This is a pre print version of the following article:

Exact methods for the traveling salesman problem with multiple drones / Cavani, S., Iori, M., Roberti, R.. - In: TRANSPORTATION RESEARCH. PART C, EMERGING TECHNOLOGIES. - ISSN 0968-090X. - 130:(2021), pp. 103280-103280. [10.1016/j.trc.2021.103280]

Terms of use:

The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

11/11/2026 17:11

(Article begins on next page)

Exact Methods for the Traveling Salesman Problem with Multiple Drones

Sara Cavani

Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, 42122 Reggio Emilia, Italy
Department of Operations Analytics, Vrije Universiteit Amsterdam, 1081HV Amsterdam, the Netherlands
217312@studenti.unimore.it

Manuel Iori

Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, 42122 Reggio Emilia, Italy
manuel.iori@unimore.it

Roberto Roberti

Department of Operations Analytics, Vrije Universiteit Amsterdam, 1081HV Amsterdam, the Netherlands
r.roberti@vu.nl

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.

Keywords: traveling salesman; drone; exact method; branch-and-cut; mixed integer linear programming

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

[1] http://www.dpdhl.com/en/media-relations/specials/dhl-parcelcopter.html

project[2]. Through its Prime Air system,[3] Amazon is also working on a system that can deliver parcels in less than 30 minutes. 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. 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 & Chu (2015). In the last type of problems, where drones work in tandem with trucks, drones and trucks are synchronized; trucks both visit customers to deliver parcels but also act as moving stations and recharging facilities, where drones can take off and land to pick up and deliver parcels to serve their own customers and can recharge their batteries (see, e.g., Murray & Chu (2015), Agatz et al. (2018), Poikonen et al. (2019), Ha et al. (2020)).

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),

[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/

Hà et al. (2020), Murray & Raj (2020), Poikonen & 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). Figure 1 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.

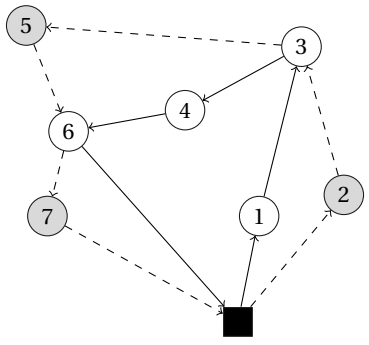


Figure 1: A feasible solution of a TSP-MD instance with seven customers

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. 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. Section 4 presents the compact MILP formulation for the TSP-MD. In Section 5, we present several families of valid inequalities for the MILP presented in Section 4. In Section 6, we describe an exact decomposition approach, based on the MILP and the valid inequalities presented in Sections 4 and 5, to solve the TSP-MD to proven optimality. 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 & 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). 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), El-Adle et al. (2019), Poikonen et al. (2019), Roberti & Ruthmair (2020), and Vásquez et al. (2020). All of these contributions 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 & 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 & 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 minutes.

El-Adle et al. (2019) 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.

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 & 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.

Vásquez et al. (2020) 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.

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 & Raj (2020), and Poikonen & Golden (2020). We revise 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 owing 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 & 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 & 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 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

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 $V = N \cup \{0, 0'\}$, 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^{T}_{ij}, and the time the drones take to traverse arc $(i, j) \in A$ is indicated by t^{D}_{ij}. We assume the drones to be faster than the truck, so $t^{\mathrm{D}}_{ij} \leq t^{\mathrm{T}}_{ij}$ 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^{T}_{ij}, and drone travel times, t^{D}_{ij}. 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 $(j, i) \in A$, $t^{\mathrm{T}}_{ij} = t^{\mathrm{T}}_{ji}$ and $t^{\mathrm{D}}_{ij} = t^{\mathrm{D}}_{ji}$. 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 drones and vehicle 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.

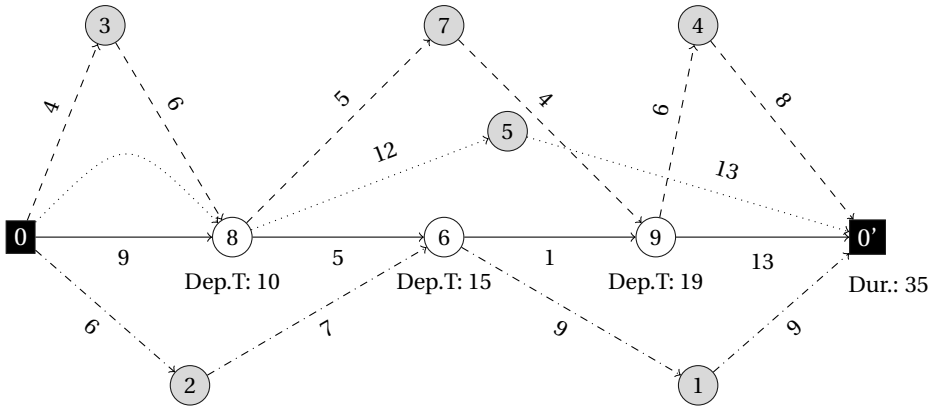


Figure 2: A feasible solution of a TSP-MD instance with nine customers and three drones

To clarify the definition of the TSP-MD and show how to compute the duration of a route while considering synchronization among vehicles, Figure 2 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^{T}_{ij}. 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 arc travel times, t^{D}_{ij}. 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.

4. Compact Formulation

In this section, we introduce a compact MILP to formulate the TSP-MD. This MILP is the basis of the exact approach that is presented in Section 6.

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 Figure 2, 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:

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

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

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

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

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

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

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

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

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

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

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

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

$$a_i + (M + t^{\mathrm{T}}_{ij})x_{ij} + \sum_{\langle k,i,j\rangle\in\mathscr{L}} \Big(M + \max\big\{t^{\mathrm{T}}_{kj} - t^{\mathrm{D}}_{ki}, t^{\mathrm{D}}_{ij}\big\}\Big) z_{kij}$$
$$+ \sum_{\langle i,j,k\rangle\in\mathscr{L}} (M + t^{\mathrm{D}}_{ij}) z_{ijk} + \sum_{\langle i,k,j\rangle\in\mathscr{L}} \max\big\{t^{\mathrm{D}}_{ik} + t^{\mathrm{D}}_{kj} - t^{\mathrm{T}}_{ij}, 0\big\} \frac{z_{ikj}}{m} \leq a_j + M \qquad (i,j) \in A \tag{1m}$$

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

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

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

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

$$w_i \in \mathbb{Z}_+ \qquad i \in V \qquad (1r)$$

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 $y_i \geq \sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} + \sum_{\langle k,j,i\rangle \in \mathscr{L}} z_{kji}$, 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 $y_i \geq \sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} + \sum_{\langle k,j,i\rangle \in \mathscr{L}} z_{kji}$.

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

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

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

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 $\boldsymbol{x}(\hat{A})$ the sum of the x-variables of the arcs belonging to the set $\hat{A} \subseteq A$, i.e., $\boldsymbol{x}(\hat{A}) = \sum_{(i,j)\in\hat{A}} x_{ij}$. Finally, let $\boldsymbol{y}(S)$ be the sum of the y-variables of the customers of the set $S \subseteq N$, i.e., $\boldsymbol{y}(S) = \sum_{i\in S} y_i$.

5. 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).

5.1. 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 $n^{\mathrm{T}} = \left\lceil \frac{n-m}{m+1} \right\rceil$. The following inequality forces the truck to serve at least n^{T} customers and is valid for (1):

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

5.2. 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 $n^{\mathrm{T}} = 1$

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

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

  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 $n^{\mathrm{T}} \geq 2$

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

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

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. (2020).

5.3. *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):

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

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

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 $(i^*, j^*) \in A$. Figure 2 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., $\mathscr{L}_{ij} = \{\langle i,k,j\rangle \in \mathscr{L} \mid t^{\mathrm{D}}_{ik} + t^{\mathrm{D}}_{kj} > t^{\mathrm{T}}_{ij}\}$. The following $O(|A|+1)$ inequalities can be added to (1) to lift (6a):

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

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

where M_{ij} is a large enough value that can be computed, for example, as $M_{ij} = \max_{k\in N}\{t^{\mathrm{D}}_{ik} + t^{\mathrm{D}}_{kj} - t^{\mathrm{T}}_{ij}\}$. 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 $|\mathscr{L}_{ij}| \geq 1$).

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

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

inequality (7b) can be lifted as follows

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

$$(i,j) \in A : |\mathscr{L}_{ij}| \geq 1 \tag{8}$$

5.4. 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

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

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

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., $y_i = 1$).

5.5. *Relationship between* $\boldsymbol{x}(V)$ *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)

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

5.6. *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)

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

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

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.

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

Proof. 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 $\sum_{\langle s,r,j\rangle \in \mathscr{L}: s \neq i,0} z_{srj} + z_{0ij} \leq m \cdot y_j$. 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.□

5.7. 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 Figure 3. 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, $n^{\mathrm{T}} = \left\lceil \frac{n-m}{m+1} \right\rceil = 3$. 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 $x_{02} + x_{23} + x_{30'} \leq y_2 + y_3$, which can be lifted as $x_{02} + x_{03} + x_{23} + x_{32} + x_{20'} + x_{30'} \leq y_2 + y_3$.

In general terms, TSPC can be described as follows:

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

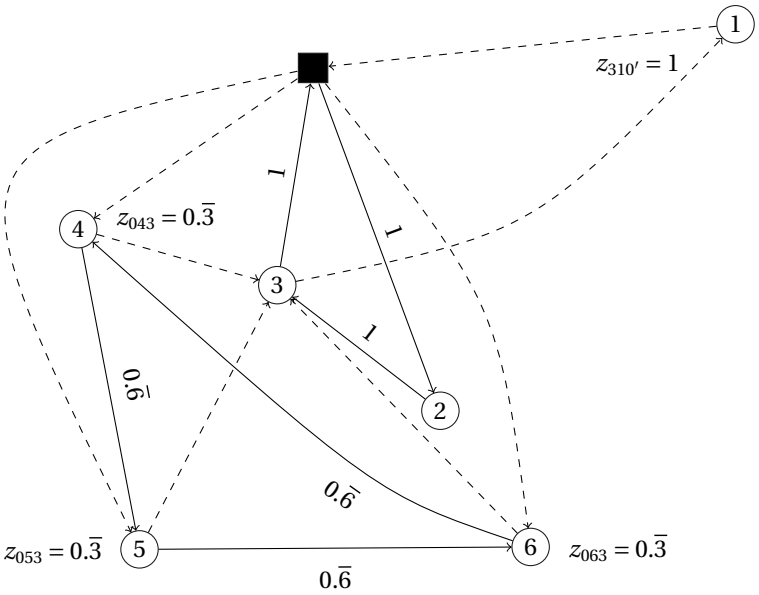


Figure 3: An example of a fractional solution featuring a violated Too-Short Path cut for an instance with six customers ($n = 6$) and a single drone ($m = 1$). Solid lines represent strictly positive x_{ij} variables. Dashed lines represent strictly positive z_{ijk} variables. The violated Too-Short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

5.8. *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

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

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)

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

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

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

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

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

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

Second, we show the correctness of constraints (14b). Because $|S| > \overline{n}^{\mathrm{D}}$, then at least one of the customers of the set S must be served by the truck. Therefore, $\boldsymbol{y}(S)$ cannot be lower than 1 in any feasible solution. For any feasible TSP-MD solution, $\boldsymbol{y}(S) - 1$ 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. □

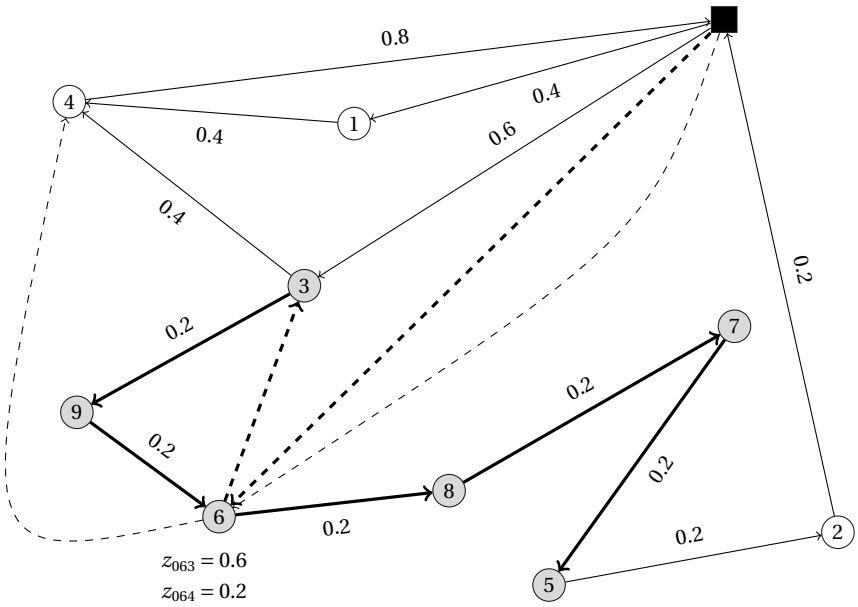


Figure 4: An example of a fractional solution featuring a violated LGSEC (14a) for an instance with nine customers ($n = 9$) and three drones ($m = 3$). The violated LGSEC corresponds to the thick straight lines, the thick dashed lines, the set $S = \{3,5,6,7,8,9\}$, and $r = 6$

Figure 4 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

$$\underbrace{x_{39} + x_{96} + x_{68} + x_{87} + x_{75}}_{1} + \underbrace{z_{063}}_{0.6} \not\leq \underbrace{y_3 + y_5 + y_6 + y_7 + y_8 + y_9}_{1.6} - \underbrace{y_6}_{0.2}$$

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.

5.9. 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 Figure 5, 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:

$$\underbrace{x_{12} + x_{21}}_{1} + \underbrace{\frac{1}{m}(z_{130'} + z_{250'} + z_{146})}_{\frac{1}{2}(1+1+0.5)=1.25} \not\leq \underbrace{y_1 + y_2}_{2}$$

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

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

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

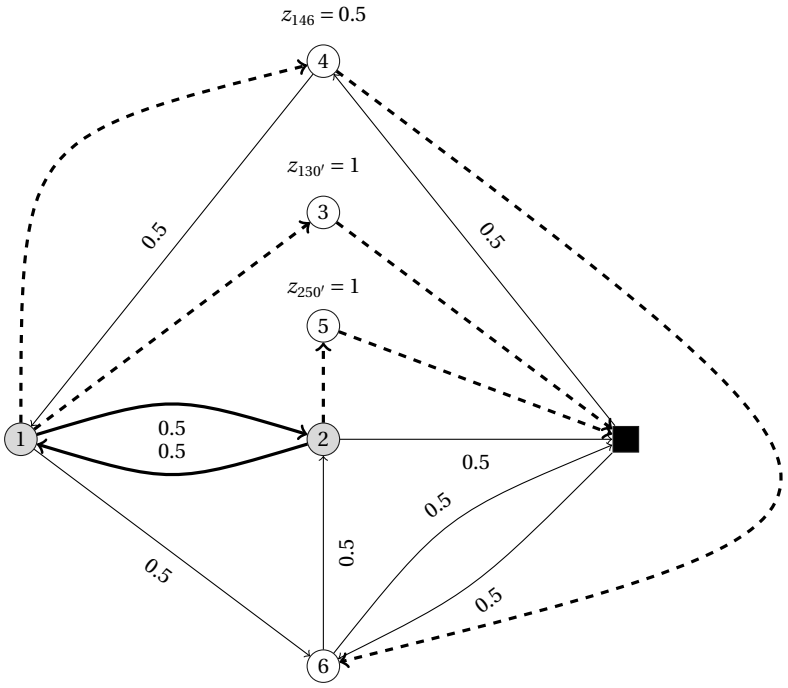


Figure 5: An example of a fractional solution featuring a violated MODL cut for an instance with six customers ($n = 6$) and two drones ($m = 2$). The violated MODL cut corresponds to the thick solid lines, the thick dashed lines, and the set $S = \{1, 2\}$

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

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

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

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

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

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.

5.10. Separation Procedures

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

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., $S \subseteq N_{0.05}(\hat{\boldsymbol{y}})$); 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 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 $S \subseteq N_{0.05}(\hat{\boldsymbol{y}})$. 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) $1 \leq |S| \leq 3$, and (b) $S \subseteq N_{0.05}(\hat{\boldsymbol{y}})$. MODLC (15)-(17b) are separated only if no violated LGSEC (14) was found, and the most violated MODLC is added (if any).

6. 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 5 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

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

In our computational experience, most of the times t^* coincides with $t^*(m)$. Therefore, in the following (see 6.3), we describe an exact method that first computes $t^*(m)$ by solving a MILP similar to (1) (see Section 6.1) 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 6.2).

6.1. Computing $t^*(m)$

Vásquez et al. (2020) 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^T_{ij}, and drone travel times, t^D_{ij}, satisfy the triangle inequality; (ii) the truck is not faster than the drones, i.e., $t^T_{ij} \leq t^D_{ij}$ for each $(i, j) \in A$; and (iii) each drone serves at least two customers.*

Proof. We prove Proposition 3 by showing the example displayed in Figure 6 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 $t^{\mathrm{D}}_{06} \leq t^{\mathrm{D}}_{01} + t^{\mathrm{D}}_{16} \leq t^{\mathrm{T}}_{01} + t^{\mathrm{D}}_{16}$, $t^{\mathrm{D}}_{63} \leq t^{\mathrm{D}}_{62} + t^{\mathrm{D}}_{23} \leq t^{\mathrm{D}}_{62} + t^{\mathrm{T}}_{23}$, and

$t^{\mathrm{D}}_{70'} \le t^{\mathrm{D}}_{75} + t^{\mathrm{D}}_{50'} \le t^{\mathrm{D}}_{75} + t^{\mathrm{T}}_{50'}$, 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.

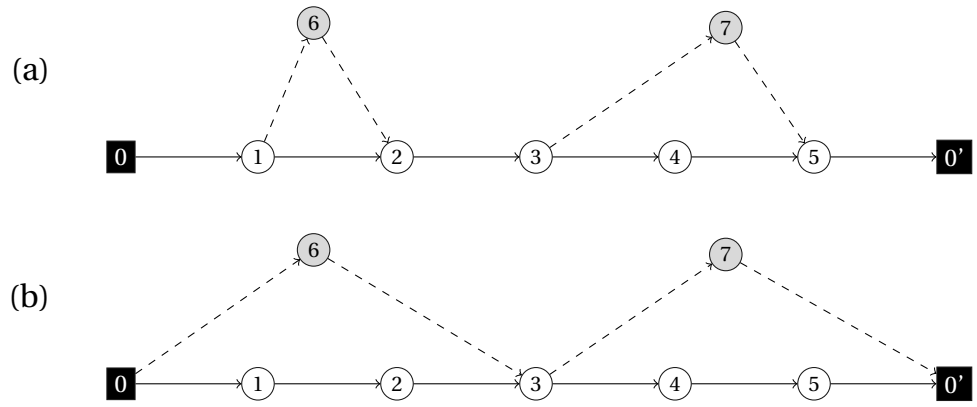


Figure 6: Example of a feasible TSP-MD solution where the drones traverse some arcs on the truck (panel (a)) and the corresponding solution where the drones are always airborne when the truck is moving (panel (b))

In general, 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). □

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

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

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

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

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

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:

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

Notice that all valid inequalities introduced in Section 5 are still valid and can be added to (19).

6.2. *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:

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

$$\text{s.t. } (1b)-(1d),\ (1f),\ (1l)-(1q),\ (18d)$$

$$y_j + \sum_{\langle i,j,k\rangle \in \mathscr{L}} z_{ijk} \le 1 \qquad j \in N \tag{20b}$$

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

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

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

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 5, 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 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 $n^{\mathrm{T}}(\gamma) = \left\lceil \frac{n-m}{\gamma+1} \right\rceil$; thus, constraint (3) is replaced by $\boldsymbol{y}(N) \ge n^{\mathrm{T}}(\gamma)$. 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 Figure 7.

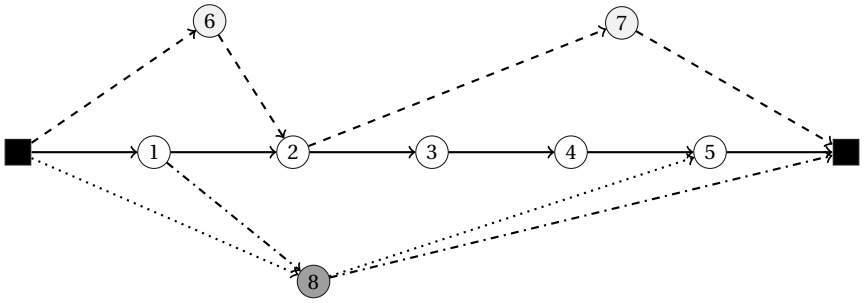


Figure 7: Example of the optimality check on an optimal solution of formulation (20)

The top part of Figure 7 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 $t^{\mathrm{D}}_{08} + t^{\mathrm{D}}_{85} \le a_5^*$, $t^{\mathrm{D}}_{18} + t^{\mathrm{D}}_{80'} \le lb(\gamma) - a_1^*$, $t^{\mathrm{D}}_{08} + t^{\mathrm{D}}_{85} + t^{\mathrm{T}}_{50'} \le lb(\gamma)$ 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 $(\boldsymbol{x}^*, \boldsymbol{y}^*, \boldsymbol{z}^*, \boldsymbol{a}^*)$ 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

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

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

$$\max\left\{ t^{\mathrm{D}}_{0i} + t^{\mathrm{D}}_{ii_\ell} - \overline{a}^*_{i_\ell},\ t^{\mathrm{D}}_{i_f i} + t^{\mathrm{D}}_{i0'} - lb(\gamma) + \underline{a}^*_{i_f}, \right\} \le 0,$$

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

6.3. 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 $\boldsymbol{y}(N) = n - 2 \cdot m$ and the valid inequalities described in Section 5 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 5 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 5 and 6.2. If the corresponding optimal solution does not pass the optimality check described in Section 6.2, 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

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

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 5, 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.

7. Computational Results

In this section, we discuss the computational results achieved by the exact solution method described in Section 6, provide managerial insights, and report a sensitivity analysis on key parameters and components of the proposed solution method. 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 insight to investigate the importance of the drone speed. Section 7.4 compares the performance of our exact solution method with the performance of the compact formulation (1) presented in Section 4. Section 7.5 shows how TSPC (12), LGSEC (14), and MODLC (15)-(17b) affect the performance of the solution method.

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^{T}_{ij}, according to the Manhattan metric (i.e., $t^{\mathrm{T}}_{ij} = \lceil |x_i - x_j| + |y_i - y_j| \rceil$) and drone travel times, t^{D}_{ij}, according to the Euclidean metric (i.e., $t^{\mathrm{D}}_{ij} = \sqrt{\lceil ((x_i - x_j)^2 + (y_i - y_j)^2)\alpha \rceil}$), 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 $\alpha = 0.333$ 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 6 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, over the instances not solved to proven optimality, of the best lower bound computed at Step 2, 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 $\%\mathsf{T}_{\mathsf{sep}}$ 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: 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	1 809	41.0	26	0.0	7.0	25.2	239	145	679	1 783	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	1 787	62.8	28	0.0	7.7	29.8	329	231	783	1 750	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	1 370	51.4	5	0.0	14.5	32.3	393	51	555	1 340	1	0	26	0	0
24	1	13.1	4 734	72.6	23	0.0	11.3	38.0	356	137	965	4 662	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 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 $\alpha = 0.333$ (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: Computational Results on the Instances with $\alpha = 0.333$

	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	2 027	80.0	38	0.0	9.2	33.5	281	160	647	1 984	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	1 583	110.1	30	0.0	10.3	35.0	270	78	429	1 533	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 2 shows that 109 instances can be solved to optimality when $\alpha = 0.333$, 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 $\alpha = 0.333$, 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.

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.

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	9	1.7	3 933	11.1	7	0.0	8.2	21.8	452	116	842	3 926	35	0	0	0	0
24	0	8.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
All	29	7.5	1 242	6.2	2	0.0	8.5	21.5	176	53	352	1 240	16	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

In Figure 8, we show how the average optimal solution cost changes when the number of available drones and their speed change. In the left panel of Figure 8, we compare the average solution cost over the 9-customer instances solved to optimality for any combination of values of $\alpha = 0.333, 0.5, 1$ and $m = 1, 2, 3$; values are ratios over the average cost with $m = 1$ and $\alpha = 1$. A similar comparison is provided in the right panel for instances with 14 customers.

Figure 8 shows that the higher the number of drones and their speed, the lower the optimal solution costs. When $\alpha = 0.5$, the optimal solution cost decreases by 17 to 26% compared to when $\alpha = 1$, and, when $\alpha = 0.333$, there is a further decrease of 6 to 10%. Having two drones instead of one allows to decrease the solution cost by 13 to 21%, and adding a third drone allows to further decrease it by 9-21%. In terms of solution cost, having a single fast drone (with $\alpha = 0.333$) 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 4, when solved with `CPLEX` with the addition of all the valid inequalities presented in in Section 5, with the performance of the exact decomposition method (hereafter called Dec) presented in Section 6.

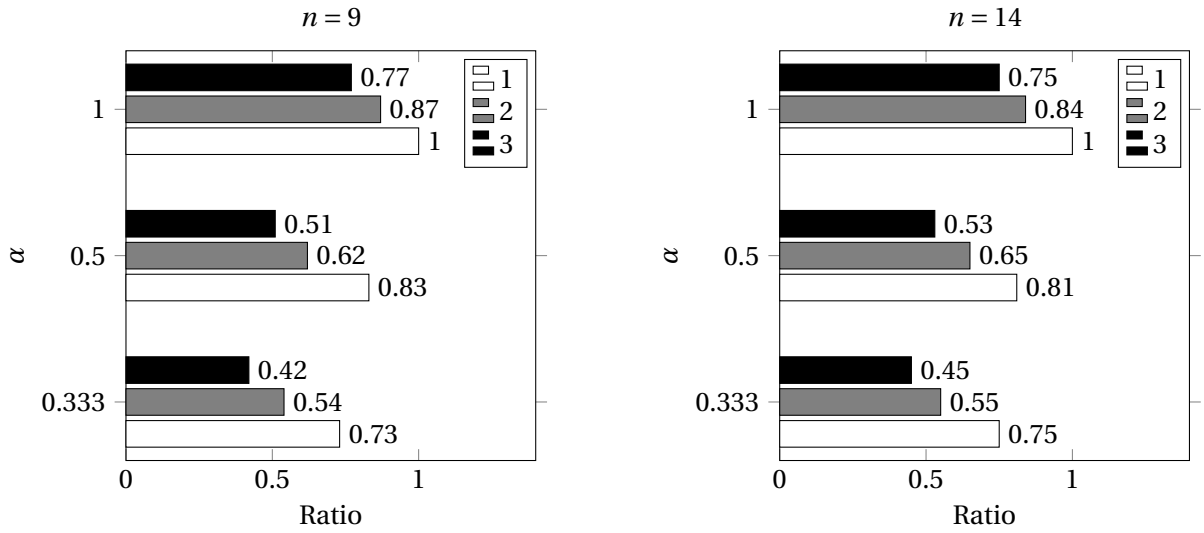


Figure 8: Ratios between Average Solution Costs for Different Values of α (1, 0.5, 0.333) and m (1, 2, 3)

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: 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	1 157	1 157	8	815	414
24	2	1 729	1 729	2	1 787	1 787

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. These results indicate that Dec, on average, outperforms CF.

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: 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	1 096	8	5.1	1 071	4	6.5	2 884
24	2	6.8	1 787	1	15.2	2 856	2	6.6	1 403	1	17.7	4 037
All	30	5.8	361	27	13.0	373	30	6.3	411	25	13.2	691

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.

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.

Our study has been aimed at investigating the implications and challenges of having multiple drones in a truck-and-drones distributing setting, so we have focused our attention on a *neat* problem, without considering side constraints that arise in real-life applications and have been recently explored in the scientific literature. We expect that the proposed solution method can be adjusted to handle many side constraints, such as the flying range of the drones (possibly weight-dependent), the unavailability of customers to be served by drones, launch and rendezvous times for the drones to take off and land, etc. As many of such side constraints could be modeled by properly defining the set of feasible drone legs and the number of variables of the formulation we have proposed would be limited by these side constraints, we are confident that our exact solution method could provide comparable performance to those illustrated in our paper on instances of similar size.

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. 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.

References

Agatz, N., Bouman, P., & Schmidt, M. (2018). Optimization Approaches for the Traveling Salesman Problem with Drone. *Transportation Science, 52*, 965–981.

Baldacci, R., Mingozzi, A., & Roberti, R. (2011). New Route Relaxation and Pricing Strategies for the Vehicle Routing Problem. *Operations Research, 59*, 1269–1283.

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

Chung, S. H., Sah, B., & Lee, J. (2020). Optimization for Drone and Drone-truck Combined Operations: A Review of the State of the Art and Future Directions. *Computers & Operations Research, 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. *Computers & Operations Research, 88*, 306–315.

Dell'Amico, M., Montemanni, R., & Novellani, S. (2019). Drone-Assisted Deliveries: New Formulations for the Flying Sidekick Traveling Salesman Problem. *Optimization Letters, 13*, 1–32.

El-Adle, A. M., Ghoniem, A., & Haouari, M. (2019). Parcel Delivery by Vehicle and Drone. *Journal of the Operational Research Society, Forthcoming*.

Fischetti, M., Salazar González, J. J., & Toth, P. (1998). Solving the Orienteering Problem through Branch-and-Cut. *INFORMS Journal on Computing*, *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. *Journal of Heuristics*, *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. *Transportation Research Interdisciplinary Perspectives*, *4*, 100088.

Kitjacharoenchai, P., Min, B.-C., & Lee, S. (2020). Two Echelon Vehicle Routing Problem with Drones in Last Mile Delivery. *International Journal of Production Economics*, *225*, 107598.

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. *Transportation Research Part C: Emerging Technologies*, *120*, 102762.

Murray, C. C., & Chu, A. G. (2015). The Flying Sidekick Traveling Salesman Problem: Optimization of Drone-Assisted Parcel Delivery. *Transportation Research Part C: Emerging Technologies*, *54*, 86–109.

Murray, C. C., & Raj, R. (2020). The Multiple Flying Sidekicks Traveling Salesman Problem: Parcel Delivery with Multiple Drones. *Transportation Research Part C: Emerging Technologies*, *110*, 368–398.

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. *Operations Research*, *68*, 180–198.

Park, Y., Nielsen, P., & Moon, I. (2020). Unmanned Aerial Vehicle Set Covering Problem Considering Fixed-Radius Coverage Constraint. *Computers & Operations Research*, *119*, 104936.

Poikonen, S., & Golden, B. (2020). Multi-Visit Drone Routing Problem. *Computers & Operations Research*, *113*, 104802.

Poikonen, S., Golden, B., & Wasil, E. A. (2019). A Branch-and-Bound Approach to the Traveling Salesman Problem with a Drone. *INFORMS Journal on Computing*, *31*, 335–346.

Roberti, R., & Ruthmair, M. (2020). Exact Methods for the Traveling Salesman Problem with Drone. *Transportation Science*, *Forthcoming*.

Schermer, D., Moeini, M., & Wendt, O. (2019). A Hybrid VNS/Tabu Search Algorithm for Solving the Vehicle Routing Problem with Drones and En Route Operations. *Computers & Operations Research*, *109*, 134–158.

Vásquez, S. A., Angulo, G., & Klapp, M. A. (2020). An Exact Solution Method for the TSP with Drone Based on Decomposition. *Computers & Operations Research*, .