Transportation Research Part C 165 (2024) 104691

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Exact and heuristic approaches for the Truck–Drone Team Logistics Problem

Maurizio Boccia [a], Andrea Mancuso [a], Adriano Masone [a,*], Claudio Sterle [a,b]

[a] Department of Electrical Engineering and Information Technology, University of Naples Federico II, Via Claudio, 21, Naples, 80125, Naples, Italy

[b] Istituto di analisi dei sistemi ed informatica Antonio Ruberti, Consiglio Nazionale delle Ricerche, Via dei Taurini, 19, Rome, 00185, Rome, Italy

ARTICLE INFO

Keywords:
Drones
Logistics
MILP
Branch-and-cut

ABSTRACT

Truck-and-drone routing problems are a topic of increasing interest due to advancements in drone technology. Initially, drones were thought to serve only one customer per flight; however, recent improvements in drone technology have led to the definition of multi-visit truck-and-drone routing problems, where a drone can serve multiple customers in each flight. These routing problems, which require a careful synchronization between the truck and the drone, are challenging to solve, even with a small number of customers. In this work, we address a multi-visit, single-truck, single-drone routing problem also known in the literature as the Truck–Drone Team Logistics Problem (*TDTLP*). In particular, we study two versions of the *TDTLP* arising from the possibility for the drone to land or hover at the recollection points. Our study proposes a novel mixed-integer linear programming formulation for the *TDTLP*, solved by a Branch-and-Cut (*B&C*) algorithm. Then, we introduce a matheuristic approach that incorporates our *B&C* algorithm to solve medium- and large-size instances. Computational results on benchmark instances demonstrate the robustness and efficacy of the proposed methods.

1. Introduction

The rapid expansion of e-commerce has created an increasing demand for fast and efficient delivery services. Emerging technologies, such as unmanned aerial vehicles or drones, have received significant interest for their potential in parcel delivery within logistics distribution. Indeed, in comparison to conventional delivery with trucks, drones offer numerous advantages: quicker delivery times, less prone to traffic congestion, lower transportation costs, and no need for human drivers. However, drones are not ideal for long-distance deliveries due to their restricted carrying and battery capacities, which limits the feasibility of drone-only deliveries. Consequently, to fully leverage the capabilities of drones, truck-and-drone hybrid delivery systems have been proposed for logistics distribution (Carlsson and Song, 2018; Murray and Chu, 2015). In such hybrid delivery systems, the truck serves a dual role: it delivers parcels to customers and functions as a mobile launching/landing platform for one or more drones. As the truck travels to different customer locations for deliveries, the drone is simultaneously dispatched from the truck to serve additional customers, subsequently returning to the truck at a rendezvous location along its route. The series of drone operations (take-off, deliveries, and landing) is commonly referred to as 'sortie' in the literature.

The utilization of such delivery systems and the resulting routing problems have emerged as a cutting edge research topic within the Operations Research community. The related literature has expanded rapidly, introducing a variety of truck-and-drone routing

* Corresponding author.
E-mail address: adriano.masone@unina.it (A. Masone).

https://doi.org/10.1016/j.trc.2024.104691
Received 23 February 2024; Received in revised form 30 April 2024; Accepted 28 May 2024
Available online 20 June 2024

problems that differ in terms of operational conditions and the structure of hybrid truck-and-drone delivery systems (Chung et al., 2020; Liang and Luo, 2022). Currently, most studies have assumed that a drone can serve only one customer per sortie due to technological limitations. However, with the rapid advancement of drone technologies, there have been significant improvements in flight endurance and carrying capacity. This progress allows to serve multiple customers in a single drone sortie. Considering these developments, a new category of truck-and-drone routing problems, termed multi-visit truck-and-drone routing problems, has emerged (Poikonen and Golden, 2020).

Despite the considerable interest from the scientific community, these hybrid systems have not been extensively deployed by logistics companies, partly due to regulatory limitations and the high investment costs associated with these innovative systems (Tavares, 2021; Zwickle et al., 2019). Specifically, many logistics companies are not fully aware of the potential benefits of deploying such systems, largely because of the difficulties in determining effective routes due to the complexity of truck-and-drone routing problems. Indeed, these problems, which include synchronization requirements, represent a more complex variant of Traveling Salesman Problems or Vehicle Routing Problems. Thus, they are NP-hard problem as well, and they are extremely challenging to solve optimally and/or effectively, even for instances involving few customers. Due to this complexity, the majority of contributions in the literature address these problems through heuristic approaches. To the best of our knowledge, only two contributions in the literature propose tailored exact solution methods for multi-visit truck-and-drone routing problems (Wang and Sheu, 2019; Yin et al., 2023). However, both studies focus on a delivery problem that minimizes the sum of operational costs. Yet, as indicated in Schermer et al. (2019), the more common performance indicators in hybrid drone–truck scenarios are time-related. These include the minimal time required to serve all locations or the earliest time at which both vehicles return to the depot at the end of operations.

In light of this, our study has developed an exact solution method for a specific type of multi-visit truck-and-drone routing problem, known in the literature as Truck–Drone Team Logistics Problem (*TDTLP*) (Gonzalez-R et al., 2020). The *TDTLP* is the first routing problem defined in the literature where a single truck and a single drone, able of visiting multiple customers within the same sortie, are utilized in tandem to serve a subset of customers with the objective of minimizing the delivery completion time. In particular, in this work, we will focus on two versions of the *TDTLP* differing in the drone operation while waiting for the truck at the rendezvous point of a sortie. In the *TDTLP* with landing (*TDTLP-L*), the drone can safely land at the rendezvous location, thus saving energy as assumed in Gonzalez-R et al. (2020). In the *TDTLP* with hovering (*TDTLP-H*), we assume that the drone has to hover at the rendezvous location, with the energy expended contributing to its battery depletion.

The main contributions of our paper are summarized as follows:

- We introduce a new mixed-integer linear programming (*MILP*) formulation for the *TDTLP* with a polynomial number variables and without *Big-M* constraints. Additionally, we present two families of valid inequalities and a Branch-and-Cut (B&C) algorithm developed to optimally solving the proposed formulation.
- We propose a matheuristic approach in order to highlight how the proposed *MILP* formulation can be effectively exploited to heuristically solve medium- and large-size *TDTLP* instances. The matheuristic approach is based on the solution of the *TDTLP* on a reduced graph, obtained by fixing the precedence of customer service in the solution.
- We conduct extensive experiments on benchmark instances for the *TDTLP*. For the *TDTLP-L*, our computational results demonstrate that our approach is either competitive with or superior to the existing state-of-the-art methods (Gonzalez-R et al., 2020). Notably, our method either finds optimal solutions for some instances previously unsolved or yields improved upper bounds for those that remain unsolved. As this study is the first one to tackle the *TDTLP-H*, a direct comparison with other solution methods in the literature is not possible. Therefore, we show the impact of the hovering assumption on both the delivery completion time and the efficacy of the proposed solution methods.

The remainder of the paper is organized as follows: Section 2 provides a brief review of related literature to position our study. In Section 3, we recall the *TDTLP* assumptions and features and introduce the *MILP* formulation. Section 4 details the *B&C* algorithm and the valid inequalities. The matheuristic approach, based on the proposed *MILP* formulation, is discussed in Section 5. Section 6 is dedicated to presenting the computational results. Finally, the conclusions and perspectives for future work on this topic are discussed in Section 7.

2. Literature review

The first truck-and-drone routing problem studied in the literature, known as the Flying Sidekick Traveling Salesman Problem (*FSTSP*), established the foundational framework for a truck-and-drone delivery system. Following this seminal work, research into truck-and-drone delivery systems has significantly increased in recent years. However, each study typically concentrates on a distinct hybrid delivery system, each with its unique characteristics and operational features, often representing variations of the basic framework defined by the *FSTSP*. Consequently, the literature on this topic shows considerable heterogeneity. Based on this and categorizing the literature according to the composition of the delivery fleet, readers aiming at a comprehensive review are referred to Masone et al. (2022), Najy et al. (2023) for single-truck single-drone problems, Cavani et al. (2021), Murray and Raj (2020) for single-truck multi-drone, and Schermer et al. (2019), Wang et al. (2022) for multi-truck single/multi-drone. In this work, our literature review is concentrated on multi-visit truck-and-drone routing problems, as the *TDTLP* falls into this category. Similar to the literature on truck-and-drone routing problems, the contributions on multi-visit hybrid systems are also quite different. Therefore, to provide a clear overview of the differences between these studies, we summarize their key characteristics in Table 1. The first two columns list the authors and the publication year. Subsequent three columns detail the number of trucks utilized in the delivery

Table 1
Summary of the contributions on multi-visit truck-and-drone routing problems.

Authors	Year	#T	#D	Objective	Battery capacity	Energy consumption	Payload capacity	Hover	Truck serve	P&D	Exact approach	Heuristic approach	Size exact	Size heuristic
Karak and Abdelghany (2019)	2019	1	m	Routing costs	Distance	Distance	✓	–	–	✓	Solver	Clark-and-Wright based heuristics	8	100
Wang and Sheu (2019)	2019	k	m	Routing and vehicle dispatchment costs	Time	Time	✓	–	✓	–	*B&P*	–	13	–
Gonzalez-R et al. (2020)	2020	1	1	Completion time	Time	Time	–	–	✓	–	Solver	Greedy and Simulated Annealing	10	250
Kitjacharoenchai et al. (2020)	2020	k	m	Completion time	Time	Time	✓	–	✓	–	Solver	Large Neighborhood Search	8	76
Poikonen and Golden (2020)	2020	1	m	Completion time	Energy	Time and weight (Non-linear)	–	✓	–	–	–	Graph transformation, Greedy heuristic and Local Search	–	50
Luo et al. (2021)	2021	1	m	Completion time	Energy	Time and weight	✓	✓	✓	–	Solver	Tabu Search	10	100
Gu et al. (2022)	2022	k	1	Completion time and vehicle dispatchment costs	Energy	Time and weight	–	✓	✓	–	Solver	Iterative Local Search and Variable Neighborhood Descent based on segment representation	10	200
Leon-Blanco et al. (2022)	2022	1	m	Completion time	Time	Time	–	–	✓	–	–	Agent based heuristic	–	500
Luo et al. (2022)	2022	k	m	Completion time and vehicle dispatchment costs	Energy	Time and weight	✓	✓	✓	✓	Solver	Iterative Local Search based on segment representation	10	100
Masone et al. (2022)	2022	1	1	Completion time	Energy	Time and weight (Non-linear)	✓	✓	–	–	–	Iterative Local Search with Discrete and Continuous Improvements	–	70
Meng et al. (2023b)	2023	1	m	Routing and vehicle dispatchment costs	Energy	Time and weight	✓	✓	✓	✓	Solver	Two-stage simulated annealing	11	101
Meng et al. (2024)	2023	k	m	Routing, drone energy and vehicle dispatchment costs	Energy	Time and weight	✓	✓	✓	✓	Solver	Adaptive Large Neighborhood Search	10	50
Yin et al. (2023)	2023	k	1	Routing and vehicle dispatchment costs	Energy	Time	✓	–	✓	–	*B&C&P*	–	45	–
Jiang et al. (2024)	2024	k	m	Routing costs	Distance	Distance	✓	–	✓	✓	Solver	Adaptive Large Neighborhood Search	10	100
Boccia et al. (This work)	2024	1	1	Completion time	Time	Time	✓	✓	✓	–	*B&C*	Matheuristic	20	75

system, the number of drones mounted on each truck, and the objective function considered. The following six columns highlight key features of the delivery system: the unit of measurement for drone battery capacity, the physical parameter affecting energy consumption, whether the drone payload capacity is considered, whether the drone is required to hover at collection points for safety reasons while waiting for the truck, if the truck can directly serve customers, and if both pick-up and delivery operations are included. The final four columns provide some details about the solution method and the largest instance size solved, in terms of customers considered, categorized into exact and heuristic approaches.

Given Table 1, in terms of delivery system features, it is evident that almost no contribution tackles the very same problem due to variations in fleet composition, objective functions, or operating conditions. Additionally, the contributions can be categorized based on their objective function: those aiming to minimize completion time (Poikonen and Golden, 2020; Gonzalez-R et al., 2020; Masone et al., 2022; Kitjacharoenchai et al., 2020; Luo et al., 2021; Gu et al., 2022; Leon-Blanco et al., 2022; Luo et al., 2022), and those focusing on reducing operational costs (routing costs, number of vehicles used, etc.) (Wang and Sheu, 2019; Yin et al., 2023; Karak and Abdelghany, 2019; Meng et al., 2023b, 2024; Jiang et al., 2024). In terms of kind of optimization problems, the former represents min–max problems, while the latter are min-sum problems. Specifically, when multiple tandems are available, the typical objective is to minimize operational costs. Conversely, with only a single tandem available (even with multiple drones mounted on the truck), the usual objective is the minimization of completion time. It can be also noticed that drone flight is always constrained by battery capacity, generally quantified in terms of maximum energy or time. The energy consumption is usually directly proportional to travel time or a function of both travel time and carried weight. Additionally, some studies do not consider the drone hovering while waiting for the truck, assuming it can land at rendezvous locations (Wang and Sheu, 2019; Yin et al., 2023; Gonzalez-R et al., 2020; Karak and Abdelghany, 2019; Kitjacharoenchai et al., 2020; Leon-Blanco et al., 2022; Jiang et al., 2024). The customers that can be served within the same sortie are also often limited by the maximum payload, especially in problems involving both pick-up and delivery operations (Karak and Abdelghany, 2019; Luo et al., 2022; Meng et al., 2023b, 2024; Jiang et al., 2024). Furthermore, a few contributions consider delivery systems where the truck cannot directly serve customers, acting only as a mobile depot for the drone (Karak and Abdelghany, 2019; Poikonen and Golden, 2020; Masone et al., 2022).

In terms of solution approaches, the majority of contributions propose a mathematical formulation solved using an optimization solver, such as Cplex or Gurobi. This trend can be attributed to the fact that the main contribution of these works lies in their heuristic approach, generally based on the large neighborhood search framework (Pisinger and Ropke, 2019). Regarding exact methods, only two works have developed tailored exact solution approaches, specifically, a Branch-and-Price (*B&P*) algorithm and a Branch-and-Cut-and-Price (*B&C&P*) algorithm. However, both the approaches tackle min-sum optimization problems.

Based on this discussion, it is evident that our work is the only one proposing an ad-hoc exact solution method for a min–max version of a multi-visit truck-and-drone routing problem. Other methods are tailored for very specific problem variants, limiting their applicability to the problem under study without significant modification and no guaranteed effectiveness.

On this basis, the only two works relevant for comparison in our study are those considering the truck–drone team logistics setting as a framework for the delivery system, specifically Gonzalez-R et al. (2020) and Leon-Blanco et al. (2022). The first defines the truck–drone team logistics setting, extending the one considered in the *FSTSP* by including multiple visits per sortie, but simplifies synchronization aspects by allowing the drone to land at the rendezvous location. To address the problem, the authors present a *MILP* formulation and an iterative local search heuristic. The *MILP* formulation is solved using Gurobi which manages to determine the optimal solution for instances with up to 10 customers, while the heuristics is tested on instance with up to 250 customers. The second study, while operating within the same truck–drone team logistics setting but with multiple drones on the truck, does not propose an exact approach or a mathematical formulation, but rather a heuristic approach. This heuristic was also tested in scenarios involving a single drone, revealing that the heuristic method from Gonzalez-R et al. (2020) performs slightly better. Consequently, the approaches in Gonzalez-R et al. (2020) represent the state-of-the-art for the *TDTLP*. Therefore, we will compare our solution methods with theirs in the computational results section. Specifically, we will compare our proposed *B&C* approach with the exact method introduced in Gonzalez-R et al. (2020) for small-sized instances. For instances of a more realistic size, we will compare our *Matheuristic* approach with the heuristic method detailed in Gonzalez-R et al. (2020).

3. Problem description and formulation

In this section, we briefly recall the *TDTLP* description. Since the *TDTLP* can be viewed as an extension of the *FSTSP*, the description employs the same assumptions found in Murray and Chu (2015) for the *FSTSP*, with the only modification being the number of customers the drone can serve within a single sortie. We also introduce an original formulation for the *TDTLP* that represents an advancement over the formulation proposed in Boccia et al. (2023) for the *FSTSP*. The proposed formulation can be applied to solve both versions of the *TDTLP*, with the only difference being in the drone endurance constraints.

3.1. Problem description

The *TDTLP* considers a set of customers, C, that have to be served only once, either by the truck or the drone. The two vehicles depart from and return to a single depot once. The proposed formulation splits the depot into a source node, s, and a destination node, t. Moreover, let V be the set of nodes, $V = C \cup \{s,t\}$, and A the set of arcs, $A = \{(i,j) : i \in C \cup \{s\}, j \in C \cup \{t\}\}$. The travel time of the truck and the drone on each arc $(i,j), (i,j) \in A$, is indicated with t_{ij} and d_{ij}, respectively.

The truck has a sufficiently large capacity, enabling it to serve all the customers met along its route. Moreover, it acts as a mobile depot for the drone providing the packages and dispatching the drone to serve the customers.

The drone can serve more than one customer for each sortie. Therefore, a sortie is a drone path from the launch node to the rendezvous node. The launch and the rendezvous nodes can be either the depot or a customer node. In addition, they must be different except if there is only one sortie starting from and ending at the depot. The duration of a sortie is limited by the drone endurance (Dtl), moreover there is a maximal number of clients $Cmax$ that can be served in a single sortie. In the *TDTLP-L*, the drone can land at the rendezvous node while waiting for the truck, so the flight time, which cannot exceed its endurance, is equal to its travel time. On the other hand, in the *TDTLP-H*, the drone cannot land due to safety reasons and must hover while waiting for the truck, meaning the drone hovering contributes to its flight time and, as a consequence, to its energy expenditure. In both versions, it is clear that the two vehicles must be synchronized at the beginning and at the end of a sortie.

Moreover, service times arise when the drone is launched (launch time, SL) or retrieved (recovery time, SR). We point out that launch operations are performed when the drone is still on the truck, while rendezvous operations are conducted while the drone is airborne. Therefore, the recovery time is included in the endurance computation to ensure that the drone battery is not depleted during these operations. Moreover, if a drone is launched from the depot, there is no launch time since the related operations can be performed in advance.

The objective is to minimize the duration of the delivery process, i.e., the time needed by the two vehicles to serve all the customers and return to the depot.

Based on this description, we can observe that in any feasible solution, the number of sorties cannot exceed the value $s_{max} = \lfloor (|V| - 1)/2 \rfloor$. This value is derived from the assumption that the simplest sortie consists of just one truck arc and two drone arcs, used to serve a single client. Therefore, the number of nodes served by the truck, equivalent to the number of sorties, cannot be greater than the number of nodes traversed by the truck. Following this, we will denote by $S = \{1, \dots, s_{max}\}$ the set of all potential

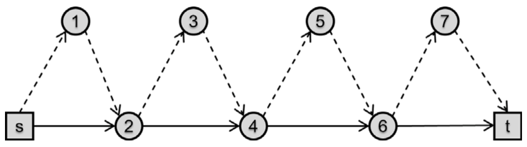

Fig. 1. Feasible solution with the maximal number of smallest sorties.

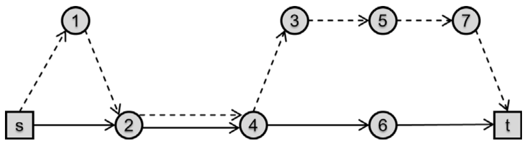

Fig. 2. An example of feasible solution.

sorties. In Fig. 1 an example of feasible solution for an instances of 9 nodes with the maximal number of sorties is reported. The continuous arcs represent the truck path, while the dashed arcs indicate the drone path.

In conclusion, to solve a *TDTLP*, we must determine: the number of sorties and, for each sortie, the customers served by the drone; the drone path between the launch and rendezvous nodes for each sortie; and the truck route. For the sake of clarity, Fig. 2 presents an example of a generic feasible solution for an instance with 9 nodes.

3.2. Problem formulation

To model the *TDTLP* as a *Mixed Integer Linear Programming (MILP)* problem, we introduce the following sets of variables based on the previous problem description:

- y_{ij}, $(i,j) \in A$, is equal to 1 if the arc (i,j) is traveled by the truck, 0 otherwise.
- y_{ij}^k, $(i,j) \in A$, $k \in S$, is equal to 1 if the arc (i,j) is traveled by the truck while the drone is performing the kth sortie, 0 otherwise.
- x_{ij}, $(i,j) \in A$, is equal to 1 if the arc (i,j) is traveled by the drone, 0 otherwise.
- x_{ij}^k, $(i,j) \in A$, $k \in S$, is equal to 1 if the arc (i,j) is traveled by the drone in its kth sortie, 0 otherwise.
- θ^h, $h \in C$, is equal to 1 if customer h is served by the drone, 0 otherwise.
- ω_i^k, $i \in C \cup \{s\}, k \in S$, is equal to 1 if node i is the launch node of the kth sortie, 0 otherwise.
- δ_j^k, $j \in C \cup \{t\}, k \in S$, is equal to 1 if node j is the rendezvous node of the kth sortie, 0 otherwise.
- σ^k, $k \in S$, is a continuous variable indicating the truck waiting time at the destination node of the kth sortie. If sortie k does not happen, then $\sigma^k = 0$.

For clarity, Fig. 3 depicts one of the sorties of a feasible solution, along with the corresponding values of the binary variables involved in the sortie. As we can observe from the example, it is important to note that there is a subtle difference between the x_{ij} and x_{ij}^k variables. The values of these two types of variables are equal when the drone is actively serving a customer during a sortie. For instance, in the example, $x_{jh} = x_{jh}^k = 1$ indicates that the drone travels the arc (j,h) and serves customer h. However, for the arc (i,j), while $x_{ij} = 1$, there is no x_{ij}^k equal to 1 for any k because the drone travels the arc (i,j) atop the truck, as indicated by $y_{ij} = 1$, which shows that the truck also travels the same arc. For the nodes visited by both vehicles, we recall that the truck, due to its larger capacity, will serve them. On this basis, we observe that there is no difference between visiting and serving with respect to the y_{ij} and y_{ij}^k variables. However, we introduce the y_{ij}^k variables to track the truck path for computing the waiting time of the truck during sortie k and to assess the feasibility of the sortie in terms of drone endurance. Regarding the truck waiting time in sortie k, represented by the value of the variable σ^k, Fig. 3 also presents its expression. As we can see, the waiting time of the truck is computed as the maximum of 0 and the difference between the duration of the drone path and the truck path during sortie k. We recall that, as also shown in Boccia et al. (2023), due to the synchronization of the two vehicles at drone launch and rendezvous locations, the total completion time (i.e., the makespan) of the delivery process can be expressed as a sum of three components: the truck routing time, the overhead time related to the launch and rendezvous locations, and the truck waiting time for the drone. It is also possible to express the completion time as a sum of the drone routing time, the overhead time, and the drone waiting time for the truck. However, it is preferable to use the truck as the reference vehicle for these components since it is generally the slower of the two vehicles.

Consistently with the notation and the set of variables described above, the makespan of the delivery process, i.e., the objective function, can be written as:

$$\sum_{(i,j)\in A} t_{ij} y_{ij} + \sum_{k\in S} \sum_{i\in V\setminus\{s,t\}} (SL + SR)\, \omega_i^k + SR\, \omega_s^1 + \sum_{k\in S} \sigma^k$$

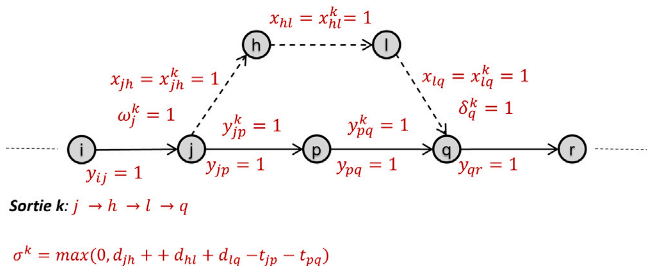

Fig. 3. kth sortie of a feasible solution.

where the first term measures the duration of the truck path, the second and the third term evaluate the total launch and recovery service time (the launch service time at the depot is equal to 0), and the fourth term assesses the total truck waiting time.

Then, the set of constraints can be divided into the following families.

Truck routing constraints

$$\sum_{j:(s,j)\in A} y_{sj} = \sum_{i:(i,t)\in A} y_{it} = 1 \tag{3.1}$$

$$\sum_{j:(i,j)\in A} y_{ij} = \sum_{j:(j,i)\in A} y_{ji} \le 1 \qquad i \in C \tag{3.2}$$

$$\sum_{i,j\in H|(i,j)\in A} y_{ij} \le \sum_{h\in H\setminus\{q\}} (1-\theta^h) \qquad H \subseteq V \quad q \in H \tag{3.3}$$

Constraints (3.1) impose that the truck path starts from the origin node s and ends in the destination node t. Constraints (3.2) are flow conservation constraints for the truck path and ensure that each customer node is visited no more than once by the truck. Constraints (3.3) are the subtour elimination constraints. As in the formulation of the *orienteering problem* (Gunawan et al., 2016), given a subset of nodes H, constraints (3.3) impose that the number of arcs with origin and destination in H traveled by the truck must be less than the number of nodes served by the truck in the subset H.

Drone routing constraints

$$\sum_{j:(s,j)\in A} x_{sj} = \sum_{i:(i,t)\in A} x_{it} = 1 \tag{3.4}$$

$$\sum_{j:(i,j)\in A} x_{ij} = \sum_{j:(j,i)\in A} x_{ji} \le 1 \qquad i \in C \tag{3.5}$$

$$\sum_{i,j\in H|(i,j)\in A} x_{ij} \le \sum_{h\in H\setminus\{q\}} \theta^h \qquad H \subseteq V \quad q \in H \tag{3.6}$$

Constraints (3.4) impose that the drone path starts from the origin node s and ends in the destination node t. Constraints (3.5) are flow conservation constraints for the drone path and ensure that each customer node is visited no more than once by the drone. Constraints (3.6) are the subtour elimination constraints.

Truck path and launch/rendezvous linking constraints

$$\sum_{j:(i,j)\in A} y^k_{ij} - \sum_{j:(i,j)\in A} y^k_{ji} = \omega^k_i - \delta^k_i \qquad i \in V \quad k \in S \tag{3.7}$$

They impose that there is a truck path without the drone onboard from the launch node to the rendezvous node of each drone sortie k.

Drone path and launch/rendezvous linking constraints

$$\sum_{j:(i,j)\in A} x^k_{ij} - \sum_{j:(i,j)\in A} x^k_{ji} = \omega^k_i - \delta^k_i \qquad i \in V \quad k \in S \tag{3.8}$$

They impose that there is a drone path from the launch node to the rendezvous node of each drone sortie k.

Single assignment constraints

$$\sum_{j:(h,j)\in A} y_{hj} + \sum_{j:(h,j)\in A} x_{hj} \ge 1 \qquad h \in C \tag{3.9}$$

$$\sum_{j:(h,j)\in A} y_{hj} + \theta^h = 1 \qquad h \in C \tag{3.10}$$

Constraints (3.9) impose that an arc departing from client h must be traversed by the truck alone, the drone alone, or the truck carrying the drone. Constraints (3.10) impose that each customer must be served either by the truck or by the drone.

Drone endurance constraints

$$\sum_{(i,j)\in A} t_{ij} y^k_{ij} \le (Dtl - SR) \qquad k \in S \tag{3.11}$$

$$\sum_{(i,j)\in A} d_{ij} x^k_{ij} \le (Dtl - SR) \qquad k \in S \tag{3.12}$$

Constraints (3.11) and (3.12) respectively set an upper bound on the duration of the truck and drone path for the kth sortie. The upper bound corresponds to the drone endurance minus the recovery time, as the drone is airborne during these operations. In the *TDTLP-H*, both sets of constraints are considered, while in the *TDTLP-L*, only the constraints (3.12) are considered.

Capacity constraints

$$\sum_{(i,j)\in A} x^k_{ij} \le C_{max} + 1 \qquad k \in S \tag{3.13}$$

They impose a maximal number of clients served in a single sortie. We can observe that in the *FSTSP* problem C_{max} is set to 1.

Waiting time constraints

$$\sum_{(i,j)\in A} d_{ij} x^k_{ij} - \sum_{(i,j)\in A} t_{ij} y^k_{ij} \le \sigma^k \qquad k \in S \tag{3.14}$$

Constraints (3.14) are needed to consider the truck waiting time in the objective function.

Consistency constraints

(i) Linking constraints between truck variables

$$\sum_{k\in S} y^k_{ij} \le y_{ij} \qquad (i,j) \in A \tag{3.15}$$

They are variable upper bounds imposing that, during a generic sortie k, the variable y^k_{ij} can be equal to 1 only if the arc (i,j) is traversed by the truck.

(ii) Linking constraints between drone variables

$$\sum_{k\in S} x^k_{ij} \le x_{ij} \qquad (i,j) \in A \tag{3.16}$$

They impose that, during a generic sortie k, the variable x^k_{ij} can be equal to 1 only if the arc (i,j) is traversed by the drone.

(iii) Linking constraints between truck and drone variables

$$\sum_{k\in S} y^k_{ij} \le 1 - x_{ij} \qquad (i,j) \in A \tag{3.17}$$

$$\sum_{k\in S} x^k_{ij} \le 1 - y_{ij} \qquad (i,j) \in A \tag{3.18}$$

$$x_{ij} \le y_{ij} + \sum_{k\in S} x^k_{ij} \qquad (i,j) \in A \tag{3.19}$$

Constraints (3.17) guarantee that if the drone travels an arc, then the truck path during any sortie k cannot include that arc. Similarly, constraints (3.18) impose that if the truck travels an arc, then the drone path during any sortie k cannot include that arc. Finally, constraints (3.19) ensure that a drone can travel an arc only if it is carried by the truck along that arc or because the arc is part of the drone path during a generic sortie k.

(iv) Linking constraints between origin and destination sortie variables

$$\sum_{i\in V\setminus t} \omega^k_i = \sum_{i\in V\setminus s} \delta^k_i \qquad k \in S \tag{3.20}$$

$$\omega^k_i + \delta^k_i \le 1 \qquad k \in S \quad i \in V \setminus \{s,t\} \tag{3.21}$$

Constraints (3.20) guarantee that, if a sortie k exists, it must have both an origin and a destination node. Constraints (3.22) ensure that a node cannot simultaneously be the origin and destination of the same sortie.

(v) Linking constraints between drone and sortie variables

$$\omega^k_i \ge \sum_{j|(i,j)\in A} x^k_{ij} - \theta^i \qquad k \in S \quad i \in V \setminus \{s\} \tag{3.22}$$

$$\delta^k_j \ge \sum_{i|(i,j)\in A} x^k_{ij} - \theta^j \qquad k \in S \quad j \in V \setminus \{t\} \tag{3.23}$$

$$\omega^k_i \le \sum_{j|(i,j)\in A} x^k_{ij} \qquad k \in S \quad j \in V \setminus \{t\} \tag{3.24}$$

$$\delta^k_j \le \sum_{i|(i,j)\in A} x^k_{ij} \qquad k \in S \quad j \in V \setminus \{t\} \tag{3.25}$$

If the drone traverses the arc (i,j) during its kth sortie, then constraints (3.22) ensure that node i is either the origin of the sortie or it is served by the drone. Likewise, constraints (3.23) guarantee that node i is either the destination of the sortie or it is served by the drone. Finally, if the arc (i, j) is not traveled by the drone in its kth sortie, (3.24) and (3.25) guarantee that the nodes i and j cannot be the origin and the destination of the kth sortie.

Symmetry breaking constraints

$$\sum_{i \in V \setminus \{s\}} \omega_i^k \leq \sum_{i \in V \setminus \{s\}} \omega_i^{k-1} \qquad k \in S \setminus \{1\} \tag{3.26}$$

$$\sum_{j \in V \setminus \{t\}} \delta_j^k \leq \sum_{j \in V \setminus \{t\}} \delta_j^{k-1} \qquad k \in S \setminus \{1\} \tag{3.27}$$

They are used to reduce the dimension of the problem by eliminating the symmetric solutions while preserving the existence of at least one of them.

4. Branch-and-Cut approach

The proposed formulation includes an exponential number of constraints due to the subtour elimination constraints (3.3), (3.6). To address this, we developed a *B&C* algorithm to solve the formulation. Furthermore, a set of valid inequalities are integrated within the *B&C* to strengthen the formulation and enhance the quality of the lower bound. In this section, we will first present the proposed valid inequalities and then detail the implementation strategies employed for our *B&C*.

4.1. Valid inequalities

Proposition 1. *Given the graph $G(V, A)$ and two nodes p and q, p, $q \in V$, let $Cut(p, q)$ be a generic cut on the graph G between the nodes p and q, and let $h, h \in C$, be a generic customer. Then, the truck cut inequalities:*

$$\sum_{(i,j) \in Cut(s,h)} y_{ij} \geq 1 - \theta^h \tag{4.1}$$

and

$$\sum_{(i,j) \in Cut(h,t)} y_{ij} \geq 1 - \theta^h \tag{4.2}$$

are valid for the TDTLP problem formulation.

Proof. If the drone does not serve customer h ($\theta^h = 0$), there must be a truck path from the origin node s to h and from h to the destination node t in any feasible solution. Therefore, an arc must cross any cut from s to h and from h to t. $\square$

Analogously to the *truck cut* inequalities, it is possible to define the following *drone cut* inequalities, which are also valid for the *TDTLP*.

$$\sum_{(i,j) \in Cut(s,h)} x_{ij} \geq \theta^h \tag{4.3}$$

and

$$\sum_{(i,j) \in Cut(h,t)} x_{ij} \geq \theta^h \tag{4.4}$$

Finally, it is also simple to prove that constraints (4.1), (4.2), (4.3) and (4.4) can be used as *subtour elimination* constraints.

4.2. Implementation details

The implementation of the *B&C* algorithm follows a classical framework, which includes solving a relaxed problem and introducing violated cuts. Therefore, it begins with the solution of the linear programming relaxation consisting in:

- the set of y_{ij}, y_{ij}^k, x_{ij}, x_{ij}^k, θ^h, ω_i^k, δ_j^k, and σ^k variables; the subset of truck routing constraints (3.1) and (3.2);
- the subset of drone routing constraints (3.4) and (3.5);
- the truck and drone path and launch/rendezvous linking constraints (3.7) and (3.8);
- the single assignment constraints (3.9) and (3.10);
- the drone endurance constraints (3.11) and (3.12) if solving the *TDTLP-H*, or only the constraints (3.12) if solving the *TDTLP-L*;
- the capacity constraints (3.13);
- the waiting time constraints (3.14);
- the consistency constraints (3.15)–(3.25);
- the symmetry breaking constraints (3.26) and (3.27).

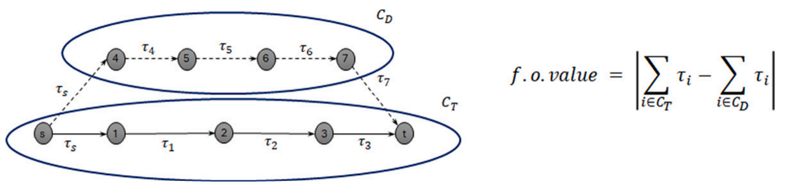

Fig. 4. Example of feasible solution when $t_{ij} = d_{ij} = \tau_i \quad \forall (i,j) \in A$.

Then, at each node belonging to the first three depths of the enumeration tree, we separate and add to the formulation the *truck (drone) cut* inequalities (4.1)–(4.4). The separation procedure solves the *minimum cut problem* (Karger and Stein, 1996), from the origin node s to h and from h to the destination node t on the weighted graph $G'(V,A,W)$ where the weight w_{ij} on the generic arc $(i,j),(i,j) \in A$, is given by the value of the corresponding y_{ij} (x_{ij}) variable. If the value of the minimum cut is less than $1-\theta^h$ (θ^h), then the violated cut inequality is added to the formulation. Successively, at each other node of the enumeration tree, if an integer solution is found, we separate the *lazy constraints* on truck and drone subtour (3.3) and (3.6).

In Section 6.1.1, the *B&C* algorithm was experimented on small-size instances, up to 20 nodes, and the results were compared with the exact approach described in Gonzalez-R et al. (2020) which, to the best of the authors knowledge, currently represents the state of the art solution for this problem. The experimentation reveals that, within a 15-minute time limit, small instances of 10 nodes can be solved to the optimality, while for 20-node instances, the proposed approach is able to obtain feasible solutions with an average gap of 20%. For medium and large-sized instances, it becomes necessary to develop a heuristic approach able to obtain "good" solutions within reasonable computation times.

5. Matheuristic approach

In this section, a *Matheuristic* approach for solving the *TDTLP* on medium-size instances is described. The proposed approach is based on two observations derived from the experimental analysis:

- the order in which customers are visited in an optimal solution of the *TDTLP* is generally *very similar* to the visitation order obtained by solving the *Hamiltonian Path* problem on the same instance, from the origin node s to the destination node t;
- solving the *TDTLP* on an acyclic graph is *much simpler* than on a fully connected graph.

To build an acyclic graph, we can consider an ordered list of all the nodes from the set V, $L = \{p_1, p_2, \dots, p_n\}$, where $p_i \in V$ indicates the node in position i in the list L. In this ordered list, the relation *a comes before b* ($a \prec_L b$) is a strict total order relation on the set of nodes.

Let the *L-Ordered Graph* $G_L(V, A_L)$ be a sub-graph of $G(V,A)$ with the same set of nodes, but only a selected subset of its edges ($A_L \subseteq A$). An arc (i,j) belongs to A_L iff $i \prec_L j$. $G_L(V,A_L)$ is an acyclic graph.

Proposition 2. *The TDTLP defined on a Directed Acyclic Graph is a NP-hard problem.*

Proof. To prove the thesis, let us examine a particular instance of the problem in which:

- the drone has infinite endurance ($Dtl = \infty$);
- the time required for both the drone and the truck to traverse any arc $(i,j) \in A$ is identical;
- the traversal time for an arc (i,j) is determined exclusively by its tail node i ($t_{ij} = d_{ij} = \tau_i \quad \forall (i,j) \in A$).

Let C be the set of clients of this instance and let $C_T, C_D \subseteq C$ be a partition of C. It can be straightforwardly demonstrated that there is exactly one feasible solution of the problem in which all clients in C_T are served by the truck, and all clients in C_D are served by the drone. The paths taken by both the truck and the drone to serve their respective clients are predetermined by the topological ordering of the nodes within the Directed Acyclic Graph. Moreover, as depicted in Fig. 4, each node i belonging to either the truck or drone path increases the path length by τ_i. Since the problem objective is to minimize the difference between the lengths of the two paths, finding the optimal solution requires to solve a partition problem. This entails identifying two subsets within C, C_T (clients served by the truck) and C_D (clients served by the drone), with the goal of minimizing the difference between the sum of the weights in the two subsets. The problem that arises is a partition problem. Given that partition problems are NP-hard, as shown in Mertens (2005), Garey and Johnson (1979), the proposition holds. □

Even though the problem remains NP-hard, the *MILP* formulation of the *TDTLP* on a Directed Acyclic Graph can be greatly simplified. Furthermore, our computational experiments, as detailed in Section 6, show that by considering the ordered list L obtained from the optimal solution of the Hamiltonian Path on the original graph $G(V,A)$, we achieve "good" upper bounds.

In other words, to derive an upper bound for the optimal solution of the *TDTLP*, we define the ordered list L as follows: the first element of L is the origin node, and the order of the subsequent nodes is determined by their position within the minimum Hamiltonian Path, from the origin node to the destination node on the original graph $G(V,A)$. Based on this, we construct the *L-Ordered Graph* $G_L(V,A_L)$ and then we solve the *TDTLP* on this acyclic graph.

In Fig. 5, the main steps of the heuristic are described:

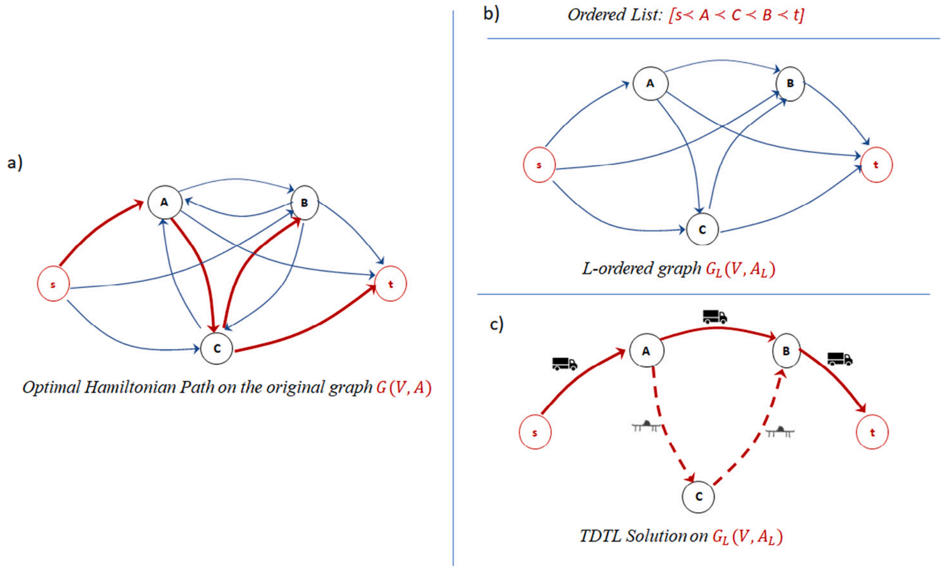
a)
Optimal Hamiltonian Path on the original graph $G(V, A)$
b)
Ordered List: [s≺ A ≺ C ≺ B ≺ t]
L-ordered graph $G_L(V, A_L)$
c)
TDTL Solution on $G_L(V, A_L)$

Fig. 5. Matheuristic scheme.

(a) starting from the original graph $G(V, A)$ in which the depot has been splitted into an origin node with only outgoing arcs and a destination node with only incoming edges, the minimum Hamiltonian Path is computed;
(b) considering the ordered list defined by the minimum Hamiltonian Path, the *L-ordered graph* $G_L(V_L A_L)$ is generated;
(c) the *TDTLP* on the acyclic *L-ordered graph* $G_L(V_L A_L)$ is solved.

In the computational experience, described in Section 6, the minimum Hamiltonian Path is computed by using a "classical" ILP formulation in which the subtour elimination constraints are separated as lazy constraints. Finally, to solve the *TDTLP* on the acyclic graph, we used the formulation described in the following subsection together with new families of valid inequalities and fixed strategies.

5.1. MILP formulation on a L-Ordered Graph

Given a *L-Ordered Graph* $G_L(V, A_L)$, we can simplify the formulation of the *TDTLP* (as outlined in the previous section) as reported below. For the sake of readability and simplicity, we slightly abuse notation by replacing p_i (the node in position i in the list L) with i.

$$\min z = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} t_{ij} y_{ij} + \sum_{k=1}^{s_{max}}\sum_{i=1}^{n-1}(SL + SR)\,\omega_i^k + SR\,\omega_s^1 + \sum_{k=1}^{s_{max}} \sigma^k$$

$$\sum_{j=2}^{n} y_{1j} = \sum_{i=1}^{n-1} y_{in} = 1 \tag{5.1}$$

$$\sum_{j=i+1}^{n} y_{ij} = \sum_{j=1}^{i-1} y_{ji} \le 1 \qquad i \in \{2, \dots, n-1\} \tag{5.2}$$

$$\sum_{j=i+1}^{n} y_{ij}^k - \sum_{j=1}^{i-1} y_{ji}^k = \omega_i^k - \delta_i^k \qquad i \in \{1, \dots, n\} \quad k \in \{1, \dots, s_{max}\} \tag{5.3}$$

$$\sum_{j=i+1}^{n} x_{ij}^k - \sum_{j=1}^{i-1} x_{ji}^k = \omega_i^k - \delta_i^k \qquad i \in \{1, \dots, n\} \quad k \in \{1, \dots, s_{max}\} \tag{5.4}$$

$$\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} t_{ij} y_{ij}^k \le Dtl - SR \qquad k \in \{1, \dots, s_{max}\} \tag{5.5}$$

$$\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} d_{ij} x_{ij}^k \le Dtl - SR \qquad k \in \{1, \dots, s_{max}\} \tag{5.6}$$

$$\sum_{i=1}^{n-1}\sum_{j=i+1}^{n} d_{ij} x_{ij}^k - \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} t_{ij} y_{ij}^k \le \sigma^k \qquad k \in \{1, \dots, s_{max}\} \tag{5.7}$$

$$\sum_{i=1}^{j-1} y_{ij} + \sum_{k=1}^{s_{max}} \sum_{i=1}^{j-1} x_{ij}^k \geq 1 + \sum_{k=1}^{s_{max}} \delta_j^k \qquad j \in \{2, \dots, n\} \tag{5.8}$$

$$\sum_{k=1}^{s_{max}} y_{ij}^k \leq y_{ij} \qquad i, j = \{1, \dots, n\},\ i < j \tag{5.9}$$

$$\sum_{k=1}^{s_{max}} x_{ij}^k \leq 1 - y_{ij} \qquad i, j = \{1, \dots, n\},\ i < j \tag{5.10}$$

$$\omega_i^k \leq \sum_{j=1}^{i-1} \omega_j^{k-1} \leq 1 \qquad i \in V\ k \in \{2, \dots, s_{max}\} \tag{5.11}$$

$$\delta_i^k \leq \sum_{j=1}^{i-1} \delta_j^{k-1} \leq 1 \qquad i \in V\ k \in \{2, \dots, s_{max}\} \tag{5.12}$$

$$\delta_i^k \leq \sum_{j=1}^{i-1} \omega_j^k \leq 1 \qquad i \in V\ k \in \{2, \dots, s_{max}\} \tag{5.13}$$

The objective function measures the makespan given by the duration of the truck path (first term), the total launch and recovery service time where the launch service time at the depot is equal to 0 (second and third terms), and the total truck waiting time (fourth term).

Constraints (5.1)–(5.2) are referred to as *Truck routing constraints*. They require that a truck path exists from the origin node *1* to the destination node n in any feasible solution. Since the graph $G_L(V, A_L)$ is acyclic, subtour elimination constraints are not necessary.

Constraints (5.3) impose that there must be a truck path, without the drone on-board, from the launch node to the rendezvous node for each drone sortie k. In a similar vein, constraints (5.4) impose that a drone path must be present from the launch node to the rendezvous node for each sortie k.

Constraints (5.5) and (5.6) impose an upper bound for each sortie on the duration of the truck and drone path, respectively. In the *TDTLP-L*, as assumed in Gonzalez-R et al. (2020), the drone can wait for the truck on the ground at the recovery node, the set of constraints (5.5) is not considered.

Constraints (5.7) determine the truck waiting time (σ^k) for each sortie. This waiting time is given by the difference between the duration of the truck path and the duration of the drone path.

Constraints (5.8) impose that each node must be visited either by the truck or by the drone. Moreover, if it is a recovery node, both the drone and the truck must reach it.

Constraints (5.9) guarantee that, during a particular sortie k, the variable y_{ij}^k can be equal to 1 only if the arc (i, j) is traversed by the truck. Furthermore, constraints (5.10) guarantee that the drone path should be distinct from the truck path during any sortie k.

Finally constraints (5.11), (5.12) and (5.13) are not strictly necessary but can be added to the formulation to strengthen it. Specifically, constraints (5.11) ensure that a generic node i can serve as the launch node for sortie k only if there exists another node j preceding i in the relation ($j \prec_L i$), which is the launch node for the prior sortie $k - 1$. Constraints (5.12) ensure that a generic node i can serve as the recovery node for sortie k only if there exists another node j preceding i, which is the recovery node for the prior sortie $k - 1$. Constraints (5.13) ensure that a generic node i can serve as the recovery node for sortie k only if there exists another node j preceding i, which is the launch node for the same sortie k.

5.2. Variable fixing strategies

To the aim of reducing the dimension of the problem we can set to 0 the following variables:

(1) ω_i^k with $i \leq 2 \times (k - 1)$
(2) δ_j^k with $j \leq 2 \times k$
(3) y_{ij}^k with $i \leq 2 \times (k - 1)$ or $j \leq 2 \times k$
(4) y_{ij} if $\sum_{p=i}^{j-1} d_{p,(p+1)} > Dtl - SR$

The first set of variables can be set to 0 because each sortie involves at least two nodes, excluding the recovery node from the previous sortie (or the origin node for the first sortie). Therefore, in any feasible solution, sortie k cannot start before node $i = (k - 1) \times 2$ (including node i). Similarly, the second set of variables can be set to 0 because each sortie cannot end before node $j = k \times 2$ (including node j). The third set of variables can be set to 0 because the generic sortie k cannot start before node $(k - 1) \times 2$ (inclusive) and cannot end before node $k \times 2$ (inclusive).

The last set of variables is composed of truck variables associated with arcs from node i to node j such that the shortest drone path serving all the nodes between them exceeds the drone endurance. Since any feasible solution must serve all nodes, such arcs cannot be traveled by a truck, and thus $y_{ij} = 0$.

6. Computational results

This section presents and discusses the computational results of the experimentation performed to evaluate and validate the proposed Exact (*B&C* algorithm) and *Matheuristic* approaches on both the *TDTLP-L* and *TDTLP-H*. Both the algorithms have been coded in Python language using Gurobi 10.7.3 with default setting as MILP solver. The experiments are performed on an Intel(R) Core(TM) i7-8700, 3.20 GHz, 16.00 GB of RAM.

We consider the same set of instances as those considered in Gonzalez-R et al. (2020), where the *TDTLP* was first defined. These instances are built upon those originally outlined in Agatz et al. (2018) and are available in Bouman et al. (2018). They are derived through a process of randomly positioning each node within a plane, with the time of travel between nodes calculated proportionally to the Euclidean distance, and held constant for each pair of nodes. The test bed is divided into three categories: (i) uniform, (ii) 1-center, and (iii) 2-center. In the 'uniform' category, the node positions are randomly distributed according to a uniform integer distribution ranging from 1 to 100. The '1-center' and '2-center' categories are designed to simulate customer densities in urban landscapes with one or two circular city centers, respectively. Moreover, a constant battery life Q is considered for all instances. This life is calculated as proportional to the average drone travel time between each pair of nodes ($Q = 2/|A| \sum_{(i,j)\in A} d_{ij}$). With regard to the truck and drone speed we consider three cases: in the first one ($\alpha = 1$) the drone and truck have equal speed on all the edges; in the second one ($\alpha = 2$) the drone is twice as fast as the truck on all edges; finally, in the third case, ($\alpha = 3$) the drone is three times as fast as the truck. Lastly, in every instance, the starting node is the same as the depot, and the destination node is defined as the last node in each instance specification.

The results of the experimentation are divided into two distinct subsections, one for each *TDTLP* version. In particular, we first report the results for the *TDTLP-L*, as this version is the one addressed in Gonzalez-R et al. (2020). Moreover, the *TDTLP-L* serves as a lower bound for the *TDTLP-H*. Consequently, in the section related to the *TDTLP-H*, we will also compare the solutions of the two problems in terms of delivery completion time and evaluate the solution approaches in terms of running times.

To avoid any bias related to parameter settings, we use the same settings for both experiments as those considered in Gonzalez-R et al. (2020). In particular, no launch and recovery times are considered ($SR = SL = 0$), and there is no limit to the maximum number of clients for a single sortie ($C_{\max} = \infty$). The only limit to a drone path is its endurance (Dtl) that is set equal to the battery life Q. All the detailed results for both the *TDTLP-L* and the *TDTLP-H* versions can be found in Appendix A.

6.1. *TDTLP-L experiments*

In this subsection, we present the results of the proposed approaches for the *TDTLP-L* and compare them with the results obtained by the solution methods proposed in Gonzalez-R et al. (2020). In particular, we first discuss the results achieved using the proposed *Branch-and-Cut* algorithm and compare these with the results from the exact approach proposed in Gonzalez-R et al. (2020). Then, we evaluate the performance of the proposed *Matheuristic* and compare it with the *Simulated Annealing* heuristic, also presented in Gonzalez-R et al. (2020) that represents the current state-of-the-art for the *TDTLP*.

For the exact approach, we consider instances with a number of nodes n equal to 10 and 20. In contrast, for the heuristic approach experimentation, we use instances of size n equal to 20 again, but also instances with n equal to 50 and 75.

Since the test bed used for the study includes 10 different instances for each combination of *instance category*, value of α, and size n, it totals 360 instances. Out of these, 180 instances were used for testing the exact approach, and 270 instances were used for the heuristic approach, with an overlap of 90 instances being used in both approaches.

6.1.1. *Exact approach*

Tables 2 and 3 present a comparison between the exact approach outlined in Gonzalez-R et al. (2020) and the proposed Branch-and-Cut method. The comparison is made on instances with 10 and 20 nodes, imposing a computation time limit of 1800 secs. The columns within the macro-column *Gonzales et al. (2020) Exact Approach* are related to the results described in [3], while the *Branch-and-Cut* macro-column presents the results of the proposed exact approach. The test bed used for the study includes 10 different instances for each combination of *instance category*, value of α, and size n. Consequently, each row in both tables presents information derived from the solution of the 10 instances using the exact approach. In particular, the *Avg Gap* column reports the average gap value between lower and upper bound, computed as $(UB - LB)/UB * 100$; the *Max Gap* column shows the maximum gap value over the 10 instances; the *Avg Time* column reports the average time taken to solve each instance; and the *#Opt* column indicates the number of instances solved to optimality within a time limit of 1800 secs.

As can be seen from Table 2, the proposed exact approach successfully solves to optimality all instances with 10 nodes within a maximum time of 18 seconds. On the other hand, within a time limit of 1800 seconds, the exact approach proposed by Gonzalez-R et al. (2020) manages to solve only 55 out of the 90 instances of size $n = 10$, and in several cases, the resulting gap is quite high (more than 90%).

Solving instances of size $n = 20$ to optimality is much more challenging compared to those of size $n = 10$. Our *B&C* approach successfully solves to optimality only three 'double-center' instances with $\alpha = 1$. In contrast, the exact approach detailed in Gonzalez-R et al. (2020) does not achieve optimality for any instance. Consequently, in Table 2, we have replaced the *#Opt* columns with two columns named *#UB*. These columns report the number of instances for which a feasible solution has been identified. Our *B&C* approach, within a time limit of 1800 seconds, manages to find feasible solutions for all $n = 20$ instances. On the other hand, the exact approach proposed in Gonzalez-R et al. (2020) is able to find feasible solutions for only 56 out of 90 instances. Additionally, when examining the gaps between lower and upper bounds, our approach consistently achieves an average gap of approximately

Table 2
Exact approach - $n = 10$.

Type	α	Branch and Cut				Gonzales et al. (2020) exact approach			
		Avg Gap	Max Gap	Avg Time	#Opt	Avg Gap	Max Gap	Avg Time	#Opt
Uniform	1	0,00	0,00	10,89	10	14,94	75,12	1018,42	8
	2	0,00	0,00	14,73	10	13,75	83,97	1170,75	8
	3	0,00	0,00	19,25	10	25,06	85,91	1255,62	6
1-center	1	0,00	0,00	15,69	10	9,11	91,12	939,23	9
	2	0,00	0,00	15,49	10	32,12	89,99	1239,93	4
	3	0,00	0,00	14,43	10	45,87	92,77	1322,15	3
2-center	1	0,00	0,00	11,93	10	14,03	60,33	1313,12	5
	2	0,00	0,00	18,04	10	21,38	77,20	1465,93	6
	3	0,00	0,00	18,21	10	20,99	87,04	1078,95	6
All		0,00	0,00	15,41	90	21,92	91,12	1200,46	55

Table 3
Exact approach - $n = 20$.

Type	α	Branch and Cut				Gonzales et al. (2020) exact approach			
		Avg Gap	Max Gap	Avg Time	#UB	Avg Gap	Max Gap	Avg Time	#UB
Uniform	1	11,41	21,86	1800,00	10	88,61	93,62	1800,00	10
	2	17,69	26,75	1800,00	10	85,27	92,72	1800,00	9
	3	24,20	39,20	1800,00	10	–	–	1800,00	0
1-center	1	22,59	30,09	1800,00	10	91,42	97,90	1800,00	10
	2	22,64	33,95	1800,00	10	91,43	97,54	1800,00	3
	3	23,82	42,71	1800,00	10	–	–	1800,00	0
2-center	1	6,05	15,85	1564,82	10 (3*)	93,83	96,89	1800,00	10
	2	17,12	37,78	1800,00	10	91,55	96,42	1800,00	10
	3	22,67	55,79	1800,00	10	88,76	96,54	1800,00	4
All		18,69	55,79	1773,87	90	90,12	97,90	1800,00	56

20% across all instance types, while the exact approach in Gonzalez-R et al. (2020) exhibits significantly higher average gaps, approximately around 90%.

Due to the high gap values already observed for instances of 20 nodes, the *Matheuristic* approach described in Section 5 was experimented for these instances and for even larger ones. The results of this experimentation are detailed in the following subsection.

6.1.2. Heuristic approach

To demonstrate the effectiveness of our *Matheuristic* approach, in Table 4, for the smaller instances of 10 nodes, we compare the feasible solutions provided by it with the optimal solutions found by the *B&C* algorithm. The *Avg Gap* column reports the average gap between the upper bound computed by the *Matheuristic* and the value of the optimal solution for each class of instances; the *#Opt* column indicates the number of instances for which the upper bound provided by the *matheuristic* coincides with the optimal solution; finally, the *Avg Time* column indicates the average computational time taken by the *Matheuristic* to find the upper bound. The *Matheuristic* approach solves all the $n = 10$ instances in a negligible amount of time, achieving an average gap from the optimal solution of less than 4%. Moreover, it provides the optimal solution for 38 out of 90 instances.

For instances of size n = 20, Table 5 compares the results obtained using the *B&C* algorithm with a time limit of 1800 s, with those obtained using the proposed *Matheuristic* and the heuristic described in [3]. The *#Best Sol* column in each macro column indicates the number of instances for which the corresponding approach found the best upper bound. The *Avg Gap from Best* columns report the average gap between the upper bound provided by the corresponding method and the best upper bound, calculated using the formula $(UB - UB_{Best})/UB * 100$. The *Avg Time* columns show the average computational times. It is important to note that in the experiments reported in [3], the heuristic was executed 10 times for each instance. As a result, the data in the *#Best Sol* and *Avg Gap from Best* columns under the *Gonzales et al. 2020* macro-column reflect the results of these 10 runs, and in the*Avg Time* column each item was computed considering the cumulative computational time for these runs for each instance. Table 5 does not include the *Avg Time* column for the *B&C* algorithm as it reaches the imposed time limit of 1800 seconds for all instances.

Table 5 shows how the proposed *Matheuristic* is able to provide the best solution for 15 out of 90 instances of size $n = 20$, while the approach proposed in *Gonzales et al. 2020* provides the best solution for only 4 instances. Moreover, the average gap compared to the best solution is equal to 4.4% when using our heuristic, whereas it is equal to 6.6% with the second approach.

The effectiveness of the *Matheuristic* approach is further confirmed when considering larger instances of 50 and 75 nodes. To effectively solve these instances, we set a time limit equal to 300 and 600 secs respectively when the *L-ordered Graph* (5.1)–(5.13) is solved. The solution of the Hamiltonian Path on the same instances to define the *L ordered list* takes always less than 1 s for the

Table 4
Matheuristic approach - $n = 10$.

Type	α	Matheuristic		
		Avg Gap	#Opt	Avg Time
Uniform	1	1,46	7	0,21
	2	5,56	4	0,12
	3	7,13	4	0,12
1-center	1	1,88	4	1,11
	2	3,32	2	0,1
	3	4,88	2	0,08
2-center	1	1,86	5	0,23
	2	2,9	6	0,11
	3	5,72	4	0,09
All		3,86	38	0,24

Table 5
Matheuristic approach - $n = 20$.

Type	α	Branch and Cut		Matheuristic			Gonzalez et al. 2020		
		Avg Gap from best	#Best Sol	Avg Gap from best	#Best Sol	Avg Time	Avg Gap from best	#Best Sol	Avg Time
Uniform	1	0,17	8	3,87	2	18,16	11,25	0	38,25
	2	0,10	9	5,14	2	4,78	9,82	0	34,23
	3	0,61	9	8,48	1	2,95	8,61	1	35,59
1-center	1	0,42	7	3,5	3	7,31	6,9	0	34,7
	2	0,37	8	5,25	3	2,88	6,02	1	35,62
	3	0,00	10	5,92	1	2,71	6,79	1	34,97
2-center	1	0,00	10	2,36	2	12,66	4,97	0	37,58
	2	0,16	9	2,84	1	7,39	2,69	0	34,18
	3	0,12	9	2,2	0	3,7	2,29	1	35,35
All		0,22	79	4,39	15	6,95	6,59	4	35,61

Table 6
Matheuristic approach - $n = 50$.

Type	α	Matheuristic			Gonzalez et al. 2020		
		Avg Gap from best	#Best Sol	Avg Time	Avg Gap from best	#Best Sol	Avg Time
Uniform	1	0,00	10	300,96	17,08	0	476,33
	2	0,00	10	300,94	7,66	0	481,87
	3	2,39	7	300,95	5,15	3	490,73
1-center	1	0,00	10	300,88	9,66	0	463,51
	2	0,79	8	300,86	6,16	2	504,04
	3	2,96	7	298,97	6,52	3	506,82
2-center	1	0,00	10	300,88	12,74	0	462,77
	2	0,45	8	300,89	3,55	2	496,27
	3	1,13	3	300,9	0,17	7	487,65
All		0,86	73	300,69	7,63	17	485,55

instances of dimension $n = 50$ and less than 3 s for instances with $n = 75$. The results obtained for these instances are reported in Tables 6 and 7. These Tables do not report the results obtained using the *B&C* algorithm, since, for instances of more than 20 nodes, that approach does not provide any significant Upper Bound. The results of the Matheuristic approach are therefore compared only with those obtained from Gonzalez heuristic.

On both the 90 instances of 50 nodes and the 90 instances of 75 nodes, the heuristic approach provides the best solution in 75 instances, and generally, the average gap from the best solution is less than 1.3%. These results, in addition to confirming the effectiveness of the proposed approach, encourage the search for improved heuristic approaches that, starting from the solution provided by the Matheuristic, can attempt to further improve it within reasonable times

6.2. *TDTLP-H experiments*

In this subsection, we report the results of the proposed approaches for the *TDTLP-H*, employing the same experimental design previously reported for the *TDTLP-L* in terms of instances and parameter settings (e.g., time limit). Furthermore, to underscore the

Table 7
Matheuristic approach - $n = 75$.

Type	α	Matheuristic			Gonzalez et al. 2020		
		Avg Gap from best	#Best Sol	Avg Time	Avg Gap from best	#Best Sol	Avg Time
Uniform	1	0,00	10	602,63	20,25	0	1632,01
	2	0,00	10	602,65	9,48	0	1631,64
	3	0,77	8	602,75	4,79	2	1660,98
1-center	1	0,00	10	602,57	11,67	0	1575,19
	2	2,33	6	602,72	3,75	4	1703,96
	3	4,71	6	602,66	1,36	4	1711,06
2-center	1	0,00	10	602,38	16,35	0	1629,73
	2	0,66	8	602,39	7,02	2	1726,86
	3	3,13	7	602,45	2,72	3	1682,56
All		1,29	75	602,58	8,6	15	1661,55

Table 8
Exact and matheuristic approaches - $n = 10$.

Type	α	Branch and Cut				Matheuristic				Hover vs No hover			
		Avg Gap	Max Gap	#Opt	Avg Time	Avg Gap	Max Gap	#Opt	Avg Time	Avg Delta	Max Delta	Avg Rat_E	Avg Rat_H
Uniform	1	0,00	0,00	10	15,99	3,08	9,43	2	0,27	7,79	33,08	2,02	1,39
	2	0,00	0,00	10	7,41	8,48	20,32	3	0,07	6,80	27,78	0,72	0,64
	3	0,00	0,00	10	4,34	16,79	45,91	0	0,06	20,75	33,34	0,35	0,55
Single	1	0,00	0,00	10	19,45	2,99	6,65	2	0,16	8,64	30,50	1,95	1,60
	2	0,00	0,00	10	7,62	4,96	14,67	2	0,07	5,62	17,06	0,57	0,77
	3	0,00	0,00	10	5,49	9,35	24,93	0	0,06	10,42	21,18	0,38	0,83
Double	1	0,00	0,00	10	36,54	4,11	13,37	3	0,26	5,77	21,33	2,69	1,27
	2	0,00	0,00	10	7,39	6,01	21,15	2	0,07	7,83	23,23	0,53	0,81
	3	0,00	0,00	10	4,83	13,34	44,59	1	0,07	13,83	33,41	0,40	0,81
All		0,00	0,00	90	12,12	7,68	45,91	15	0,12	9,72	33,41	1,07	0,96

impact associated with drone hovering, we provide a comparative analysis of the solutions for the *TDTLP-H* and *TDTLP-L* across all instance sets. Specifically, we highlight the difference between the delivery times of the solutions obtained for the two problems, and we also emphasize the differences in the running times of the solution approaches.

Table 8 presents the results of both proposed approaches for instances where n equals 10. We observe that the *B&C* approach solves all instances to optimality with an average time of about ten seconds. Conversely, the *Matheuristic* approach achieves good solutions (with an average gap of less than 8%) in less than a second. Furthermore, the *Matheuristic* approach determines the optimal solutions for 15 instances. Concerning the difference between the *TDTLP-L* and *TDTLP-H*, we calculate for each instance the percentage difference between the best solution obtained by any method for the *TDTLP-H* and the *TDTLP-L*, referred to as *Delta*. Specifically, *Delta* is computed as $(\mathrm{BUB}_{\mathrm{TDTLP\text{-}H}} - \mathrm{BUB}_{\mathrm{TDTLP\text{-}L}})/\mathrm{BUB}_{\mathrm{TDTLP\text{-}L}} \cdot 100$, where $\mathrm{BUB}_{\mathrm{TDTLP\text{-}H}}$ and $\mathrm{BUB}_{\mathrm{TDTLP\text{-}L}}$ represent the best upper bound computed by any method for the *TDTLP-H* and *TDTLP-L*, respectively. We also calculate an indicator to highlight the differences in running times for both algorithms when solving the two versions. Specifically, we compute the ratio of the running time for solving the *TDTLP-H* to the running time for solving the *TDTLP-L*, for both the exact and the heuristic methods, referred to as *Rat_E* and *Rat_H*, respectively. We observe that, on average, drone hovering leads to a 10% increase in delivery time, with a worst-case scenario of about 30%. Conversely, examining the ratios seems that the average running times are similar, with a ratio equal to 1. However, it is noted that when the two vehicles have the same speed, the running times for both approaches when solving the *TDTLP-H* are higher than those for the *TDTLP-L*. Conversely, when the drone is faster than the truck, the situation is reversed. This trend can be explained by considering how the battery life is computed for these instances. Indeed, battery life depends on the drone travel time; thus, if the drone is faster, the travel times are shorter, and the battery life decreases as a consequence. Therefore, a low value of battery life leads to limited endurance, resulting in a reduced solution space. This reduction is even more pronounced if hovering contributes to energy expenditure.

Table 9 shows the results of both proposed approaches for instances with n equal to 20. Unlike the *TDTLP-L*, where none of the instances could be solved within the time limit, the *B&C* approach manages to optimally solve 11 instances for the *TDTLP-H*. Conversely, the *Matheuristic* method achieves a better or equivalent solution to the *B&C* only in 5 instances. However, it is noted that, on average, the quality of the solutions from the *Matheuristic* is only 9% higher than those obtained by the *B&C* (the *Diff* indicates the percentage difference between the upper bound computed by the *Matheuristic* and the one computed by the *B&C*). The running times for the *Matheuristic* remain very low, averaging less than 10 s. In terms of comparing the *TDTLP* versions, the same trends observed previously are confirmed for these instances as well.

Finally, Tables 10 and 11 present only the comparison between the *TDTLP-H* and *TDTLP-L* for instances with n equal to 50 and 75, respectively. This focus is due to the fact that for these instances, only the results from the *Matheuristic* approach are available, as the *B&C* method is unable to find a solution within the time limit.

Table 9
Exact and matheuristic approaches - $n = 20$.

Type	α	Branch and Cut				Matheuristic				Hover vs No hover			
		Avg Gap	Max Gap	#Best Sol	Avg Time	Avg Diff	Max Diff	#Best Sol	Avg Time	Avg Delta	Max Delta	Avg Rat_E	Avg Rat_H
	1	15,90	29,73	8	1800,00	5,44	16,07	2	38,21	5,07	13,33	1,00	3,13
Uniform	2	17,89	27,63	10	1800,00	7,29	23,17	0	2,33	2,41	8,24	1,00	0,57
	3	21,37	38,94	10	1800,00	20,43	41,63	0	0,29	8,03	22,35	1,00	0,18
	1	22,18	33,04	10	1800,00	4,52	12,56	0	14,62	11,23	24,22	1,00	5,02
Single	2	8,11	23,81	10(2*)	1699,61	7,11	21,93	1	0,48	4,94	11,73	0,94	0,35
	3	3,86	8,97	10(3*)	1389,33	16,11	41,82	0	0,20	9,05	25,70	0,77	0,21
	1	6,55	18,27	10(1*)	1676,73	3,53	17,23	0	20,09	1,57	8,33	1,51	1,64
Double	2	11,10	27,92	10(2*)	1483,46	4,53	14,32	1	1,61	2,62	7,27	0,82	0,28
	3	13,39	49,39	9(2*)	1383,58	10,01	33,19	1	0,41	5,82	19,79	0,77	0,26
All		13,37	49,39	87	1648,08	8,77	41,82	5	8,69	5,64	25,70	0,98	1,29

Table 10
Exact and matheuristic approaches - $n = 50$.

Instance	α	Hover vs No hover			
		Avg Delta	Max Delta	#Worst Sol	Avg Rat_H
	1	2,09	5,23	1	1,00
Uniform	2	−0,69	1,13	4	1,00
	3	2,73	7,82	2	0,84
	1	5,41	9,85	0	1,00
Single	2	1,33	9,16	2	0,67
	3	13,66	27,52	0	0,30
	1	−0,43	1,51	3	1,00
Double	2	0,57	5,85	2	1,00
	3	4,90	11,30	0	0,69
All		3,36	27,52	15	0,84

Table 11
Exact and matheuristic approaches - $n = 75$.

Instance	α	Hover vs No hover			
		Avg Delta	Max Delta	#Worst Sol	Avg Rat_H
	1	1,61	7,25	1	1,00
Uniform	2	0,25	2,99	4	1,00
	3	6,52	16,91	0	1,00
	1	3,54	9,26	2	1,00
Single	2	3,37	9,70	1	1,00
	3	13,27	29,49	0	0,72
	1	1,91	11,02	4	1,00
Double	2	0,41	3,96	3	1,00
	3	5,30	20,06	1	1,00
All		4,02	29,49	16	0,97

The increase in delivery time for these larger instances remains at about 25% in the worst case. However, we note that on average, the difference is around 4%, slightly lower than that observed in smaller instances. This variation is attributed to the complexity of these instances. Indeed, there are instances where the objective function value of the solution for the *TDTLP-L* is higher than that for the *TDTLP-H*. The column *# Worst Sol* indicates the number of instances where the *Matheuristic* approach on the *TDTLP-L* resulted in a worse solution compared to the one obtained for the *TDTLP-H*. In conclusion, the results of our experimentation with the *TDTLP-H* indicate that drone hovering substantially affects both the delivery time and the running times of the solution approaches.

7. Conclusions

In this study, we study a multi-visit single-truck single-drone routing problem known as *TDTLP*. We specifically investigate two versions of the problem: *TDTLP-H* and *TDTLP-L*. To tackle these versions, we develop a mathematical programming formulation and introduce a *B&C* approach. This method is enhanced with two distinct families of valid inequalities, providing a tailored exact solution for the *TDTLP*. Additionally, we propose a matheuristic approach that solves the *TDTLP* on an acyclic graph. This graph is constructed by fixing the precedence between customers based on the order determined by the Hamiltonian Path.

Computational results from different benchmark instances validate the robustness of the proposed methods and the efficacy of both the *B&C* and the matheuristic approaches. Indeed, the experiments demonstrate that our methods are competitive with or surpass existing state-of-the-art approaches, providing either optimal solutions or improved bounds for numerous previously unsolved instances.

Future research will naturally extend the proposed formulation to other multi-visit truck-and-drone routing problems, with a particular focus to those with multiple vehicles. Additionally, as done in Boccia et al. (2024), we plan to explore integrating the proposed matheuristic approach with machine learning and data science techniques, aiming to compute effective solutions while minimizing computational effort.

CRediT authorship contribution statement

Maurizio Boccia: Writing – review & editing, Writing – original draft, Supervision, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Andrea Mancuso:** Writing – review & editing, Writing – original draft, Supervision, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Adriano Masone:** Writing – review & editing, Writing – original draft, Supervision, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Claudio Sterle:** Writing – review & editing, Writing – original draft, Supervision, Software, Methodology, Investigation, Formal analysis, Conceptualization.

Data availability

Data will be made available on request.

Acknowledgments

Research presented in the paper has been partially funded by the National Centre for Sustainable Mobility “CN MOST” (MUR code CN00000023 - CUP UNINA code: E63C22000930007), by the MUR research program PRIN 2022 founded by European Union - Next Generation EU (Project “ACHILLES, eco-sustAinable effiCient tecH-drIven Last miLE logiStics”, Prot. 2022BC2CW7, CUP: E53D23005630006), and by the MUR research program PRIN 2022 PNRR founded by European Union - Next Generation EU (Project “COSMO, Co-Opetitive Sustainable Mobility Optimization”, Prot. P20224YN23 CUP: F53D23009980001).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.trc.2024.104691.

References

Agatz, N., Bouman, P., Schmidt, M., 2018. Optimization approaches for the traveling salesman problem with drone. Transp. Sci. 52 (4), 965–981.
Boccia, M., Mancuso, A., Masone, A., Murino, T., Sterle, C., 2024. New features for customer classification in the flying sidekick traveling salesman problem. Expert Systems with Applications 247, 123106.
Boccia, M., Mancuso, A., Masone, A., Sterle, C., 2023. A new MILP formulation for the flying sidekick traveling salesman problem. Networks 82 (3), 254–276.
Bouman, P., Agatz, N., Schmidt, M., 2018. Instances for the TSP with Drone (and Some Solutions). Zenodo, London, UK, p. v1.
Carlsson, J.G., Song, S., 2018. Coordinated logistics with a truck and a drone. Manage. Sci. 64 (9), 4052–4069.
Cavani, S., Iori, M., Roberti, R., 2021. Exact methods for the traveling salesman problem with multiple drones. Transp. Res. C 130, 103280.
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. Comput. Oper. Res. 123, 105004.
Garey, M., Johnson, D., 1979. Computers and intractability: A guide to the theory of NP-completeness. In: Mathematical Sciences Series, Freeman, URL https://books.google.it/books?id=fjxGAQAAIAAJ.
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. Transp. Res. C 114, 657–680.
Gu, R., Poon, M., Luo, Z., Liu, Y., Liu, Z., 2022. A hierarchical solution evaluation method and a hybrid algorithm for the vehicle routing problem with drones and multiple visits. Transp. Res. C 141, 103733.
Gunawan, A., Lau, H.C., Vansteenwegen, P., 2016. Orienteering problem: A survey of recent variants, solution approaches and applications. European J. Oper. Res. 255 (2), 315–332.
Jiang, J., Dai, Y., Yang, F., Ma, Z., 2024. A multi-visit flexible-docking vehicle routing problem with drones for simultaneous pickup and delivery services. European J. Oper. Res. 312 (1), 125–137.
Karak, A., Abdelghany, K., 2019. The hybrid vehicle-drone routing problem for pick-up and delivery services. Transp. Res. C 102, 427–449.
Karger, D.R., Stein, C., 1996. A new approach to the minimum cut problem. J. ACM 43 (4), 601–640.
Kitjacharoenchai, P., Min, B.-C., Lee, S., 2020. Two echelon vehicle routing problem with drones in last mile delivery. Int. J. Prod. Econ. 225, 107598.
Leon-Blanco, J.M., Gonzalez-R, P.L., Andrade-Pineda, J.L., Canca, D., Calle, M., 2022. A multi-agent approach to the truck multi-drone routing problem. Expert Syst. Appl. 195, 116604.
Liang, Y.-J., Luo, Z.-X., 2022. A survey of truck–drone routing problem: Literature review and research prospects. J. Oper. Res. Soc. China 10 (2), 343–377.
Luo, Z., Gu, R., Poon, M., Liu, Z., Lim, A., 2022. A last-mile drone-assisted one-to-one pickup and delivery problem with multi-visit drone trips. Comput. Oper. Res. 148, 106015.
Luo, Z., Poon, M., Zhang, Z., Liu, Z., Lim, A., 2021. The multi-visit traveling salesman problem with multi-drones. Transp. Res. C 128, 103172.
Masone, A., Poikonen, S., Golden, B.L., 2022. The multivisit drone routing problem with edge launches: An iterative approach with discrete and continuous improvements. Networks 80 (2), 193–215.

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

Meng, S., Guo, X., Li, D., Liu, G., 2023b. The multi-visit drone routing problem for pickup and delivery services. Transp. Res. Part E: Logist. Transp. Rev. 169, 102990.

Mertens, S., 2005. The easiest hard problem: Number partitioning. In: Computational Complexity and Statistical Physics. Oxford University Press, http://dx.doi.org/10.1093/oso/9780195177374.003.0012.

Murray, C.C., Chu, A.G., 2015. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. Transp. Res. C 54, 86–109.

Murray, C.C., Raj, R., 2020. The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones. Transp. Res. C 110, 368–398.

Najy, W., Archetti, C., Diabat, A., 2023. Collaborative truck-and-drone delivery for inventory-routing problems. Transp. Res. C 146, 103791.

Pisinger, D., Ropke, S., 2019. Large neighborhood search. In: Handbook of metaheuristics. Springer, pp. 99–127.

Poikonen, S., Golden, B., 2020. Multi-visit drone routing problem. Comput. Oper. Res. 113, 104802.

Schermer, D., Moeini, M., Wendt, O., 2019. A matheuristic for the vehicle routing problem with drones and its variants. Transp. Res. C 106, 166–204.

Tavares, T., 2021. Comparing the cost-effectiveness of drones v ground vehicles for medical, food and parcel deliveries. Unmanned Airspace.

Wang, Z., Sheu, J.-B., 2019. Vehicle routing problem with drones. Transp. Res. Part B: Methodol. 122, 350–364.

Wang, Y., Wang, Z., Hu, X., Xue, G., Guan, X., 2022. Truck–drone hybrid routing problem with time-dependent road travel time. Transp. Res. C 144, 103901.

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

Zwickle, A., Farber, H.B., Hamm, J.A., 2019. Comparing public concern and support for drone regulation to the current legal framework. Behav. Sci. Law 37 (1), 109–124.