Modeling the Flying Sidekick Traveling Salesman Problem with Multiple Drones

SPECIAL ISSUE: NEW NETWORK MODELS AND APPROACHES FOR LOGISTICS AND TRANSPORTATION

Networks

Modeling the Flying Sidekick Traveling Salesman Problem with Multiple Drones

Mauro Dell'Amico[1] | **Roberto Montemanni**[1] | **Stefano Novellani**[1]

[1]DISMI, University of Modena and Reggio Emilia, Reggio Emilia, 42122, Italy

Correspondence
Stefano Novellani, DISMI, University di Modena and Reggio Emilia, Reggio Emilia, 42122, Italy
Email: stefano.novellani@unimore.it

Funding information

This paper considers a version of the flying sidekick traveling salesman problem with multiple drones in which a set of customers can be served with either a truck or a set of homogeneous drones, whose flights must respect a battery endurance. Each flight is made of a launch, a service to a customer, and a return: launch and return must happen at the customers or at the depot. These operations require time and when multiple drones are launched and/or collected at the same node, their order becomes relevant. The proposed model takes into account the order of these operations as a scheduling problem, because ignoring it could lead to infeasible solutions in the reality due to the possible exceeding of drones endurance. We propose a set of novel formulations for the problem that can improve the size of the largest solved instances in the literature. We provide a comparison among the formulations, between the multiple drones solutions and the single drone ones, and we compare different variants of the original model.

KEYWORDS
UAV, aerial drones, formulations, MILP, branch-and-cut, parcel delivery, covid19

1 | INTRODUCTION

Retail e-commerce sales worldwide from 2014 to 2019 have doubled from 1,336 to 3,535 billion US dollars and are forecasted to almost double once more to 6,542 billions in 2023 (see e.g. [34]). The e-commerce boom requires new investment and ideas in parcel delivery, indeed a recent Boston Consulting Group study revealed that parcel and express delivery startups founding have increased 20 times between 2014 and 2016 (see, e.g. [38]). Another recent study, this one by McKinsey (see, e.g. [18]), shows the relevance of ever fast delivery request: 20 to 25% of customers are willing to pay more for same day delivery and 2% for instant delivery. The same study confirms that this request could be fulfilled by autonomous vehicles and forecasts that autonomous vehicles will deliver 80% of all parcels in the next ten years. Among autonomous vehicles one could find aerial drones, also called Unmanned Aerial Vehicles (UAV) or, in the following, just drones. The most important e-commerce and parcel delivery companies have considered, implemented, or are willing to implement parcel delivery via drones. We can mention Alibaba, Alphabet, Amazon, DHL, and JD.com, among the others. Moreover, one should not forget the boom of food delivery; indeed, Uber Eats is going to test the use of drones for at least one portion of the food delivery process in summer 2020 in San Diego (see e.g. [3]). During the 2020 Covid-19 outbreak many drone delivery services have been considered to increase social distancing, in particular for medical, parcel, or grocery delivery in order to avoid centralizing the services and to avoid people meeting at the distribution centers, or to deliver needed goods and medications to quarantined people (see e.g. [33]). We report a few examples: Drone Delivery Canada is looking for Covid-19 use cases for drone applications (see e.g. [2]); Zipline wants to expand to the US their expertise built in Africa for blood and medical supply (see e.g. [5]); and robots have being used to deliver grocery in Washington D.C. (see e.g. [1]).

The interest of the commercial sector justifies the increasing interest of researchers in the operational use of drones. This leads to studies devote to optimization problems accounting for the use of drones, such as this paper that studies a version of the Flying sidekick traveling salesman problem with multiple drones (MFSTSP). The MFSTSP considers the combined used of a truck and a set of homogeneous drones to deliver parcels so to diminish the completion time of the delivery operations. Each drone can leave the truck (or the depot) to serve a customer at a time and must return to the truck (or the depot). Multiple drones can take off and return in a certain point and the scheduling of these operations is included in our model.

The contribution of the paper is the following:

- We propose a set of novel formulations for the MFSTSP and a comparative study, including valid inequalities.
- We show the possible advantages of using multiple drones for delivery activities.
- We study how our model works on different versions of the problem and we provide an extensive comparison among the several versions.

In the reminder of the paper we provide a review on recent literature in Section 2. The problem is formally defined in Section 3 and four formulations are presented in Section 4. Section 5 shows the implementation details and all the numerical results and comparisons. Section 6 concludes the paper.

2 | RELATED LITERATURE

Drones are an emerging technology and, as we have seen, their use in day-to-day operations is becoming more and more common; therefore, it is not surprising that many recent works have been focused on the use of drones. In

particular, the amount of literature that considers optimization problems related to drones, and trucks and drones has increased remarkably in the last few years. A recent survey by Otto et al. [23] reports more than 300 papers and classify them in four categories: (i) Vehicles supporting operations of drones; (ii) Drones supporting operations of vehicles; (iii) Drones and vehicles performing independent tasks; (iv) Drones and vehicles as synchronized working units. The MFSTSP belongs to the last category and in particular refers to a single truck and multiple drones subcategory.

We consider, from now on, the category *Drones and vehicles as synchronized working units*, that includes routing problems in which trucks are equipped with drones, both vehicles can be used to deliver packages to customers, and synchronization is important. We firstly examine the case with one truck and one drone and subsequently the cases that treat multiple trucks and/or drones.

2.1 | Single truck and drone

The first appeared problem involving the use of one truck and one drone is the *flying sidekick traveling salesman problem* (FSTSP), that was firstly defined and solved by Murray and Chu [21]. In the FSTSP the truck and the drone cooperate to serve all customers, the drone can leave the truck to serve one customer and returns to the truck when this is stationary. The truck and the drone must be synchronized and thus wait for each other, but no scheduling is needed, because only one drone is used. The objective function is to minimize the completion time. In the FSTSP, customers can be serviced only once by the truck or by the drone; however, some customers can be visited only by the truck, because their request cannot be fulfilled by the drone. In this form of the problem the drones cannot return at the launching point and each drone flight is limited by a finite endurance. In their work, Murray and Chu propose a mixed integer linear programming (MILP) formulation and three heuristic methods based on the nearest neighbor, savings, and sweep algorithms, modified to include the drone flights in the solution.

The *TSP with drone* (TSP-D), presented by Agatz et al. [6] for the first time, shares the main characteristics with the FSTSP. However, in the TSP-D: the customers can be visited more than once by the truck if it is convenient for launching and collecting a drone; the drone can return to the same place it has been launched; endurance is unlimited and launching and rendezvous times are considered negligible. The authors present an integer linear programming model and route first-cluster second heuristics. Bouman et al. [7] extend the work by Agatz et al. [6]: they solve the TSP-D with dynamic programming, that is also use to define a heuristic algorithm.

Ha et al. [15] solve a FSTSP for which they propose two heuristic algorithms: a route first-cluster second and a cluster first-route second. In [14], the same authors solve a similar problem with a different objective function, made of four cost components that depend on the total distance traveled by the truck, the one travelled by the drone, and the waiting time of the truck and the drone. They present a MILP formulation based on Murray and Chu's one and two heuristics. Ponza's thesis [28] tackles the FSTSP proposing a modified MILP formulation with respect to the Murray and Chu's [21] one and solve it with a simulated annealing algorithm. The Ponza's formulation does not allow the drone to wait on the ground at customers nodes to save battery, so the time spent in waiting for the truck is included in the computation of the used energy. De Freitas and Penna [10] propose a randomized variable neighborhood descent for the FSTSP. In [11], the same authors propose a hybrid general variable neighborhood search algorithm for the FSTSP proposed by Murray and Chu [21] and the TSP-D proposed by Agatz et al. [6]. Poikonen et al. [26] propose a branch-and-bound (B&B) for a TSP-D version that considers a maximum endurance and drone flights that can start end end in the same node, but not multiple visits by the truck. They also propose three heuristic algorithms, two derived from the B&B and the third that is a divide-and-conquer one. Yurek and Ozmutlu [39] propose an iterative algorithm based on a decomposition approach for the FSTSP. Dell'Amico et al. [13] solve two variants of the FSTSP, one where the drones can wait on the ground at customer nodes and the waiting time is not included in the drone

flight and one in which the drone can wait only if hovering. They propose two MILP formulations that improve the Murray and Chu's one, one where flights are represented with three-index variables and one in which the flights are represented with two-index variables. The authors consider an objective function that is equivalent to the Murray and Chu's one, but that provides higher lower bounds, they also propose a set of inequalities to be separated in a branch-and-cut algorithm in order to be able to decrease the large number of variables and constraints needed to represent synchronization. The same authors [12] propose improved formulations that could solve larger instances and a matheuristc for the same problem.

2.2 | Single truck and multiple drones

Phan et al. [35] solve the *traveling salesman problem with multiple drones*, a problem in which one truck is equipped with multiple drones whose aim is to serve all the customers either with one vehicle or the others, in order to minimize the transportation costs of both the truck and the drones. Each node must be visited once and the drones cannot return to their launching point. The problem includes a battery endurance but the drones can wait for the truck on the ground and thus that time is not included in the endurance computation, as opposed to the MFSTSP consdiered in this paper. The authors also impose a maximum time for which drones and truck can wait each other. They adapt the heuristic algorithm proposed by Ha et al. [14] and propose an adaptive large neighborhood search that is proved being more efficient. They heuristically solve instances with up to 100 customers and at most three drones.

Karak and Abdelghany [19] present the *hybrid vehicle-drone routing problem for pickup and delivery services*, a problem that considers a single truck equipped with multiple drones that can serve more than one customer during their flight. Each node can be visited by the vehicle only once, while the drones can return in their launching node. Customers are served only by drones, the truck is used only as drone depot. Both pickups and deliveries are considered, so as weight capacity and maximum endurance. The authors propose a MILP formulation and a Clarke and Wright algorithm (see, e.g., [8]) extended for the proposed problem. They solve exactly instances with six and eight customers and three stations within 1 hour and heuristically with up to 100 customers and 24 stations.

Murray and Raj [22] propose the *multiple flying sidekick traveling salesman problem* (also called MFSTPS) for parcel delivery with multiple drones, which is an extension of the FSTSP including multiple drones. The problem considers one truck and an heterogeneous fleet of drones, that may have different speed, capacity, service times, and flight and endurance limitations. Endurance is a function of the payload weight, of the travel distance without carrying a parcel and the travel distance with a parcel; however, the authors also consider versions with linear and fixed endurance (e.g. a maximum time or a maximum distances). The scheduling of arrivals and departures of drones is considered and also a service time in a node. Only a set of customers can be served with drones, also depending on the type of drone and a drone cannot go back to its launching location. In the original version of the problem the drones cannot wait on the ground at the customer nodes. They also study two additional versions of the problem: one in which the drone can leave the depot even if the truck is not there anymore and another in which launches and rendezvous do not need any intervention from the driver. The authors propose a MILP for the problem solved directly with the Gurobi solver. They also propose a three-phase iterative heuristic: in the first phase the customers are divided into two groups, one to be served by the truck and one by the drones, a TSP is solved heuristically to determine the truck route. In the second phase drone flights to serve the other customers are generated. In the third phase the timing of the activities is determined and local search procedures are used to improve the solution. The heuristic is tested on instances with up to 100 customers and four drones, while the MILP could solve to optimality 220 out 360 generated instances with up to eight customers.

Seifried [32] propose the *traveling salesman problem with multiple drones*. All the customers can be served either by

the truck or by a set of homogeneous drones. Similarly to Murray and Raj, the drones can be launched and retrieved only at the customers or depot and can serve only one customer at a time and they cannot return to their launching point. Drones are assumed to be at least as fast as the truck. In this version the truck can wait at the nodes without consuming energy and no scheduling is considered at the launching and rendezvous at every node. The author present a MILP model that solves instances with up to 10 customers and consider with up to nine drones.

Peng et al. [25] consider a two-echelon type of problem, in which the truck start and ends at the depot, and stops in anchor points where the drones are launched to serve customers. The decision of the anchor points and the routing decisions are part of the decision process. Drones can serve multiple customers, and if the battery is finished before all customers are served, they must return to the truck to recharge its battery. The aim is to minimize the overall delivery time. They present a hybrid genetic algorithm for the problem. Hu et al. [17] solve a similar two-echelon problem for which they present an iterative heuristic. In Hu et al. [16] the authors solve a problem where drones are used for inspection and not delivery. This means that the drones can visit multiple customers in their sorties. They propose heuristic algorithms.

2.3 | Multiple trucks and drones

Multiple trucks and drones problems have been treated and we report here a brief selection. Wang et al. [36] define the *vehicle routing problem with drones* (VRPD), Poikonen et al. [27] treat thets these problems theoretically, Daknama and Kraus [9] heuristically, Sacramento et al. [30] propose a MILP and a adaptive large neighborhood search for the VRPD, Schermer et al. [31] propose a formulation and a matheuristic for the VRPD, Kitjacharoenchai et al. [20] a MILP formulation and an adaptive insertion heuristic. Wang and Sheu [37] solve a VRP with drones, in which each drone can serve multiple customers during one flight because the parcels are delivered by means of parachutes. Di Puglia Pugliese and Guerriero [29] propose and solve the *VRP with time windows in which each truck is equipped with drones*.

3 | PROBLEM DESCRIPTION

The MFSTSP is a multiple drone version of the FSTSP originally defined by Murray and Chu [21] and thus its inherits the NP-hardness. The MFSTSP is the problem of serving a set of customers $C = \{1, \dots, c\}$ with either a truck or a drone. The truck starts from the depot 0, returns to the final depot $c+1$, and is equipped with a set D of homogeneous flying drones that can be used to serve one customer at a time, in parallel to the truck. A drone service is called *sortie*, defined by a launching node, a served customer, a and rendezvous node. All customers of C can be served by the truck, but only a subset $C' \subseteq C$ can be served by a drone with a sortie. The problem is built on a digraph $G = (N, A)$, where the set $N = \{0, 1, \dots, c+1\}$ represents all the nodes, while we define $N_0 = \{0, 1, \dots, c\}$ and $N_+ = \{1, \dots, c+1\}$. Let A be the set of all the arcs $(i, j), i \in N_0, j \in N_+, i \neq j$. Each arc (i, j) is associated with two non-negatives traveling times: τ_{ij}^T and τ_{ij}^D, that represent the time for traveling that arc by the truck and by the drones, respectively. The travel time matrices of the drones and the truck are normally different. Nodes 0 and $c+1$ represent the same physical point, the depot, and the traveling time between them is set to 0. Serving times at customers for both drone and truck are included in the travel times, while the time for preparing the drones at launch is given by σ^L and the rendezvous time is given by σ^R. No launch time is considered when the sortie starts form the depot. The drones have a battery limit (endurance) of E time units, that constraints their use. Rendezvous time σ^R contributes to the endurance computation while σ^L does not, since the drone lies on the truck when it is prepared for the launch.

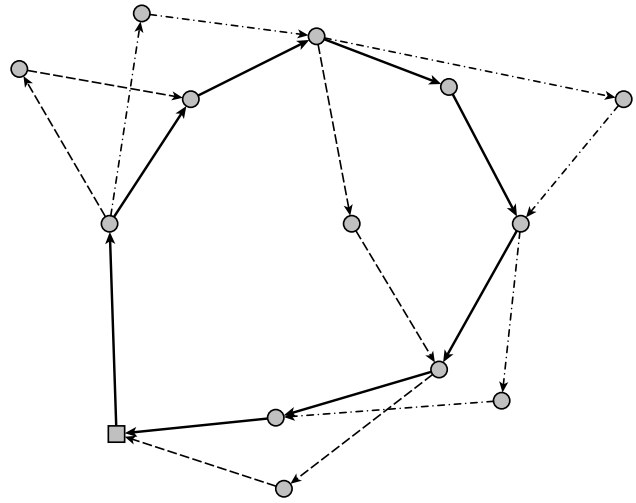

FIGURE 1 Example of a feasible solution of the MFSTSP with two drones, the dashed lines represent two different drones, while the solid arrow represent the truck path. The square is the depot and the circles are the customers.

A sortie performed by a drone $v \in D$ is formally defined by a tuple (i, j, k, v), ($i \neq j, k$ and $j \neq k$) where $i \in N_0$ is the launching node, $j \in C'$ the customer to serve, and $k \in N_+$ the rendezvous node. Let F be the set of all sorties that can be performed within the endurance time E ($\tau^D_{ij} + \tau^D_{jk} + \sigma^R \leq E$).

The drones can be launched from the truck only when the truck is stopped at a customer or at the depot; the drones cannot leave the depot before the truck starts its route. The truck can keep serving customers while the drones are performing their sorties; however, a synchronization is required. First of all, if a drone arrives before the truck at the rendezvous point it waits for the truck, and endurance has to allow this. If the truck arrives at the rendezvous before a drone, the truck must wait but it can perform other operations, such as launching or collecting other drones. Indeed, the scheduling of the operations in a node is one of the main novelties of this problem: the sequence of the operations becomes crucial. If a drone is collected and a second drone arrives in the meanwhile, the second drone should wait and this can influence its flying time and the endurance limit can bee exceeded. We thus consider all the operations sequences in each node to ensure that the launching times, the rendezvous times, and the endurance limits are respected. The objective of the optimization is to minimize the completion time, that is the moment when the last vehicle arrives at the depot. A feasible solution of a problem with two drones is depicted in Figure 1.

4 | MATHEMATICAL FORMULATIONS

4.1 | Four-index formulation

The first formulation we propose, that we call *4I-BC*, makes use of a set of binary variables x_{ij} that equal 1 if the arc $(i, j) \in A$ is used by the truck, 0 otherwise. We recall that the set of arcs A includes only arcs with (i, j) such that $i \in N_0$ and $j \in N_+$. The second set of variables represents the sorties: y^v_{ijk} equals 1 if the sortie $(i, j, k, v) \in F$ is performed, 0 otherwise. To impose synchronization, a set of time variables is needed. First of all we introduce the

truck arrival and departure time variables t_i^A, t_i^P for each node $i \in N$. Moreover, we introduce the drone arrival and departure time t_i^{vA}, t_i^{vP} in each node $i \in N$ for each drone $v \in D$. Time variables are not enough to guarantee the good management of precedence at each node, we thus introduce new binary variables: λ_i^{vw} equals 1 if drone $v \in D$ is launched before drone $w \in D, w \neq v$ in node $i \in N_0$, 0 otherwise. ρ_i^{vw} equals 1 if drone $v \in D$ returns before drone $w \in D, w \neq v$ in node $i \in N_+$, 0 otherwise. α_i^{vw} equals 1 if drone $v \in D$ is launched before $w \in D$ is returned to $i \in N$, 0 otherwise. β_i^{vw} equals 1 if drone $v \in D$ returns before drone $w \in D$ is launched from $i \in N$, 0 otherwise. The resulting formulation is described step by step in the following.

Objective function.

As by the problem description, the objective function aims to minimize the completion time, the time of the last vehicle arriving at the final depot. This can be expressed as the minimization of the truck departure time in the final depot: min t_{c+1}^P; however, this objective function results in being very noneffective in terms of provided lower bounds (see e.g. Dell'Amico et al. [13]). To overcome this problem, we decompose the objective function: the first component is the one given by the traveling times of the truck route, indeed we need to wait at least that time before returning to the depot. The second component is given by the difference between the truck departure time and arrival time in each node; we will see in the following that this difference includes the launching and rendezvous times and the waiting times. The second component can be written as $\sum_{i \in N}(t_i^P - t_i^A)$; however, this difference can lead to bad lower bounds and thus we define a set of non-negative variables w_i, one for each node $i \in N$ to represent each of these differences. This rationale is expressed by the objective function (1) and the constraint (2).

$$\min \sum_{(i,j) \in A} \tau_{ij}^T x_{ij} + \sum_{i \in N} w_i \tag{1}$$

$$w_i \geq (t_i^P - t_i^A) \qquad i \in N \tag{2}$$

Routing and sortie constraints.

In the following we report the constraints that represent the truck routing and the drone flights.

$$\sum_{i:(i,j) \in A} x_{ij} + \sum_{i,k,v:(i,j,k,v) \in F} y_{ijk}^v = 1 \qquad j \in C \tag{3}$$

$$\sum_{i:(j,i) \in A} x_{ji} + \sum_{i,k,v:(i,j,k,v) \in F} y_{ijk}^v = 1 \qquad j \in C \tag{4}$$

$$\sum_{j \in N_+} x_{0j} = \sum_{i \in N_0} x_{i,c+1} = 1 \tag{5}$$

$$\sum_{i:(i,j) \in A} x_{ij} = \sum_{k:(j,k) \in A} x_{jk} \qquad j \in C \tag{6}$$

$$\sum_{j,k:(i,j,k,v) \in F} y_{ijk}^v \leq 1 \qquad i \in N_0, v \in D \tag{7}$$

$$\sum_{i,j:(i,j,k,v) \in F} y_{ijk}^v \leq 1 \qquad k \in N_+, v \in D \tag{8}$$

$$2y_{ijk}^v \leq \sum_{h \in N_0, h \neq k} x_{hi} + \sum_{l \in C, l \neq k} x_{lk} \qquad i,j,k,v : (i,j,k,v) \in F \tag{9}$$

$$y_{0jk}^v \leq \sum_{h \in N_0, h \neq k} x_{hk} \qquad j,k,v : (0,j,k,v) \in F \tag{10}$$

Constraints (3) and (4) impose that each customer must be visited once and only once by the truck or one of the drones, if the drone visit is possible. Note that the linear combination of these constraint impose also the flow conservation constraint (6). Constraints (5) guarantee that the truck starts form the starting depot 0 and returns to the ending depot. Note that if the variable $x_{0,c+1} = 1$ only drones are used to serve customers, but one drone can be used only once in such a case. Constraints (7) and (8) guarantee that a drone can leave and return only once in a certain node. Constraint (9) guarantees that a sortie is performed only if its launching and rendezvous nodes are visited by the truck. Constraint (10) imposes the same concept for sorties starting from the depot. One should note that a typical subtour elimination constraint is missing, this is because subtours are eliminated thanks to the timing constraints; however, we can easily avoid two nodes subtorus by imposing $x_{ij} + x_{ji} \leq 1, i, j \in N, i \neq j$.

Truck timing and drone timing.

To impose synchronization of the operations we include the following constraints in our formulation.

$$t_j^A \geq t_i^P + \tau_{ij}^T - M(1 - x_{ij}) \qquad (i,j) \in A \qquad (11)$$

$$t_j^A \leq t_i^P + \tau_{ij}^T + M(1 - x_{ij}) \qquad (i,j) \in A \qquad (12)$$

$$t_k^{vA} \geq t_i^{vP} + \tau_{ij}^D + \tau_{jk}^D + \sigma^R - M(1 - y_{ijk}^v) \qquad (i,j,k,v) \in F \qquad (13)$$

$$t_k^{vA} - t_i^{vP} \leq E + M(1 - \sum_{j \in C'} y_{ijk}^v) \qquad i \in N_0, k \in N_+, v \in D \qquad (14)$$

Constraints (11) and (12) guarantee that if an arc (i,j) is traveled, then the arrival time of the truck in j is the departure time of the truck in i plus the time to travel the arc. The combination of constraints (11) and (12) imposes the equality $t_j^A = t_i^P + \tau_{ij}^T$ if the arc (i,j) is traveled: thanks to that the waiting times are included between the departure and the arrival times at every node. If drone $v \in D$ performs the sortie $(i,j,k,v) \in F$ then the arrival time in k should include the departure time in i, the traveling times τ_{ij}^D and τ_{jk}^D and the rendezvous time: this is guaranteed by constraint (13). We recall that the launching time is not considered in the flying time and thus it must be considered before t_i^{vA}. If the sortie is performed then the difference between the arrival time and the departure time of the drone should not exceed the endurance E as imposed by constraint (14).

Timing in a vertex.

Here we consider what happens between the arrival and the departure of the tuck in a node. Constraint (15) imposes that the arrival time of a drone is subsequent to the arrival time of the truck and if a sortie arrives in the node then rendezvous time should also be included. Note that if the drone finishes its sortie before the truck is arrived at the meeting point, the drone arrival will be extended to include this (endurance constraint is taken into account). In terms of costs in the objective function, the drone waiting time is absorbed by the truck time. In case the truck waits for one or more drones, this is included between the arrival and departure time of the vertex where the truck is waiting. Constraint (16) states that the departure time of the drones in node 0 is grater than or equal to the arrival time; in constraint (17) it is also imposed that the departure time must be grater than the arrival time of the drone plus a launching time, if the drone is launched in that node. Constraint (18) guarantees that the truck departs always after all the drone handling is completed.

$$t_i^{vA} \geq t_i^A + \sigma^R \sum_{k,j,v:(k,j,i,v) \in F} y_{kji}^v \qquad i \in N, v \in D \qquad (15)$$

$$t_0^{vP} \geq t_0^{vA} \qquad v \in D \qquad (16)$$

$$t_i^{vP} \geq t_i^{vA} + \sigma^L \sum_{j,k,v:(i,j,k,v)\in F} y_{ijk}^v \qquad i \in N_+, v \in D \tag{17}$$

$$t_i^P \geq t_i^{vP} \qquad i \in N, v \in D \tag{18}$$

Scheduling in a vertex.

With the following constraints we impose that the drone times respect the sequencing of launches and rendezvous. Constraint (19) imposes the following: if drone w is launched before drone v in node i then the departure time of drone v should be at least the departure time of drone w plus the launching time (not included in the flight time). Note that the launching time is not included if the drone is launched from the starting depot. Constraint (20) states that, if the drones w and v are returning in node i and drone w returns before drone v then the time of arrival of drone v should be at least the time of arrival of drone w plus the rendezvous time (which is included in the flying time). If drone w returns in i before the drone v is launched then the departure time of v should be at least the arrival time of w plus the launching time; while, if the drone w is launched from i before drone v is returned then the arrival time of v should be greater than the departure time of w plus the rendezvous time. The two latter conditions are expressed by constraints (21) and (22), respectively.

$$t_i^{vP} \geq t_i^{wP} + \sigma^L - M(1-\lambda_i^{wv}) \qquad i \in N_+, v, w \in D, v \neq w \tag{19}$$

$$t_i^{vA} \geq t_i^{wA} + \sigma^R - M(1-\rho_i^{wv}) \qquad i \in N_+, v, w \in D, v \neq w \tag{20}$$

$$t_i^{vP} \geq t_i^{wA} + \sigma^L - M(1-\beta_i^{wv}) \qquad i \in N_+, v, w \in D, v \neq w \tag{21}$$

$$t_i^{vA} \geq t_i^{wP} + \sigma^R - M(1-\alpha_i^{wv}) \qquad i \in N_+, v, w \in D, v \neq w. \tag{22}$$

Linking constraints.

The following constraints are needed to guarantee that variables used to express the scheduling sequence in each node are linked to the sortie variables.

$$\rho_k^{vw} + \rho_k^{wv} \leq 1/2 \sum_{i\in N_0} \sum_{j\in C'} (y_{ijk}^v + y_{ijk}^w) \qquad k \in N_+, w, v \in D, w \neq v \tag{23}$$

$$\rho_k^{vw} + \rho_k^{wv} + 1 \geq \sum_{i\in N_0} \sum_{j\in C'} (y_{ijk}^v + y_{ijk}^w) \qquad k \in N_+, w, v \in D, w \neq v \tag{24}$$

$$\lambda_i^{vw} + \lambda_i^{wv} \leq 1/2 \sum_{j\in C'} \sum_{k\in N_+} (y_{ijk}^v + y_{ijk}^w) \qquad i \in N_0, w, v \in D, w \neq v \tag{25}$$

$$\lambda_i^{vw} + \lambda_i^{wv} + 1 \geq \sum_{j\in C'} \sum_{k\in N_+} (y_{ijk}^v + y_{ijk}^w) \qquad i \in N_0, w, v \in D, w \neq v \tag{26}$$

$$\alpha_i^{vw} + \alpha_i^{wv} \leq 1 \qquad i \in N, v, w \in D, v \neq w \tag{27}$$

$$\alpha_i^{vw} + \beta_i^{wv} \leq 1/2(\sum_{j\in C'} \sum_{k\in N_+} y_{ijk}^v + \sum_{k\in N_0} \sum_{j\in C'} y_{kji}^w) \qquad i \in N, v, w \in D, v \neq w \tag{28}$$

$$\alpha_i^{vw} + \beta_i^{wv} + 1 \geq (\sum_{j\in C'} \sum_{k\in N_+} y_{ijk}^v + \sum_{k\in N_0} \sum_{j\in C'} y_{kji}^w) \qquad i \in N, v, w \in D, v \neq w \tag{29}$$

To explain constraint (23) we recall that ρ_k^{vw} equals one if the two drones return to the same node and drone v returns before w. In that constraint we impose that at most one between ρ_k^{vw} and ρ_k^{wv} can equal one and this can happen only when two sorties, one for v and one for w, return in k. Constraint (24) completes the reasoning by imposing that one drone must return before the other if two drones return in the same node. Constraints (25) and (26) ensure the same

for launches represented by variables λ. To explain the remaining constraints we recall that β_i^{vw} equals one if drone v returns in i before w is launched in the same node, and that α_i^{vw} equals one if drone v is launched from node i before drone w returns to the same node. This said, in constraint (**??**) we impose that either v returns before w is launched in i, or the other way round, if a sortie involving v and w happens in that node. On the other hand, in constraint (27) we impose that either v is launched before w returns in i, or the other way round, if a sortie involving v and w happens in that node. In constraint (28) and (29) we impose that, if drone v is returning to i and drone w is launched in the same node, than either one or the other happens first.

Crossing sortie elimination.

The formulation we described up to now could lead to infeasible solutions because it allows the so-called *crossing sorties*, defined properly as follows. Let $i \in N_0, I \in C$ be two nodes visited by the truck while running along a path $P(v)$ from i to I, and assume that two sorties of the same drone $v \in D$ (i, j, k, v) and (I, m, n, v) with $k \notin P(v)$ exist. In this case, the second sortie starts before the first one is returned and the overall solution is therefore infeasible. Let us define $\mathcal{P}(v)$ as the set of all the paths with these characteristics for each drone $v \in D$. The following *crossing sorties elimination constraints* (30) can be used to eliminate this type of infeasibility.

$$\sum_{h=1}^{|P(v)|-1} x_{s(h),s(h+1)} + \sum_{\substack{(i,j,k,v)\in F \\ k\notin P(v)}} y_{ijk}^v + \sum_{(I,m,n,v)\in F} y_{Imn}^v \leq |P(v)| \qquad P(v) \in \mathcal{P}(v), v \in D \tag{30}$$

where $P(v) = \{s(1), s(2), \ldots, s(|P(v)|)\}$, $s(1) = i, s(|P(v)|) = I$. These constraints impose that one of the arcs of the path or one of the sorties should be set to zero to make the solution feasible.

We can strengthen (30) using the idea of "tournament constraint". Since the path enters at most once in each nodes, we can include in the constraint also the arcs $(h, j) \in A$ such that $h, j \in P(v)$ and h precedes j in $P(v)$. We obtain the *tournament crossing constraints* (TCS), given by (31).

$$\sum_{h=1}^{|P(v)|-1} \sum_{j=h+1}^{|P(v)|} x_{s(h)s(j)} + \sum_{\substack{(i,j,k,v)\in F \\ k\notin P(v)}} y_{ijk}^v + \sum_{(I,m,n,v)\in F} y_{Imn}^v \leq |P(v)| \qquad P \in \mathcal{P}(v), v \in D \tag{31}$$

Backward sortie elimination.

Let $\mathcal{B}(v)$ denote the set of all truck paths $P(v) = \{s(1), s(2), \ldots, s(q)\}$ with $s(1) = 0$ and $s(q) \in C$ such that it exists a sortie $(i, j, s(q), v)$ with $i \notin P(v)$, $v \in D$. The *backward sortie elimination constraints* are given by (32), and impose that at least one of the arcs of the path or the sortie is eliminated.

$$\sum_{i=1}^{|P(v)|-1} x_{s(i)s(i+1)} + \sum_{(i,j,s(q),v)\in F} y_{ijs(q)}^v \leq |P(v)| - 1 \qquad P(v) \in \mathcal{B}(v), v \in D \tag{32}$$

Constraint (32) can be strengthen by considering a tournament type constraint on path $P(v)$, that imposes that at most one arc enters a node of the path, and adding all sorties of drone $v \in D$ that terminate in $P(v)$, but start at nodes outside $P(v)$. We obtain the *tournament backward constraints* (TBS), given by (33).

$$\sum_{i=1}^{|P(v)|-1} \sum_{j=i+1}^{|P(v)|} x_{s(i)s(j)} + \sum_{\substack{(i,j,k,v)\in F \\ i\notin P(v), k\in P(v)}} y_{ijk}^v \leq |P(v)| - 1 \qquad P(v) \in \mathcal{B}(v), v \in D \tag{33}$$

Finally note that backward sorties happen only when the time constraints are not respected, in our case in fractional solutions.

Bounds.

$$t_0^A = t_0^P = 0 \tag{34}$$

$$t_i^A, t_i^P \geq 0 \qquad i \in N_+, v \in D \tag{35}$$

$$t_i^{vA}, t_i^{vP} \geq 0 \qquad i \in N, v \in D \tag{36}$$

$$x_{ij} \in \{0,1\} \qquad (i,j) \in A \tag{37}$$

$$y_{ijk}^v \in \{0,1\} \qquad (i,j,k,v) \in F \tag{38}$$

$$w_i \geq 0 \qquad i \in N \tag{39}$$

$$\alpha_i^{vw}, \beta_i^{vw}, \rho_i^{vw}, \lambda_i^{vw} \in \{0,1\} \qquad i \in N, v, w \in D, v \neq w \tag{40}$$

4.2 | Three-index formulation

The second formulation we present, called *3I-BC*, uses a smaller number of variables similarly to what done by Dell'Amico et al. [13]. In particular, we treat each sortie with two sets of three-index variables, instead of one four-index set: the binary variable $\overrightarrow{g}_{ij}^v$ that equals 1 if a sortie of drone $v \in D$ starts in node $i \in N_0$ and serves node $i \in C'$, 0 otherwise; and the binary variable $\overleftarrow{g}_{ij}^v$ that equals 1 if a sortie of drone $v \in D$ returns in node $j \in N_+$ after serving node $i \in C'$, 0 otherwise. The other variables remain the same with respect to the previous formulation. We need to define some bounds on the variables.

To further reduce the number variables we preliminary fix to zero all the variables corresponding to arcs with flying time exceeding the battery limit, i.e., we set $\overrightarrow{g}_{ij}^v = 0$ for all $(i,j) \in A : \tau_{ij}^D > E$ and $\overleftarrow{g}_{jk}^v = 0$ for all $(j,k) \in A : \tau_{jk}^D + \sigma_R > E$, $v \in D$. We also fix to zero variables that do not allow to complete a feasible drone fly: $\overrightarrow{g}_{ij}^v = 0$, $(i,j) \in A, j \notin C'$; $\overleftarrow{g}_{jk}^v = 0$, $(j,k) \in A, j \notin C'$; $\overrightarrow{g}_{ic+1}^v = 0$ $i \in N$ and $\overleftarrow{g}_{j0}^v = 0$ $j \in N$, $v \in D$.

The new formulation inherits some components from the previous one, first of all the objective function (1) and the constraints needed to respect the waiting time (2); the routing constraints (5) and (6); the timing constraints (11), (12), (16), (18)–(22); the linking constraints (**??**), (27); and the variable bounds: (34)–(37), (39), and (40).

Routing and sortie constraints.

Similarly to what done for the previous formulation, we need to express the routing and sortie constraints with the new variables.

$$\sum_{i:(i,j)\in A} x_{ij} + \sum_{i:(i,j)\in A} \overrightarrow{g}_{ij}^v = 1 \qquad j \in C, v \in D \tag{41}$$

$$\sum_{j:(i,j)\in A} x_{ij} + \sum_{j:(i,j)\in A} \overleftarrow{g}_{ij}^v = 1 \qquad i \in C, v \in D \tag{42}$$

$$\sum_{j:(i,j)\in A} \overrightarrow{g}_{ij}^v \leq \sum_{h:(i,h)\in A} x_{ih} \qquad i \in N_0, v \in D \tag{43}$$

$$\sum_{i:(i,j)\in A} \overleftarrow{g}_{ij}^v \leq \sum_{h:(h,j)\in A} x_{hj} \qquad j \in N_+, v \in D \tag{44}$$

$$\sum_{i:(i,j)\in A} \overrightarrow{g}^v_{ij} = \sum_{k:(j,k)\in A} \overleftarrow{g}^v_{jk} \qquad j \in C', v \in D \tag{45}$$

Constraint (41) and (42) are needed to impose that each customer is served either by the truck or by a drone. Constraints (43) and (44) impose that a sortie starts and returns only if on nodes traveled by the truck; while constraint (45) is needed to impose the flow conservation at every node served by drones.

Truck timing and drone timing.

Also the truck and drone timing constraints need to be updated to embrace the new variables.

$$t^{vA}_k \geq t^{vP}_i + \tau^D_{ij} + \tau^D_{jk} + \sigma^R - M(2 - \overrightarrow{g}^v_{ij} - \overleftarrow{g}^v_{jk}) \qquad i \in N_0, j \in C', k \in N_+, i \neq j, k, j \neq k, v \in D \tag{46}$$

$$t^{vA}_k - t^{vP}_i \leq E + M(2 - \overrightarrow{g}^v_{ij} - \overleftarrow{g}^v_{jk}) \qquad i \in N_0, j \in C', k \in N_+, v \in D \tag{47}$$

Constraint (46) states that if a sortie is traveled the arrival time of the sortie in the rendezvous node is grater or equal the departure time plus the traveling and the rendezvous time. Constraint (47) guarantees that the endurance is not exceeded once a sortie is performed.

Time in a vertex.

Similarly to what we proposed for the previous formulation, we need to consider the times in a vertex.

$$t^{vA}_i \geq t^A_i + \sigma^R \sum_{j\in C'} \overleftarrow{g}^v_{ji} \qquad i \in N, v \in D \tag{48}$$

$$t^{vP}_i \geq t^{vA}_i + \sigma^L \sum_{j\in C'} \overrightarrow{g}^v_{ij} \qquad i \in N_+, v \in D \tag{49}$$

Constraint (48) imposes that the arrival time of a drone includes the rendezvous. Constraint (49) includes the launching time in a node if a sortie is launched.

Linking constraints.

As in the previous formulation we need to represent the sequence of sorties arrival and departure and thus we need to link the needed variables with those defining the sorties.

$$\rho^{vw}_k + \rho^{wv}_k \leq 1/2 \sum_{j\in C'} (\overleftarrow{g}^v_{jk} + \overleftarrow{g}^w_{jk}) \qquad k \in N_+, w, v \in D, w \neq v \tag{50}$$

$$\rho^{vw}_k + \rho^{wv}_k + 1 \geq \sum_{j\in C'} (\overleftarrow{g}^v_{jk} + \overleftarrow{g}^w_{jk}) \qquad k \in N_+, w, v \in D, w \neq v \tag{51}$$

$$\lambda^{vw}_i + \lambda^{wv}_i \leq 1/2 \sum_{j\in C'} (\overrightarrow{g}^v_{ij} + \overrightarrow{g}^w_{ij}) \qquad i \in N_0, w, v \in D, w \neq v \tag{52}$$

$$\lambda^{vw}_i + \lambda^{wv}_i + 1 \geq \sum_{j\in C'} (\overrightarrow{g}^v_{ij} + \overrightarrow{g}^w_{ij}) \qquad i \in N_0, w, v \in D, w \neq v \tag{53}$$

$$\alpha^{vw}_i + \beta^{wv}_i \leq 1/2(\sum_{j\in C'} \overrightarrow{g}^v_{ij} + \sum_{j\in C'} \overleftarrow{g}^w_{ji}) \qquad i \in N, v, w \in D, v \neq w \tag{54}$$

$$\alpha^{vw}_i + \beta^{wv}_i + 1 \geq (\sum_{j\in C'} \overrightarrow{g}^v_{ij} + \sum_{j\in C'} \overleftarrow{g}^w_{ji}) \qquad i \in N, v, w \in D, v \neq w \tag{55}$$

Constraint (50) and (51) impose that if two sorties are returning at the same node than one should arrive first. Constraints (52) and (53) impose that if two sorties are launched from the same node, one is launched before the other. Constraints (54) and (55) impose that, if one sortie is returning in one node and the other one is leaving than one action should happen before the other.

Crossing sortie elimination.

TCS (31) must be rewritten as follows. Let $i \in N_0, l \in C$ be two nodes encountered by the truck while running along a path P from i to l, and assume that exist two sorties of drone $v \in D$ defined by $\overrightarrow{g}^v_{ij} > 0$ and $\overrightarrow{g}^v_{lm} > 0$ and such that there is no node $k \in P \setminus \{i\}$ with $\overleftarrow{g}^v_{hk} > 0$. In this case the second sortie starts before the first sortie is terminated an the following tournament crossing constraint (TCS2) holds:

$$\sum_{h=1}^{|P|-1} \sum_{j=h+1}^{|P|} x_{s(h)s(j)} + \sum_{\substack{(i,j)\in A,\\ j\notin P}} \overrightarrow{g}^v_{ij} + \sum_{\substack{(l,j)\in A,\\ j\notin P}} \overrightarrow{g}^v_{lj} \le |P| \qquad P \in \mathcal{P}(v), v \in D \tag{56}$$

where $P = \{s(1), s(2), \ldots, s(q)\}$ with $s(1) = i, s(q) = l$, and $\mathcal{P}$(v) now defines the set of all the paths with the described characteristics with respect to drone $v \in D$.

Backward sorties elimination.

Backward sorties elimination are modified as follows. Let $i \in N_0, j \in N_+$ be two nodes encountered by the truck while traveling on a path P, suppose that exists a sortie launch identified by $\overleftarrow{g}^v_{k,i} > 0$ and such that $k \notin P$ and a sortie rendezvous identified by $\overrightarrow{g}^v_{v(q),k} > 0$ and such that $k \notin P$, that, together determine an infeasible solution. Let $\mathcal{B}(v)$ now denote the set of all truck paths $\{s(1), \ldots, s(m), \ldots, s(q)\}$ with $s(1) = 0$, $s(m) = i$, and $s(q) = j$ with such a sortie as described for drone $v \in D$. The tournament version of the backward sorties elimination constraints (TBS2) is:

$$\sum_{h=1}^{|P|-1} \sum_{l=h+1}^{|P|} x_{s(h)s(l)} + \overrightarrow{g}^v_{jk} + \sum_{\substack{(l,k)\in A\\ l\notin P}} \overrightarrow{g}^v_{lk} + \sum_{\substack{(k,l)\in A\\ l\notin P}} \overleftarrow{g}^v_{kl} \le |P| \qquad P \in \mathcal{B}(v), v \in D \tag{57}$$

Bounds and other constraints.

We finally need to impose the following constraints to avoid infeasibilities:

$$\overrightarrow{g}^v_{ij} + \overleftarrow{g}^v_{ij} \le 1 \qquad (i,j) \in A, v \in D \tag{58}$$

$$\overrightarrow{g}^v_{ij} + \overleftarrow{g}^v_{ji} \le 1 \qquad (i,j) \in A, v \in D \tag{59}$$

$$\overrightarrow{g}^v_{ij}, \overleftarrow{g}^v_{ij} \in \{0,1\} \qquad (i,j) \in A, v \in D \tag{60}$$

4.3 | Crossing Sortie Variables formulation with four-index variables

We define a new formulation based on the 4I-BC one that we call *4I-z*. Mainly we replicate most of the four-index constraints and the objective function, in particular (1)–(29) and (34)–(40). Similarly to what done in Dell'Amico et al. [12], we introduce the *crossing sortie variables*, new binary variables z^v_i for each node $i \in N$ and each drone $v \in D$ that take value 1 if the drone v is available for launch at node i, 0 otherwise. To guarantee this we introduce the following

constraints:

$$\sum_{j,k:(i,j,k,v)\in F} y^v_{ijk} \le z^v_i \qquad i \in N_0, v \in D \tag{61}$$

$$z^v_i \le \sum_{j:(j,i)\in A} x_{ji} \qquad i \in N_+, v \in D \tag{62}$$

$$z^v_j \le z^v_i - x_{ij} + \sum_{l,k:(l,k,j,v)\in F} y^v_{lkj} - \sum_{k,l:(i,k,l,v)\in F} y^v_{ikl} \qquad (i,j) \in A, v \in D. \tag{63}$$

$$z^v_i \in \{0,1\} \quad i \in N, v \in D \tag{64}$$

Constraint (61) guarantees that variable z^v_i equals 1 if a sortie of drone $v \in D$ starts in node $i \in N_0$. Constraint (62) guarantees that a variable z^v_i assumes value 1 only if the truck in entering in node $i \in N_+$. Constraint (63) allows the z to be 1 until a sortie starts, after that point the z variable is enforced to be 0 on the truck path until a sortie of the same drone returns to the truck, at that point the variable z is allowed to assume value 1 once more. This constraint imposes that the variable z is 0 when the corresponding drone is not on the truck. In (64) the variables z^v_i are defined binary.

Note that no sortie elimination constraint is needed because the new variables guarantee the avoidance of crossing sorties; however, one could include TCS (31) and also TBS (33) to strengthen this formulation by cutting infeasible fractional solutions.

4.4 | Crossing Sortie Variables formulation with three-index variables

Similarly to the previous formulation, we apply the crossing sortie variables to the three-index one obtaining a new formulation called *3I-z*. In particular we reproduce the following: objective function (1) and the constraints needed to respect the waiting time (2); the routing constraints (5) and (6); the timing constraints (11), (12), (16), (18)–(22); the linking constraints (**??**), (27); and the variable bounds: (34)–(37), (39)–(55), and (58)–(60). The variables z^v_i are the same as in the previous formulation and we also include in the current formulation the linking constraint (62) and the variable bound (64). We thus add constraints (65) and (66), that work similarly to constraints (61) and (63), respectively.

$$\sum_{j\in C'} \overrightarrow{g}^v_{ij} \le z^v_i \qquad i \in N_0, v \in D \tag{65}$$

$$z^v_j \le z^v_i - x_{ij} + \sum_{k\in C'} (\overleftarrow{g}^v_{kj} - \overrightarrow{g}^v_{ik}) + 1 \qquad (i,j) \in A, v \in D \tag{66}$$

Note that, also for this formulation, no sortie elimination constraint is needed because the new variables guarantee the avoidance of crossing sorties; however, we can use TCS2 (56) and also TBS2 (57) to strengthen this formulation by cutting infeasible fractional solutions.

5 | COMPUTATIONAL EXPERIMENTS

To test the proposed methods we have implemented them in C++ to run on a computer equipped with an Intel Core 3.10 GHz i3-2100 CPU and with 8.00 GB of RAM, running Windows 7 operating system. CPLEX 12.71 was used

as MILP solver, and only a single thread was utilized during the testing. Full implementation details are given in the next section. We tested our methods on the well-known instances presented by Murray and Chu [21] for the FSTSP, that can be used also for the MFSTSP without any effort. The 36 randomly generated instances have endurance $E = 20$ minutes or $E = 40$ minutes resulting in 72 instances, each with 10 customers located in an 8x8 area. In our experiments the truck could carry from two to five drones. The depot is located in four possible positions: in the first position 'a' the depot is almost in the barycenter of the customers, in 'b', 'c' and 'd' it is close to the right border of the square, with a vertical position which is, respectively, in the barycenter, at the bottom border and below the bottom border at a distance which is equal to the distance of the barycenter from this border. Moreover, the truck speed is 25 miles/h and based on Manhattan distances, the drones speed can be either 15, 25, 35 miles/h and is based on Euclidean distances. We refer to Murray and Chu [21] for full details. We used the benchmark instances proposed by Murray and Chu to compare the proposed algorithms and select the best performing one; afterwards, we compared our best algorithm on a second set, proposed by Murray and Raj [22].

5.1 | Implementation

Formulations 4I-BC and 3I-BC required a branch-and-cut implementation, since they include TCS (31) and TCS2 (56), respectively, that are exponentially many. In Table 1, these constraints are summarized under the names TCSint and TCSfrac, when generated from integer and fractional solutions, respectively. To separate these constraints, we have considered the residual graph $G' = (N, A')$ obtained from G by selecting the only arcs associated with a non zero variable (x, y, g) in the continuous relaxation of the model. For separating those cuts we explore the graph starting from depot 0, until a truck path violating one of the constraints is identified, if any. During the same graph exploration we also check if TBS (33) and TBS2 (57) are violated for fractional solutions (TBSfrac in Table 1) and, when one is found, we insert it (note that the elimination of backward sorties is always guaranteed by the time variables for integer solutions). In such a case the overall cut separation procedure has a time complexity $O(|A'|)$. Constraints (31) and (56) are not needed to solve formulations 4I-z and 3I-z; however, for both formulations we separate the fractional solutions violating those inequalities (TCSfrac in Table 1). We also include fractional *subtours elimination constraints* (SECfrac in Table 1) in all formulations, they are separated with the standard approach that requires to solve at most $O(n^2)$ max flow problems (see, e.g. [24]) on the residual graph. As branching strategy we used the CPLEX solver 'strong branching', and an initial heuristic upper bound is provided to the solver.

In Table 1 one can see the number of cuts used in the branch-and-cut (up to optimally or when the 1 hour time limit is reached) with respect to different number of drones and formulation averaged on all the 72 instances. We inserted one violated cut at a time in the order in which they are displayed in the table. We cut SECfrac first because the separating procedure is computationally less expensive than the procedure that identifies violated TCSfrac and TBSfrac cuts (which are detected in the same procedure). Preliminary tests guaranteed the profitability of the fractional cuts for all formulations with respect to implementations without fractional cuts. In particular, for all formulations the current setting improves the lower bound value at the root node by about 5%, in average, with respect to the case with no cuts, being the SECfrac the cut providing the higher contribution among all. This justifies the choice of separating SECfrac first.

One can see that, TCSint are the most used cuts at the end of the iterations; however, the number of these cuts is relatively small, being that they are needed in formulations 4I-BC and 3I-BC to make the solution feasible. The number of needed TCSint is larger for the 3I-BC formulation than for the 4I-BC, while it is the opposite when it comes to the fractional TCSfrac. Also TBSfrac shows to be relevant, being the second most important cut for 4I-BC and the firs one for 4I-z and for 3I-z. A relevant number of SECfrac is inserted and its number is similar for all formulations.

TABLE 1 Average number of inserted cuts for type with respect to the number of drones and the used method on the 72 Murray and Chu instances.

|D|	Form.	TCSint	SECfrac	TCSfrac	TBSfrac
2	4I-BC	55.86	6.78	9.26	21.08
	3I-BC	71.65	8.72	3.74	5.13
	4I-z	-	8.40	0.53	32.17
	3I-z	-	8.28	0.36	6.11
3	4I-BC	24.21	6.35	6.18	14.14
	3I-BC	41.88	7.17	3.39	3.44
	4I-z	-	7.13	0.26	16.19
	3I-z	-	7.01	0.08	3.19
4	4I-BC	10.78	5.90	5.65	11.92
	3I-BC	30.25	6.39	1.86	2.33
	4I-z	-	6.17	0.04	10.97
	3I-z	-	6.11	0.01	2.29
5	4I-BC	5.85	5.29	3.15	6.68
	3I-BC	12.56	5.10	0.97	1.07
	4I-z	-	5.35	0.07	5.01
	3I-z	-	5.21	0.00	1.17

Moreover, one can see that the number of inserted cuts, especially the TCS ones, integer and fractional, diminishes with the increase of the available drones, because infeasible crossing sorties are less likely to appear when a large number of drones is available.

5.2 | Numerical Results for Murray and Chu benchmark

In this section we present the results on the benchmark instances by Murray and Chu in order to decide which one is the best among the proposed methods. Note that the solutions for $|D| = 1$ are treated in detail in other papers (see e.g. Dell'Amico et al. [13]) and we do not report them in this work.

In Figures 2 and 3 we summarize, in two histograms, the results for the four proposed algorithms (4I-BC, 3I-BC, 4I-Z, and 3I-z) by varying the number of available drones $|D|$ from two to five and the endurance E assuming the values of 20 and 40 minutes. In Figure 2 we show the average percentage gap computed as follows: $\%gap = 100 \cdot (UB - LB)/UB$ (where UB an LB are the upper and the lower bounds at the end of the iterations for each algorithm). In Figure 3 one can find the number of optimal solutions that each of the algorithms could gather at the end of the iterations (ether the optimal solution is found or the 1 hour time limit is reached). One can notice that the average percentage gap increases with the number of available drones and by increasing the endurance. The first point is justified by the increase of the number of variables due to a larger number of available drones, while the second one is due to the larger number of feasible sorties that are guaranteed by a larger endurance. That is linked to the trend of the number of optima found by the proposed algorithms, which follows the opposite direction. A clear conclusion that one can draw is that formulation 4I-z is the one providing the best results, and thus in the following we will consider only this one.

Details are displayed in Tables 2 and 3. In Table 2 we show the results of formulation 4I-z for different number of available drones aggregated by endurance and depot location, while in Table 3 we show the same results aggregated by endurance and drone speed. In both tables we report the number of available drones $|D|$, the endurance E, the depot location dep or the drone speed $speed$, the averaged $\%gap$, the computing time (averaged on all instances),

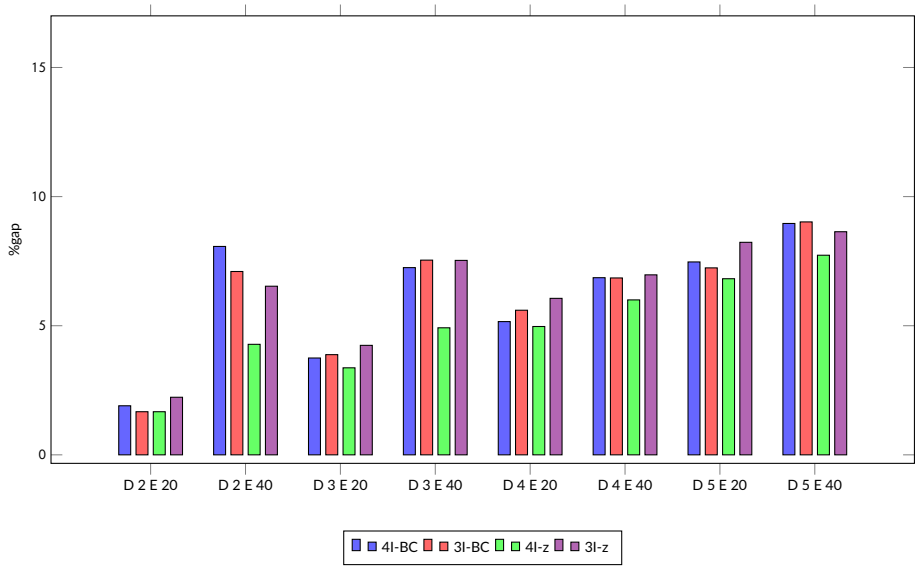

FIGURE 2 Percentage gap averaged over all instances computed for the four proposed algorithms. A comparison is shown with respect to the number of available drones D and the endurance E.

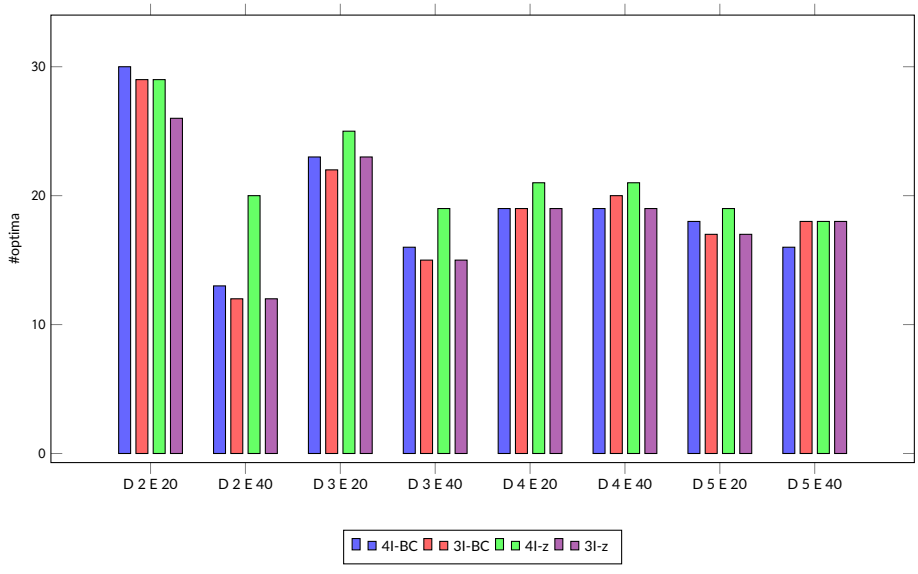

FIGURE 3 Number of optimal solutions obtained by the four proposed algorithms within the 1 hour time limit. A comparison is shown with respect to the number of available drones D and the endurance E.

TABLE 2 Results for the 4I-z formulation per depot location.

|D|	E	dep	%gap	time	opt	|D|	E	dep	%gap	time	opt
2	20	a	3.95	1376.21	6	4	20	a	4.47	2054.18	4
		b	0.73	668.60	8			b	4.39	1861.98	5
		c	1.13	736.17	8			c	6.37	1645.54	6
		d	0.85	985.21	7			d	4.66	1452.79	6
		avg/sum	1.67	941.55	29			avg/sum	4.97	1753.62	21
	40	a	5.27	2343.97	4		40	a	1.16	1290.87	7
		b	2.71	1559.32	6			b	4.21	1631.04	5
		c	3.80	1760.54	6			c	11.43	2018.69	4
		d	5.36	2017.14	4			d	7.20	2001.63	5
		avg/sum	4.28	1920.24	20			avg/sum	6.00	1735.56	21
	avg/sum		2.97	1430.89	49		avg/sum		5.49	1744.59	42
3	20	a	3.88	1668.48	6	5	20	a	5.39	1717.49	5
		b	2.99	1386.77	6			b	10.25	2058.28	4
		c	3.48	1115.18	7			c	7.13	1830.07	5
		d	3.11	1361.13	6			d	4.52	2077.75	5
		avg/sum	3.37	1382.89	25			avg/sum	6.82	1920.90	19
	40	a	2.60	2075.76	4		40	a	2.01	1466.69	6
		b	2.27	1511.27	6			b	9.60	2005.54	4
		c	7.54	1775.47	5			c	10.61	2006.31	4
		d	7.26	2007.41	4			d	8.69	2011.27	4
		avg/sum	4.92	1842.48	19			avg/sum	7.73	1872.45	18
	avg/sum		4.14	1612.68	44		avg/sum		7.28	1896.67	37

TABLE 3 Results for the 4I-z formulation per drone speed.

|D|	E	speed	%gap	time	opt	|D|	E	speed	%gap	time	opt
2	20	15	0.00	107.32	12	4	20	15	1.10	1005.36	10
		25	4.29	2025.29	6			25	10.21	2445.39	4
		35	0.70	692.03	11			35	3.61	1810.12	7
		avg/sum	1.67	941.55	29			avg/sum	4.97	1753.62	21
	40	15	9.11	3083.75	3		40	15	10.64	2795.35	3
		25	2.00	1542.53	8			25	4.96	1486.33	8
		35	1.74	1134.45	9			35	2.39	924.99	10
		avg/sum	4.28	1920.24	20			avg/sum	6.00	1735.56	21
	avg/sum		2.97	1430.89	49		avg/sum		5.49	1744.59	42
3	20	15	0.57	551.73	11	5	20	15	2.68	1359.46	9
		25	7.50	2317.19	5			25	11.25	2512.42	4
		35	2.03	1279.75	9			35	6.54	1890.81	6
		avg/sum	3.37	1382.89	25			avg/sum	6.82	1920.90	19
	40	15	8.88	2757.01	4		40	15	12.30	3005.77	2
		25	3.13	1536.26	7			25	7.81	1403.94	8
		35	2.74	1234.15	8			35	3.07	1207.64	8
		avg/sum	4.92	1842.48	19			avg/sum	7.73	1872.45	18
	avg/sum		4.14	1612.68	44		avg/sum		7.28	1896.67	37

having a 1 hour time limit, and the number of optimal solutions obtained opt.

In Figure 4 we show the improvement (cost decrease) in percentage for the cases with two, three, four, and five available drones with respect to the single drone solution. These values are computed on the 35 instances that could be solved to optimality for every D. The availability of multiple drones on the truck makes the solution cost decrease as the number of drones increases. Indeed, by increasing the number of drones, several sorties can be performed in parallel and this makes the objective function value decrease. Moreover, since the launches from the depot do not pay the launching time, the larger is the number of available drones the larger is the advantage of launching sorties from the depot. In Figure 4 one can see that by having five available drones the solution can improve over the single drone one by almost 9%, in average.

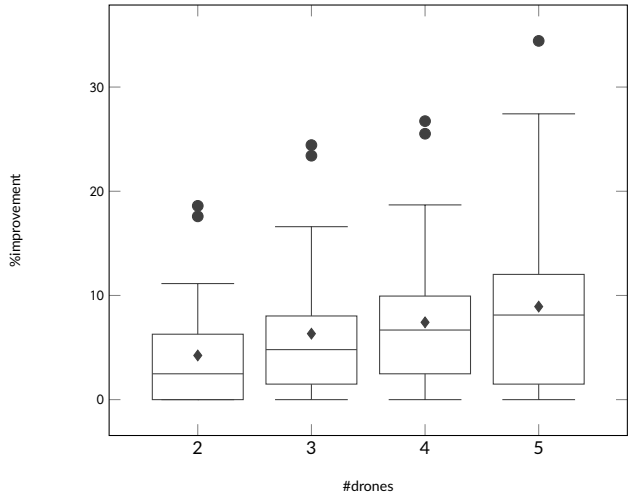

FIGURE 4 Improvement on the optimal solution value with multiple drones with respect to the optimal solution value with one drone.

5.3 | Variants of the original model

In this section we introduce three variants of the problem described in the previous part of the paper, to which we refer now as original model. The first model we propose is a variant that considers the launching time also when the drones are launched from the depot. One should note that we decided to follow the literature and consider no launching time at the depot, since the truck and the drone can leave the depot uncoupled. On the other hand, when increasing the number of drones, this can lead to a large amount of drones launched from the depot in the optimal solution, since that implies an advantage in terms of total time and thus in the objective function. We believe it can be interesting to evaluate the influence of this component on the solution. The second variant that we consider, already considered in the literature even if in the minority of the papers, allows the drones to wait on the ground so to save energy. The third variant recalls other methods proposed in the literature that do not consider the correct scheduling of activities in each node, and thus leads to problems that might be easier to solve, but with solution that might turn to be infeasible in practice. With these motivations we describe these variants in the following.

5.3.1 | Model with launching time from the depot

Including in mathematical formulation the launching times also when the drones are launched from the depot is straightforward. For the variation of the 4I-BC and 4I-z we need to remove to each of the models constraints (16) and (34) and change the following constraints, rewriting them for all $i \in N$ instead of N_+: (17), (19)–(22), (35). For 3I-BC and 3I-z we need to remove from each of the models constraints (16) and (34) and change the following constraints (19)–(22), (35), (48)–(49) rewriting them for all $i \in N$ instead of N_+.

5.3.2 | Model with drones that can wait on the ground

This variant, that allows the drones to wait on the ground at customer nodes so to save battery has already been considered (see e.g. Phan et al. [35]) and also for the single drone case (see e.g. Ponza [28] and Dell'Amico et al. [13]). To include this variant in the formulations we only need to modify constraints (13) and (14) in 4I-BC and 4I-z and (46) and (47) in 3I-BC and 3I-z.

5.3.3 | Model with no scheduling of activities in the nodes

Recent works (e.g. Seifried [32]) consider the launching and rendezvous times included in the drone traveling times, and thus do not consider scheduling activities at the nodes where multiple launches or rendezvous happen. This may lead to infeasible solutions in reality; however, we intend to investigate the variant where parallel operations are allowed by technology. This can also be the case of non-aerial drones that can be sent from the truck and return to it with less restrictive limitation in numbers. To do so we need to change the previously introduced formulations. The formulations 4I-BC and 4I-z can be rewritten without the following constraints: (19)–(29), and (40). The formulations 3I-BC and 3I-z can be rewritten without the following constraints: (19)–(22), (**??**), (27), (40), and (50)–(55).

Note that in case no scheduling is considered, there is no need of including drone depending variables in the formulations and the crossing sortie elimination constraints could also be simplified and provide improved computational results; however, that is not the scope of this publication.

5.3.4 | Comparison: different variant with single drone and multiple drones

A comparison among the different proposed variants is given in Figure 5, in which we display the percentage difference in the objective function for the multiple drones case with respect to the single drone one for all the considered variants. For each variant and instance we compute the gap as follows $100 \cdot (z_1 - z_D)/z_1$, being z_1 tho objective function value of the single drone case and z_D the multiple drone one, with the current number of drones in set D. This is computed for all the instances for which the optimal solution value was at hand for all the number of available drones. These gaps for all instances for which the optimal solution was obtained are displayed in classical box-plots, in which the lower and the upper bounds are the first and the third quartile, the middle line shows the median, and the diamond shows the average. The points represent the outliers, that are considered so, and thus depicted outside the whiskers, if they are above the the third quartile or below the first quaritle by more than the interquartile range multiplied by 1.5. In dark grey one can find the original model, in blue the case with launching time from the depot, in red the wait case, and in light grey the model with no scheduling.

One can see that most of the times the increment of the number of available drones leads to an improvement in the objective function value; indeed, the larger is the number of the available drones, more are the possibilities to

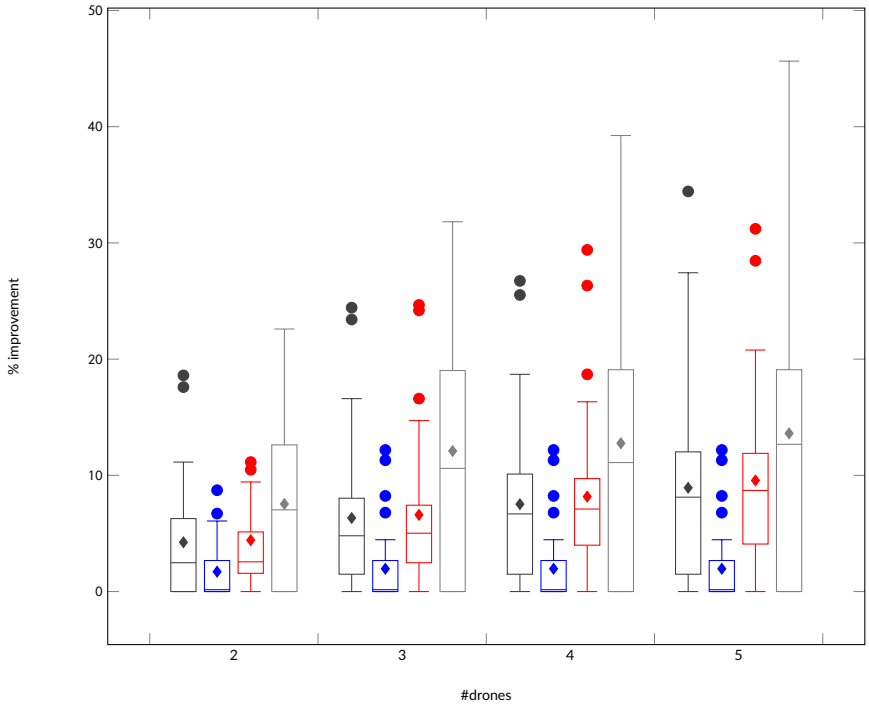

FIGURE 5 Percentage improvement of the optimal solution value with multiple drones ($|D| = 2, \ldots, 5$) with respect to the optimal solution value with a single drone.

serve customers in parallel and less is the total service time. The original model (dark grey in the figure) follows this pattern. The wait case (in red in the figure) has a similar behavior, even if the improvement in the objective function is more visible, especially in the lower bound of the box-plot. This can be justified by the fact that more sorties become feasible when the drone can wait on the ground to save battery. The variant with no scheduling (grey in the figure) allows more drones to depart and serve customers and this makes the cost of the solution decrease with the increase of the number of available drones. This variant shows the largest improvements when increasing the number of drones, reaching up to more than 13% improvement in average in the case with $|D| = 5$, with respect to the single drone case. This is justified by the fact that we can launch and collect many drones in parallel. The version in which the launches from the depot must pay the launching time (blue in the figure) shows that this component is very important; indeed, by increasing the number of available drones the solution value does not improve remarkably. In the graph, one can see that, for this variant, having more than three drones available leads to no improvement for the instances we solved to optimality. This is true for those instances, but it is not necessarily true in every case; however, a large number of drones does not improve the solution value as for the other variants, because those variants take great advantage of multiple drones departing from the depot without paying the launching time.

5.3.5 | Comparison: different variant with respect to the original model

In Figure 6 we show the percentage improvement between the solution value of the original model ($z_{original}$) and each one of the other analysed variants ($z_{variant}$), respectively, the variant with no launching time for sortie starting from the depot (in blue), the wait case variant (in red), and the variant with no scheduling (in grey). We display these values in a classical box-plot for different numbers of available drones, considering all the instances for which we obtained the optimal solution. For each drone number and instance the improvement is computed as $100 \cdot (z_{original} - z_{variant})/z_{original}$.

One can see that the percentage improvement of the version with launching time at the depot decreases strongly with the increase of the number of available drones. This is due to the fact that the original model takes advantage of multiple depot departures; indeed, by increasing the number of available drones, in the original model a larger and larger number of drones can be launched from the depot without paying the launching time. In this variant, launching multiple drones from the depot is less favourable. This difference between the cost of the two versions is more and more visible with the increase of the number of available drones. Considering the wait case the solutions have a lower cost with respect to the original case; however, the improvement is small and slowly decreases with the increase of the number of available drones. In particular, when adding more available drones, the gap between the solution values of the two variants is zero most of the times. This means that by increasing the available drones in the original model we can compensate the lack of possibility of waiting, and this produces similar solutions to those of the wait case. Note also that the outliers values decrease as well with the increase of the number of available drones. One can see that the variant with no scheduling provides solution values that largely improve the original model ones. This is so because some operations with drones can be done in parallel and this produces less costly solutions. The solution value increase decelerates when increasing the number of available drones because the the maximum number of customers serviced with drones is reached for many instances. The values for $|D| = 5$ appear lower than those of $|D| = 4$ because the baseline is reduced given that the number of instance for which the optimal solution was available (over which we computed the gap) is lower; however, the lower bound of the box-plot for $|D| = 5$ is largely higher than that of $|D| = 4$.

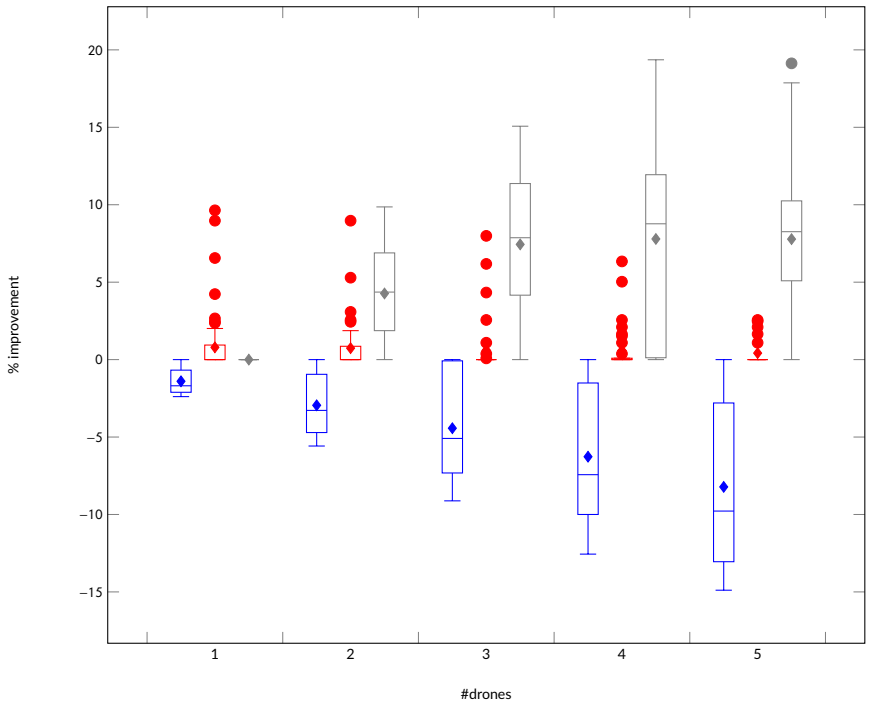

FIGURE 6 Percentage improvement on the optimal solution value with respect to the original model for the variants, grouped per number of available drones.

5.4 | Numerical Results for Murray and Raj Benchmark

We tested the best performing method of ours, 4I-BC, on the instances provided by Murray and Raj [22]. They use a set of instances with eight customers that they solve with an exact method and a set of instances with 10 customers that they solve with heuristic methods. Murray and Raj, in their paper, use different versions of endurance and different drone settings, we decided to use a fixed endurance represented by a maximum amount of flight time of 700, 1400, 350, and 700 seconds, each linked to its corresponding drone setting, depending on the speed and range. This resulted in 80 instances for each number of available drones with eight customer and 80 for each number of available drones with 10 customers. In the problem they solve, they consider a service time when serving the customers, that we do not include in our model, and they use a launching time of 60 seconds and a rendezvous time of 30 seconds which we have also used. In Table 4 we report the averaged results on the 640 instances we run, reporting the number of customers, the number of drones, the percentage gap, the solving time in seconds (having set a time limit of 1 h) and percentage of optimal solutions obtained for each subset. We also report the time that Murray and Raj needed to solve the instances on a PC with an 8-core Intel i7-6700 processor and 16 GB RAM ($MRtime$). Note that their machine, compared to ours, is much faster according to [4]. The author declare that they could solve only the 66% of the eight customer instances, for which we could provide an optimal solution the 88% of the times. One interesting fact is that the instances are divided into two sets called Buffalo and Seattle and Murray and Raj found easier to solve the Seattle instances, which we also do for the eight customer instances, while our method found it was easier to solve the Buffalo 10 customer instances.

TABLE 4 Results on Murray and Raj instances with eight and 10 customers.

c	|D|	%gap	time	%opt	MRtime
8	1	0.00	25.41	100	51.4
	2	0.00	227.30	100	1075.5
	3	2.44	829.19	85	2155.6
	4	4.25	1579.84	65	2430.5
	Buffalo	2.64	977.37	80	2349.1
	Seattle	0.71	353.50	95	507.4
	avg	1.67	665.44	88	1428.3
10	1	0.88	654.48	93	-
	2	2.95	1246.12	78	-
	3	8.07	2126.31	53	-
	4	7.87	2160.09	46	-
	Buffalo	3.45	1473.91	69	-
	Seattle	6.43	1619.59	65	-
	avg	4.94	1546.75	67	-

6 | CONCLUSIONS

In this paper we considered the version of the flying Sidekick Traveling Salesman Problem with multiple drones, where a truck is equipped with a set of homogeneous drones and all can work together to serve a set of customers. In the studied version the drones need time to be launched and collected when the truck is stationary at some node of the considered network and the scheduling of launching and rendezvous operations of multiple drones is an important

component of the problem. To solve the problem we proposed four novel formulations implemented as branch-and-cut algorithms that improved upon the related literature. We studied the effect of having up to five available drones on the truck and with up to 10 customers to serve. We also considered three variants of the problem and evaluated the influence of each of these on the solution. This kind of problems is very challenging even when it comes to solve small size instances, and we believe there is space for new improvements; in particular we think that future works should focus on the scheduling component of this problem.

References

[1] 2020. Coronavirus: Grocer uses robots to delivery grocery orders amid covid-19 pandemic. https://6abc.com/shopping/grocer-using-delivery-robots-during-coronavirus-pandemic/6060134/. (Accessed: 2020-05-29).

[2] 2020. Drone delivery canada asks for covid-19 use cases. https://www.gpsworld.com/drone-delivery-canada-asks-for-covid-19-use-cases/. (Accessed: 2020-05-29).

[3] 2020. Here's what the uber eats delivery drone looks like. https://techcrunch.com/2019/10/28/heres-what-the-uber-eats-delivery-drone-looks-like/.. (Accessed: 2020-05-29).

[4] 2020. Userbenchmark, https://cpu.userbenchmark.com/. (Accessed: 2020-05-29).

[5] 2020. Zipline will bring its medical delivery drones to the u.s. to help fight the coronavirus. https://www.fastcompany.com/90483592/zipline-will-use-its-medical-delivery-drones-to-the-u-s-to-help-fight-the-coronavirus. (Accessed: 2020-05-29).

[6] N. Agatz, P. Bouman, and M. Schmidt, *Optimization approaches for the traveling salesman problem with drone*, Transp. Sci. (2018).

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

[8] G. Clarke and J.W. Wright, *Scheduling of vehicles from a central depot to a number of delivery points*, Oper. research **12** (1964), 568–581.

[9] R. Daknama and E. Kraus, *Vehicle routing with drones*, arXiv preprint arXiv:1705.06431 (2017).

[10] J.C. de Freitas and P.H.V. Penna, *A randomized variable neighborhood descent heuristic to solve the flying sidekick traveling salesman problem*, Electron. Notes Discr. Math. **66** (2018), 95–102.

[11] J.C. de Freitas and P.H.V. Penna, *A variable neighborhood search for flying sidekick traveling salesman problem*, Int. Trans. Oper. Res **27** (2020), 267–290.

[12] M. Dell'Amico, R. Montemanni, and S. Novellani, *Models and algorithms for the flying sidekick traveling salesman problem*, arXiv preprint arXiv:1910.02559 (2019).

[13] M. Dell'Amico, R. Montemanni, and S. Novellani, *Drone-assisted deliveries: new formulations for the flying sidekick traveling salesman problem*, Optim. Lett. (2019), 1–32.

[14] Q.M. Ha, Y. Deville, and Q.D. Pham, *On the min-cost traveling salesman problem with drone*, Transp. Research. Part C: Emerging Technologies (2017).

[15] Q.M. Ha, Y. Deville, Q.D. Pham, and M.H. Hà, *Heuristic methods for the traveling salesman problem with drone*, Comput. Sci (2015).

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

[17] M. Hu, W. Liu, K. Peng, X. Ma, W. Cheng, J. Liu, and B. Li, *Joint routing and scheduling for vehicle-assisted multi-drone surveillance*, IEEE Internet Things J. (2018).

[18] M. Joerss, J. Schröder, F. Neuhaus, C. Klink, and F. Mann, *Parcel delivery - the future of last mile*, Travel, Transport Logist. (2016).

[19] A. Karak and K. Abdelghany, *The hybrid vehicle-drone routing problem for pick-up and delivery services*, Transp. Res. Part C: Emerging Technologies **102** (2019), 427–449.

[20] P. Kitjacharoenchai, M. Ventresca, M. Moshref-Javadi, S. Lee, J.M. Tanchoco, and P.A. Brunese, *Multiple traveling salesman problem with drones: Mathematical model and heuristic approach*, Comput. Indus. Eng. **129** (2019), 14–30.

[21] C.C. Murray and A.G. Chu, *The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery*, Transp. Res. Part C: Emerging Technologies **54** (2015), 86–109.

[22] C.C. Murray and R. Raj, *The multiple flying sidekicks traveling salesman problem: Parcel delivery with multiple drones*, Transp. Res. Part C: Emerging Technologies **110** (2020), 368–398.

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

[24] M. Padberg and G. Rinaldi, *Facet identification for the symmetric traveling salesman polytope*, Math. programming **47** (1990), 219–257.

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

[26] S. Poikonen, B. Golden, and E.A. Wasil, *A branch-and-bound approach to the traveling salesman problem with a drone*, INFORMS J. Comput. **31** (2019), 335–346.

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

[28] A. Ponza, Optimization of drone-assisted parcel delivery, Master's thesis, Univ. Degli Studi Di Padova, 2016.

[29] L.D.P. Pugliese, F. Guerriero, D. Zorbas, and T. Razafindralambo, *Modelling the mobile target covering problem using flying drones*, Optim. Lett. **10** (2016), 1021–1052.

[30] D. Sacramento, D. Pisinger, and S. Ropke, *An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones*, Transp. Res. Part C: Emerging Technologies **102** (2019), 289–315.

[31] D. Schermer, M. Moeini, and O. Wendt, *A matheuristic for the vehicle routing problem with drones and its variants*, Transp. Res. Part C: Emerging Technologies **106** (2019), 166–204.

[32] K. Seifried, *The traveling salesman problem with one truck and multiple drones*, Available at SSRN 3389306 (2019).

[33] B. Skorup and H. C., 2020. How drones can help fight the coronavirus. `https://www.mercatus.org/publications/covid-19-policy-brief-series/how-drones-can-help-fight-coronavirus`. (Accessed: 2020-05-29).

[34] Statista, 2020. Retail e-commerce sales worldwide from 2014 to 2021 (in billion u.s. dollars), `https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/`. (Accessed: 2020-05-29).

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

[36] X. Wang, S. Poikonen, and B. Golden, *The vehicle routing problem with drones: Several worst-case results*, Optim. Lett. **11** (2017), 679–697.

[37] Z. Wang and J.B. Sheu, *Vehicle routing problem with drones*, Transp. research part B: methodological **122** (2019), 350–364.

[38] R. Wolleswinkel, V. Lukic, W. Jap, R. Chan, J. Govers, and S. Banerjee, *An onslaught of new rivals in parcel and express*, BCG website (2018).

[39] E.E. Yurek and C.H. Ozmutlu, *A decomposition-based iterative optimization algorithm for traveling salesman problem with drone*, Transp. Res. Part C: Emerging Technologies **91** (2018), 249–262.