Transportation Research Part C 130 (2021) 103280

Contents lists available at ScienceDirect

Transportation Research Part C

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

Exact methods for the traveling salesman problem with multiple drones

Sara Cavani [a,b], Manuel Iori [a], Roberto Roberti [b,*]

[a] Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, 42122 Reggio Emilia, Italy
[b] Department of Operations Analytics, Vrije Universiteit Amsterdam, 1081HV Amsterdam, the Netherlands

ARTICLE INFO

Keywords:
Traveling salesman
Drone
Exact method
Branch-and-cut
Mixed integer linear programming

ABSTRACT

Drone delivery is drawing increasing attention in last-mile delivery. Effective solution methods to solve decision-making problems arising in drone delivery allow to run and assess drone delivery systems. In this paper, we focus on delivery systems with a single traditional vehicle and multiple drones working in tandem to fulfill customer requests. We address the Traveling Salesman Problem with Multiple Drones (TSP-MD) and investigate the modeling challenges posed by the presence of multiple drones, which have proven to be hard to handle in the literature. We propose a compact Mixed-Integer Linear Programming (MILP) model to formulate the TSP-MD and several families of valid inequalities. Moreover, we illustrate an exact decomposition approach based on the compact MILP and a branch-and-cut algorithm. We show that this exact approach can solve instances with up to 24 customers to proven optimality, improving upon existing exact methods that can solve similar problems with up to ten customers only.

1. Introduction

The idea of delivering parcels with drones is becoming more and more popular. Since 2013, major transport and delivery companies, such as DHL and UPS, have been carrying out experiments to test the feasibility of parcel drone delivery, see, e.g., DHL Parcelcopter project[1] and UPS Flight Forward project.[2] Through its Prime Air service,[3] Amazon is also working on a system that can deliver parcels in less than 30 min. Moreover, Google with its Alphabet section has designed X-Wing,[4] an unmanned aerial vehicle specifically designed for parcel delivery. Replacing traditional vehicles with drones to deliver parcels can allow to reduce transportation costs, decrease delivery times and carbon footprint, as well as reach areas inaccessible with traditional vehicles (see, e.g., Joerss et al., 2016; Lee et al., 2016; Kellermann et al., 2020).

Parcel delivery with drones is also drawing increasing attention in the research community, particularly among researchers who develop solution methods for distribution problems. Distribution problems arising in parcel drone delivery can be classified into three main categories depending on how drones interact (if any) with traditional vehicles (i.e., trucks): (1) drones operate on their own, (2) drones cooperate with trucks, and (3) drones work in tandem with trucks - see Fig. 1 for a visual representation of distribution plans in

* Corresponding author.
E-mail addresses: 217312@studenti.unimore.it (S. Cavani), manuel.iori@unimore.it (M. Iori), r.roberti@vu.nl (R. Roberti).

[1] http://www.dpdhl.com/en/media-relations/specials/dhl-parcelcopter.html.
[2] http://www.ups.com/us/en/services/shipping-services/flight-forward-drones.page.
[3] https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011.
[4] https://x.company/projects/wing/.

https://doi.org/10.1016/j.trc.2021.103280
Received 9 November 2020; Received in revised form 25 May 2021; Accepted 23 June 2021
Available online 10 July 2021

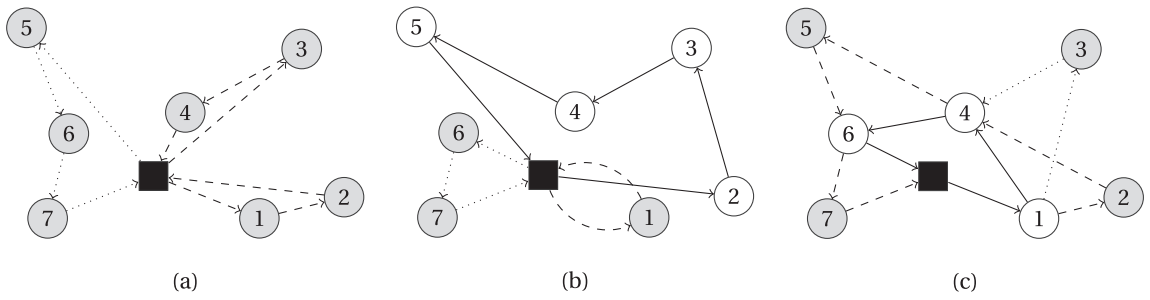

Fig. 1. Examples of distribution plans in the three types of systems: (a) drones operating on their own, (b) drones cooperating with trucks, and (c) drones working in tandem with trucks. White circles represent customers served by trucks, grey circles customers served by drones; straight lines represent the routes of the trucks, and dashed and dotted lines routes of two different drones.

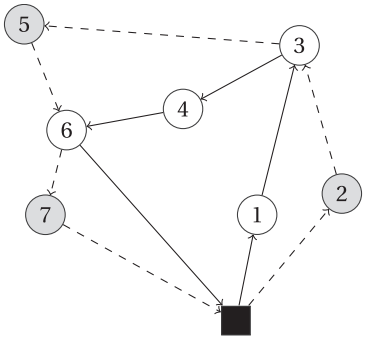

Fig. 2. A feasible solution of a TSP-MD instance with seven customers.

the three systems for an instance with seven customers. When drones operate on their own, they perform all deliveries without the support of other vehicles; in this type of delivery systems, the main decisions concern locating drone's facilities, such as warehouses and recharging points (see, e.g., Coelho et al., 2017; Paradiso et al., 2020; Park et al., 2020). When drones cooperate with trucks, they work independently, in the sense that drones travel their own routes and serve some of the customers while trucks travel other routes to serve other customers; the cooperation between the two types of vehicles consists of making sure that all customers are served efficiently. An example of cooperation between drones and trucks without the two being synchronized is the Parallel Drone Scheduling Traveling Salesman Problem investigated by Murray and Chu (2015). In the last type of problem, drones work in tandem with trucks, and the two types of vehicles are synchronized. Trucks visit customers to deliver parcels and act as moving stations, where drones can take off and land to pick up and deliver parcels to serve their own customers. Trucks also act as recharging facilities, where drone batteries are recharged while drones are airborne so that when drones are launched they are equipped with fully charged batteries (see, e.g., Murray and Chu, 2015; Agatz et al., 2018; Poikonen et al., 2019; Ha et al., 2020). As indicated, among others, by Toksoz et al. (2011) and Ure et al. (2015), swapping batteries takes just a few seconds, so such time is negligible from an optimization perspective. Therefore, we will ignore the battery swapping time in the rest of the paper as commonly done in the optimization-related literature.

In this paper, we focus on the last type of delivery systems, where trucks and drones work in tandem, and we focus on solution methods to design optimal combined routes for the two types of vehicles. In particular, we investigate systems where there are a single truck and multiple drones. This topic is more and more popular in the scientific literature also due to the growing interest shown by different organizations, as shown by Otto et al. (2018). To the best of our knowledge, in the scientific literature currently available, no exact methods that can solve to proven optimality instances with more than ten customers have been proposed. Several authors (see, e.g., Schermer et al., 2019; Hà et al., 2020; Murray and Raj, 2020; Poikonen and Golden, 2020) propose Mixed Integer Linear Programming (MILP) models to solve small instances of different distribution problems involving a truck and multiple drones, but the main contributions of these papers lie in the proposal of heuristics and metaheuristics that are suitable to solve medium and large instances of the corresponding problems.

The problem we investigate can be described as follows. A set of customer locations is given. Each customer requires the delivery of a single package. Packages are initially located at a depot. A truck and one or more drones work in tandem to deliver all packages to the customers. The truck and the drones start and end their route at the depot. The truck can serve customers along the route. At the depot and at all the customer locations visited by the truck, the drones can take off to fly towards a customer to serve and can land after delivering a package to a customer. A single customer can be served by a drone in between each take-off/landing operation. While the drones are airborne, the truck moves and can fulfill other deliveries. When the drones are not airborne, they are housed on the truck. We call this problem the Traveling Salesman Problem with Multiple-Drones (TSP-MD). Fig. 2 shows a feasible solution of a TSP-MD

with seven customers, a single drone, and a truck. The depot is represented with the rectangle. Customers are represented with circles. White customers are served by the truck, and grey customers are served by the drone. Straight lines represent the movements of the truck while dashed lines represent the movements of the drone. The truck leaves the depot while the drone takes off to serve Customer 2. While the drone is airborne, the truck serves Customers 1 and 3. At Customer 3, the two vehicles rejoin. The drone is launched to serve Customer 5 while the truck serves Customers 4 and 6. At Customer 6, the drone lands on the truck and is launched to serve Customer 7. The two vehicles rejoin at the depot, where their route ends.

In this work, we aim at contributing to the literature on exact methods to solve distribution problems where trucks and drones work in tandem and multiple drones are used. The existing literature shows that the presence of multiple drones makes the exact solutions of these problems particularly challenging, so we study the TSP-MD as previously described, and we do not consider other side constraints that have been studied in the literature, such as limited drone flying range or the impossibility of serving some customers with the drones. As commonly done in the literature, the objective of the TSP-MD is to minimize the time required to complete all the deliveries and return to the depot. Moreover, we also assume that drones can take off and land at customer locations or the depot only; this assumption is made in most papers from the literature, with a few exceptions (e.g., Carlsson and Song, 2018) where drones and trucks can meet at any location.

The main contributions of this paper are: (*a*) we formulate the TSP-MD with a compact MILP model; (*b*) we propose numerous sets of valid inequalities for the compact MILP; (*c*) we propose an exact solution method that decomposes the TSP-MD into easier problems and solves each of these problems with a branch-and-cut method; (*d*) we present extensive computational experiments that show that the proposed decomposition method can solve instances with up to 24 customers to proven optimality.

The rest of the paper is organized as follows. Section 2 provides an overview of the relevant literature on the TSP-MD. Section 3 formally introduces the TSP-MD and presents the compact MILP formulation for the TSP-MD. In Section 4, we present several families of valid inequalities for the MILP presented in Section 3. In Section 5, we describe an exact decomposition approach, based on the MILP and the valid inequalities presented in Sections 3 and 4, to solve the TSP-MD to proven optimality. Section 6 illustrates how different side constraints addressed in the literature can be embedded in the compact formulation and the exact solution approach. Section 7 discusses the computational results achieved by the exact decomposition approach. Finally, Section 8 summarizes the main insights of our paper and outlines potential future research avenues.

2. Literature review

As shown by Macrina et al. (2020), the academic literature on decision-making problems for last-mile delivery with combined trucks and drones has steadily increased since the seminal paper of Murray and Chu (2015). For an in-depth analysis of the literature on the topic, we refer the reader to the recent exhaustive reviews of Otto et al. (2018), Chung et al. (2020), and Macrina et al. (2020). As the focus of this paper is more on the computational side of the topic, we limit the scope of our literature review accordingly. However, there are several interesting studies on the more theoretical aspects of the topic, such as Carlsson and Song (2017), Wang et al. (2017), and Bergmann et al. (2020). In the next two sections, we review only the literature on exact solution techniques for last-mile delivery problems where drones and trucks work in tandem and are synchronized (Section 2.1) and papers investigating synchronized routing problems with one or more trucks and multiple drones (Section 2.2).

2.1. Exact solution techniques

Whereas the literature on heuristic methods to solve drone-related last-mile delivery problems where drones and trucks work in tandem and are synchronized is rich, few contributions on exact solution techniques are available. The main contributions, listed in chronological order, are Bouman et al. (2018), Dell'Amico et al. (2019), Poikonen et al. (2019), Roberti and Ruthmair (2020), Bakir and Özbaygin (2020), El-Adle et al. (2021), and Vásquez et al. (2021). All of these contributions, except for Bakir and Özbaygin (2020), address problems featuring a single truck and a single drone but different operational constraints and solution approaches, ranging from dynamic programming recursions to branch-and-bound, branch-and-cut, and branch-and-price methods.

Bouman et al. (2018) investigate the Traveling Salesman Problem with Drone (TSP-D), where a single truck and a single drone are available to serve a set of customers. The TSP-D is the single-drone version of the TSP-MD. They propose dynamic programming recursions to solve the TSP-D to optimality and test them on instances with up to 20 nodes. The results indicate that instances with 16 nodes can be solved to proven optimality and that the addition of the restriction that the truck can visit a limited number of customers while the drone is airborne significantly reduces the solution time.

Dell'Amico et al. (2019) study the Flying Sidekick TSP introduced by Murray and Chu (2015). They propose a three-index and a two-index formulation and different sets of valid inequalities that are embedded in branch-and-cut algorithms. These algorithms are tested on the 72 10-customer instances introduced by Murray and Chu (2015). The results show that the two-index formulation is the better formulation as it can solve 59 out of the 72 benchmark instances with an average computing time of less than 20 min.

The TSP-D is investigated by Poikonen et al. (2019), who also consider additional constraints such as drone battery capacity and incompatibilities between customers and drones. They propose a heuristic branch-and-bound approach that provides a provably optimal solution of the problem under some circumstances. At each node of the search tree, a bound corresponding to a potential order to deliver a subset of packages is computed by solving a dynamic program. Even though the main contribution of the paper is represented by four heuristics based on branch-and-bound, the computational results show that instances with ten nodes can be solved to proven optimality.

Another contribution on the TSP-D is owed to Roberti and Ruthmair (2020), who consider a basic TSP-D and show how to model

and address different side constraints, such as drone battery capacity, incompatibilities between drones and customers, launch and rendezvous times, etc. The authors present an exact branch-and-price algorithm based on a set-partitioning formulation of the problem, where the pricing problem is solved by a dynamic programming recursion that builds upon the *ng*-route relaxation introduced by Baldacci et al. (2011). Computational results show that benchmark instances with up to 39 customers can be solved to proven optimality.

Bakir and Özbaygin (2020) study a hybrid delivery system with multiple trucks and multiple drones operating in tandem and address a problem called the Vehicle Routing Problem with Flexible Drones (VRPFD). The problem aims at minimizing the time required to serve all customers. In the VRPFD, drones are not dedicated to specific trucks but can be exchanged, and trucks are allowed to visit the same location multiple times. The problem is formulated as a MILP on a time-space network and solved with an algorithm using the dynamic discretization discovery paradigm. The computational experiments indicate that the proposed method can solve instances with ten customers to proven optimality. Primal and dual bounds on larger instances with 25 customers are also reported.

El-Adle et al. (2021) study the TSP-D of Bouman et al. (2018) with some additional features, such as the battery life of the drone. They illustrate a MILP model with binary variables and present a combination of valid inequalities, pre-processing techniques, and other bound tightening strategies. The computational results demonstrate that optimal solutions of instances with up to 24 nodes can be found and encouraging results can be achieved on larger instances with 28 and 32 nodes.

Vásquez et al. (2021) propose a MILP formulation for the TSP-D and an exact two-stage decomposition method. In the first stage, the sequence of customers served by the truck is defined. In the second stage, the operations of the drone are defined based on the solution of the first stage. The authors design a Benders-like decomposition algorithm strengthened by different families of valid inequalities. The computational results on benchmark instances demonstrate that the optimal solution of instances with up to 25 nodes can be found and that the drone speed strongly affects the computational performance of the proposed method - the faster the drone, the lower the computing time of the exact method.

When examining the literature on exact methods for drone-related last-mile delivery problems, we can see that a variety of different methodologies have been investigated: dynamic programming (Bouman et al., 2018), branch-and-cut algorithms (Dell'Amico et al., 2019, 2021), branch-and-bound algorithms (Poikonen et al., 2019), branch-and-price algorithms (Roberti and Ruthmair, 2020), MILPs on time-space networks (Bakir and Özbaygin, 2020), and Benders-like decomposition methods (Vásquez et al., 2021). The computational results of these papers suggest that the most promising approach to solve the TSP-D is the branch-and-price algorithm of Roberti and Ruthmair (2020), so an extension of this algorithm could lead to competitive results also on the TSP-MD. Indeed, the set-partitioning formulation used in Roberti and Ruthmair (2020) can be easily adjusted to formulate the TSP-MD, and the corresponding lower bound derived from its linear relaxation would most likely be tight. However, adjusting the dynamic programming recursion proposed by Roberti and Ruthmair (2020) to generate columns when multiple drones are present can be challenging and may require long computing times within a branch-and-price algorithm. Therefore, in this paper, we investigate a different approach for the TSP-MD based on MILP, valid inequalities, and a decomposition method.

2.2. Solution approaches for multi-drone problems

The literature on problems featuring multiple drones is diverse both in terms of problem settings and solution methods. The papers that address problems that are closer to the TSP-MD are owed to Schermer et al. (2019), Kitjacharoenchai et al. (2020), Murray and Raj (2020), and Poikonen and Golden (2020). We review these four contributions in the following.

Schermer et al. (2019) study the Vehicle Routing Problem with Drones and En Route Operations, where a fleet of vehicles, each equipped with a set of drones, is used to serve a set of customers with the goal of minimizing the makespan to serve all of them. Drones can take-off and land at particular points on the arcs that connect the customers. The problem is formulated as a MILP and some valid inequalities are introduced. However, this model can solve small instances only, so the authors propose a heuristic method based on Variable Neighborhood Search and Tabu Search to tackle medium and large-sized instances.

Kitjacharoenchai et al. (2020) introduce a problem that features multiple trucks, each one having limited capacity and carrying multiple drones that can be launched to serve one or multiple customers. The problem addresses two echelons of delivery: in the first level, trucks are routed from the depot to serve some customers, and, in the second level, drones perform routes starting and ending at the trucks. The authors propose a MILP formulation to solve small instances and two heuristics to solve large-sized instances - a route construction heuristic and a Large Neighborhood Search.

Murray and Raj (2020) develop a MILP formulation for the TSP-MD with additional side constraints. In particular, they consider that if multiple drones are launched and retrieved at the same customer location, they must be synchronized in order to avoid collisions. The MILP they propose can solve small instances only. Therefore, the authors propose a three–phase heuristic approach to solve instances with more than eight customers. First, the set of customers is partitioned into two subsets (the customers served by the truck, and those served by the drones), and a feasible TSP-tour for the truck is built. Then, the paths of the drones are added to the TSP-tour. In the third phase, they determine the correct coordination of drones at each truck node considering launch, recovery, and service times.

Poikonen and Golden (2020) consider the *k*-Multi-visit Drone Routing Problem, where a truck and multiple drones work in tandem. Each drone is allowed to carry multiple packages whose weight is taken into account to design the corresponding routes. They propose a constructive heuristic approach based on the solution of the TSP for the truck and the concept of operation, which is defined as a set of actions that start with the truck and all the drones being at the same (launching) location and end with all the drones rejoining the truck.

Even though the literature on solution approaches to solve problems featuring truck(s) and multiple drones working in tandem is richer and richer, to the best of our knowledge, no method is available to optimally solve instances with more than ten customers. The

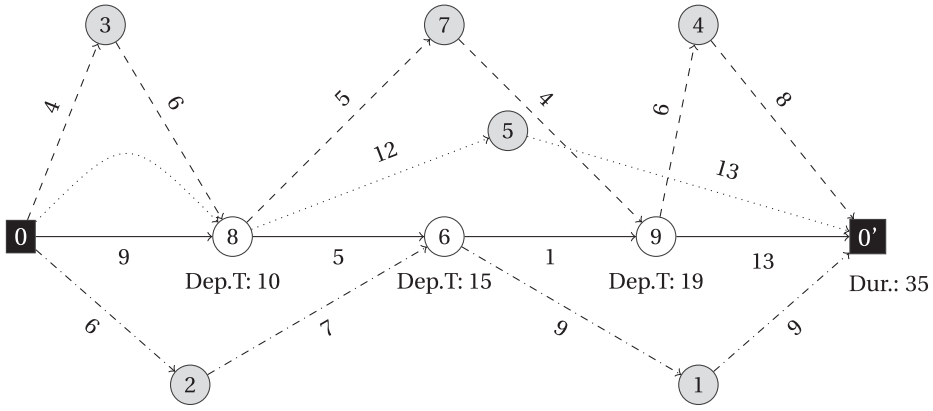

Fig. 3. A feasible solution of a TSP-MD instance with nine customers and three drones.

main contribution of the available papers lie in the proposition of heuristic and metaheuristic techniques. We aim at partly filling this gap by proposing a decomposition method based on the branch-and-cut paradigm to solve instances of the TSP-MD with up to 24 customers.

3. Problem description and compact formulation

The TSP-MD can be formally described as follows. A complete directed graph $G = (V, A)$ is given. The vertex set V is defined as $V = Nfv \cup \{0, 0'\}$, where N represents a set of n customers to serve and 0 and $0'$ are two copies of the depot, indicating the initial and final vertex of the vehicles' route, respectively – in the following, we also use the notation $N_0 = N \cup \{0\}$ and $N_{0'} = N \cup \{0'\}$. The arc set A is defined as $A = \{(0,j) \mid j \in N\} \cup \{(i,j) \mid i,j \in N : i \neq j\} \cup \{(i,0') \mid i \in N\}$. A truck is located at the depot. The truck is equipped with m identical drones. The truck and the drones are used to serve all customers. The time the truck takes to traverse arc $(i,j) \in A$ is indicated by t_{ij}^{T}, and the time the drones take to traverse arc $(i,j) \in A$ is indicated by t_{ij}^{D}. We assume the drones to be faster than the truck, so $t_{ij}^{\mathrm{D}} \leqslant t_{ij}^{\mathrm{T}}$ holds for each arc $(i,j) \in A$. As commonly done in the literature, we also assume that the triangle inequality holds for both truck travel times, t_{ij}^{T}, and drone travel times, t_{ij}^{D}. We also assume that both truck travel times and drone travel times are symmetric, i.e., for each arc $(i,j) \in A$ such that $(j,i) \in A, t_{ij}^{\mathrm{T}} = t_{ji}^{\mathrm{T}}$ and $t_{ij}^{\mathrm{D}} = t_{ji}^{\mathrm{D}}$. Each customer can be visited by the truck while some or all drones are airborne and the others are on-board. As commonly done in the literature, we neglect service times at the customers, but they can be easily included, if needed, by updating the drone and truck traveling times. The drones have limited carrying capacity, so they can serve a single customer before rejoining the truck. The drones can take off and rejoin the truck at customer locations and the depot only. While drones are airborne, the truck is traveling, so a drone cannot take off and land at the same location while the truck stands still. The goal of the TSP-MD is to find the route of minimum duration to serve all customers by the truck and the drones while considering the synchronization between the truck and the drones.

Illustrative Example. To clarify the definition of the TSP-MD and show how to compute the duration of a route while considering synchronization among vehicles, Fig. 3 shows a feasible solution of a TSP-MD instance with nine customers and three drones. In this figure, the position of each node is not meant to represent the real position of the corresponding customer, but nodes are positioned so as to highlight the path traversed by each vehicle. The truck travels the solid path $0-8-6-9-0'$. Numbers close to the solid arcs are the truck travel times, t_{ij}^{T}. The first drone traverses the dashed path $0-3-8-7-9-4-0'$ and serves Customers 3, 7, and 4. The second drone traverses the dotted path $0-8-5-0'$ and serves Customer 5 - when traveling from 0 to 8, the second drone is on the truck. The third drone traverses the dash-dotted path $0-2-6-1-0'$ and serves Customers 2 and 1. Numbers above or below the dashed, dotted, and dash-dotted lines represent the drone travel times, t_{ij}^{D}. To compute the duration of the solution, the departure time (*Dep.T*) at each node visited by the truck has to be computed. We assume that all vehicles leave the depot at time 0. The departure time from Customer 8 is 10 because the truck arrives at Customer 8 at time 9 but has to wait for the first drone to rejoin after serving Customer 3. The departure time from Customer 6 is 15 because the truck arrives there at time 15 and does not have to wait for the third drone, which arrives at Customer 6 at time 13. The departure time from Customer 9 is 19 because the truck arrives there at time 16 but has to wait on the first drone to arrive after serving Customer 7. Finally the duration of the solution is 35 units because the truck returns to $0'$ at time 32, the first drone at time 33, the second drone at time 35, and the last drone at time 33.

We now introduce a compact MILP to formulate the TSP-MD. This MILP is the basis of the exact approach that is presented in Section 5.

Let $x_{ij} \in \{0,1\}$ be a binary variable equal to 1 if the truck traverses arc $(i,j) \in A$ (no matter if there is any drone on-board or all of them are airborne), and let $y_i \in \{0,1\}$ be a binary variable equal to 1 if the truck visits node $i \in N$. Moreover, let us call *drone leg* a sequence of three nodes $\langle i,j,k \rangle$ $(i \in N_0, j \in N, k \in N_{0'})$, where i is the node where a drone takes off from the truck, j is a customer served by the drone on its own, and k is the node where the drone rejoins the truck; for example, in Fig. 3, the first drone performs three drone

legs: $\langle 0,3,8\rangle$, $\langle 8,7,9\rangle$, and $\langle 9,4,0'\rangle$. Let $\mathscr{L} = \{\langle i,j,k\rangle \mid i \in N_0, j \in N, k \in N_{0'} : i \neq j \neq k\} \setminus \{\langle 0,j,0'\rangle \mid j \in N\}$ be the set of all feasible drone legs, and let $z_{ijk} \in \{0,1\}$ be a binary variable equal to 1 if one of the drones performs drone leg $\langle i,j,k\rangle \in \mathscr{L}$ to serve customer j. Finally, let $a_i \in \mathbb{R}_+$ be the departure time of the truck from node $i \in V$ ($a_{0'}$ is actually the maximum arrival time at $0'$ over all vehicles), and let $w_i \in \mathbb{Z}_+$ be a commodity variable representing the number of drones airborne when the truck leaves node $i \in V$. Then, the TSP-MD can be formulated as follows:

$$t^* = \min a_{0'} \tag{1a}$$

$$\text{s.t.} \sum_{(j,i)\in A} x_{ji} = \sum_{(i,j)\in A} x_{ij} \, i \in N \tag{1b}$$

$$\sum_{(i,j)\in A} x_{ij} = y_i \, i \in N \tag{1c}$$

$$\sum_{(0,j)\in A} x_{0j} = \sum_{(i,0')\in A} x_{i0'} = 1 \tag{1d}$$

$$y_j + \sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} = 1 \, j \in N \tag{1e}$$

$$y_i \geqslant \sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} + \sum_{\langle k,j,i\rangle \in \mathscr{L}} z_{kji} + \sum_{\langle i,k,j\rangle \in \mathscr{L}} \frac{z_{ikj}}{m} + x_{ji} \, i,j \in N : i \neq j \tag{1f}$$

$$w_0 = \sum_{\langle 0,j,k\rangle \in \mathscr{L}} z_{0jk} \tag{1g}$$

$$w_{0'} = 0 \tag{1h}$$

$$w_i \leqslant m \sum_{(i,j)\in A} x_{ij} \, i \in N_0 \tag{1i}$$

$$w_0 + \sum_{\langle j,r,s\rangle \in \mathscr{L}} z_{jrs} - \sum_{\langle 0,r,j\rangle \in \mathscr{L}} z_{0rj} \leqslant w_j + m\left(1 - x_{0j}\right) j \in N \tag{1j}$$

$$w_i + \sum_{\langle j,r,s\rangle \in \mathscr{L}} z_{jrs} - \sum_{\langle r,s,j\rangle \in \mathscr{L}} z_{rsj} \leqslant w_j + m\left(1 - x_{ij}\right) i \in N \quad j \in N_{0'} \tag{1k}$$

$$a_0 = 0 \tag{1l}$$

$$\begin{aligned} &a_i + \left(M + t^{\mathrm{T}}_{ij}\right) x_{ij} + \sum_{\langle k,i,j\rangle \in \mathscr{L}} \left(M + \max\left\{t^{\mathrm{T}}_{kj} - t^{\mathrm{D}}_{ki}, t^{\mathrm{D}}_{ij}\right\}\right) z_{kij} \\ &+ \sum_{\langle i,j,k\rangle \in \mathscr{L}} \left(M + t^{\mathrm{D}}_{ij}\right) z_{ijk} + \sum_{\langle i,k,j\rangle \in \mathscr{L}} \max\left\{t^{\mathrm{D}}_{ik} + t^{\mathrm{D}}_{kj} - t^{\mathrm{T}}_{ij}, 0\right\} \frac{z_{ikj}}{m} \leqslant a_j + M \quad (i,j) \in A \end{aligned} \tag{1m}$$

$$x_{ij} \in \{0,1\}(i,j) \in A \tag{1n}$$

$$y_i \in \{0,1\} i \in N \tag{1o}$$

$$a_i \in \mathbb{R}_+ i \in V \tag{1p}$$

$$z_{ijk} \in \{0,1\}\langle i,j,k\rangle \in \mathscr{L} \tag{1q}$$

$$w_i \in \mathbb{Z}_+ i \in V \tag{1r}$$

The objective function (1a) aims at minimizing the total duration of the route that serves all customers. Constraints (1b) are flow conservation constraints of the truck. Constraints (1c) link the arc variables with the y-variables. Constraints (1d) ensure that the truck leaves from and returns to the depot. Constraints (1e) guarantee that each customer is served exactly once.

Constraints (1f) ensure that drone legs start and end only at locations visited by the truck. Notice that the last two terms of the right-hand side are not necessary for the correctness of the constraints. Indeed, imposing that $y_i \geqslant \sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} + \sum_{\langle k,j,i\rangle \in \mathscr{L}} z_{kji}$, for each i,

$j \in N : i \neq j$, is sufficient to guarantee that a drone leg is selected if and only if its corresponding launching and landing locations are visited by the truck. If the sum of the last two terms of (1f) is strictly positive, y_i must be one and the first two terms of the right-hand side must be zero. Thus, constraints (1f) is a valid lifting of $y_i \geqslant \sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} + \sum_{\langle k,j,i\rangle \in \mathscr{L}} z_{kji}$.

Constraint (1g) sets w_0 equal to the number of airborne drones upon leaving 0. Constraint (1h) sets $w_{0'}$ equal to zero to ensure that all drones rejoins the truck upon returning to the depot. Constraints (1i) guarantee that the number of airborne drones upon leaving each location visited by the truck does not exceed the number of available drones. Constraints (1j)–(1k) allow to correctly set the values of the *w*-variables. Constraint (1l) sets the departure time upon leaving the depot. Constraints (1m) set the departure time from each node and act as subtour elimination constraints. Constraints (1n)–(1r) define the domain of the variables.

Single-Drone Case. Being developed for the TSP-MD, model (1) is also valid for the special case where there is a single drone (i.e., $m = 1$). However, in this special case, the $n \times (n-1)$ constraints (1f) can be replaced by the following $2n$ constraints

$$y_i \geqslant \sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} \qquad i \in N \tag{2a}$$

$$y_i \geqslant \sum_{\langle k,j,i\rangle \in \mathscr{L}} z_{kji} \qquad i \in N \tag{2b}$$

Constraints (2a) (constraints (2b), resp.) guarantee that the drone can perform a drone leg starting from (ending at, resp.) a given customer $i \in N$ only if customer i is visited by the truck (i.e., $y_i = 1$).

Additional Notation. The following additional notation is used in the rest of the paper. Let $A(S) \subset A, S \subseteq N$, denote the set of arcs whose endpoints belong to the set S, i.e., $A(S) = \{(i.j) \in A \mid i.j \in S\}$; in other words, $A(S)$ is the set of arcs induced by the set of customers S. Moreover, let $A(S_1, S_2)$ be the set of arcs starting from any node of the set $S_1 \subset V$ and ending at any node of the set $S_2 \subset V$, i.e., $A(S_1, S_2) = \{(i,j) \in A \mid i \in S_1, j \in S_2\}$. We also denote by $x(\widehat{A})$ the sum of the x-variables of the arcs belonging to the set $\widehat{A} \subseteq A$, i.e., $x(\widehat{A}) = \sum_{(i,j)\in \widehat{A}} x_{ij}$. Finally, let $y(S)$ be the sum of the y-variables of the customers of the set $S \subseteq N$, i.e., $y(S) = \sum_{i \in S} y_i$.

A summary of the mathematical notation introduced so far is provided in Table 8 of the appendix.

4. Valid inequalities

Our computational experience about using formulation (1) to solve TSP-MD shows that its linear relaxation is as weak as it can be. Indeed, the lower bound provided by the linear relaxation of (1) is 0 in all the test instances we have used. Indeed, the bigM-coefficients in constraints (1m) make it feasible to set all variables a equal to 0 for highly fractional solutions of the x and z variables. Therefore, in this section, we present different sets of valid inequalities that strengthen the linear relaxation of formulation (1).

4.1. Minimum number of customers to serve with the truck

We can observe that, while the truck traverses an arc, m customers can be served with the drones if all of them are airborne. Moreover, all but the last arc traversed by the truck end at a customer served by the truck. Therefore, we can compute the minimum number of customers that the truck has to serve, n^{T}, as $n^{\mathrm{T}} = \lceil \frac{n-m}{m+1} \rceil$. The following inequality forces the truck to serve at least n^{T} customers and is valid for (1):

$$y(N) \geqslant n^{\mathrm{T}} \tag{3}$$

4.2. Symmetry breaking

Because of the assumption that both truck and drone travel times are symmetric (see Section 3), we can observe that every feasible solution of (1) has a corresponding *symmetric* solution obtained by traversing its arcs in the opposite direction. The two solutions have the same duration. The following *Symmetry-Breaking Constraints* are valid for (1):

- If $n^{\mathrm{T}} = 1$

$$x_{0i} \leqslant \sum_{j=i}^{n} x_{j0'} \qquad i \in N \tag{4a}$$

$$x_{j0'} \leqslant \sum_{i=1}^{j} x_{0i} \qquad j \in N \tag{4b}$$

  Constraints (4) halve the set of feasible solutions of (1) by forcing the index of the first customer served by the truck not to be greater than the index of the last customer visited by the truck;
- If $n^{\mathrm{T}} \geqslant 2$

$$x_{0i} \leqslant \sum_{j=i+1}^{n} x_{j0'} \qquad i \in N \tag{5a}$$

$$x_{j0'} \leqslant \sum_{i=1}^{j-1} x_{0i} \qquad j \in N \tag{5b}$$

Constraints (5) force the index of the first customer served by the truck to be strictly lower than the index of the last customer served by the truck.

Symmetry-breaking constraints (4)–(5) have been introduced in Vásquez et al. (2021).

4.3. Lower bound to $a_{0'}$

Schermer et al. (2019) observe that the total duration of any feasible route, $a_{0'}$, is bounded from below by the total travel time of the truck and the average travel time of the drones. Therefore, the following inequalities are valid for (1):

$$a_{0'} \geqslant \sum_{(i,j)\in A} t^{\mathrm{T}}_{ij} x_{ij} \tag{6a}$$

$$m \cdot a_{0'} \geqslant \sum_{\langle i,j,k\rangle \in \mathscr{L}} \max\left\{ t^{\mathrm{T}}_{ik},\ t^{\mathrm{D}}_{ij} + t^{\mathrm{D}}_{jk} \right\} z_{ijk} \tag{6b}$$

Valid inequality (6a) can be strengthened by considering (part of) the time the truck has to wait for the drones. In particular, we lift valid inequality (6a) by considering the average delay, if any, caused by the drones performing drone legs $\langle i^*, k, j^*\rangle$, for a given arc $(i^*, j^*) \in A$, when the truck traverses arc $(i^*, j^*) \in A$. Fig. 3 shows two examples of this case. The first case is when the truck traverses arc $(0,8)$ (i.e., $x_{08} = 1$) and the first drone performs drone leg $\langle 0,3,8\rangle$ (i.e., $z_{038} = 1$); as $t^{\mathrm{D}}_{03} = 4, t^{\mathrm{D}}_{38} = 6$, and $t^{\mathrm{T}}_{08} = 9$, the unit of waiting time given by $t^{\mathrm{D}}_{03} + t^{\mathrm{D}}_{38} - t^{\mathrm{T}}_{08} = 4 + 6 - 9 = 1$ can safely be added to the right-hand side of (6a). Similarly, another unit of waiting time can be added to the right-hand side of (6a) by considering the last arc traversed by the truck, $(9, 0')$ and the last drone leg performed by the first drone, $\langle 9,4,0'\rangle$.

To strengthen inequality (6a) by considering this waiting time, let us introduce a non-negative continuous variable $d_i \in \mathbb{R}_+$ representing a lower bound to the minimum time the truck has to wait for the drone(s) after traversing an arc leaving from node $i \in N_0$. Moreover, let $\mathscr{L}_{ij} \subset \mathscr{L}$ be the set of drone legs that start from node $i \in N_0$, end at node $j \in N_{0'}$, and make the truck wait for the drones if the truck traverses arc $(i,j) \in A$, i.e., $\mathscr{L}_{ij} = \{ \langle i,k,j\rangle \in \mathscr{L} \,\big|\, t^{\mathrm{D}}_{ik} + t^{\mathrm{D}}_{kj} > t^{\mathrm{T}}_{ij} \}$. The following $O(|A|+1)$ inequalities can be added to (1) to lift (6a):

$$a_{0'} \geqslant \sum_{(i,j)\in A} t^{\mathrm{T}}_{ij} x_{ij} + \sum_{i\in N_0} d_i \tag{7a}$$

$$M_{ij} x_{ij} + \sum_{\langle i,k,j\rangle \in \mathscr{L}_{ij}} \frac{t^{\mathrm{D}}_{ik} + t^{\mathrm{D}}_{kj} - t^{\mathrm{T}}_{ij}}{\min\{m, |\mathscr{L}_{ij}|\}} z_{ikj} \leqslant d_i + M_{ij} y_i \qquad \left(i,j\right) \in A : |\mathscr{L}_{ij}| \geqslant 1 \tag{7b}$$

where M_{ij} is a large enough value that can be computed, for example, as $M_{ij} = \max_{k\in N}\left\{ t^{\mathrm{D}}_{ik} + t^{\mathrm{D}}_{kj} - t^{\mathrm{T}}_{ij} \right\}$. Constraint (7a) lifts inequality (6a) by adding the set of d-variables to its right-hand side. Constraints (7b) set $d_i, i \in N_0$, equal to the average delay caused by the drones performing drone legs of type $\langle i,k,j\rangle$ if $x_{ij} = 1$ (and, consequently, $y_i = 1$). Notice that constraints (7b) are meaningful, for each arc $(i,j) \in A$, only if there exists at least one drone leg $\langle i,k,j\rangle$ that makes the truck waiting on the drones after traversing arc $(i,j) \in A$ (i.e., only if $|\mathscr{L}_{ij}| \geqslant 1$).

By observing that, for each arc $(i,j) \in A$, the following inequality holds

$$x_{ij} + x_{ji} + \sum_{\langle i,j,s\rangle \in \mathscr{L}} z_{ijs} + \sum_{\langle s,j,i\rangle \in \mathscr{L}} z_{sji} \leqslant 1$$

inequality (7b) can be lifted as follows

$$M_{ij}\left(x_{ij} + x_{ji} + \sum_{\langle i,j,s\rangle \in \mathscr{L}} z_{ijs} + \sum_{\langle s,j,i\rangle \in \mathscr{L}} z_{sji} \right) + \sum_{\langle i,k,j\rangle \in \mathscr{L}_{ij}} \frac{t^{D}_{ik} + t^{D}_{kj} - t^{T}_{ij}}{\min\{m, |\mathscr{L}_{ij}|\}} z_{ikj} \leqslant d_i + M_{ij} y_i \tag{8}$$

$$\left(i,j\right) \in A : |\mathscr{L}_{ij}| \geqslant 1$$

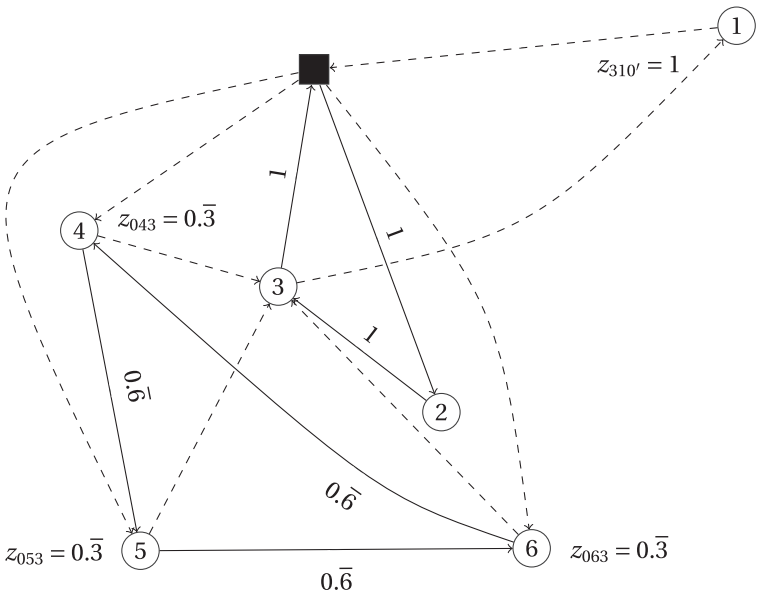

Fig. 4. An example of a fractional solution featuring a violated Too-Short Path cut for an instance with six customers ($n = 6$) and a single drone ($m = 1$). Solid lines represent strictly positive x_{ij} variables. Dashed lines represent strictly positive z_{ijk} variables. The violated Too-Short Path cut correspond to the set $S = \{2, 3\}$.

4.4. Maximum drone legs leaving from a node

We can establish the following relationship between the number of drone legs $\langle i, j, k\rangle \in \mathscr{L}$ departing from node $i \in N_0$ and the variables w_i and y_i

$$\sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} \leqslant w_i \qquad i \in N_0 \tag{9a}$$

$$\sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} \leqslant m y_i \qquad i \in N \tag{9b}$$

Constraints (9a) stipulate that the number of drones airborne w_i when the truck leaves node $i \in N_0$ is greater than or equal to the number of selected drone legs departing from node i. Constraints (9b) are on-off constraints stating that one or more drone legs departing from node $i \in N$ can be selected if and only if i is served by the truck (i.e., $y_i = 1$).

4.5. Relationship between $\boldsymbol{x}(A)$ and $\boldsymbol{y}(N)$

We can easily observe that the number of arcs traversed by the truck must be equal to the number of customers served by the truck plus 1. Therefore, the following inequality holds for (1)

$$\boldsymbol{x}(A) = \boldsymbol{y}(N) + 1 \tag{10}$$

4.6. Drone legs incompatible with first and last truck arcs

The next two sets of valid inequalities are based on the relationship between the first (or the last) two arcs traversed by the truck and the drone legs that are incompatible with them. The two following sets of inequalities are valid for (1)

$$x_{0i} + x_{ij} + \frac{1}{m} \sum_{\substack{\langle s,r,j\rangle \in \mathscr{L}: \\ s \neq i,0}} z_{srj} + \frac{z_{0ij}}{m} \leqslant y_i + y_j \qquad i, j \in N : i \neq j \tag{11a}$$

$$x_{ij} + x_{j0'} + \frac{1}{m} \sum_{\substack{\langle i,r,s\rangle \in \mathscr{L}: \\ s \neq j,0}} z_{irs} + \frac{z_{ij0'}}{m} \leqslant y_j + y_i \qquad i, j \in N : i \neq j \tag{11b}$$

Constraints (11a) relate the first two arcs traversed by the truck and the incompatible drone legs whereas constraints (11b) are about the last two arcs traversed by the truck.

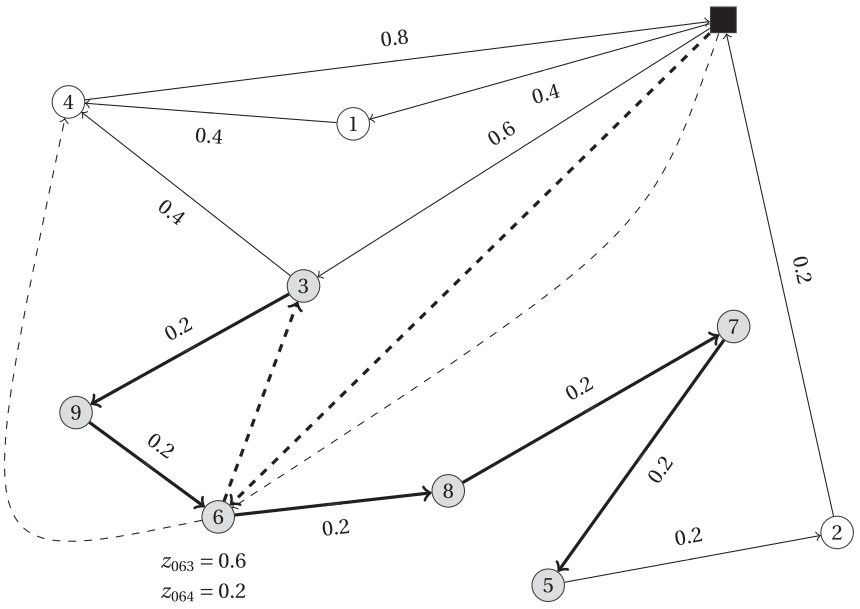

Fig. 5. An example of a fractional solution featuring a violated LGSEC (14a) for an instance with nine customers ($n = 9$) and three drones ($m = 3$). The violated LGSEC corresponds to the thick straight lines, the thick dashed lines, the set $S = \{3, 5, 6, 7, 8, 9\}$, and $r = 6$.

Proposition 1. Inequalities (11) are valid for formulation (1).

Proof. See B. □

4.7. Too-short paths cuts

The idea behind *Too-Short Paths Cuts* (TSPC) is that, for any subset of customers $S \subset N$ such that $|S| < n^{\mathrm{T}}$, at most $|S|$ x_{ij} variables such that $i,j \in S \cup \{0,0'\}, i \neq j$, can be selected; indeed, as the truck must serve at least n^{T} customers, if more than $|S|$ of these x_{ij} variables are selected, the truck makes a close path serving less than n^{T} customers.

To formally describe TSPC, let us refer to the fractional solution depicted in Fig. 4. The TSP-MD instance features six customers ($n = 6$) and a single drone ($m = 1$). Solid lines show strictly positive x_{ij} variables, namely, $x_{02} = x_{23} = x_{30'} = 1$ and $x_{45} = x_{56} = x_{64} = \frac{2}{3}$. The values of the x_{ij} variables also determine the values of the y-variables, which are $y_2 = y_3 = 1$ and $y_4 = y_5 = y_6 = \frac{2}{3}$. Dashed lines represent strictly positive drone legs, namely, $z_{043} = z_{053} = z_{063} = \frac{1}{3}, z_{310'} = 1$. Because $n = 6$ and $m = 1$, the truck must visit at least three customer, indeed, $n^{\mathrm{T}} = \lceil \frac{n-m}{m+1} \rceil = 3$. By looking at variables x_{02}, x_{23}, and $x_{30'}$ (all of which are equal to 1), we can see that they form a close path $0-2-3-0'$ that serves two customers only. Such a path can be cut off by adding inequality $x_{02} + x_{23} + x_{30'} \leqslant y_2 + y_3$, which can be lifted as $x_{02} + x_{03} + x_{23} + x_{32} + x_{20'} + x_{30'} \leqslant y_2 + y_3$.

In general terms, TSPC can be described as follows:

$$\boldsymbol{x}\left(S \cup \{0,0'\}\right) \leqslant \boldsymbol{y}(S) \qquad S \subset N : |S| < n^{\mathrm{T}} \tag{12}$$

4.8. Lifted generalized subtour-elimination constraints

Generalized Subtour-Elimination Constraints (GSEC) have been extensively used as valid inequalities for variants of the TSP where not all customers are to be visited, e.g., in the Undirected Selective TSP (Gendreau et al., 1998) and the Orienteering Problem (Fischetti et al., 1998). GSEC can be stated, among others, in the following form

$$\boldsymbol{x}(A(S)) \leqslant \boldsymbol{y}(S) - y_r \qquad S \subset N : 2 \leqslant |S| \leqslant n-2, \quad r \in S \tag{13}$$

As in the TSP-MD, the truck is not supposed to serve all customers, constraints (13) are valid also for formulation (1). However, constraints (13) can be lifted as follows.

Let $\overline{n}^{\mathrm{D}} = n - n^{\mathrm{T}}$ be the maximum number of customers that can be served by the drones alone. Constraints (13) can be replaced by the following *Lifted GSEC* (LGSEC)

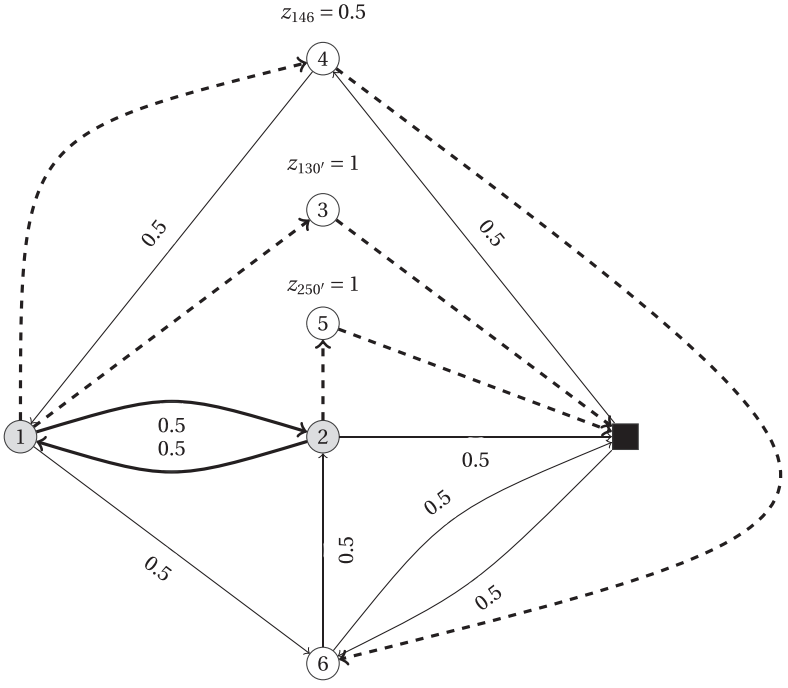

Fig. 6. An example of a fractional solution featuring a violated MODL cut for an instance with six customers ($n = 6$) and two drones ($m = 2$). The violated MODL cut corresponds to the thick solid lines, the thick dashed lines, and the set $S = \{1, 2\}$.

$$\boldsymbol{x}\left(A\left(S\right)\right) + \sum_{\substack{(i,k)\in\mathcal{L}:\\ \{i,k\}\cap S\neq\emptyset}} z_{ik} \leqslant \boldsymbol{y}\left(S\right) - y_r \quad S\subset N:\ 2\leqslant|S|\leqslant\overline{n}^{\mathrm{D}},\ r\in S \tag{14a}$$

$$\boldsymbol{x}\left(A\left(S\right)\right) \leqslant \boldsymbol{y}\left(S\right) - 1 \quad S\subset N:\ \overline{n}^{\mathrm{D}} < |S| < n-2 \tag{14b}$$

Proposition 2. LGSEC (14) are valid inequalities for formulation (1).

Proof. See C. □

Fig. 5 shows an example of a fractional solution that violates an LGSEC constraint (14a). The instance features nine customers ($n = 9$) and three drones ($m = 3$). The fractional solution consists of the following strictly positive x-variables (represented with solid lines): $x_{01} = x_{14} = x_{34} = 0.4, x_{20'} = x_{39} = x_{52} = x_{68} = x_{75} = x_{87} = x_{96} = 0.2, x_{03} = 0.6$, and $x_{40'} = 0.8$. Hence, the following y-variables have strictly positive values: $y_5 = y_6 = y_7 = y_8 = y_9 = 0.2$ and $y_3 = 0.6$. Finally, among others, two z-variables have strictly positive values, namely, $z_{063} = 0.6$ and $z_{064} = 0.2$ - these are represented with dashed lines. This fractional solution violates constraint (14a) for the set $S = \{3, 5, 6, 7, 8, 9\}$ and $r = 6$. Indeed, by looking at the thick solid and dashed lines, we have

$$\underbrace{x_{39} + x_{96} + x_{68} + x_{87} + x_{75}}_{1} + \underbrace{z_{063}}_{0.6} \leqslant \underbrace{y_3 + y_5 + y_6 + y_7 + y_8 + y_9}_{1.6} - \underbrace{y_6}_{0.2}$$

which is violated by 0.2. Notice that the corresponding constraint (13) obtained by removing z_{063} from the left-hand side is not violated.

4.9. Maximum outgoing drone legs cuts

Maximum Outgoing Drone Legs Cuts (MODLC) limit the number of drone legs that take off within a given subset of customers $S \subset N$ and land at nodes not contained in S. We explain the idea behind MODLC with the numerical example depicted in Fig. 6, which displays a fractional solution of an instance with six customers ($n = 6$) and two drones ($m = 2$). There are nine strictly positive x-variables (represented with solid lines) in solution, namely, $x_{04}, x_{06}, x_{12}, x_{16}, x_{21}, x_{20'}, x_{41}, x_{62}$, and $x_{60'}$; all these variables take value 0.5. Consequently, the values of the corresponding y-variables are $y_1 = y_2 = y_6 = 1, y_4 = 0.5$, and $y_3 = y_5 = 0$. Moreover, among others, there are three z-variables in solution (represented with dashed lines), namely, $z_{130'} = z_{250'} = 1$, and $z_{146} = 0.5$. It is easy to check that

Too-Short Path cuts (12) and LGSEC (14) are not violated in this fractional solution. Let us now focus on the subset of grey customers $S = \{1, 2\}$ for which $\boldsymbol{x}(S) = x_{12} + x_{21} = 1$ and $\boldsymbol{y}(S) = y_1 + y_2 = 2$. In any feasible solution where $x_{12} = 1$ (or $x_{21} = 1$), y_1 and y_2 must be equal to 1. However, we can also observe that, if $x_{12} = 1$ (or $x_{21} = 1$), there can be at most two drone legs that take off at customers 1 or 2 and end at nodes served by the truck after serving 1 or 2. Therefore, the following inequality holds for (1) and is violated by the current fractional solution:

$$\underbrace{x_{12} + x_{21}}_{1} + \underbrace{\frac{1}{m}\left(z_{130'} + z_{250'} + z_{146}\right)}_{\frac{1}{2}\left(1+1+0.5\right)=1.25} \leqslant \underbrace{y_1 + y_2}_{2}$$

This concept can be generalized for any subset of customers $S \subset N$ with the following MODLC:

$$\boldsymbol{x}\left(A\left(S\right)\right) + \frac{1}{m} \sum_{\substack{\langle k,r,s\rangle \in \mathscr{L}: \\ k \in S,\ s \notin S}} z_{krs} \leqslant \boldsymbol{y}\left(S\right) \qquad S \subset N \tag{15}$$

which state that the sum over all the arcs within the set S (i.e., $\boldsymbol{x}(A(S))$) plus the number of drone legs leaving from S and ending outside S divided by m (i.e., $\frac{1}{m}\sum_{\substack{\langle k,r,s\rangle \in \mathscr{L}: \\ k \in S,\ s \notin S}} z_{krs}$) cannot exceed the number of customers of the set S served by the truck (i.e., $\boldsymbol{y}(S)$). A similar reasoning can be applied to limit the number of drone legs that take off outside S and end within S

$$\boldsymbol{x}\left(A\left(S\right)\right) + \frac{1}{m} \sum_{\substack{\langle s,r,k\rangle \in \mathscr{L}: \\ k \in S,\ s \notin S}} z_{srk} \leqslant \boldsymbol{y}\left(S\right) \qquad S \subset N \tag{16}$$

Valid inequalities (15)–(16) can be further adjusted to account for the x-variables connecting the set S and the depot as follows:

$$\boldsymbol{x}\left(A\left(S\right)\right) + \boldsymbol{x}\left(S, \left\{0'\right\}\right) + \frac{1}{m} \sum_{\substack{\langle k,r,s\rangle \in \mathscr{L}: \\ k \in S,\ s \in N \setminus S}} z_{krs} \leqslant \boldsymbol{y}\left(S\right) \qquad S \subset N \tag{17a}$$

$$\boldsymbol{x}\left(A\left(S\right)\right) + \boldsymbol{x}\left(\left\{0\right\}, S\right) + \frac{1}{m} \sum_{\substack{\langle s,r,k\rangle \in \mathscr{L}: \\ k \in S,\ s \in N \setminus S}} z_{srk} \leqslant \boldsymbol{y}\left(S\right) \qquad S \subset N \tag{17b}$$

MODLC (15)–(17b) are similar to the *Crossing Sorties Elimination Constraints* (CSEC) and the *Backward Sorties Elimination Constraints* (BSEC) proposed by Dell'Amico et al. (2019), who also propose a lifting of such constraints called *Tournament Crossing Constraints* and *Tournament Backward Constraints*, respectively. The difference between these four families of cuts proposed by Dell'Amico et al. (2019) and MODLC is that MODLC are defined on subsets of customers whereas CSEC, BSEC, and their lifted counterparts are defined on paths of arcs traversed by the truck.

4.10. Separation procedures

In this section, we summarize how we separate the valid inequalities previously described for formulation (1). Given a fractional solution $(\widehat{x}, \widehat{y}, \widehat{z}, \widehat{a}, \widehat{w})$ of the linear relaxation of (1), let $N_\epsilon(\widehat{y})$ be the set of customers whose y-variables have values greater than ϵ in the solution $(\widehat{x}, \widehat{y}, \widehat{z}, \widehat{a}, \widehat{w})$, i.e., $N_\epsilon(\widehat{y}) = \{i \in N \,|\, \widehat{y}_i > \epsilon\}$.

Valid inequalities (3), (4), (5), (6), (7a), (8), (9), (10), and (11) can be easily separated by inspection. Our computational experience shows that it is computationally convenient to add all these valid inequalities to (1) as constraints already from the beginning. The other valid inequalities (i.e., TSPC (12), LGSEC (14), and MODLC (15)–(17b)) are exponentially many in the number of customers, so they are separated in a cutting plane fashion at each node of the search tree and added as global cuts as follows.

TSPC (12) are separated by inspection for subsets of customers $S \subset N, |S| < n^{\mathrm{T}}$, that satisfy two conditions: (a) the y-variables of all customers of the set S in the incumbent fractional solution are greater than 0.05 (i.e., $S \subseteq N_{0.05}(\widehat{y})$); and (b) at least one of the variables x_{0i} or $x_{i0'}$, with $i \in S$, is at least 0.05 in the incumbent fractional solution, meaning that the truck must either enter the set S straight

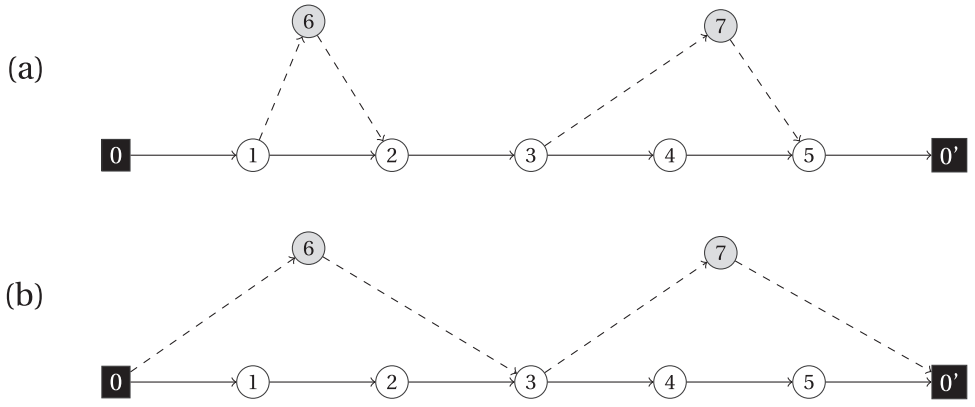

Fig. 7. Example of a feasible TSP-MD solution where the drones traverse some arcs on the truck (panel (a)) and the corresponding solution where the drones are always airborne when the truck is moving (panel (b)).

from the depot or return to the depot directly from the set S. The number of sets of customers that satisfy these two conditions is usually low, so the separation of (12) by inspection is computationally tractable even though better separation procedures could be developed. The most violated too-short path (12) cut is added (if any).

Whenever no violated TSPC exists, LGSEC (14) are separated by inspection over the subsets of customers S such that $S \subseteq N_{0.05}(\hat{y})$. The number of these sets can be exponentially high, but our computational experience shows that such a separation is computationally tractable on the instances we have used and the time spent on the separation of LGSEC is limited when solving formulation (1) with a branch-and-cut algorithm. However, more clever separation procedures (even potentially running in polynomial time) can be devised based on the rich literature on algorithms to separate GSEC (13). Again, the most violated LGSEC (14) cut is added (if any).

As to the separation of MODLC (15)–(17b), we have noticed that they are rarely violated for sets of customers of cardinality greater than three, so we separate them, by inspection, for subsets of customers S that satisfy two conditions: (a) $1 \leqslant |S| \leqslant 3$, and (b) $S \subseteq N_{0.05}(\hat{y})$. MODLC (15)–(17b) are separated only if no violated LGSEC (14) was found, and the most violated MODLC is added (if any).

5. Solving the TSP-MD by decomposition

Finding an optimal solution of the TSP-MD by solving formulation (1) with the different valid inequalities presented in Section 4 is challenging even for instances with 15–20 customers. Therefore, in this section, we propose a decomposition method that finds an optimal TSP-MD solution by solving $m+1$ MILPs similar to (1), where each MILP is easier to solve than (1), and the overall required computational effort is usually lower than solving (1) as it is.

The key observation of this decomposition approach is that the set of feasible solutions of the TSP-MD can be partitioned into $m+1$ sets, where each index set γ ($\gamma = 0,1,\ldots,m$) is characterized by the fact that exactly γ out of m drones serve at least two customers each. For example, if $m = 2$, the feasible TSP-MD solutions can be partitioned into three sets: in the first set of solutions, both drones serve at least two customers each (i.e., $\gamma = 2$); in the second set of solutions, one drone serves at least two customers, and the other drone serves no more than one customer (i.e., $\gamma = 1$); in the third and last set of solutions, both drones serve no more than one customer each (i.e., $\gamma = 0$).

Let $t^*(\gamma)$ be an optimal solution cost of the TSP-MD where exactly γ ($\gamma = 0,1,\ldots,m$) drones serve at least two customers each and the other $m-\gamma$ drones serve at most one customer each. Then, the optimal TSP-MD cost can be expressed as

$$t^* = \min\{t^*(\gamma) \,|\, \gamma = 0, 1, \ldots, m\}$$

In our computational experience, most of the times t^* coincides with $t^*(m)$. Therefore, in the following (see 5.3), we describe an exact method that first computes $t^*(m)$ by solving a MILP similar to (1) (see Section 5.1) and then computes a lower bound to each value $t^*(\gamma)$, $\gamma = 0,1,\ldots,m-1$, by solving another MILP derived from (1) (see 5.2).

5.1. Computing $t^*(m)$

Vásquez et al. (2021) present some properties of the optimal solution(s) of a problem similar to the TSP-MD but with a single drone. Inspired by these properties, we present some properties of the optimal solution(s) of the TSP-MD under the assumption that each drone is supposed to serve at least two customers. We use these properties to derive a MILP to compute $t^*(m)$ by adjusting (1).

Proposition 3. There exists an optimal solution of the TSP-MD where all the drones are airborne when the truck is moving if the following three conditions hold: (i) truck travel times, t_{ij}^{T}, and drone travel times, t_{ij}^{D}, satisfy the triangle inequality; (ii) the truck is not faster than the drones, i.e., $t_{ij}^{\mathrm{T}} \leqslant t_{ij}^{\mathrm{D}}$ for each $(i,j) \in A$; and (iii) each drone serves at least two customers.

Proof. Whenever a drone performs a drone leg that is preceded or followed by an arc traversed by the drone on-board of the truck, such a drone leg can be replaced by another drone leg such that the drone takes off and lands at the same node (in two subsequent drone legs) without increasing the duration of the whole solution. This is the case no matter how many drones are available. Notice that this is true if and only if each drone serves at least two customers, as imposed by Condition (*iii*). □

We illustrate the idea behind Proposition 3 by showing the example displayed in Fig. 7 with seven customers and a drone. Panel (*a*) shows a feasible solution involving two drone legs $\langle 1,6,2\rangle$ and $\langle 3,7,5\rangle$. The duration of solution $(a), t_a$, is $t_a = t^{\mathrm{T}}_{01} + \max\{t^{\mathrm{T}}_{12}, t^{\mathrm{D}}_{16} + t^{\mathrm{D}}_{62}\} + t^{\mathrm{T}}_{23} + \max\{t^{\mathrm{T}}_{34} + t^{\mathrm{T}}_{45}, t^{\mathrm{D}}_{37} + t^{\mathrm{D}}_{75}\} + t^{\mathrm{T}}_{50'}$. Let us assume the two drone legs $\langle 1,6,2\rangle$ and $\langle 3,7,5\rangle$ are replaced by drone legs $\langle 0,6,3\rangle$ and $\langle 3,7,0'\rangle$, respectively (see panel (*b*)). Because of Conditions (*i*) and (*ii*), we have $t^{\mathrm{D}}_{06} \leqslant t^{\mathrm{D}}_{01} + t^{\mathrm{D}}_{16} \leqslant t^{\mathrm{T}}_{01} + t^{\mathrm{D}}_{16}, t^{\mathrm{D}}_{63} \leqslant t^{\mathrm{D}}_{62} + t^{\mathrm{D}}_{23} \leqslant t^{\mathrm{D}}_{62} + t^{\mathrm{T}}_{23}$, and $t^{\mathrm{D}}_{70'} \leqslant t^{\mathrm{D}}_{75} + t^{\mathrm{D}}_{50'} \leqslant t^{\mathrm{D}}_{75} + t^{\mathrm{T}}_{50'}$, so in the solution of panel (*b*) the arrival times of the drone at Customer 3 and at the depot 0′ cannot be higher than the corresponding arrival times in the solution of panel (*a*). This implies that the duration of solution $(b), t_b$, given by $t_b = \max\{t^{\mathrm{T}}_{01} + t^{\mathrm{T}}_{12} + t^{\mathrm{T}}_{23}, t^{\mathrm{D}}_{06} + t^{\mathrm{D}}_{63}\} + \max\{t^{\mathrm{T}}_{34} + t^{\mathrm{T}}_{45} + t^{\mathrm{T}}_{50'}, t^{\mathrm{D}}_{37} + t^{\mathrm{D}}_{70'}\}$, cannot be higher than t_a.

Because of Proposition 3, to compute $t^*(m)$, the following constraints can be added to (1):

$$my_i = w_i \quad i \in N \tag{18a}$$

$$\boldsymbol{y}(N) \leqslant n - 2m \tag{18b}$$

$$\sum_{\langle 0,j,k\rangle \in \mathscr{L}} z_{0jk} = \sum_{\langle i,j,0'\rangle \in \mathscr{L}} z_{ij0'} = m \tag{18c}$$

$$\sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} = \sum_{\langle k,j,i\rangle \in \mathscr{L}} z_{kji} \quad k \in N \tag{18d}$$

Constraints (18a) state that if customer $i \in N$ is visited by the truck, then m drones must be airborne when the truck leaves i. Constraint (18b) ensures that the truck does not visit more than $n-2m$ customers as each drone is supposed to serve at least two customers. Constraints (18c) ensure that all drones take off and land at the depot at the beginning and the end of the route. Constraints (18d) are flow conservation constraints for the drones guaranteeing that all drones are airborne when the truck leaves each served customer.

If constraints (18) are added to (1), because of constraints (18a), the w-variables can be removed from the model and constraints (1g)–(1k) become redundant. Therefore, $t^*(m)$ can be computed as follows:

$$t^*(m) = \min\{\, a_{0'} \text{ s.t. } (1b) - (1f), (1l) - (1q), (18b) - (18d)\,\} \tag{19}$$

Notice that all valid inequalities introduced in Section 4 are still valid and can be added to (19).

5.2. Computing a lower bound to $t^*(\gamma)$ with $\gamma < m$

To compute a lower bound, $lb(\gamma)$ ($\gamma = 0,1,\ldots,m-1$), to $t^*(\gamma)$, we solve a MILP derived from (19) by allowing $m-\gamma$ customers not to be visited at all and γ drones to serve at least two customers each. The MILP to compute $lb(\gamma)$ is the following:

$$lb(\gamma) = \min a_{0'} \tag{20a}$$

$$\text{s.t. } (1b) - (1d), (1f), (1l) - (1q), (18d) \qquad y_j + \sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} \leqslant 1 \quad j \in N \tag{20b}$$

$$y\left(N\right) + \sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} = n - m + \gamma \tag{20c}$$

$$y(N) \leqslant n - m - \gamma \tag{20d}$$

$$\sum_{\langle 0,j,k\rangle \in \mathscr{L}} z_{0jk} = \sum_{\langle i,j,0'\rangle \in \mathscr{L}} z_{ij0'} = \gamma \tag{20e}$$

Constraints (20b) replace constraints (1e) by stating that every customer is served at most once. Constraint (20c) guarantees that $n-(m-\gamma)$ customers are served. Constraint (20d) replaces constraint (18b) and ensures that at most $n-(m-\gamma)-2\cdot\gamma = n-m-\gamma$ customers are served by the truck. Constraints (20e) replace constraints (18c) to ensure that exactly γ drones take off and land at the depot at the beginning and the end of the route. Moreover, notice that, in constraints (1f) and (1m), m can be replaced with $\max\{1,\gamma\}$ to tighten the formulation.

As to the valid inequalities presented in Section 4, constraints (7a) and (10) are valid for (20) and can be separated without any changes. Constraints (6) and (9) are valid for (20), too, and can be separated, but m can also be replaced by γ to tighten the formulation. Constraints (8), (11), and MODLC (15)–(17b) are also valid for (20), but m can also be replaced by $\max\{1,\gamma\}$. Inequality (3) must be

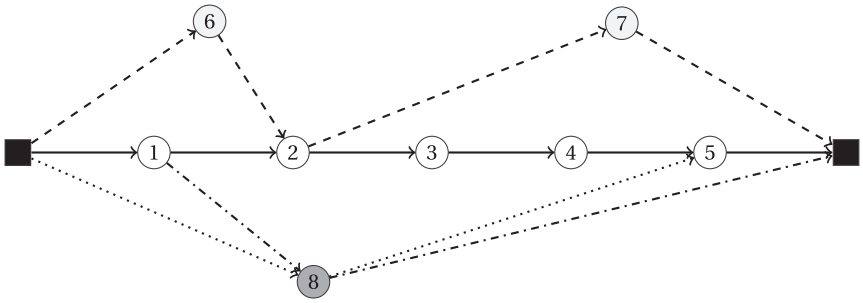

Fig. 8. Example of the optimality check on an optimal solution of formulation (20).

changed as the minimum number of customers that must be served by the truck if γ drones are used out of m (let us call such a value $n^{\mathrm{T}}(\gamma)$) depends on γ and can be computed as $n^{\mathrm{T}}(\gamma) = \lceil \frac{n-m}{\gamma+1} \rceil$; thus, constraint (3) is replaced by $y(N) \geqslant n^{\mathrm{T}}(\gamma)$. Similarly, in Symmetry Breaking constraints (4), (5) and in TSPC (12), n^{T} must be replaced by $n^{\mathrm{T}}(\gamma)$. Finally, in LGSEC (14), $\overline{n}^{\mathrm{D}}$ must be replaced by $n - n^{\mathrm{T}}(\gamma)$.

Optimality Check. In general, $lb(\gamma)$ is just a lower bound to $t^*(\gamma)$, and there is no guarantee that the two values coincide. However, it can be the case that $lb(\gamma) = t^*(\gamma)$ and is possible to derive an optimal solution corresponding to $lb(\gamma)$ from the optimal solution of (20). We show this case in Fig. 8.

The top part of Fig. 8 shows an optimal solution of formulation (20) for an instance with eight customers ($n = 8$), two drones ($m = 2$), and $\gamma = 1$. Customer 8 is not served in this optimal solution. The truck serves Customers 1, 2, 3, 4, 5, in this sequence, and the only available drone performs drone legs $\langle 0,6,2 \rangle$ and $\langle 2,7,0' \rangle$. Let a_1^* and a_5^* be the departure time of the truck from Customers 1 and 5 in this solution. If at least one of the three inequalities $t_{08}^{\mathrm{D}} + t_{85}^{\mathrm{D}} \leqslant a_5^*$, $t_{18}^{\mathrm{D}} + t_{80'}^{\mathrm{D}} \leqslant lb(\gamma) - a_1^*$, $t_{08}^{\mathrm{D}} + t_{85}^{\mathrm{D}} + t_{50'}^{\mathrm{T}} \leqslant lb(\gamma)$ holds, $lb(\gamma)$ is equal to $t^*(\gamma)$. Indeed, one of the two drone legs $\langle 0,8,5 \rangle$ and $\langle 1,8,0' \rangle$ can be added to the solution without affecting the duration of the tour.

In general terms, let (x^*, y^*, z^*, a^*) be an optimal solution of formulation (20) of cost $lb(\gamma)$. Let i_f and i_ℓ be the first and the last customer served by the truck in such a solution. Let $\underline{a}_{i_f}^*$ ($\overline{a}_{i_\ell}^*$, resp.) be the minimum (maximum, resp.) departure time of the truck from node i_f (i_ℓ, resp.) such that the duration of the tour is $lb(\gamma)$ computed as

$$\underline{a}_{i_f}^* = \max \left\{ t_{0i_f}^{\mathrm{T}}, \max_{\substack{\langle 0,i,i_f \rangle \in \mathscr{Z}: \\ z^*_{0ii_f}=1}} \left\{ t_{0i}^{\mathrm{D}} + t_{ii_f}^{\mathrm{D}} \right\} \right\} \qquad \overline{a}_{i_\ell}^* = lb\left(\gamma\right) - \max \left\{ t_{i_\ell 0'}^{\mathrm{T}}, \max_{\substack{\langle i_\ell,i,0' \rangle \in \mathscr{Z}: \\ z^*_{i_\ell i 0'}=1}} \left\{ t_{i_\ell i}^{\mathrm{D}} + t_{i0'}^{\mathrm{D}} \right\} \right\}$$

Moreover, let $\overline{N}^*$ be the set of $m - \gamma$ customers that are not served in (x^*, y^*, z^*, a^*). If, for all customers $i \in \overline{N}^*$, the following inequality holds

$$\max \left\{ t_{0i}^{\mathrm{D}} + t_{ii_\ell}^{\mathrm{D}} - \overline{a}_{i_\ell}^*, \; t_{i_f i}^{\mathrm{D}} + t_{i0'}^{\mathrm{D}} - lb\left(\gamma\right) + \underline{a}_{i_f}^*, \right\} \leqslant 0,$$

then $lb(\gamma) = t^*(\gamma)$.

5.3. Exact solution method

The exact method we propose to solve the TSP-MD consists of the following steps:

Step 1. *Compute an upper bound to t^*.* Solve formulation (19) with the addition of constraint $y(N) = n - 2 \cdot m$ and the valid inequalities described in Section 4 to compute an upper bound to t^*. Let *ub* be such an upper bound.

Step 2. *Compute $t^*(m)$.* Solve formulation (19) with the addition of the valid inequalities described in Section 4 and by replacing M with *ub*. Use the optimal solution of Step 1 to hot-start the solver.

Step 3. *Compute $lb(\gamma)$ with $\gamma = 0, 1, \ldots, m-1$.* For each value of γ between 0 and $m-1$, solve formulation (20) with the addition of the valid inequalities presented in Sections 4 and 5.2. If the corresponding optimal solution does not pass the optimality check described in Section 5.2, then go to Step 5.

Step 4. *Compute t^*.* Because $t^*(m)$ and $lb(\gamma)$, with $\gamma = 0, 1, \ldots, m-1$, have been computed, and, for every γ, $lb(\gamma)$ is equal to $t^*(\gamma)$, then compute the optimal TSP-MD cost t^* as

$$t^* = \min\{ t^*(m), \min\{ lb(\gamma) \mid \gamma = 0, 1, \ldots, m \} \}.$$

Return t^*, and stop.

Step 5. *Compute t^* by using formulation (1).* Solve formulation (1) with the addition of the valid inequalities described in Section 4, and by replacing M with ub. Return t^*.

In our computational experience, Step 5 has proved to be necessary only in a few small instances with nine customers and at least two drones. However, in these cases, the optimal solution is quickly found by solving (1) because of the small size of the instances.

6. Additional side constraints

The TSP-MD addressed in this paper is a "core" problem and does not feature different side constraints discussed in the literature on similar problems where drones work in tandem with trucks. There are three main types of side constraints that are often studied in the literature: loops, incompatible customers, and drone flying range. In this section, we describe if and how these three types of side constraints can be embedded in model (1), how the valid inequalities presented in Section 4 should be adjusted to take these side constraints into account, and how the decomposition approach presented in Section 5 should be changed because of these additional side constraints.

6.1. Loops

In the TSP-MD, we have assumed that the truck is traveling while drones are airborne, so a drone cannot take off and land at the same location while the truck stands still. This means that drone legs of type $\langle i,j,i\rangle$, sometimes called *loops* in the literature (see, e.g., Roberti and Ruthmair, 2020), where $i \in N_0$ and $j \in N\backslash\{i\}$, are not allowed. Adding loops to formulation (1) is not straightforward. Indeed, computing the arrival times a_i at each node $i \in V$ as done by constraints (1m) does not allow to compute the correct value of $a_{0'}$ if loops are performed because the time the truck waits for the drones performing some loops must be taken into account. If we assume that the truck stands still at location i while some drones perform some loops $\langle i,j,i\rangle$, the minimum waiting time of the truck at i from launching the drones until all these drones rejoin the truck before leaving location i can be computed if the loops are explicitly assigned to the drones. This requirement calls for the definition of extra three-index binary variables of type φ_{ijk} indicating if drone $k = 1,\ldots,m$ performs loops $\langle i,j,i\rangle$ or not. Moreover, some extra constraints are also necessary to derive a correct model. We leave the addition of loops to formulation (1) as future research avenue and do not further elaborate on it.

6.2. Incompatible customers

One of the assumptions of the TSP-MD is that all customers can be served by the drone as the corresponding packages can be carried by the drones. However, this assumption may not hold in practice (e.g., the package required by a specific customer may be too heavy for a drone), and just a subset of customers $N_d \subseteq N$ may be eligible for drone delivery. Formulation (1) can embed this feature by simply redefining the set of drone legs $\mathscr{L}$ as $\mathscr{L} = \{\langle i,j,k\rangle \mid i \in N_0, j \in N_d, k \in N_{0'} : i \neq j \neq k\}\backslash\{\langle 0,j,0'\rangle \mid j \in N\}$, and setting y_i equal to 1 for each customer $i \in N\backslash N_d$. The minimum number of truck customers, n^{T}, can also be computed as $n^{\mathrm{T}} = \max\left\{\lceil\frac{n-m}{m+1}\rceil,\ n - \left|N_d\right|\right\}$ instead of $n^{\mathrm{T}} = \max\lceil\frac{n-m}{m+1}\rceil$. The other valid inequalities described in Section 4 are still valid, and it might be possible to lift some of them. Finally, the exact solution method can be applied as it is. A computational evaluation of this method is provided in Section 7.

6.3. Drone flying range

In the TSP-MD, we have assumed that drones have unlimited flying range, i.e., every drone leg $\langle i,j,k\rangle$ with $i \in N_0, j \in N, k \in N_{0'}$, and $i \neq j \neq k$ is feasible. However, drones can usually travel up to a maximum distance because of their limited battery capacity, so they have a maximum *drone flying range*. Under the assumption that the speed of the drones is constant (as commonly assumed in the literature), we can express the drone flying range in terms of maximum traveling distance or maximum traveling time. Embedding the drone flying range into formulation (1) is straightforward as we just have to define the set $\mathscr{L}$ as the set of *feasible* drone legs. However, we cannot solve the TSP-MD with the drone flying range by using the exact solution method described in Section 5.3 because Proposition 3 does not hold anymore; indeed, it is easy to observe that there can be optimal solutions of the TSP-MD where some drones are on-board while the truck is moving if drones are subject to a maximum flying range. In Section 7, we report the computational results of the compact formulation (1) with the addition of the valid inequalities introduced in Section 4 on the TSP-MD with the drone flying range.

7. Computational results

In this section, we discuss the computational results achieved by the exact solution method described in Section 5, provide managerial insights, report a sensitivity analysis on key parameters and components of the proposed solution method, and illustrate the computational behaviour of the exact method when side constraints are considered. This section is organized as follows. In Section 7.1, we describe the test instances used in the computational experiments. Section 7.2 summarizes the results obtained on the baseline TSP-MD test instances we have selected. In Section 7.3, we show how the speed of the drone affects the performance of the solution method, and we offer some managerial insights to investigate the importance of the drone speed. Section 7.4 compares the

Table 1
Computational Results on the Baseline Test Instances with $\alpha = 0.5$.

	Overall			Step 1		Step 2								Step 3		Step 5	
n	Opt	%Gap	T	% UB	T	% UB	% Root	% LP	LGSEC	TSPC	MODLC	T	% T_{sep}	Opt	T	Call	T
								(m = 1)									
9	10	0.0	1	22.9	0	0.0	8.2	23.5	10	7	29	1	2	0	0	0	0
14	10	0.0	8	31.8	0	0.0	6.5	21.9	32	29	125	8	6	0	0	0	0
19	10	0.0	183	33.0	2	0.0	7.8	25.1	132	71	437	182	21	0	0	0	0
24	9	0.8	1809	41.0	26	0.0	7.0	25.2	239	145	679	1783	60	0	0	0	0
All	39	0.8	467	31.9	7	0.0	7.4	23.9	100	61	308	460	21	0	0	0	0
								(m = 2)									
9	10	0.0	1	16.5	0	0.0	9.8	20.5	12	4	16	1	1	0	0	0	0
14	10	0.0	72	35.6	1	0.0	12.8	28.4	115	26	193	71	2	0	1	0	0
19	8	1.8	815	48.8	4	0.0	11.2	31.3	271	114	617	808	2	0	4	0	0
24	2	6.8	1787	62.8	28	0.0	7.7	29.8	329	231	783	1750	6	0	10	0	0
All	30	5.8	361	34.6	3	0.0	11.0	26.6	136	56	287	356	2	0	2	0	0
								(m = 3)									
9	10	0.0	1	4.9	0	0.0	19.6	14.6	4	1	3	0	1	0	0	1	0
14	10	0.0	46	38.5	1	0.0	13.7	27.5	89	13	113	41	2	0	3	0	0
19	7	5.5	1370	51.4	5	0.0	14.5	32.3	393	51	555	1340	1	0	26	0	0
24	1	13.1	4734	72.6	23	0.0	11.3	38.0	356	137	965	4662	1	0	49	0	0
All	28	11.2	528	31.0	3	0.0	15.9	24.4	144	23	215	516	1	0	10	1	0

Table 2
Computational Results on the Instances with $\alpha = 0.33$.

	Overall			Step 1		Step 2								Step 3		Step 5	
n	Opt	%Gap	T	% UB	T	% UB	% Root	% LP	LGSEC	TSPC	MODLC	T	% T_{sep}	Opt	T	Call	T
								(m = 1)									
9	10	0.0	1	40.4	0	0.0	6.1	22.9	10	7	25	1	2	0	0	0	0
14	10	0.0	6	43.3	0	0.0	6.7	26.1	24	26	88	6	3	0	0	0	0
19	10	0.0	85	47.6	2	0.0	7.0	28.4	90	53	247	84	19	0	0	0	0
24	10	0.0	668	55.3	25	0.0	6.6	26.9	137	114	382	644	63	0	0	0	0
All	40	0.0	190	46.7	7	0.0	6.6	26.1	65	50	186	183	22	0	0	0	0
								(m = 2)									
9	10	0.0	1	37.7	0	0.0	7.3	17.7	7	3	6	1	1	0	0	0	0
14	10	0.0	15	61.8	1	0.0	10.5	28.0	57	17	94	14	2	0	1	0	0
19	10	0.0	245	71.5	4	0.0	10.9	34.2	171	79	383	239	2	0	2	0	0
24	6	5.9	2027	80.0	38	0.0	9.2	33.5	281	160	647	1984	6	0	5	0	0
All	36	5.9	410	60.8	8	0.0	9.5	27.8	112	54	242	401	2	0	2	0	0
								(m = 3)									
9	10	0.0	1	12.1	0	0.0	15.7	10.2	1	0	1	0	0	1	0	1	0
14	10	0.0	20	67.4	1	0.0	15.3	27.1	56	11	62	17	1	0	2	0	0
19	10	0.0	292	84.4	5	0.0	13.5	34.2	207	29	286	279	2	0	8	0	0
24	3	4.1	1583	110.1	30	0.0	10.3	35.0	270	78	429	1533	2	0	19	0	0
All	33	4.1	239	59.6	5	0.0	14.4	24.8	104	19	144	229	1	1	5	1	0

Table 3
Computational Results on the Instances with $\alpha = 1$.

	Overall			Step 1		Step 2								Step 3		Step 5	
n	Opt	%Gap	T	% UB	T	% UB	% Root	% LP	LGSEC	TSPC	MODLC	T	% T_{sep}	Opt	T	Call	T
									(m = 1)								
9	10	0.0	1	1.7	0	0.0	8.3	21.4	14	7	32	1	3	0	0	0	0
14	10	0.0	62	6.5	1	0.0	9.0	21.1	88	44	232	62	12	0	0	0	0
19	7	1.5	3000	12.4	4	0.0	8.2	23.1	438	99	814	2996	33	0	0	0	0
24	0	6.5	–	–	–	–	–	–	–	–	–	-	–	–	–	–	–
All	27	7.5	801	6.2	1	0.0	8.5	21.8	151	45	309	800	14	0	0	0	0
									(m = 2)								
9	10	0.0	3	0.3	1	0.0	13.4	25.5	19	10	28	2	2	0	1	0	0
14	10	0.0	488	6.4	4	0.0	12.2	25.0	352	37	417	483	2	0	2	0	0
19	0	8.8	–	–	–	–	–	–	–	–	–	-	–	–	–	–	–
24	0	15.6	–	–	–	–	–	–	–	–	–	-	–	–	–	–	–
All	20	12.2	246	3.4	2	0.0	12.8	25.2	185	24	222	242	2	0	1	0	0
									(m = 3)								
9	10	0.0	4	1.0	1	1.0	13.7	19.2	24	3	9	1	1	3	1	2	1
14	10	0.0	585	3.4	71	0.0	15.0	27.5	345	24	358	495	1	0	20	0	0
19	0	14.2	–	–	–	–	–	–	–	–	–	-	–	–	–	–	–
24	0	23.1	–	–	–	–	–	–	–	–	–	-	–	–	–	–	–
All	20	18.7	295	2.2	36	0.2	14.4	23.3	184	13	183	248	1	3	10	2	1

performance of our exact solution method with the performance of the compact formulation (1) presented in Section 3. Section 7.5 shows how TSPC (12), LGSEC (14), and MODLC (15)–(17b) affect the performance of the solution method. Section 7.6 presents the computational results of the exact solution method to handle different side constraints as described in Section 6. Finally, Section 7.7 compares the results achieved by the proposed exact solution method with those achieved by the branch-and-price algorithm proposed by Roberti and Ruthmair (2020) on the single-drone instances.

Both the exact solution method and the compact formulation (1) are coded in C, compiled with `g++ 7.5.0`, and solved with `CPLEX 12.8`. All experiments are conducted on a single thread of an Intel Xeon E3-1245v5 machine running at 3.50 GHz. We use the default parameters of `CPLEX`, except for `CPX_PARAM_THREADS` (set to one to use a single thread) and `CPX_PARAM_EPINT` (set to 1e-10 to decrease integrality tolerance). A time limit of two hours is imposed to solve each instance. All computing times reported in this section are in seconds.

7.1. Test instances

We use a set of 40 baseline instances with 10, 15, 20, and 25 nodes (ten instances per size) derived from the instances introduced by Poikonen et al. (2019). The instances are created by randomly locating the depot and the n customers on a 50-by-50 grid, where node coordinates are uniformly distributed in the two dimensions. The geographical x-y coordinates of each node i are given as (x_i, y_i). As commonly done in the literature, for each arc $(i,j) \in A$, we compute truck travel times, t_{ij}^{T}, according to the Manhattan metric (i.e., $t_{ij}^{\mathrm{T}} = \left\lceil \left|x_i - x_j\right| + \left|y_i - y_j\right| \right\rceil$) and drone travel times, t_{ij}^{D}, according to the Euclidean metric (i.e., $t_{ij}^{\mathrm{D}} = \sqrt{\left\lceil ((x_i - x_j)^2 + (y_i - y_j)^2)\alpha \right\rceil}$), where α is a parameter that represents the ratio between the speed of the truck and the speed of the drones and is set equal to 0.5. Travel times are rounded up to the nearest integer to ensure that the triangle inequality holds. Each instance is tested with one, two, and three drones, leading to a baseline set of 120 instances. These 120 instances are used in Section 7.2. In Section 7.3, where we investigate the importance of parameter α, we use 240 more instances, still derived from the baseline instances, but by setting $\alpha = 0.33$ and $\alpha = 1$.

7.2. Computational results on the baseline test instances

Table 1 summarizes the results achieved by the exact solution method presented in Section 5 on the baseline test instances described in Section 7.1. The results are grouped by the number of drones ($m = 1,2,3$) and size of the instance ($n = 9,14,19,24$). Each row reports averages over the corresponding ten instances. All the gaps are computed, in percentage, with respect to the best upper bound available.

Column n indicates the number of customers. Under label Overall, column Opt indicates the number of instances solved to proven optimality (out of 10), %Gap the average percentage gap of the best lower bound computed at Step 2 over the instances not solved to proven optimality, and T the average computing time over the instances closed to proven optimality. About Step 1, column %UB reports the average percentage gap of the optimal solution of Step 1, and T indicates the average total computing time. About Step 2, column %UB reports the average percentage gap of the optimal solution of Step 2, %Root (%LP, resp.) the average gap of the lower bound at the root node (at the root node before adding TSPC (12), LGSEC (14), MODLC (15)–(17b), resp.), LGSEC, TSPC, MODLC the number of cuts added of the corresponding type, T the average total computing time, and $\%T_{sep}$ the average amount of time spent on separating cuts in percentage over T. About Step 3, Opt shows how many times Step 3 provides a solution that is better than the optimal solution of Step 2, and T is the average computing time. About Step 5, Call indicates how many times Step 5 is called, and T is the average computing time.

Table 1 shows that the exact method finds an optimal solution for 97 out of the 120 instances, with an average computing time of less than ten minutes. We can see that increasing the number of customers and/or drones makes the TSP-MD more difficult to solve; this is due to higher average gaps (see column %Root) in Step 2. By looking at the difference between the gaps of columns %Root and %LP, we can also appreciate the significant contribution of separating LGSEC, TSPC, and MODLC - further details about the effect of adding these cuts are provided in Section 7.5. The right-most columns of the table show that Step 3 never finds a better solution than Step 2, and Step 5 is necessary only once (on an instance with nine customers and three drones).

7.3. Computational results with different drone speeds

In this section, we show how the speed of the drone affects the performance of the exact solution method and provide some managerial insights on the importance of this parameter.

Tables 2 and 3 summarize the results achieved by the exact solution method on the test instances when setting $\alpha = 0.33$ (i.e., drones are three times faster than the truck) and $\alpha = 1$ (i.e., drones are as fast as the truck), respectively. The format of these two tables is the same of Table 1.

Table 2 shows that 109 instances can be solved to optimality when $\alpha = 0.33$, i.e., 12 more instances than when $\alpha = 0.5$, and the average computing time is less than five minutes. Moreover, by comparing Tables 1 and 2, we can see that all instances with up to 19 customers can be solved to optimality when $\alpha = 0.33$, which is not the case when $\alpha = 0.5$. As in Table 1, increasing the number of customers and/or drones makes the instances harder to solve due to higher gaps in Step 2. Step 3 provides the optimal solution only once, and Step 5 is necessary only on an instance with nine customers and three drones. All in all, we can conclude that the higher the speed of the drones, the higher the chance that an instance can be solved to optimality within lower computing times.

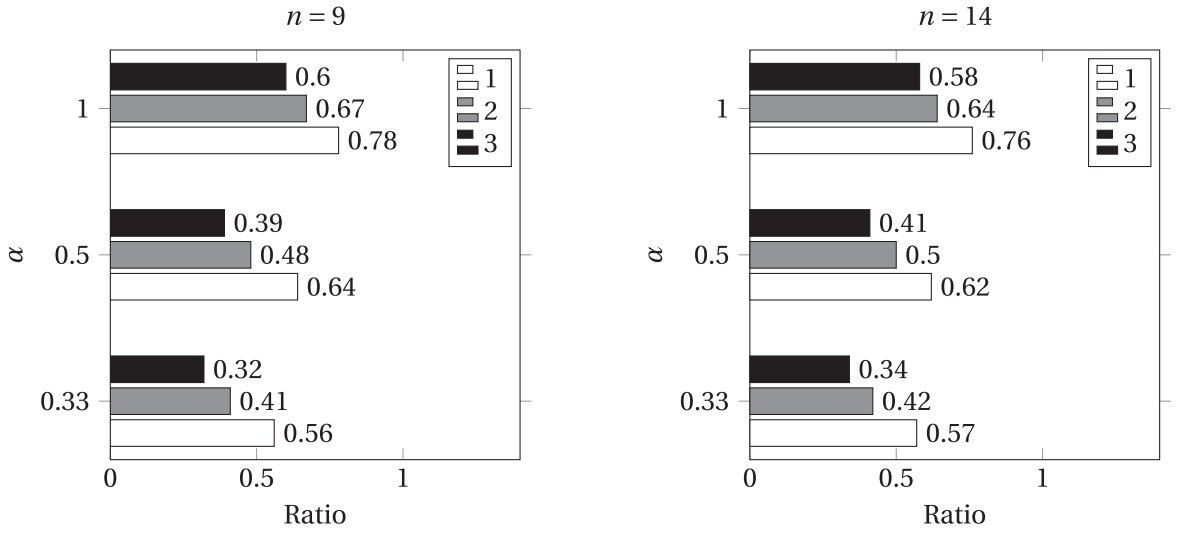

Fig. 9. Ratios between Average Optimal Values for Different Values of α (1, 0.5, 0.33) and m (1, 2, 3) compared with the TSP without Drones.

Table 4
Comparison between CF and Dec on the Baseline Instances with $m = 2$.

	CF			Dec		
n	Opt	T	T_{both}	Opt	T	T_{both}
9	10	1	1	10	1	1
14	10	647	647	10	72	72
19	5	1157	1157	8	815	414
24	2	1729	1729	2	1787	1787

Table 3 shows that, when the truck and the drones are equally fast, all instances with up to 14 customers and only nine instances with 19 customers can be solved to optimality, but none of the instances with 24 customers or 19 customers and at least two drones can be closed. By comparing these results with Tables 1 and 2, we can further confirm that decreasing the speed of the drones makes the instances harder to solve. As to the effectiveness of LGSEC, TSPC, and MODLS, and the results of Steps 3 and 5 on instances with $\alpha = 1$, we can draw similar conclusions to the ones from Tables 1 and 2.

In Fig. 9, we show how the average optimal value changes when the number of available drones and their speed change. In the left panel of Fig. 9, we compare the average optimal value over the 9-customer instances solved to optimality for any combination of values of $\alpha = 0.33, 0.5, 1$ and $m = 1, 2, 3$; values are ratios over the average cost of the TSP without using drones. A similar comparison is provided in the right panel for instances with 14 customers.

Fig. 9 shows that the higher the number of drones and their speed, the lower the optimal value. When a single drone is available, the optimal value decreases between 22 and 42% compared to not having drones at all. When multiple drones are available, such a decrease is between 36 and 61% with two drones and 43 and 68% with three drones. In terms of optimal value, having a single fast drone (with $\alpha = 0.33$) is similar to having three slow drones (with $\alpha = 1$).

7.4. Comparison between the exact solution method and the compact formulation

In this section, we compare the performance of the compact formulation (1) (hereafter called CF) presented in Section 3, when solved with CPLEX with the addition of all the valid inequalities presented in Section 4, with the performance of the exact decomposition method (hereafter called Dec) presented in Section 5.

As a benchmark set, we use the 40 baseline instances with two drones. Table 4 reports, for both CF and Dec, the number of instances solved to optimality (Opt), the average computing time (T), and the average computing time over the instances solved to optimality by both CF and Dec (T_{both}).

Table 4 shows that CF and Dec have similar performances on 9-customer instances. However, the benefits of Dec over CF stand out on larger instances with 14 and 19 customers. Indeed, both methods can solve all 14-customer instances, but Dec is nine times faster than CF. Moreover, Dec can solve three 19-customer instances more than CF and is almost three times faster. On 24-customer instances, the two approaches have similar performances, with Dec being slightly slower than CF; this behaviour might suggest that Dec does not scale well compared to CF, but no conclusions can be drawn as the comparison is based on just two instances solved to proven optimality by both methods.

Table 5
Comparison on the Baseline Instances with $m = 2$ with and without LGSEC, TSPC, and MODLC.

	All Cuts			no LGSEC			no TSPC			no MODLC		
n	Opt	%Gap	T	Opt	%Gap	T	Opt	%Gap	T	Opt	%Gap	T
9	10	0.0	1	10	0.0	1	10	0.0	1	10	0.0	1
14	10	0.0	72	10	0.0	64	10	0.0	94	10	0.0	170
19	8	1.8	815	6	8.1	1096	8	5.1	1071	4	6.5	2884
24	2	6.8	1787	1	15.2	2856	2	6.6	1403	1	17.7	4037
All	30	5.8	361	27	13.0	373	30	6.3	411	25	13.2	691

Table 6
Summary of the Computational Results on the Baseline Instances with and without Side Constraints (Incompatible Customers and Drone Flying Range).

	Baseline			Inc20			Inc40			Range20			Range40		
n	Opt	%Gap	T	Opt	%Gap	T	Opt	%Gap	T	Opt	%Gap	T	Opt	%Gap	T
							(m = 1)								
9	10	0.0	1	10	0.0	1	10	0.0	0	10	0.0	0	10	0.0	0
14	10	0.0	8	10	0.0	8	10	0.0	4	10	0.0	0	10	0.0	5
19	10	0.0	183	10	0.0	283	10	0.0	45	10	0.0	11	10	0.0	177
24	9	0.8	1809	9	1.0	973	10	0.0	296	10	0.0	410	7	2.7	2763
All	39	0.8	467	39	1.0	299	40	0.0	86	40	0.0	105	37	2.7	572
							(m = 2)								
9	10	0.0	1	10	0.0	1	10	0.0	0	10	0.0	0	10	0.0	0
14	10	0.0	72	10	0.0	14	10	0.0	2	10	0.0	0	10	0.0	6
19	8	1.8	815	10	0.0	260	10	0.0	21	10	0.0	8	10	0.0	240
24	2	6.8	1787	6	4.9	1603	10	0.0	244	10	0.0	46	8	9.6	2441
All	30	5.8	361	36	4.9	343	40	0.0	67	40	0.0	14	38	9.6	579
							(m = 3)								
9	10	0.0	1	10	0.0	0	10	0.0	0	10	0.0	0	10	0.0	0
14	10	0.0	46	10	0.0	12	10	0.0	2	10	0.0	0	10	0.0	3
19	7	5.5	1370	10	0.0	121	10	0.0	17	10	0.0	4	10	0.0	101
24	1	13.1	4734	8	6.9	1203	10	0.0	176	10	0.0	24	7	6.1	1865
All	28	11.2	528	38	6.9	288	40	0.0	49	40	0.0	7	37	6.1	381

7.5. Computational results of the exact solution method without LGSEC, TSPC, and MODLC

This section shows how separating LGSEC, TSPC, and MODLC affects the performance of the exact solution method. As we have done in Section 7.4, we use the 40 baseline instances with two drones as a benchmark set.

Table 5 summarises the results obtained by the exact solution method when all the three types of cuts are separated (label All Cuts), without LGSEC (no LGSEC), without TSPC (no TSPC), and without MODLC (no MODLC). Under each of these four labels, column Opt indicates the number of instances solved to optimality, %Gap the average percentage gap, over the instances not solved to optimality, of the best lower bound computed at Step 2, and T the average computing time over the instances solved to optimality.

Table 5 shows that, on instances with up to 14 customers, the exact solution method features similar computational performance with or without cuts. On larger instances with 19 or 24 customers, removing one of the three families of valid inequalities results in fewer instances solved to optimality, larger gaps for the open instances, and usually larger computing times to find an optimal solution. These results illustrate the benefits of separating the three families of valid inequalities.

7.6. Computational results on the TSP-MD with side constraints

This section presents the computational results achieved by the proposed solution method adjusted as described in Section 6 on the TSP-MD with incompatible customers and the drone flying range.

We have tested our solution method on the 120 baseline instances with $\alpha = 0.5$ in four different settings. The first two settings (called Inc20 and Inc40) feature $\lfloor 0.2n \rfloor$ and $\lfloor 0.4n \rfloor$ customers that are incompatible with drone delivery; these customers are the first customers of the set N. The last two settings (called Range20 and Range40) feature a maximum drone flying range equal to $0.2^{*}\max_{(i,j,k)\in\mathcal{L}}\{t^{D}_{ij}+t^{D}_{jk}\}$ and $0.4^{*}\max_{(i,j,k)\in\mathcal{L}}\{t^{D}_{ij}+t^{D}_{jk}\}$, respectively.

Table 6 summarizes the computational results achieved on the baseline instances and these four different settings with side constraints. As in Table 5, columns Opt indicate the number of instances solved to optimality, %Gap the average percentage gap, over the instances not solved to optimality, of the best lower bound computed at Step 2, and T the average computing time over the instances solved to optimality.

Tabel 6 shows that the addition of side constraints make the TSP-MD easier to solve. Whereas only 97 of the 120 baseline instances

Table 7
Comparison between Dec and the branch-and-price of Roberti and Ruthmair (forthcoming) on the instances with $m = 1$

	Dec		BP		
n	Opt	T	Opt	T	T_{both}
		($\alpha = 0.33$)			
9	10	1	10	1	1
14	10	6	10	2	2
19	10	85	10	24	24
24	10	668	10	148	148
All	40	190	40	44	44
		($\alpha = 0.5$)			
9	10	1	10	1	1
14	10	8	10	2	2
19	10	183	10	29	29
24	9	1809	10	162	157
All	39	467	40	48	44
		($\alpha = 1$)			
9	10	1	10	1	1
14	10	62	10	7	7
19	7	3000	10	141	117
24	0	–	10	1090	–
All	27	801	40	310	33

can be solved to proven optimality, 113 instances can be solved in setting Inc20 and all 120 instances in setting Inc40. Moreover, all 120 instances can be solved in setting Range20 and 112 instances in setting Range40. This computational behaviour indicates that the lower the number of feasible drone legs, the higher the chances of solving the problem to optimality. We can also observe that the average computing time is lower in setting Inc40 compared with setting Inc20 and in setting Range20 compared with setting Range40.

7.7. Comparison between the exact solution method and the branch-and-price of Roberti and Ruthmair (2020) on the single-drone instances

In this section, we compare the performance of the exact decomposition method (Dec) with the performance of the branch-and-price algorithm (hereafter called BP) proposed by Roberti and Ruthmair (2020), which, to the best of our knowledge, is the state-of-the-art algorithm for the single-drone instances of the TSP-MD.

As already indicated in Section 2, the set-partitioning formulation used in Roberti and Ruthmair (2020) can be easily adjusted to formulate the TSP-MD, but adjusting the dynamic programming recursion to generate columns when multiple drones are present is challenging. Therefore, nothing can be said about the performance on the TSP-MD when several drones are available of an extension of the branch-and-price of Roberti and Ruthmair (2020). On the other hand, as this branch-and-price is tailored for the single-drone case, it is easy to expect that it outperforms Dec on the single-drone instances. For the sake of completeness, we have tested both methods on the 40 baseline instances (i.e., with $\alpha = 0.5$) with $m = 1$ and the corresponding 80 instances with $\alpha = 0.33$ and $\alpha = 1.0$, already used in the previous sections.

Table 7 summarizes, for each value of $\alpha = 0.33, 0.5, 1.0$ and each value of $n = 9, 14, 19, 24$, the number of instances solved to optimality (Opt) by each of the two methods, the average computing time (T) over all instances solved, and the average computing time (T_{both}) over the instances solved by both Dec and BP.

Table 7 shows that BP outperforms Dec both in terms of instances solved to optimality and average computing time. BP can close all 120 instances whereas Dec cannot solve one of 24-customer instances with $\alpha = 0.5$ and 13 instances with $\alpha = 1.0$. The results indicate that BP outperforms Dec especially when the drone is slow.

8. Conclusions and discussion

In this paper, we have investigated the Traveling Salesman Problem with Multiple Drones (TSP-MD). We have proposed a compact formulation to model the problem and several sets of valid inequalities to improve the continuous relaxation of this compact formulation. Finding an optimal solution of test instances with just 15 nodes might take more than an hour of computing time if the compact formulation is solved as it is with an off-the-shelf solver. Therefore, we have proposed an exact decomposition approach that solves the TSP-MD to optimality by decomposing the problem into $m+1$ simpler problems, where m stands for the number of drones available. This exact approach provides encouraging results and allows to solve instances with up to 24 customers and three drones in less than two hours of computing time, more than doubling the size of solvable instances of related problems with similar approaches from the literature. We have also conducted a sensitivity analysis to shed light on the advantages and computational implications of varying the number of drones available and their speed. The main insights we have offered are that (*a*) increasing the number of drones and their speed can significantly decrease the time to serve all customers and (*b*) the faster the drones, the higher the chances that the exact method can find a provably optimal solution. Moreover, we have discussed how common side constraints, such as loops,

incompatible customers, and drone flying range, can be embedded in the proposed solution method. Finally, we have shown that the presence of incompatible customers and drone flying range makes the TSP-MD easier to solve to optimality.

Although our method scales better than most of the exact methods proposed in the literature, it can solve small-size instances with 25 nodes only. The current literature shows that medium- and large-size instances should be solved with metaheuristics to achieve good solutions. We are not aware of particularly good results achieved by applying matheuristics, for example, by limiting the solution space of a formulation with some variable fixing or decomposition, or heuristically reducing the search tree of a branch-and-bound. Nonetheless, these techniques might be promising solution methods to tackle large instances.

An important extension of the TSP-MD that would be interesting to study arises when multiple trucks are available. The resulting problem is challenging to model in a sharp way. An extra index referring to the truck can obviously be added to the x and z variables, but the corresponding mathematical model would quickly grow in terms of number of variables and constraints. The size of such a model and the symmetries of the corresponding feasible solutions in the set of trucks would most likely make the problem difficult to solve. The valid inequalities introduced in Section 4 must also be properly revised and may be less effective in decreasing the integrality gap. Therefore, the multi-vehicle extension of the TSP-MD requires a thorough and tailored study.

We envision several future research directions to extend our study beyond the scope of this paper. First, richer problems with different side constraints could be considered, and the effectiveness of our solution approach could be investigated - an example of a relevant additional feature is to allow the truck and the drones to meet on-road and not only at customer locations. Then, other objective functions, for example taking environmental considerations into account, could be examined, potentially in a multi-objective optimization setting. Finally, uncertainty could be considered to explore settings that are as close as possible to real-life settings.

CRediT authorship contribution statement

Sara Cavani: Conceptualization, Methodology, Software, Validation, Investigation, Writing - original draft. **Manuel Iori:** Conceptualization, Methodology. **Roberto Roberti:** Conceptualization, Methodology, Writing - original draft.

Acknowledgments

The authors are grateful to the anonymous Associate Editor and the two Referees for their valuable comments.

Appendix A. Notation

Table 8
Summary of the notation used in Sections 3,2,3,4,5.

Symbol	Meaning
G	complete directed graph $G = (V, A)$ of vertices V and arcs A
V	vertex set defined as $V = N \cup \{0, 0'\}$
N	set of n customers
0	(initial) depot
$0'$	(final) depot - copy of 0
N_0	set of customers plus 0, i.e., $N_0 = N \cup \{0\}$
$N_{0'}$	set of customers plus $0'$, i.e., $N_{0'} = N \cup \{0'\}$
A	arc set defined as $A = \{(0,j) \mid j \in N\} \cup \{(i,j) \mid i,j \in N : i \neq j\} \cup \{(i,0') \mid i \in N\}$
m	number of drones available
t_{ij}^{T}	truck travel time of arc $(i,j) \in A$
t_{ij}^{D}	drone travel time of arc $(i,j) \in A$
$x_{ij} \in \{0,1\}$	binary variable equal to 1 if the truck traverses arc $(i,j) \in A$ (0 otherwise)
$y_i \in \{0,1\}$	binary variable equal to 1 if the truck visits node $i \in N$ (0 otherwise)
$z_{ijk} \in \{0,1\}$	binary variable equal to 1 if a drone performs drone leg $\langle i,j,k \rangle \in \mathscr{L}$
$a_i \in \mathbb{R}_+$	departure time of the truck from node $i \in V$
$w_i \in \mathbb{Z}_+$	number of drones airborne when the truck leaves node $i \in V$
$A(S) \subset A$	set of arcs whose endpoints belong to the set $S \subseteq N$, i.e., $A(S) = \{(i,j) \in A \mid i,j \in S\}$
$A(S_1, S_2)$	set of arcs starting from any node of the set $S_1 \subset V$ and ending at any node of the set $S_2 \subset V$, i.e., $A(S_1, S_2) = \{(i,j) \in A \mid i \in S_1, j \in S_2\}$
$x(\widehat{A})$	sum of the x-variables of the arcs of the set $\widehat{A} \subseteq A$, i.e., $x(\widehat{A}) = \sum_{(i,j) \in \widehat{A}} x_{ij}$
$y(S)$	sum of the y-variables of the customers of the set $S \subseteq N$, i.e., $y(S) = \sum_{i \in S} y_i$

Appendix B. Proof of Proposition 1

We prove the validity of (11a) by considering all possible cases; the validity of (11b) can be proven with similar argumentation. To prove the validity of (11a), we consider four cases determined by all possible values of its right-hand side:

1. $y_i = 0$ and $y_j = 0$: as the truck does not visit customers i and j, then (a) the truck cannot traverse arc $(0,i)$ (i.e., $x_{0i} = 0$), (b) it cannot traverse arc (i,j) (i.e., $x_{ij} = 0$), (c) no drone legs starting from a node $s \in N_0 \setminus \{0,i\}$ and ending at j can be selected (i.e., $\sum_{\langle s,r,j\rangle \in \mathscr{L}: s \neq i,0} z_{srj} = 0$), and ($d$) drone leg $\langle 0,i,j\rangle$ cannot be selected. Therefore, whenever $y_i + y_j = 0$, the left-hand side cannot be strictly positive.
2. $y_i = 1$ and $y_j = 0$: as the truck serves customer i (i.e., $y_i = 1$), it can traverse arc $(0,i)$ (i.e., $x_{0i} = 1$); however, because $y_j = 0$, then (a) the truck cannot traverse arc (i,j) (i.e., $x_{ij} = 0$), and (b) no drone legs ending at j can be selected, so both $\sum_{\langle s,r,j\rangle \in \mathscr{L}: s \neq i,0} z_{srj}$ and z_{0ij} must be 0. Therefore, if $y_i = 1$ and $y_j = 0$, the highest value the left-hand side can take is 1, which corresponds to $x_{0i} = 1$.
3. $y_i = 0$ and $y_j = 1$: as the truck does not serve customer i (i.e., $y_i = 0$), it cannot traverse arcs $(0,i)$ and (i,j) (i.e., $x_{0i} = 0$ and $x_{ij} = 0$); moreover, because $y_j = 1$, the number of drone legs ending at j cannot be greater then the number of drones, so $\sum_{\langle s,r,j\rangle \in \mathscr{L}: s \neq i,0} z_{srj} + z_{0ij} \leqslant m \cdot y_j$. Therefore, if $y_i = 0$ and $y_j = 1$, the highest value the left-hand side can take is 1.
4. $y_i = 1$ and $y_j = 1$: as the truck serves both i and j, we show that the highest value the left-hand side can take in any feasible solution is 2, which happens in the following three cases:
   - $x_{0i} = 1$ and $x_{ij} = 1$: as the first two arcs traversed by the truck are $(0,i)$ and $(i.j)$, no drone legs starting at a node different from 0 and i can land at j, so $\sum_{\langle s,r,j\rangle \in \mathscr{L}: s \neq i,0} z_{srj} = 0$. Furthermore, as the truck serves i, z_{0ij} must be zero.
   - $x_{0i} = 1, x_{ij} = 0$, and $\frac{1}{m}\sum_{\langle s,r,j\rangle \in \mathscr{L}: s \neq i,0} z_{srj} = 1$: this happens if the first arc traversed by the truck is $(0,i)$, j is served by the truck later, and m drone legs ending at j are selected. However, because $x_{0i} = 1, z_{0ij}$ must be zero, so the left-hand side is equal to 2.
   - $x_{0i} = 0, x_{ij} = 1$, and $\frac{1}{m}\sum_{\langle s,r,j\rangle \in \mathscr{L}: s \neq i,0} z_{srj} = 1$: this happens when the truck does not traverse arc $(0,i)$, but it traverses arc (i,j) and m drone legs starting from a node different from 0 and i and ending at j are selected. Because $x_{ij} = 1$, so z_{0ij} must be zero, and the left-hand side is equal to 2. □

Appendix C. Proof of Proposition 2

First, we show the correctness of constraints (14a). To simplify the exposition, let us define

$$\zeta_r\left(S\right) = \sum_{\substack{\langle i,r,k\rangle \in \mathscr{L}: \\ \{i,k\} \cap S \neq \emptyset}} z_{irk}.$$

Because of (1e), $\zeta_r(S) + y_r$ cannot be greater than 1 in any feasible solution, so we have to consider two cases:

1. If $\zeta_r(S) = 0$, constraint (14a) reduces to constraint (13), which is valid for (1);
2. If $\zeta_r(S) = 1$, then $x(A(S)) + 1 \leqslant y(S)$. Because $y_r = 0$ (and $x((A(\{r\}, S\setminus\{r\}))) + x((A(S\setminus\{r\}, \{r\}))) = 0$), we have $x(A(S)) = x(A(S\setminus\{r\}))$ and $y(S) = y(S\setminus\{r\})$. If we define $S' = S\setminus\{r\}$, constraint $x(A(S)) + 1 \leqslant y(S)$ reduces to $x(A(S')) \leqslant y(S') - 1$, which corresponds to the well-known SEC for the set S' and is valid because $\zeta_r(S) = 1$ implies that $y(S') \geqslant 1$, so $y(S') - 1$ cannot be negative.

Second, we show the correctness of constraints (14b). Because $|S| > \bar{n}^{\mathrm{D}}$, then at least one of the customers of the set S must be served by the truck. Therefore, $y(S)$ cannot be lower than 1 in any feasible solution. For any feasible TSP-MD solution, $y(S) - 1$ is equal to the number of customers of the set S served by the truck minus one, which is also the maximum number of arcs of the set $A(S)$ that can be traversed by the truck without creating subtours. □

References

Agatz, N., Bouman, P., Schmidt, M., 2018. Optimization Approaches for the Traveling Salesman Problem with Drone. Transp. Sci. 52, 965–981.

Bakir, I., Özbaygin, G., 2020. Optimizing Drone-Assisted Last-Mile Deliveries: The Vehicle Routing Problem with Flexible Drones. Optimiz. Online.

Baldacci, R., Mingozzi, A., Roberti, R., 2011. New Route Relaxation and Pricing Strategies for the Vehicle Routing Problem. Oper. Res. 59, 1269–1283.

Bergmann, F.M., Wagner, S.M., Winkenbach, M., 2020. Integrating first-mile pickup and last-mile delivery on shared vehicle routes for efficient urban e-commerce distribution. Transp. Res. Part B: Methodol. 131, 26–62.

Bouman, P., Agatz, N., Schmidt, M., 2018. Dynamic Programming Approaches for the Traveling Salesman Problem with Drone. Networks 72, 528–542.

Carlsson, J.G., Song, S., 2017. Coordinated logistics with a truck and a drone. Manage. Sci. 64, 4052–4069.

Carlsson, J.G., Song, S., 2018. Coordinated logistics with a truck and a drone. Manage. Sci. 64, 4052–4069.

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.

Coelho, B.N., Coelho, V.N., Coelho, I.M., Ochi, L.S., Haghnazar, R., Zuidema, D., Lima, M.S., da Costa, A.R., 2017. A Multi-Objective Green UAV Routing Problem. Comput. Oper. Res. 88, 306–315.
Dell'Amico, M., Montemanni, R., Novellani, S., 2019. Drone-Assisted Deliveries: New Formulations for the Flying Sidekick Traveling Salesman Problem. Optimiz. Lett. 13, 1–32.
El-Adle, A.M., Ghoniem, A., Haouari, M., 2021. Parcel Delivery by Vehicle and Drone. J. Oper. Res. Soc. 72, 398–416.
Fischetti, M., Salazar González, J.J., Toth, P., 1998. Solving the Orienteering Problem through Branch-and-Cut. INFORMS J. Comput. 10, 133–148.
Gendreau, M., Laporte, G., Semet, F., 1998. A Branch-and-Cut Algorithm for the Undirected Selective Traveling Salesman Problem. Networks 32, 263–273.
Hà, M.H., Vu, L., Vu, D.M., 2020. The Two-Echelon Routing Problem with Truck and Drones. arXiv: 2004.02275.
Ha, Q.M., Deville, Y., Pham, Q.D., Hà, M.H., 2020. A Hybrid Genetic Algorithm for the Traveling Salesman Problem with Drone. J. Heurist. 26, 219–247.
Joerss, M., Schröder, J., Neuhaus, F., Klink, C., Mann, F., 2016. Parcel Delivery: The Future of Last Mile. Technical Report McKinsey & Company.
Kellermann, R., Biehle, T., Fischer, L., 2020. Drones for Parcel and Passenger Transportation: A Literature Review. Transp. Rese. Interdiscipl. Perspect. 4, 100088.
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.
Lee, H.L., Chen, Y., Gillai, B., Rammohan, S., 2016. Technological Disruption and Innovation in Last-Mile Delivery. Technical Report Value Chain Innovation Initiative.
Macrina, G., Di Puglia Pugliese, L., Guerriero, F., Laporte, G., 2020. Drone-Aided Routing: A Literature Review. Transp. Res. Part C: Emerg. Technol. 120, 102762.
Murray, C.C., Chu, A.G., 2015. The Flying Sidekick Traveling Salesman Problem: Optimization of Drone-Assisted Parcel Delivery. Transp. Res. Part C: Emerg. Technol. 54, 86–109.
Murray, C.C., Raj, R., 2020. The Multiple Flying Sidekicks Traveling Salesman Problem: Parcel Delivery with Multiple Drones. Transp. Res. Part C: Emerg. Technol. 110, 368–398.
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.
Paradiso, R., Roberti, R., Laganá, D., Dullaert, W., 2020. An exact solution framework for multitrip vehicle-routing problems with time windows. Oper. Res. 68, 180–198.
Park, Y., Nielsen, P., Moon, I., 2020. Unmanned Aerial Vehicle Set Covering Problem Considering Fixed-Radius Coverage Constraint. Comput. Oper. Res. 119, 104936.
Poikonen, S., Golden, B., 2020. Multi-Visit Drone Routing Problem. Comput. Oper. Res. 113, 104802.
Poikonen, S., Golden, B., Wasil, E.A., 2019. A Branch-and-Bound Approach to the Traveling Salesman Problem with a Drone. INFORMS J. Comput. 31, 335–346.
Roberti, R., Ruthmair, M., 2020. Exact methods for the traveling salesman problem with drone. Transp. Sci. 55 (2), 315–335.
Schermer, D., Moeini, M., Wendt, O., 2019. A Hybrid VNS/Tabu Search Algorithm for Solving the Vehicle Routing Problem with Drones and En Route Operations. Comput. Oper. Res. 109, 134–158.
Toksoz, T., Redding, J., Michini, M., Michini, B., How, J., Vavrina, M., Vian, J., 2011. Automated Battery Swap and Recharge to Enable Persistent UAV missions. In: Infotech@Aerospace 2011.
Ure, N., Chowdhary, G., Toksoz, T., How, J., Vavrina, M., Vian, J., 2015. An automated battery management system to enable persistent missions with multiple aerial vehicles. IEEE/ASME Trans. Mechatron. 20, 275–286.
Vásquez, S.A., Angulo, G., Klapp, M.A., 2021. An Exact Solution Method for the TSP with Drone Based on Decomposition. Comput. Oper. Res. 127, 105127.
Wang, X., Poikonen, S., Golden, B., 2017. The vehicle routing problem with drones: several worst-case results. Optimiz. Lett. 11, 679–697.