I., & Sanchis, J. M. (2018a). Drone arc routing problems. *Networks*, *72*, 543–559.

Campbell, J. F., Sweeney, I., Donald, C., Zhang, J., Pan, D. et al. (2018b). *Strategic Design for Delivery with Linked Transportation Assets: Trucks and Drones*. Technical Report Midwest Transportation Center.

Carlsson, J. G., & Song, S. (2018). Coordinated logistics with a truck and a drone. *Management Science*, *64*, 4052–4069.

Chiang, W.-C., Li, Y., Shang, J., & Urban, T. L. (2019). Impact of drone delivery on sustainability and cost: Realizing the uav potential through vehicle routing optimization. *Applied Energy*, *242*, 1164–1175.

Chung, S. H., Sah, B., & Lee, J. (2020). Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions. *Computers & Operations Research*, (p. 105004).

Coelho, L. C., & Laporte, G. (2014). Improved solutions for inventory-routing problems through valid inequalities and input ordering. *International Journal of Production Economics*, *155*, 391–397.

Cojocaru, M. G., Thommes, E. W., & Gillies, S. (2017). A model of residential mail delivery by drones. In *Modeling and Optimization: Theory and Applications* (pp. 1–15). Springer.

Darvish, M., Coelho, L. C., & Jans, R. (2020). *Comparison of symmetry breaking and input ordering techniques for routing problems*. Technical Report CIRRELT.

Das, D. N., Sewani, R., Wang, J., & Tiwari, M. K. (2020). Synchronized truck and drone routing in package delivery logistics. *IEEE Transactions on Intelligent Transportation Systems*, .

Dell'Amico, M., Montemanni, R., & Novellani, S. (2019). Drone-assisted deliveries: new formulations for the flying sidekick traveling salesman problem. *Optimization Letters*, (pp. 1–32).

El-Adle, A. M., Ghoniem, A., & Haouari, M. (2019). Parcel delivery by vehicle and drone. *Journal of the Operational Research Society*, (pp. 1–19).

Evans, J. (1992). *Optimization algorithms for networks and graphs*. CRC Press.

de Freitas, J. C., & Penna, P. H. V. (2018). A randomized variable neighborhood descent heuristic to solve the flying sidekick traveling salesman problem. *Electronic Notes in Discrete Mathematics*, *66*, 95–102.

de Freitas, J. C., & Penna, P. H. V. (2020). A variable neighborhood search for flying sidekick traveling salesman problem. *International Transactions in Operational Research*, *27*, 267–290.

Gendreau, M., Hertz, A., & Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem. *Management Science*, *40*, 1276–1290.

Gonzalez-R, P. L., Canca, D., Andrade-Pineda, J. L., Calle, M., & Leon-Blanco, J. M. (2020). Truck-drone team logistics: A heuristic approach to multi-drop route planning. *Transportation Research Part C: Emerging Technologies*, *114*, 657–680.

Ha, Q. M., Deville, Y., Pham, Q. D., & Ha, M. H. (2018). On the min-cost traveling salesman problem with drone. *Transportation Research Part C: Emerging Technologies*, *86*, 597–621.

Ha, Q. M., Deville, Y., Pham, Q. D., & Hà, M. H. (2020). A hybrid genetic algorithm for the traveling salesman problem with drone. *Journal of Heuristics*, *26*, 219–247.

Hu, M., Liu, W., Lu, J., Fu, R., Peng, K., Ma, X., & Liu, J. (2019). On the joint design of routing and scheduling for vehicle-assisted multi-uav inspection. *Future Generation Computer Systems*, *94*, 214–223.

IBM CPLEX (2017). *IBM ILOG CPLEX 12.8.0 callable library*.

Jeong, H. Y., Song, B. D., & Lee, S. (2019). Truck-drone hybrid delivery routing: Payload-energy dependency and no-fly zones. *International Journal of Production Economics*, *214*, 220–233.

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.

Kitjacharoenchai, P., Ventresca, M., Moshref-Javadi, M., Lee, S., Tanchoco, J. M., & Brunese, P. A. (2019). Multiple traveling salesman problem with drones: Mathematical model and heuristic approach. *Computers & Industrial Engineering*, *129*, 14–30.

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*, .

Liu, Y., Luo, Z., Liu, Z., Shi, J., & Cheng, G. (2019). Cooperative routing problem for ground vehicle and unmanned aerial vehicle: The application on intelligence, surveillance, and reconnaissance missions. *IEEE Access*, *7*, 63504–

63518.

López-Ibáñez, M., Dubois-Lacoste, J., Cáceres, L. P., Birattari, M., & Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, *3*, 43–58.

Luo, Z., Liu, Z., & Shi, J. (2017). A two-echelon cooperated routing problem for a ground vehicle and its carried unmanned aerial vehicle. *Sensors*, *17*, 1144.

Luo, Z., Liu, Z., Shi, J., Wang, Q., Zhou, T., & Liu, Y. (2018). The mathematical modeling of the two-echelon ground vehicle and its mounted unmanned aerial vehicle cooperated routing problem. In *2018 IEEE Intelligent Vehicles Symposium (IV)* (pp. 1163–1170). IEEE.

Macrina, G., Pugliese, L. D. P., Guerriero, F., & Laporte, G. (2020). Drone-aided routing: A literature review. *Transportation Research Part C: Emerging Technologies*, *120*, 102762.

Marinelli, M., Caggiani, L., Ottomanelli, M., & Dell'Orco, M. (2017). En route truck–drone parcel delivery for optimal vehicle routing strategies. *IET Intelligent Transport Systems*, *12*, 253–261.

Moeini, M., & Salewski, H. (2019). A genetic algorithm for solving the truck-drone-atv routing problem. In *World Congress on Global Optimization* (pp. 1023–1032). Springer.

Moshref-Javadi, M., Hemmati, A., & Winkenbach, M. (2020a). A truck and drones model for last-mile delivery: A mathematical model and heuristic approach. *Applied Mathematical Modelling*, *80*, 290–318.

Moshref-Javadi, M., Lee, S., & Winkenbach, M. (2020b). Design and evaluation of a multi-trip delivery model with truck and drones. *Transportation Research Part E: Logistics and Transportation Review*, *136*, 101887.

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.

Murray, C. C., & Raj, R. (2020). The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones. *Transportation Research Part C: Emerging Technologies*, *110*, 368–398.

bin Othman, M. S., Shurbevski, A., Karuno, Y., & Nagamochi, H. (2017). Routing of carrier-vehicle systems with dedicated last-stretch delivery vehicle and fixed carrier route. *Journal of Information Processing*, *25*, 655–666.

Otto, A., Agatz, N., Campbell, J., Golden, B., & Pesch, E. (2018). Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey. *Networks*, *72*, 411–458.

Pan, B., Zhang, Z., & Lim, A. (2020a). A hybird algorithm for time-dependent vehicle routing problem with time windows. *Computers & Operations Research*, .

Pan, B., Zhang, Z., & Lim, A. (2020b). Multi-trip time-dependent vehicle routing problem with time windows. *European Journal of Operational Research*, .

Peng, K., Du, J., Lu, F., Sun, Q., Dong, Y., Zhou, P., & Hu, M. (2019). A hybrid genetic algorithm on routing and scheduling for vehicle-assisted multi-drone parcel delivery. *IEEE Access*, *7*, 49191–49200.

Poikonen, S., & Golden, B. (2020a). The mothership and drone routing problem. *INFORMS Journal on Computing*, *32*, 249–262.

Poikonen, S., & Golden, B. (2020b). Multi-visit drone routing problem. *Computers & Operations Research*, *113*, 104802.

Poikonen, S., Golden, B., & Wasil, E. A. (2019). A branch-and-bound approach to the traveling salesman problem with a drone. *INFORMS Journal on Computing*, *31*, 335–346.

Poikonen, S., Wang, X., & Golden, B. (2017). The vehicle routing problem with drones: Extended models and connections. *Networks*, *70*, 34–43.

Qiu, M., Fu, Z., Eglese, R., & Tang, Q. (2018). A tabu search algorithm for the vehicle routing problem with discrete split deliveries and pickups. *Computers & Operations Research*, *100*, 102–116.

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.

Schermer, D., Moeini, M., & Wendt, O. (2018). Algorithms for solving the vehicle routing problem with drones. In *Asian Conference on Intelligent Information and Database Systems* (pp. 352–361). Springer.

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

Schermer, D., Moeini, M., & Wendt, O. (2019b). A matheuristic for the vehicle routing problem with drones and its variants. *Transportation Research Part C: Emerging Technologies*, *106*, 166–204.

Solomon., M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, *35*, 254–265.

Toth, P., & Vigo, D. (2003). The granular tabu search and its application to the vehicle-routing problem. *INFORMS Journal on Computing*, *15*, 333–346.

Tu, P. A., Dat, N. T., & Dung, P. Q. (2018). Traveling salesman problem with multiple drones. In *Proceedings of the Ninth International Symposium on Information and Communication Technology* (pp. 46–53).

Vidal, T., Crainic, T. G., Gendreau, M., & Prins, C. (2014). A unified solution framework for multi-attribute vehicle routing problems. *European Journal of Operational Research*, *234*, 658–673.

Wang, D., Hu, P., Du, J., Zhou, P., Deng, T., & Hu, M. (2019a). Routing and scheduling for hybrid truck-drone collaborative parcel delivery with independent and truck-carried drones. *IEEE Internet of Things Journal*, *6*, 10483–10495.

Wang, K., Yuan, B., Zhao, M., & Lu, Y. (2019b). Cooperative route planning for the drone and truck in delivery services: A bi-objective optimisation approach. *Journal of the Operational Research Society*, (pp. 1–18).

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

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

Yoon, J. J. (2018). *The traveling salesman problem with multiple drones: an optimization model for last-mile delivery*.

Ph.D. thesis Massachusetts Institute of Technology.

Yurek, E. E., & Ozmutlu, H. C. (2018). A decomposition-based iterative optimization algorithm for traveling salesman problem with drone. *Transportation Research Part C: Emerging Technologies*, *91*, 249–262.

Zhang, J., Campbell, J. F., II, D. C. S., & Hupman, A. C. (2020). Energy consumption models for delivery drones: A comparison and assessment. Available online.

Appendix A Additional constraints

Acceleration constraints

To tighten the model's solution space, the following constraints are added to speed up the computational process of the CPLEX solver, of which some complement the constraints as outlined in 3.4, 3.5 and 3.6, while others are defined based on assumptions detailed in 3.1:

$$r_i + \sum_{l\in V}\sum_{k\in K} x_{ljk} \le r_j + \sum_{l\in V}\sum_{k} x_{jlk} + M \times \left(1 - y_{ij}\right),\ \forall i \in V,\ \forall j \in C, i \neq j \tag{A.1}$$

$$r_i + \sum_{j\in V}\sum_{k\in K} x_{j,n+1,k} \le R + M \times (1 - y_{i,n+1}),\ \forall i \in V, \tag{A.2}$$

$$w_j^U \le \sum_{l}\left(w_l \times z_{lk}^U\right) - w_j + M \times \left(3 - x_{ijk} - z_{jk}^U - h_{ik}^L\right),\ \forall i \in V, \forall j \in C, i \neq j,\ \forall k \in K \tag{A.3}$$

$$w_j^U \le w_i^U - w_j + M \times \left(3 - \sum_{k\in k} x_{ijk} - \sum_{k\in k} z_{ik}^U - \sum_{k\in k} z_{jk}^U\right),\ \forall i, j \in C, i \neq j \tag{A.4}$$

$$p_j^U \le p_i^U - \alpha \times \left(t_{ij}^U + s_j^U\right) \times (w_j^U + w^U + w_j) + M \times \left(3 - x_{ijk} - h_{ik}^L - z_{jk}^U\right),\ \forall i, j \in V, i \neq j, \forall k \in K \tag{A.5}$$

$$p_j^U \le p_i^U - \alpha \times \left(t_{ij}^U + s_j^U\right) \times (w_j^U + w^U) + M \times \left(3 - x_{ijk} - z_{ik}^U - z_{jk}^U\right),\ \forall i, j \in V, i \neq j, \forall k \in K \tag{A.6}$$

$$t_{n+1}^{U,L} = t_{n+1}^{G,L} = 0 \tag{A.7}$$

$$\sum_{k\in K} z_{n+1,k}^U + z_{n+1}^G = 0 \tag{A.8}$$

$$M \times \sum_{a\le i} z_{ak}^U \ge \sum_{a\le i}\sum_{b\ge k} z_{ab}^U,\ \forall i \in C,\ \forall k \in K \tag{A.9}$$

Constraint (A.1) supplements Constraint (20) by ensuring that $r_i + \sum_{j\in V}\sum_{k\in K} x_{jlk} = r_l + \sum_{j\in V}\sum_{k} x_{ljk}$ when the truck travels from node i to node j, while Constraint (A.2) supplements Constraint (21) in a similar fashion. This is done to enforce equation $r_i + \sum_{j\in V}\sum_{k\in K} x_{j,n+1,k} = R$ to hold true when the truck returns to the depot. Constraint (A.3) complements Constraint (23) by restricting the payload of a drone when leaves the first customer in each drone trip. Constraint (A.4) complements Constraint (24) to balance changes in payload during drone flight. Constraints (A.5) and (A.6) complement Constraints (26) and (27) respectively in enforcing limits on drone energy consumption.

To enforce our assumptions, Constraint (A.7) initialises the departure times of the truck and drones from the depot as zero, and Constraint (A.8) prevents the depot from being assigned.

The symmetric-breaking constraint proposed by Coelho & Laporte (2014) and Darvish et al. (2020) are typically used to reduce the size of solution space to avoid isomorphic solutions. Constraint (A.9) is an improvement on the typical hierarchical ordering constraint, in which customers with lower indices always have a priority on drone trips also with lower indices.

While these constraints do not affect the objective function value of the optimal solution, they help to determine the value of key decision variables under specific conditions to help commercial exact solvers such as CPLEX to obtain the optimal solution more quickly.

Readability constraints

In this section, we introduce additional constraints to the original model that help to provide bounds for unrestricted decision variables. This helps us better understand generated optimal solutions from commercial solvers in a more intuitive manner and improve its readability.

$$\sum_{i \in V} \sum_{j \in V} x_{ijk} \leq M \times \sum_{i \in V} h^L_{ik}, \ \forall k \in K \tag{A.10}$$

$$\sum_{i \in V} \sum_{j \in V} x_{ijk} \leq M \times \sum_{i \in V} h^R_{ik}, \ \forall k \in K \tag{A.11}$$

$$r_i \leq M \times \left(1 - \sum_{k \in K} z^U_{ik}\right), \ \forall i \in C \tag{A.12}$$

$$w^U_i \leq M \times \sum_{k \in k} z^U_{ik}, \ \forall i \in C \tag{A.13}$$

$$M \times \sum_{k \in K} \sum_{j \in V} x_{jik} \geq t^{U,L}_i \geq t^{U,A}_i, \ \forall i \in C \tag{A.14}$$

$$M \times \sum_{j \in V} y_{ij} \geq t^{G,L}_i \geq t^{G,A}_i, \ \forall i \in C \tag{A.15}$$

Constraints (A.10) and (A.11) ensure that a non-empty drone trip must contain both a launch node and a retrieval node. Constraint (A.12) forces r_i to be zero if customer i is not visited by the truck. Constraint (A.12) ensures the value of w^U_i is equal to zero if node $i \in V$ is not served by a drone. Constraints (A.14) and (A.15) ensure time is set to zero if a customer is not visited by either the truck or the drones.

These additional constraints, combined with the constraints outlined in the main paper, limit the values which decision variables outside of the main routes can take on, without affecting the objective value in the optimal solution, which allows for the result to be read directly.

Are supplementary constraints necessary?

As opposed to the conventional Vehicle Routing Problem (VRP), it is necessary for us to incorporate the acceleration and readability constraints into our proposed model due to the following two reasons: 1) interference when routes are parallel, and 2) inherent limitations of the objective function.

To better understand this necessity, we present the following hypothetical instance:

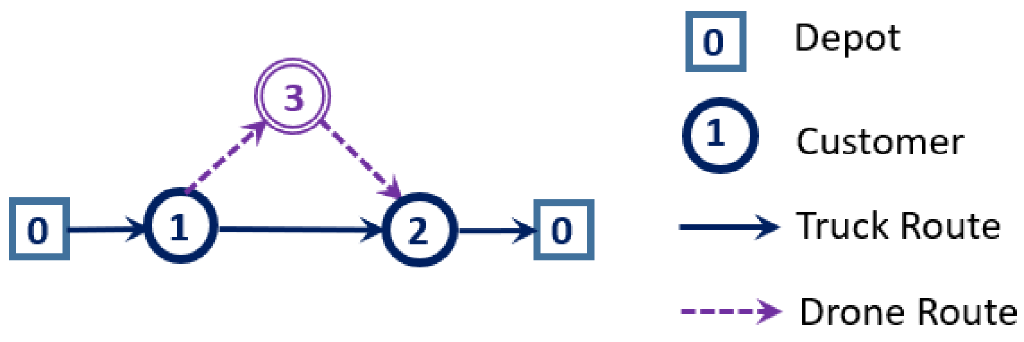

Figure A.1: Hypothetical Instance

Assume that the solution shown in A.1 is the optimal solution, and that the truck takes more time to arrive at node 2. For a conventional VRP instance, the values of decision variables at each node can be derived in reverse from the makespan of the optimal solution. However, in our problem, when a deployed drone and truck travel towards the same destination node on parallel routes with different travel times, only the travel time of the drone/truck which arrives later can be derived in reverse from the optimal solution.

In this instance, the value of $t_i^{G,A}$ and $t_i^{G,L}$ in the truck route (0-1-2-0) can be identified as a specific value based on the minimized makespan and constraints (29)-(30). Based on the assumption and constraints (32), we can then infer $t_2^{G,A} \geq t_2^{U,A} \geq t_3^{U,L} + t_{3,2}^{U}$. Thereafter, according to Constraints (32)-(35), we obtain $t_3^{U,A} \geq t_1^{U,L} + t_{1,3}^{U} = t_1^{G,L} + t_{1,3}^{U}$ and $t_3^{U,L} \geq t_3^{U,A} + s_3$. Therefore, although the optimal makespan and routes are given, we are only able to ascertain that the time when the drone leaves node 3 is within the range of ($t_2^{G,A} - t_{3,2}^{U} \geq t_3^{U,L} \geq t_1^{G,L} + t_{1,3}^{U} + s_3$).

This time range introduces randomness which in turn leads to uncertainty in other decision variables. With Constraint (26) on edge <1-3>, $p_3^U \geq \theta - \alpha \times \left(t_{1,3}^{U} + s_3\right) \times (w_3^U + w^U + w_3)$, which simplifies to $p_3^U \geq \theta - (P_{1,3} + P_3^S)$. Similarly, with Constraint (28) on edge <3-2>, we obtain $p_3^U \geq P_{3,2} + P_2^T$. Without the above supplementary constraints, the value of p_3^U instead be a random number in the range $\theta \geq p_3^U \geq max\left\{\theta - (P_{1,3} + P_3^S), P_{3,2} + P_2^T\right\}$.

This randomness expands the solution space of the optimal solution and makes it difficult to apply relevant constraints in some algorithms. Introducing Constraint (A.5) helps to determine that $p_3^U = \theta - (P_{1,3} + P_3^S)$ and reduces the solution space while improving readability of the solution. The other acceleration constraints we have defined also have similar effects on the decision variables r_i and w_i^U.

The necessity of supplementary constraints implies that the key to the solution lies in finding the longest path to complete the task. This characteristic shares similarities with the Program Evaluation and Review Technique (PERT). Hence, there might be potential to further study algorithms in project management and discover methods that may be useful for solving the MTSP-MD.

We also note that this necessity exists only when the objective function aims to minimize makespan. When the objective function is defined instead as minimizing of total costs in lieu of drone flight times and energy consumption, a reverse-derivation of decision variables for each note from the objective value becomes feasible without supplementary constraints. In future studies, the MTSP-MD model may be simplified by applying cost-minimization objective functions or using multi-objective optimization.

Appendix B Detailed construction algorithm

Algorithm B.1 Construction algorithm

1: Create a random sequence of customers V_{seq} as the truck route
2: Duplicate V_{seq} as unprocessed customer seq C_{seq}
3: **for** each customer i in G **do**
4:     Remove i from C_{seq}
5: **end for**
6: **for** each drone **do**
7:     **while true do**
8:         Let c be the first customer in C_{seq}
9:         **for** each pair of l_c and r_c in V_{seq} **do**
10:             **if** single drone trip (l_c, c, r_c) is feasible at both drone-level and solution-level **then**
11:                 Create a single drone trip (l_c, c, r_c) for the drone
12:                 Remove c from V_{seq} and C_{seq}
13:                 Remove l_c and r_c from C_{seq} if necessary
14:                 Break while statement
15:             **end if**
16:         **end for**
17:         Remove c from C_{seq}
18:     **end while**
19: **end for**
20: **while** $|C_{seq}| \geq (1 - \phi) \times n$ **do**
21:     Let c be the first customer in C_{seq}
22:     **for** each single drone trip in solution **do**
23:         **if** c can be inserted into the single drone trip **then**
24:             Insert c into the single drone trip
25:             Remove c from V_{seq}
26:             Go to Line 39
27:         **end if**
28:     **end for**
29:     **for** each drone schedule **do**
30:         **for** each pair of l_c and r_c in V_{seq} where the drone is idling **do**
31:             **if** single drone trip (l_c, c, r_c) is feasible at both drone-level and solution-level **then**
32:                 Create a new drone trip (l_c, c, r_c) and insert it into the drone schedule
33:                 Remove c from V_{seq}
34:                 Remove l_c and r_c from C_{seq} if necessary
35:                 Go to Line 39
36:             **end if**
37:         **end for**
38:     **end for**
39:     Remove c from C_{seq}
40: **end while**

Appendix C CPLEX vs MSTS

For the CPLEX solver, Table C.1 presents whether an optimal solution is found within the run time limit ("Opt?"), the best solution cost found (C_{best}^{CPLEX}), and the actual run time (T_{tot}). For the MSTS algorithm, the table presents the cost of the best-found solution (C_{best}^{MSTS}), the best running time to find C_{best}^{MSTS}, as well as the gap between the cost of solutions found by both algorithms. The gap is formally defined as $(C_{best}^{MSTS} - C_{best}^{CPLEX})/C_{best}^{CPLEX}$.

Table C.1: CPLEX vs MSTS for small instances

	CPLEX			MSTS				CPLEX			MSTS		
Inst.	Opt?	C_{best}^{CPLEX}	T_{tot}	C_{best}^{MSTS}	T_{bst}	Gap	Inst.	Opt?	C_{best}^{CPLEX}	T_{tot}	C_{best}^{MSTS}	T_{bst}	Gap
8-1-L-C101	Y	75.49	84	75.49	0.25	-	10-1-L-C101	N	88.27	7219	87.53	0.62	-0.8%
8-1-L-R101	Y	128.82	55	128.82	1.52	-	10-1-L-R101	Y	150.1	4268	150.1	0.09	-
8-1-L-RC101	Y	112.73	44	112.73	7.15	-	10-1-L-RC101	Y	149.82	3732	149.82	0.42	-
8-1-M-C101	Y	69.03	88	69.03	2.15	-	10-1-M-C101	N	81.83	7219	81.72	0.64	-0.1%
8-1-M-R101	Y	107.34	67	107.34	7.69	-	10-1-M-R101	N	136.13	7215	133.22	40.75	-2.1%
8-1-M-RC101	Y	103.16	50	103.16	0.35	-	10-1-M-RC101	Y	132.38	4656	132.38	0.6	-
8-1-H-C101	Y	66.24	164	66.24	8.04	-	10-1-H-C101	N	78.74	7200	79.36	2.12	0.8%
8-1-H-R101	Y	99.1	98	99.1	4.25	-	10-1-H-R101	N	122.96	7216	125.85	26.55	2.4%
8-1-H-RC101	Y	97.21	54	97.21	0.08	-	10-1-H-RC101	Y	116.33	5067	116.33	0.52	-
8-2-L-C101	Y	57.25	24	57.25	0.75	-	10-2-L-C101	Y	64.69	3335	65.96	0.75	2.0%
8-2-L-R101	Y	88.34	45	88.34	2.23	-	10-2-L-R101	Y	112.45	3133	112.45	0.64	-
8-2-L-RC101	Y	92.25	25	92.25	0.24	-	10-2-L-RC101	Y	138.97	3078	138.97	0.15	-
8-2-M-C101	Y	54.18	46	54.18	11.90	-	10-2-M-C101	N	58.81	7200	59.31	0.5	0.9%
8-2-M-R101	Y	73.14	62	73.14	0.22	-	10-2-M-R101	N	88.84	7200	91.02	12.54	2.5%
8-2-M-RC101	Y	79.34	24	79.34	4.31	-	10-2-M-RC101	Y	95.16	1768	95.16	0.63	-
8-2-H-C101	Y	52.84	81	52.84	3.22	-	10-2-H-C101	N	56.24	7200	55.97	21.43	-0.5%
8-2-H-R101	Y	65.98	91	65.98	0.09	-	10-2-H-R101	N	81.65	7213	82.15	6.81	0.6%
8-2-H-RC101	Y	74.92	16	74.92	0.20	-	10-2-H-RC101	Y	83.76	1623	88.11	0.25	5.2%
8-3-L-C101	Y	49.91	27	49.91	1.07	-	10-3-L-C101	Y	54.86	2234	56.96	1.51	3.8%
8-3-L-R101	Y	70.65	22	70.65	0.06	-	10-3-L-R101	Y	86.15	1895	86.15	3.58	-
8-3-L-RC101	Y	89.23	21	89.23	0.04	-	10-3-L-RC101	Y	128.91	2339	128.91	2.25	-
8-3-M-C101	Y	46.05	31	46.05	2.24	-	10-3-M-C101	Y	47.94	1957	47.94	17.19	-
8-3-M-R101	Y	61.8	63	61.8	0.14	-	10-3-M-R101	Y	73.7	1817	74.36	10.65	0.9%
8-3-M-RC101	Y	71.61	13	71.61	0.32	-	10-3-M-RC101	Y	79.84	807	79.84	0.11	-
8-3-H-C101	Y	43.38	24	43.38	1.52	-	10-3-H-C101	Y	45.35	2141	45.35	72.79	-
8-3-H-R101	Y	60.61	55	60.61	1.80	-	10-3-H-R101	Y	67.55	5402	67.55	18.23	-
8-3-H-RC101	Y	71.61	16	71.61	0.10	-	10-3-H-RC101	Y	74.92	765	74.92	1.86	-
8-4-L-C101	Y	45.69	18	45.69	21.08	-	10-4-L-C101	Y	50.07	1284	50.07	0.16	-
8-4-L-R101	Y	61.8	18	61.8	9.22	-	10-4-L-R101	Y	78.12	1256	78.12	0.18	-
8-4-L-RC101	Y	82.11	17	82.11	9.49	-	10-4-L-RC101	Y	128.91	2278	128.91	0.11	-
8-4-M-C101	Y	43.58	20	43.58	2.44	-	10-4-M-C101	Y	46.05	1544	46.05	0.32	-
8-4-M-R101	Y	55.83	26	55.83	4.75	-	10-4-M-R101	Y	62.51	1495	62.51	10.71	-
8-4-M-RC101	Y	71.61	19	71.61	0.11	-	10-4-M-RC101	Y	71.61	798	71.61	4.03	-
8-4-H-C101	Y	42.17	18	42.17	42.78	-	10-4-H-C101	Y	43.38	1407	43.38	0.1	-
8-4-H-R101	Y	51.72	24	51.72	8.38	-	10-4-H-R101	Y	61.8	3035	61.8	21.31	-
8-4-H-RC101	Y	71.61	16	71.61	0.10	-	10-4-H-RC101	Y	71.61	270	71.61	8.95	-

Appendix D Multi-visit vs Single-visit detailed table

Table D.1: Multi-vist vs single-visit scenario

Inst	SV	MV	MV/SV	Inst	SV	MV	MV/SV
100-1-H-C101	1181.13	966.70	81.8%	100-3-H-C101	770.39	570.67	74.1%
100-1-H-R101	1065.90	919.96	86.3%	100-3-H-R101	676.32	513.18	75.9%
100-1-H-RC101	1268.51	1035.51	81.6%	100-3-H-RC101	856.75	628.66	73.4%
100-1-L-C101	1242.34	1179.93	95.0%	100-3-L-C101	1046.34	1009.48	96.5%
100-1-L-R101	1182.99	1089.03	92.1%	100-3-L-R101	831.49	691.13	83.1%
100-1-L-RC101	1349.29	1259.73	93.4%	100-3-L-RC101	1167.16	975.94	83.6%
100-1-M-C101	1246.29	1042.42	83.6%	100-3-M-C101	836.93	644.98	77.1%
100-1-M-R101	1094.46	950.62	86.9%	100-3-M-R101	706.97	564.37	79.8%
100-1-M-RC101	1336.07	1133.37	84.8%	100-3-M-RC101	869.28	714.33	82.2%
100-2-H-C101	882.14	709.68	80.4%	100-4-H-C101	670.85	510.09	76.0%
100-2-H-R101	846.96	647.68	76.5%	100-4-H-R101	613.86	423.81	69.0%
100-2-H-RC101	973.45	748.37	76.9%	100-4-H-RC101	708.77	502.78	70.9%
100-2-L-C101	1057.99	995.72	94.1%	100-4-L-C101	1033.19	914.85	88.5%
100-2-L-R101	956.24	839.59	87.8%	100-4-L-R101	782.76	625.21	79.9%
100-2-L-RC101	1226.20	1018.64	83.1%	100-4-L-RC101	1015.72	943.82	92.9%
100-2-M-C101	942.19	787.57	83.6%	100-4-M-C101	710.97	507.91	71.4%
100-2-M-R101	867.34	692.18	79.8%	100-4-M-R101	643.38	466.77	72.5%
100-2-M-RC101	1033.70	803.17	77.7%	100-4-M-RC101	808.72	609.71	75.4%