Drone-assisted deliveries: new formulations for the flying sidekick traveling salesman problem

Mauro Dell’Amico¹ · Roberto Montemanni¹ · Stefano Novellani¹

Received: 29 March 2019 / Accepted: 4 October 2019 / Published online: 18 November 2019
© Springer-Verlag GmbH Germany, part of Springer Nature 2019

Abstract
In this paper we consider a problem related to deliveries assisted by an unmanned aerial vehicle, so-called drone. In particular we consider the Flying Sidekick Traveling Salesman Problem, in which a truck and a drone cooperate to deliver parcels to customers minimizing the completion time. In the following we improve the formulation found in the related literature. We propose three-indexed and two-indexed formulations and a set of inequalities that can be implemented in a branch-and-cut fashion. The methods that we propose are able to find the optimal solution for most of the literature instances. Moreover, we consider two versions of the problem: one in which the drone is allowed to wait at the customers, as in the literature, and one in which waiting is allowed only in flying mode. The solving methodologies are adapted to both versions and a comparison between the two is provided.

Keywords Aerial drones · Routing · Branch-and-cut · Parcel deliveries · Formulations

1 Introduction
Once restricted to the military domain, unmanned aerial vehicles (UAV), also known as (aerial) drones, have been adopted broadly in civil applications in the humanitarian sector and are currently gaining increasing interest in the commercial sector. This is due to their capability of accomplishing otherwise impossible tasks, performing activities more effectively, or to lead to cost saving solutions. A comprehensive definition of

¹ Dipartimento di Scienze e Metodi dell’Ingegneria (DISMI), Università di Modena e Reggio Emilia (UNIMORE), via Amendola 2, 42122 Reggio Emilia, Italy
drones is the one provided by Scott and Scott [48]: “Drones are devices which are capable of sustained flight, which do not have human on board, and are under sufficient control to perform useful functions”.

During the last years, drones have interested researchers whose effort was put especially in improving the required technology:

(a) the hardware, by reducing weight, increasing battery duration, improving the charging system, improving safety, etc.,
(b) the software, by designing better autonomous operations and guidance systems using improved GPS accuracy, localization techniques, obstacle detection and avoidance techniques, enhanced sensors, image processing, etc.,
(c) the safety and security elements, by adopting technologies to protect the flight against spoofing and hijacking of UAVs, etc.

A smaller effort has been done for solving optimization and operational problems related to the use of drones, although the number of research papers in this area has strongly increased in the very last years. In this work we treat the flying sidekick traveling salesman problem (FSTSP) an operational optimization problem in which a truck is equipped with a drone and both cooperate to deliver parcels to customers in the minimum completion time.

The paper is organized as follows: we start with Sect. 2, that summarizes the main real-life applications that use drones. In Sect. 3 we discuss upon the most relevant related literature, while in Sect. 4 we formally describe the FSTSP. Mathematical formulations and their implementations are described in Sects. 5 and 6. Extensive computational results, comparing the models and their performances, are presented in Sect. 7. Sect. 8 concludes the paper.

2 Applications

Early drone applications arose in the military sector, where most of the optimization works are focused on the path planning, normally to avoid obstacles or radars, minimizing travel length or altitude, or to improve the usage of batteries. In this paper we are not interested in listing the military applications, but we address the interested reader to the following works: Bortoff [21], Richards et al. [43], Zheng et al. [57], and Roberge et al. [44].

Besides military applications, drones can be used in both humanitarian and commercial sectors. The three sectors make use of drones in such a way that can be divided in two main applications: to collect and deliver information and to collect and deliver goods. On one hand we have surveillance, monitoring, or covering, and on the other hand deliveries. In the following we describe a large set of applications adhering to this dichotomy.

2.1 Gathering information: surveillance, monitoring, and covering

The information gathering applications include surveillance, monitoring, and covering activities, in which displacement of goods is not necessary. When operating those
activities, drones fly autonomously to monitor the environment with different sensors or cameras and communicate and exchange data and information with other drones or with a central station.

Drones can be used to optimize the coverage of an area (see, e.g., Shang et al. [49]) or to follow specific targets that can also be moving ones (see, e.g., Zorbas et al. [58] and Di Puglia Pugliese et al. [42]). This is done by defining the position of drones, their path (see, e.g., Kashuba et al. [32]), their number, etc., while maximizing the coverage or minimizing the time between an appearance of an event and its covering, the total travel length, the service costs in energy, etc. In particular, typical applications regard the following domains: traffic management, environmental monitoring, catastrophic events, remote locations surveillance, precision agriculture, building inspection, security surveillance, etc.

2.2 Moving goods: deliveries

This second set of applications is characterized by the fact that real goods need to be moved from one place to another. The main application is to provide faster, more cost efficient goods or parcel delivery, especially in the last-mile, but also mail delivery and quickly required medical items delivery have been object of interest. Indeed, in emergency and events management drones can be used when infrastructure are damaged or unreliable to access isolated regions to deliver needed goods in addition to surveillance. Drones can transport water, food, medical supplies during a crisis or and event. This point can match the medical items delivery: delivery of blood, medications, vaccines, defibrillators, insulin, oxygen, or other needed health-care items in location with difficult access due to poor infrastructure, remote areas, traffic congestion, inaccessible roads due to weather or disasters, or simply urgently needed.

Regarding commercial applications, several companies are investigating the use of drones for parcel delivery for e-commerce, such as Amazon [4], Alibaba [3], and Alphabet [17]. Amazon CEO announced Amazon Prime Air, that uses a fleet of UAVs to deliver parcels from warehouses to customers [4]. Australian textbook distributor Zookal started testing drone parcel deliveries in Australia [18]. DHL Parcel operates an autonomous drone delivery system, to deliver medications and other urgent goods to the German island of Juist [8]. In 2016 it also tested a delivery system in the Bavarian alpine region, typically carrying either sporting goods or urgently needed medicines [13]. Chinese JD.com deploys drones to extend its delivery and logistics network, using drones in areas with complex terrain and poor infrastructure for last-mile delivery. The employment of drones has started in 2016 with four rural locations of China in the outskirts of Beijing and in the provinces of Jiangsu, Shaanxi, and Sichuan. The company announced an agreement with the Shaanxi provincial government to build China’s largest low altitude drone network, serving a 300km radius area with drone stations and routes to deliver e-commerce parcels [11,12], and declared that it will build 150 drone launch facilities in China by 2020 [10]. Alphabet (Google) announced to enter the field with Project Wing [17]. UPS and DPDgroup are also testing parcel delivery with drones [7,15]. UPS and Zipline are working on a drone network to deliver vaccines and blood to 20 clinics in remote locations in Rwanda [9]. UPS and
Workhorse are testing drone and truck combo delivery [14]. Flirtey completed the first US Federal Aviation Administration approved drone delivery in July 2015, when it delivered medical supplies to a health clinic in Wise, Virginia. The company has also started a partnership with 7-Eleven for home delivery [1]. In the Netherlands and Sweden a prototype ambulance drone was tested for delivering defibrillators [5,6]. Matternet provides an on-demand delivery platform, an end-to-end solution integrating the Matternet’s drones and stations. The company provides its platform as a service to healthcare, e-commerce, and logistics organizations. In particular, it transports medical items between health-care facilities in Switzerland, and performs deliveries with drones in Zurich. In this last case Matternet and Mercedes-Benz have joined forces to create delivery solutions integrating vans and drones for siroop online shop [16]. The United Arab Emirates stated that they plan to use unmanned aerial drones to deliver official documents and packages to its citizens as part of efforts to upgrade government services [2].

This is just an incomplete list of many of the cases that arose in the last years. The large number makes us suspect an increasing interest in the field in the future.

### 3 Related literature

The FSTSP is a generalization of the *traveling salesman problem* (TSP) and the *vehicle routing problem* (VRP). A large body of literature has been dedicated to these problems; however, we are interested only in those problems that are highly related to the FSTSP, especially those in which drones and trucks are coupled and synchronized.

A problem that is conceptually related to the FSTSP is the *close enough traveling salesman problem* (see, e.g., Shuttleworth et al. [51]), that aims at finding the cheapest route for the truck without visiting every customer on its route, but only getting within a certain radius of each customer. A similar problem and even more conceptually related is the *covering salesman problem* (see, e.g., Current and Schilling [26]), that aims at finding the cheapest tour such that all nodes that are not part of the route lie within a specified radius from a node on the truck route. This reminds the maximum radius that the drone can travel.

In the FSTSP, the vehicle (truck or drone) that arrives first at the meeting point has to wait for the other, thus we can state that it lies in the class of problems that require synchronization between vehicles, in particular, among the categories defined by Drexl [29], the FSTSP can be considered under *movement synchronization en route*, in which vehicles may join and separate multiple times along a route. The author claims that this class received little attention in literature. One of these problems is also one of the most related to the FSTSP: the *truck and trailer routing problem* (TRRP) (see e.g., Chao [25]), where two different types of vehicle can serve customers: trucks and trailers. Due to practical constraints (e.g. street size), some customers can only be served by a truck, while other customers can be served by a truck or a truck pulling a trailer. Trucks are autonomous, while trailers always need to be pulled by a truck. Parking places are used to decouple trucks from trailers when convenient to serve some subset of customers that can only be served by a single truck. This resembles the idea of the drone truck decoupling. A problem similar to the FSTSP has been studied by Lin
New formulations for the FSTSP [33], in which two types of delivery resources are used, vans and foot couriers, which allows for coordination. A heavy resource (a van) may carry both delivery items and one or more units of the lighter resource (foot couriers). Foot couriers can pick up and deliver items independently or travel on a van on its outbound and/or return leg, they can serve more than one customer, in contrast to what a drone can do. Moreover, foot couriers do not need to return to the same van.

Let us now consider works that solve problems related to the use of drones. As said, the first applications of UAVs were implemented in the military sector, and the same can be said for the use of optimization techniques when optimizing operations for drones. In Sisson’s thesis [52] the author studies a tabu search coupled with Monte Carlo simulation to determine the minimum number of drones to cover a pre-selected target set based on stochastic survival probabilities that also incorporate the wind effects. In Ryan’s thesis [45] a tabu search within a discrete event simulation is applied to solve a multi TSP with time windows for UAVs, that requires to attain a level of target coverage using a minimum number of vehicles. Weather and threat are considered. A direct extension of this work is proposed in Ryan et al. [46]. O’Rourke et al. [37] consider dynamic routing of UAVs in operational use with the US Air Force. The dynamic components are wind and emerging targets. They model the problem as a VRP with time windows and use a tabu search to solve it.

Boone et al. [20] solve a multi TSP for drones, they firstly divide customers into clusters by using K-mean clustering, then they solve a TSP for each cluster with the nearest neighbor algorithm improved with the 2-opt local search. Dorling et al. [28] study a multi trip VRP for drone delivery that considers the effect of batteries and weight on energy consumption. They solve one version of the problem in which costs are minimized under a delivery time limit and a second version that aims at minimizing the delivery time subject to a budget constraint. The authors present mixed integer linear programming (MILP) formulations and a simulated annealing algorithm. Tseng et al. [53] solve a modified TSP in which an autonomous drone has to serve delivery points and may use charging stations for charging the battery. They consider wind uncertainty and variable speeds. The used algorithm is based on the Christofides one to solve the shortest spanning tree and then the minimum matching problem to obtain an Eulerian tour.

Location problems linked to drones are studied in the two following works. Scott and Scott [48] consider drone delivery models for healthcare. They consider deliveries with trucks that leave a central depot to drone nests, and then with drones from drone nests to delivery points. They use two objective functions: one minimizes the total time and the other minimizes the maximum weighted time for truck/drone delivery. Both depot and nests need to be located. No routing is considered but only direct trips. Shavarani et al. [50] studied the facility location problem for the optimization of drone delivery system evaluating the Amazon Prime Air case study in San Francisco. They locate both launching stations and recharge stations. No routing is considered but direct trips. The authors solve the problem with a genetic algorithm.

Campbell et al. [24] present some drone arc routing problems, that are continuous optimization problems. They discretized them by approximating the curves with polygonal chains. The drone rural postman problem, in which drones have no capac-
ity, is solved by applying iteratively algorithms for the rural postman problem. They discuss also other problems that consider capacitated or multiple drones.

The following papers consider the coupled interventions of drones and trucks. The first paper we consider is slightly different from the others, indeed Savuran and Karakaya [47] study a problem in which a truck follows a linear path and in the meanwhile a drone is launched from the truck and returns to the truck after performing an open TSP to visit all the targets. They solve the problem with a genetic algorithm. Ferrandez et al. [30] propose a work that couples drones and a truck. They have a set of delivery customers that are clusterized by using K-means clusterization. They thus solve a TSP to visit the centroid of each cluster, the centroids are the points where the truck stops to launch one or more drones to serve the customers of that cluster. The TSP part is solved with a genetic algorithm. Boysen et al. [23] consider a fixed truck route in which the truck represents a loading platform for the drones. The truck and the drone can wait each other. They use at most two drones and three restriction of the problem: one that imposes the drones to return to the launching point, one in which the drones can return up to the next node in the truck sequence after the launching one, and the last one in which the drones can return in one of the following nodes in the truck sequence.

In the following we examine problems that consider drones and trucks working together to complete operations, in which both can accomplish tasks. In this first part we evaluate what Otto et al. [38] classify as drones and vehicles performing independent tasks. Murray and Chu [36] propose the parallel drone scheduling TSP (PDSTSP), in which customers can be served by either one truck route or by a fleet of drones that can operate only from the depot. In this case only customers within a certain range from the depot can be served with drones. The others are served by the truck. Solving this problem can provide good results when many customers lie close to the depot. They propose MILP formulations and simple greedy heuristics. Saleu et al. [34] propose a two step iterative heuristic based on dynamic programming for the same problem. Ulmer and Thomas [54] study a dynamic variant of the PDSTSP called the same-day delivery with heterogeneous fleets of drones and vehicles, a problem in which requests arrive dynamically and need to be allocated to drones or truck maximizing the number of served customers. They solve the problem with an approximate dynamic programming known as parametric policy function approximation.

We consider, now, routing problems in which trucks are equipped with drones and both vehicles can be used to deliver packages to customers. Otto et al. [38] classify these problems under the name drones and vehicles as synchronized working units. The flight of the drone is called sortie: the drone leaves its truck in a node of the network (launch), performs a delivery to a customer, and returns to the truck (rendezvous) in a node of the network. Some problems consider the launch and rendezvous times negligible, some others take them into account; however, truck and drone must be synchronized and thus wait for each other. The objective function is normally to minimize the completion time. Murray and Chu [36] define and study the FSTSP. In the FSTSP truck and drone can cooperate to serve customers: one drone can leave the truck at a node and return to the truck at another node after completing a delivery. Customers can be visited only once, but some customers can be visited only by the truck because their request cannot be fulfilled by the drone (for various reasons, such as capacity limitations, requirement
of a signature, drone cannot safely land, etc.). Drones cannot return to the launching point. Agatz et al. [19] solve the TSP with drone (TSP-D), a problem in which one truck cooperates with one drone to make deliveries. The aim is to find the fastest method to serve customers with a truck or a drone that can leave the truck, serve a customer, and return to the truck in one of the nodes. Each customer has to be visited at least once by one of the vehicles, but they can be visited more than once by the truck if it is convenient for drone launch and return. Launching and rendezvous points of the same sortie can coincide. Some nodes cannot be visited by drones. Endurance is unlimited and launching and rendezvous times are considered negligible. The authors present an integer linear programming (ILP) formulation and propose route first-cluster second heuristics based on local search and dynamic programming. Bouman et al. [22] solve the TSP-D with dynamic programming. Ha et al. [31] also use the name TSP-D, albeit in this case nodes cannot be visited multiple times and launch and rendezvous of a sortie are not allowed to happen at the same node. We can thus say that they solve the FSTSP. The authors propose two heuristic algorithms: a route first-cluster second one and a cluster first-route second one. Ponza’s thesis [40] tackles the FSTSP proposing a modified MILP formulation with respect to the Murray and Chu’s [36] one and solve it with a simulated annealing algorithm. The new formulation, among the other few differences, does not allow the drone to wait at customers nodes. Yurek and Ozmutlu [56] propose a decomposition-based iterative heuristic for the FSTSP (which they also call TSP-D). In the first phase they generate all possible truck tours with a fixed number of customers, the other customers are assigned to the drone. In the second phase they solve a MILP to define the optimal times for the truck and the drone. They iterate while increasing the number of fixed customers to be served by the drone. For larger instances they generate truck routes with the nearest neighbor greedy to shorten computing times.

Wang et al. [55] define the vehicle routing problem with drones (VRPD), that considers a homogeneous fleet of trucks equipped with a not necessary unitary number of drones to deliver parcels to customers. Drones can be launched from trucks at depot or at any customer node. Each drone must return to the same truck also if this happens at the same node where it has been launched. One drone can carry only one parcel. The authors imagine that the drones must travel along the street network. They derive a number of worst case results based on the number of drones per truck and the difference between the drones speed and the trucks speed. Poikonen et al. [39] extend the worst-case results considering different metrics for trucks and drones, considering limited drone batteries, and evaluating different objective functions. They also evaluate connections between the VRPD and other VRP variants. Daknama and Kraus [27] present and solve a VRPD in which they minimize the average delivery times instead of the completion time. No mathematical model is presented, but they solve the problem by first solving a multiple TSP heuristically and then introducing drones. Local search procedures are thus applied to improve the solution. Di Puglia Pugliese and Guerriero [41] solve a vehicle routing problem with time windows in which each truck is equipped with a drone that can help the truck in serving the customers so to respect the time windows (VDRPTW). The authors present a MILP formulation solved with CPLEX.
In Table 1 we classify the truck and drone problems. In the ‘Paper’ column we display the authors and the reference to the paper, in the second column we display the name of the problem (we use the name FSTSP for Ha et al. [31] and Yurek and Ozmuntlu [56] in the Table). In the following columns we consider some features of the problems: the fact that launching and rendezvous points may coincide (‘l = r’); the fact that trucks may visit nodes multiple times (‘m – v’); the fact that some nodes can be visited only by trucks (‘t – v’); the objective function used (‘Of’), where L stands for objective functions whose main point is to minimize the latest arrival at the depot, D represents those functions whose main objective is to minimize the total distance of both truck and drone, and T represents those that minimize the time until a delivery is accomplished. One should note that an objective function that minimizes the latest arrival makes the problem harder to solve with respect to one minimizing the total distance. In the last column we provide a rough description of the method used to solve the proposed problems, if any. Table 1 is inspired by the table proposed by Ponza [40] that we adjusted referring only to the problems that consider drones and trucks as synchronized working units and by adding the fact that the studied problems allow the following points: (i) the possibility of having multiple visits; (ii) the possibility of having nodes to be visited only by the truck; (iii) a brief description of the used objective function.

We address the interested reader to the recent survey by Otto et al. [38] that provides a wide overview on civil applications of drones, it gives an insight into optimization approaches used to solve operational problems, in particular those that consider drones and drones combined with other vehicles, that is the class of problems talked in this paper.

4 Problem description and basic mathematical model

In this work we study the FSTSP defined by Murray and Chu [36], that is to serve a set of customers $C = \{1, \ldots, c\}$ with either a truck or a drone. The truck starts from depot 0 and returns to the final depot $c + 1$, and is equipped with a flying drone that can be used in parallel to serve one customer at a time. The problem is built on digraph $G = (N, A)$, where the set $N = \{0, 1, \ldots, c + 1\}$ represents all the nodes, while we define $N_0 = \{0, 1, \ldots, c\}$ and $N_+ = \{1, \ldots, c + 1\}$. The set $A$ is the set of all the arcs $(i, j), i \in N_0, j \in N_+, i \neq j$. The drone can perform a sortie, that we recall is defined by a launching node (where the drone leaves the truck), a served customer, and a rendezvous node (where the drone returns to the truck), that must be different from the launching one. All customers of $C$ can be served by the truck, but only a subset $C' \subseteq C$ can be served by the drone with a sortie. Each arc $(i, j)$ is associated with two non-negatives traveling times: $\tau_{ij}^T$ and $\tau_{ij}^D$, which represent the time for traveling that arc by the truck and by the drone, respectively. The travel time matrix of the drone and the truck may be different ($\tau_{ij}^T \not\leq \tau_{ij}^D, (i, j) \in A$). Nodes 0 and $c + 1$ represent the same physical point, the depot, and the truck traveling time between them is set to 0 ($\tau^T_{0,c+1} = 0$). The arc $(0, c + 1)$ can be used to accommodate the case in which there is only one customer to be served and the drone performs the service, departing from the depot and returning to the depot after the delivery.
| Paper                        | Name          | Formulation                                      | l = r | m | v | t | v | Of Solving method                                                                 |
|------------------------------|---------------|--------------------------------------------------|-------|---|---|---|---|----------------------------------------------------------------------------------|
| Murray and Chu [36]          | FSTSP         | MILP 2-index (truck arcs) 3-index (sorties)       | x     |   | x | x | x | MILP, heuristic (nearest neighbor, savings, sweep + savings for UAV routes definition) |
| Agatz et al. [19]            | TSP-D         | ILP set covering                                 |       |   | x | x | x | L Heuristic (route first cluster second)                                         |
| Bouman et al. [22]           | TSP-D         | None                                             |       |   | x | x | x | L Dynamic programming                                                            |
| Ha et al. [31]               | FSTSP         | MILP for clustering, set packing                  |       |   |   |   |   | L Heuristic (route first cluster second, cluster first route second)             |
| Ponza [40]                   | FSTSP         | MILP 2-index (truck arcs) 3-index (sorties)       | x     |   |   |   |   | L MILP, simulated annealing                                                      |
| Yurek and Ozmutlu [56]        | FSTSP         | MILP only for drone sorties                       | x     |   |   |   |   | L Decomposition-based iterative algorithm, exact and heuristic                  |
| Wang et al. [55]             | VRPD          | None                                             | x     |   |   |   |   | L None                                                                           |
| Poikonen et al. [39]          | VRPD          | None                                             | x     |   |   |   |   | L None                                                                           |
| Daknama and Kraus [27]        | VRD           | None                                             |       |   |   |   |   | T Metaheuristic, local search                                                   |
| Di Puglia Pugliese and Guerriero [41] | VDRPTW  | MILP                                             | x     |   |   |   |   | D MILP                                                                           |
We assume that the capacity of the truck is large enough to serve all customers and that the drone performs only one delivery at the time, i.e., it leaves the truck, serves a customer, and returns to the truck before possibly serving a new customer. That is a sortie, which is formally defined by a triplet \( \langle i, j, k \rangle \), \((i \neq j \neq k, j \neq k \neq i)\) where \( i \in N_0 \) is the launching node, \( j \in C' \) the customer to serve, and \( k \in N_+ \) the rendezvous node.

Serving times at customers for both drone and truck are negligible. Service times for preparing the drone at launch and rendezvous are given by \( \sigma_L \) and \( \sigma_R \). Note that the launch time in not considered when the drone is launched at the depot, because the truck and the drone are uncoupled and the depot operations are done off-line. Drones have a battery limit (endurance) of \( E \) time units, that limits drone use for a single sortie. Rendezvous time \( \sigma_R \) contributes to the endurance computation, while \( \sigma_L \) does not, since the drone lies on the truck when it is prepared for launch.

Let \( F \) be the set of all sorties that can be performed within the endurance \( E \), that means \( \tau_{Dij} + \tau_{Djk} + \sigma_R \leq E, i \in N_0, j \in C', k \in N_+, i \neq j, k, and j \neq k \).

By the definition of sortie, the drone can be launched from the truck only when the truck is stopped at customers nodes or at the depot; however, it cannot leave the depot before the truck starts its route. In the meanwhile the truck can keep serving customers and the drone can return to the truck only at another node, that is not the launching one; however, sorties launched from the initial depot and returning to the final one are allowed. This requires a certain synchronization: the vehicle (drone or truck) that arrives first at the meeting point has to wait for the other. The objective is to minimize the completion time, that is the moment when the last vehicle arrives at the depot.

### 4.1 Murray–Chu formulation

We report in this section the mathematical formulation proposed by Murray and Chu [36] (MC in the following) that is the basis for our work. A binary variable \( x_{ij} \) is created for each arc \((i, j) \in A\) and is set to 1 if the truck uses the corresponding arc, 0 otherwise. For each sortie of the drone a binary variable \( y_{ijk}, \langle i, j, k \rangle \in F \) is used and takes value 1 if the sortie is performed, 0 otherwise.

The non-negative variables \( t^T_i, i \in N \) and \( t^D_i, i \in N \) represent the time of availability at node \( i \) for the truck and the drone, respectively. For the starting depot we fix \( t^T_0 = t^D_0 = 0 \). Note that the drone and the truck may leave uncoupled at the starting depot, this is modeled allowing the drone to wait on the ground at customer nodes after the delivery. The waiting of the truck (resp. of the drone) at a node \( i \) is modeled as a delayed availability and included in \( t^T_i \) (resp. \( t^D_i \)).

The MC formulation uses two sets of auxiliary variables. Variables \( u_i, i \in N \), \( 1 \leq u_i \leq c + 2 \) is used to model subtour elimination constraints for the track route, as in Miller–Tucker–Zemlin (see, e.g., Miller et al. [35]). Finally, the binary variable \( p_{ij} \) is set to 1 if customer \( i \in C \) is visited before customer \( j \in C \) \((j \neq i)\), thus defining a total ordering among all the pair of customers. We set \( p_{0j} = 1, j \in C \) to impose at the route starts from the depot.
The resulting model for the MC formulation is described in the following, presenting one group of logically related constraints at a time.

**Objective function**
The objective function (1) minimizes the arrival time at the depot of the truck, but due to next constraints (12) and (13) this is equivalent to minimize \( \max\{t^T_{c+1}, t^D_{c+1}\} \).

\[
\min t^T_{c+1} \tag{1}
\]

**Customer covering**
The first constraints impose that all customers must be served either by the truck or by the drone:

\[
\sum_{(i, j) \in A} x_{ij} + \sum_{(i, j, k) \in F} y_{ijk} = 1 \quad j \in C \tag{2}
\]

**Truck routing constraints**
Constraints (3) and (4) force the truck to depart from depot 0 and to return to depot \( c+1 \) at the end of the trip, while constraints (5) guarantee the flow conservation at customers.

\[
\sum_{j \in N_+} x_{0j} = 1 \tag{3}
\]

\[
\sum_{i \in N_0} x_{i, c+1} = 1 \tag{4}
\]

\[
\sum_{(i, j) \in A} x_{ij} = \sum_{(j, k) \in A} x_{jk} \quad j \in C \tag{5}
\]

**Single sortie launch/rendezvous**
Constraints (6) and (7) impose that at most one launch and one rendezvous is done in any node. Note that the drone cannot start a sortie in \( c+1 \) or return to 0.

\[
\sum_{(i, j, k) \in F} y_{ijk} \leq 1 \quad i \in N_0 \tag{6}
\]

\[
\sum_{(i, j, k) \in F} y_{ijk} \leq 1 \quad k \in N_+ \tag{7}
\]

**x-y coupling constraints**
In constraints (8) we impose that, if the triplet \( \langle i, j, k \rangle \) is selected, then the truck must enter in node \( i \) and in node \( k \) to launch and collect the drone. Constraints (9) is the equivalent of (8) when the drone is launched from the depot.

\[
2y_{ijk} \leq \sum_{(h, i) \in A} x_{hi} + \sum_{(l, k) \in A} x_{lk} \quad \langle i, j, k \rangle \in F, i \neq 0 \tag{8}
\]

\[
y_{0jk} \leq \sum_{(h, k) \in A} x_{hk} \quad \langle 0, j, k \rangle \in F \tag{9}
\]
Truck-drone timing constraints

Constraints from (10) to (13) ensure time synchronization between the truck and the drone at launching and rendezvous nodes. Constraints (14) and (15) set the starting time at depot of both drone and truck to zero. Note that \( M \) is an arbitrarily large number.

\[
\begin{align*}
t^D_i & \geq t^T_i - M \left( 1 - \sum_{(i,j,k) \in F} y_{ijk} \right) & i \in C \\
t^D_i & \leq t^T_i + M \left( 1 - \sum_{(i,j,k) \in F} y_{ijk} \right) & i \in C \\
t^D_k & \geq t^T_k - M \left( 1 - \sum_{(i,j,k) \in F} y_{ijk} \right) & k \in N_+ \\
t^D_k & \leq t^T_k + M \left( 1 - \sum_{(i,j,k) \in F} y_{ijk} \right) & k \in N_+ \\
t^D_0 & = 0 \\
t^D_0 & = 0
\end{align*}
\]

Truck timing constraints

Constraints (16) state that if the arc \((h, k) \in A\) is used by the truck, then the timing variables must be consistent with the traveling times and the launch and rendezvous times, if drone is used. We will discuss these constraints in detail in Sect. 4.2.

\[
t^T_k \geq t^T_h + t^T_{hk} + \sigma_R \sum_{(i,j,k) \in F} y_{ijk} + \sigma_L \sum_{(k,l,m) \in F} y_{klm} - M(1 - x_{hk}) \quad (h, k) \in A
\]

Drone timing constraints

Constraints (17) and (18) impose consistency on the drone timing variables when a sortie \((i, j, k)\) is selected.

\[
\begin{align*}
t^D_j & \geq t^D_i + t^D_{ij} - M \left( 1 - \sum_{(i,j,k) \in F} y_{ijk} \right) & (i, j) \in A, j \in C' \\
t^D_k & \geq t^D_j + t^D_{jk} + \sigma_R - M \left( 1 - \sum_{(i,j,k) \in F} y_{ijk} \right) & (j, k) \in A, j \in C'
\end{align*}
\]

Note that these constraints allow \( t^D_j \) to be greater than the arrival time at customer \( j \), and consequently to insert a waiting time at \( j \).

Drone battery endurance constraints

Constraints (19) are the battery endurance constraints, for each sortie.

\[
t_k^D - (t_j^D - t_{ij}^D) \leq E + M(1 - y_{ijk}) \quad (i, j, k) \in F
\]
We observe that inequalities (19) do not consider the waiting at \( j \) in the computation of the energy consumption, i.e., we can therefore suppose that the drone is allowed to wait on the ground.

**Miller–Tucker–Zemlin subtour elimination**

Constraints (20) are the classical Miller–Tucker–Zemlin constraints for the truck route, while (21) are the extension to sorties, and impose that node \( i \) precedes node \( k \) if sortie \( \langle i, j, k \rangle \) is selected.

\[
\begin{align*}
  u_i - u_j + 1 &\leq (c + 2)(1 - x_{ij}) \quad (i, j) \in A, i \in N_+ \\
  u_k - u_i &\geq 1 - (c + 2) \left( 1 - \sum_{(i, j, k) \in F} y_{ijk} \right) \quad i \in C, k \in N_+, k \neq i
\end{align*}
\]

**Node total ordering**

Constraints (22) and (23) induce variables \( p_{ij} \) to define a total ordering of the nodes.

\[
\begin{align*}
  p_{ij} + p_{ji} &= 1 \quad (i, j) \in A, i < j \\
  p_{0j} &= 1 \quad j \in C
\end{align*}
\]

**u-p congruence**

These constraints impose the consistency between the ordering given by \( u \) and \( p \) variables.

\[
\begin{align*}
  u_i - u_j &\geq 1 - (c + 2)p_{ij} \quad i, j \in C, j \neq i \\
  u_i - u_j &\leq -1 + (c + 2)(1 - p_{ij}) \quad i, j \in C, j \neq i
\end{align*}
\]

**Simple crossing sorties elimination**

Constraints (26) avoid that a sortie starts before a previous sortie is terminated. More specifically, if node \( i \) is visited by the truck before node \( l \) (i.e., \( p_{il} = 1 \)) and there are two sorties starting from \( i \) and \( l \), respectively, then the rendezvous node \( k \) of the sortie launched in \( i \), must be visited before node \( l \).

\[
\begin{align*}
  t_i^D &\geq t_k^D - M \left( 3 - \sum_{(i, j, k) \in F} y_{ijk} - \sum_{(l, m, n) \in F} y_{lmn} - p_{il} \right) \\
  i &\in N_0, l \in C, k \in N_+, i \neq l, k, l \neq k
\end{align*}
\]

**Variable bounds**

\[
\begin{align*}
  t_i^T &\geq 0 \quad i \in N \\
  t_i^D &\geq 0 \quad i \in N \\
  p_{ij} &\in \{0, 1\} \quad (i, j) \in A
\end{align*}
\]
We propose a formulation called $\text{MC}$ built upon MC with just a set of modifications inserted to provide a more realistic interpretation of the problem description given by Murray and Chu.

Let us consider the truck timing constraints (16), which applies to a truck running along arc $(h, k) \in A$. Let us refer to Fig. 1, showing a solution where the drone performs a sortie which starts from $i$, a node that precedes $h$ on the truck route, terminates in $k$, and immediately starts a second sortie from $k$. If this happens, constraint (16) imposes $t_{hk}^T \geq t_h^T + \tau_{hk}^T + \sigma_R + \sigma_L$, where $\sigma_R$ and $\sigma_L$ refer, respectively, to the rendezvous and launch times arriving in $k$ and restarting from $k$. Allocating the rendezvous time to $t_{hk}^T$ is correct, but we believe that the launch time must be allocated to the time of the node visited after $k$ in the truck route, otherwise the launch time needed for sortie $\langle k, l, m \rangle$ would be included in the flying time of sortie $\langle i, j, k \rangle$, and that could erroneously make exceed its endurance. Moreover, we believe that, to be consistent with the problem definition, the launch time $\sigma_L$ must not be allocated to the truck when the drone starts from the depot, since depot operations are considered to be done off-line. Therefore, we substituted (16) with (33) and (34).

Moreover, we believe that $\sigma_L$ must be included in the drone timing when the drone is launched from a node different from the depot, being the time when the drone is on the truck and the operator prepares it. We thus substitute the first drone timing constraint (17) with (35) and (36).
New formulations for the FSTSP

\[ t_j^D \geq t_i^D + \tau_{ij}^D + \sigma_L - M \left( 1 - \sum_{(i,j,k) \in F} y_{ijk} \right) \quad (i, j) \in A, \ j \in C', i \in C \] (35)

\[ t_j^D \geq t_0^D + \tau_{0j}^D - M \left( 1 - \sum_{(0,j,k) \in F} y_{0jk} \right) \quad j \in C' \] (36)

### 4.2.1 Wait and no-wait models

As for the original MC formulation, constraints (35) and (36) allow \( t_j^D \) to be greater than the arrival time at customer \( j \), in a sortie \( \langle i, j, k \rangle \), without incurring in its computation while calculating energy consumption (19). We refer to this model as the wait model, because the drone is allowed to wait on the ground after serving a customer. To allow for the ‘wait’ case in the MC formulation we need to change the constraints (19) into

\[ t_k^D - (t_j^D - \tau_{ij}^D) \leq E + \sigma_L + M(1 - y_{ijk}) \langle i, j, k \rangle \in F, i \neq 0 \] (37)

\[ t_k^D - (t_j^D - \tau_{0j}^D) \leq E + M(1 - y_{0jk}) \langle 0, j, k \rangle \in F \] (38)

From our experience, the waiting is not always technically possible. In this case the no-wait model is obtained by substituting the drone battery endurance constraints (37) and (38) with (39) and (40).

\[ t_k^D - t_i^D \leq E + \sigma_L + M(1 - y_{ijk}) \langle i, j, k \rangle \in F, i \neq 0 \] (39)

\[ t_k^D - t_0^D \leq E + M(1 - y_{0jk}) \langle 0, j, k \rangle \in F \] (40)

Note that our reformulation is similar, but slightly different from the one proposed by Ponza [40]. Indeed, in [40] the launch time is allocated to both truck and drone, so it enters in the computation of the energy consumption, while we suppose that launching does not consume energy, being the time needed by the truck to make the drone ready, while the drone does not fly. Moreover, Ponza [40] includes the launch time in the computation also when the sortie starts from the depot, while we use the original MC problem definition that assumes depot operations are done off-line. With the ‘no-wait’ model we actually cannot make truck and drone leave uncoupled from the depot, because we set \( t_0^T = t_0^D = 0 \) and the drone is not allowed to wait at customers.

### 5 Improved formulation DMN

In this section we present a first enhanced formulation, built upon formulation MC of Sect. 4.2. The general idea is to substitute some explicit constraints with exponentially many constraints to be added in a cutting plane fashion and to provide an improved objective function.
5.1 Crossing sorties elimination

The $\text{MC}$ model uses the explicit simple crossing sorties elimination (26) which are based on timing and the use of a ‘big $M$’ constant. We propose to substitute them with the following structural constraints that directly address and prevent the unfeasible topologies. Let $i \in N_0, l \in C$ be two nodes visited by the truck while running along a path $P$ from $i$ to $l$, and assume that two sorties $\langle i, j, k \rangle$ and $\langle l, m, n \rangle$ with $k \notin P$ exist. In this case, the second sortie starts before the first one returns to the truck and the overall solution is therefore infeasible. Let us define $P$ as the set of all the paths with these characteristics. The following crossing sorties elimination constraints (CSEC) (41) can be used to eliminate this type of infeasibility.

$$\left| P \right| - 1 \sum_{h=1}^{\left| P \right| - 1} x_{v(h), v(h+1)} + \sum_{\langle i, j, k \rangle \in F \atop k \notin P} y_{ijk} + \sum_{\langle l, m, n \rangle \in F} y_{lmn} \leq \left| P \right| \quad P \in \mathcal{P} \quad (41)$$

where $P = \{v(1), v(2), \ldots, v(\left| P \right|)\}$, $v(1) = i$, $v(\left| P \right|) = l$. These constraints impose that at least one of the variables representing the arcs of the path and the sorties are set to zero to make the solution feasible.

We can strengthen (41) using the idea of “tournament constraint”. Since the path enters at most once in each node, we can include in the constraint also the arcs $(h, j) \in A$ such that $h, j \in P$ and $h$ precedes $j$ in $P$. We obtain the tournament crossing constraints (TCS), given by (42).

$$\sum_{h=1}^{\left| P \right| - 1} \sum_{j=h+1}^{\left| P \right|} x_{v(h), v(j)} + \sum_{\langle i, j, k \rangle \in F \atop k \notin P} y_{ijk} + \sum_{\langle l, m, n \rangle \in F} y_{lmn} \leq \left| P \right| - 1 \quad P \in \mathcal{P} \quad (42)$$

5.2 Backward sorties elimination

A next improvement is the avoidance of backward sorties, that are sorties with a rendezvous that happens before the departure. Let $\mathcal{B}$ denote the set of all truck paths $P = \{v(1), v(2), \ldots, v(q)\}$ with $v(1) = 0$ and $v(q) \in C$. Given one of such paths $P$ suppose that exists a sortie $\langle i, j, v(q) \rangle$ with $i \notin P$. The backward sortie elimination constraints (BSEC) are given by (43), and impose that at least one of the arcs of the path or the sortie is eliminated.

$$\sum_{i=1}^{\left| P \right| - 1} x_{v(i), v(i+1)} + \sum_{\langle i, j, v(q) \rangle \in F} y_{ijv(q)} \leq \left| P \right| - 1 \quad P \in \mathcal{B} \quad (43)$$

Constraint (43) can be strengthen by considering a tournament type constraint on path $P$, which imposes that at most one arc enters a node of the path, and adding all sorties that terminate in $P$, but start at nodes outside $P$. We obtain the tournament backward constraints (TBS), given by (44).
\[
\sum_{i=1}^{P} \sum_{j=i+1}^{P} x_{v(i)v(j)} + \sum_{(i,j,k) \in F} y_{ijk} \leq |P| - 1 \quad P \in \mathcal{B}
\] (44)

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

### 5.3 Improving the objective function

Preliminary computational results showed that formulation MC, and thus \(\overline{MC}\), provides very bad lower bounds at the root node of the decision-tree (see Sect. 7 for more details). Therefore, we reformulate the objective function \(\min_{t_{c+1}} t\) by explicating its components: the traveling time of the truck, the launching and rendezvous times, and the time in which the truck waits for the drone. The first two components can be modeled using the variables we already have available, but the waiting times must be expressed by new variables. We add variables \(w_i, i \in N\), to model the truck waiting at node \(i\), thus giving the new objective function (45).

\[
\min \sum_{(i,j) \in \mathcal{A}} \tau_{ij} x_{ij} + \sigma_R \sum_{(0,j,k) \in \mathcal{F}} y_{0jk} + (\sigma_L + \sigma_R) \sum_{(i,j,k) \in \mathcal{F}, i \neq 0} y_{ijk} + \sum_{i \in N} w_i
\] (45)

The waiting variables must be included in the truck timing constraints as in (46)–(49). Note that we use a pair of constraints with opposite sign to impose equality when an arc \((h, k)\) is selected. Non-negativity of the variables \(w_i\) must also be imposed.

\[
t_k^T \geq t_h^T + \tau_{hk}^T + \sigma_R \sum_{(i,j,k) \in \mathcal{F}} y_{ijk} + \sigma_L \sum_{(h,r,s) \in \mathcal{F}, r \neq k} y_{hrs} - M(1 - x_{hk}) + w_k \quad (h, k) \in \mathcal{A}, h \neq 0
\] (46)

\[
t_k^T \leq t_h^T + \tau_{hk}^T + \sigma_R \sum_{(i,j,k) \in \mathcal{F}} y_{ijk} + \sigma_L \sum_{(h,r,s) \in \mathcal{F}, r \neq k} y_{hrs} + M(1 - x_{hk}) + w_k \quad (h, k) \in \mathcal{A}, h \neq 0
\] (47)

\[
t_k^T \geq t_0^T + \tau_{0k}^T + \sigma_R \sum_{(0,j,k) \in \mathcal{F}} y_{0jk} - M(1 - x_{0k}) + w_k \quad (0, k) \in \mathcal{A}, k \in \mathcal{N}_+
\] (48)

\[
t_k^T \geq t_0^T + \tau_{0k}^T + \sigma_R \sum_{(0,j,k) \in \mathcal{F}} y_{0jk} + M(1 - x_{0k}) + w_k \quad (0, k) \in \mathcal{A}, k \in \mathcal{N}_+
\] (49)

\[
w_i \geq 0 \quad i \in \mathcal{N}
\] (50)
5.4 Subtour elimination constraints

When avoiding crossing sorties by using the CSEC or TCS, Miller–Tucker–Zemlin constraint are not needed; indeed, total node ordering variables are not needed and timing constrains are sufficient to prevent subtours; however, the introduction of subtour elimination constraints (51) and the two nodes subtours elimination (52) is profitable. Preliminary computational results confirmed us the reduction of the number of visited nodes by the branch-and-bound and the computing time even if applied to the MD formulation.

\[
(\text{SEC}) \sum_{i \in S} \sum_{j \in S \setminus \{i, j\} \in A} x_{ij} \leq |S| - 1 \quad S \subseteq N, S \neq \emptyset, |S| > 2 \quad (51)
\]

\[
x_{ij} + x_{ji} \leq 1 \quad (i, j) \in A \quad (52)
\]

The overall formulation, called DMN in the following, uses all the improvements of this section with the “tournament” version for crossing sorties (42) and backward sorties (44). The DMN formulation can be summarized in (2)–(15), (27)–(31), (35), (36), (42), (44)–(52) plus constraints (37) and (38) in the, ‘wait’ case and (39) and (40) in the ‘no-wait’, case.

6 Two-indexed formulations: DMN2

In the previous models we represented the sorties with three-indexed boolean variables, hereby we propose a new formulation which uses two-indexed arc variables, instead. In particular, let us consider the binary variables $\vec{g}_{ij}$ and $\vec{g}_{jk}$ which take value 1 if the drone enters, respectively leaves, customer node $j$. In this way the number of variables representing the sorties reduces from $n^3$ to $2n^2$. To further reduce the number of variables we preliminary fix to zero all the variables corresponding to arcs with flying time exceeding the battery limit, i.e., we set $\vec{g}_{ij} = 0$ for all $(i, j) \in A : \tau_{ij}^{D} > E$ and $\vec{g}_{jk} = 0$ for all $(j, k) \in A : \tau_{jk}^{D} + \sigma_{R} > E$. We also fix to zero variables that do not allow to complete a feasible drone flight: $\vec{g}_{ij} = 0, (i, j) \in A, j \notin C'$; $\vec{g}_{jk} = 0, (j, k) \in A, j \notin C'$; $\vec{g}_{i, c+1} = 0, i \in N$ and $\vec{g}_{j0} = 0, j \in N$.

Starting from model DMN, we build the new formulation DMN2 using the above two-indexed variables. The objective function is:

\[
\min \sum_{(i, j) \in A} \tau_{ij}^{T} x_{ij} + \sum_{(i, j) \in A} \sigma_{L} \sum_{i \neq 0} \vec{g}_{ij} + \sum_{(j, k) \in A} \vec{g}_{jk} + \sum_{i \in N} w_{i} \quad (53)
\]

All the constraints using $y$ variables must be rewritten using the $\vec{g}$ and $\vec{g}$ variables, as follows. Customer covering constraints (2) become

\[
\sum_{(i, j) \in A} x_{ij} + \sum_{(i, j) \in A} \vec{g}_{ij} = 1 \quad j \in C \quad (54)
\]
\[
\sum_{(i,j) \in A} x_{ij} + \sum_{(i,j) \in A} \overrightarrow{g}_{ij} = 1 \quad i \in C \quad (55)
\]

Single sortie launch/rendezvous constrains (6) and (7) are no longer necessary, since they are induced by the truck routing constraints (3)–(5) and by the variables coupling constraints, now called \textit{x-g coupling constraints}:

\[
\sum_{(i,j) \in A} \overrightarrow{g}_{ij} \leq \sum_{(i,h) \in A} x_{ih} \quad i \in N_0 \quad (56)
\]

\[
\sum_{(i,j) \in A} \overrightarrow{g}_{ij} \leq \sum_{(h,j) \in A} x_{hj} \quad j \in N_+ \quad (57)
\]

\[
\sum_{(i,j) \in A} \overrightarrow{g}_{ij} = \sum_{(j,k) \in A} \overrightarrow{g}_{jk} \quad j \in C' \quad (58)
\]

Drone-truck timing constraints (10)–(13) are substituted by

\[
t^D_i \geq t^T_i - M \left(1 - \sum_{(i,j) \in A} \overrightarrow{g}_{ij}\right) \quad i \in C \quad (59)
\]

\[
t^D_i \leq t^T_i + M \left(1 - \sum_{(i,j) \in A} \overrightarrow{g}_{ij}\right) \quad i \in C \quad (60)
\]

\[
t^D_k \geq t^T_k - M \left(1 - \sum_{(j,k) \in A} \overrightarrow{g}_{jk}\right) \quad k \in N_+ \quad (61)
\]

\[
t^D_k \leq t^T_k + M \left(1 - \sum_{(j,k) \in A} \overrightarrow{g}_{jk}\right) \quad k \in N_+ \quad (62)
\]

Truck timing constraints (46)–(49) become

\[
t^{T}_k \geq t^{T}_h + \tau^{T}_{hk} + \sigma_R \sum_{(j,k) \in A} \overrightarrow{g}_{jk} + \sigma_L \sum_{(h,l) \in A} \overrightarrow{g}_{hl} - M(1 - x_{hk}) + w_k \quad \text{if} \quad (h,k) \in A, h \neq 0 \quad (63)
\]

\[
t^{T}_k \leq t^{T}_h + \tau^{T}_{hk} + \sigma_R \sum_{(j,k) \in A} \overrightarrow{g}_{jk} + \sigma_L \sum_{(h,l) \in A} \overrightarrow{g}_{hl} + M(1 - x_{hk}) + w_k \quad \text{if} \quad (h,k) \in A, h \neq 0 \quad (64)
\]

\[
t^{T}_k \geq t^{T}_0 + \tau^{T}_{0k} + \sigma_R \sum_{(j,k) \in A} \overrightarrow{g}_{jk} - M(1 - x_{0k}) + w_k \quad k \in N_+ \quad (65)
\]

\[
t^{T}_k \leq t^{T}_0 + \tau^{T}_{0k} + \sigma_R \sum_{(j,k) \in A} \overrightarrow{g}_{jk} + M(1 - x_{0k}) + w_k \quad k \in N_+ \quad (66)
\]
Drone timing constraints (35), (36) and (18) are now

\[ t_j^D \geq t_i^D + \tau_{ij}^D + \sigma_L - M (1 - \overline{g}_{ij}) \quad j \in C', i \in C, i \neq j \]  
\[ t_j^D \geq t_0^D + \tau_{0j}^D - M (1 - \overline{g}_{0j}) \quad j \in C' \]  
\[ t_k^D \geq t_j^D + \tau_{jk}^D + \sigma_R - M \left( 1 - \sum_{(j,k) \in A} \overline{g}_{jk} \right) \quad j \in C', k \in N_+, j \neq k \]  

Drone battery endurance constraints for the ‘no-wait’ case, (39) and (40), become:

\[ t_k^D - t_i^D \leq E + \sigma_L + M (2 - \overline{g}_{ij} - \overline{g}_{jk}) \]  
\[ (i, j) \in A, (j, k) \in A, j \in C', i \in N_+, i \neq k \]  
\[ t_j^D - t_0^D \leq E + M (2 - \overline{g}_{0j} - \overline{g}_{jk}) \quad (j, k) \in A, j \in C' \]  

By changing the left hand side of (70) with \( t_k^D - (t_j^D - \tau_{ij}^D) \) and that of (71) with \( t_k^D - (t_j^D - \tau_{0j}^D) \) we obtain the drone battery endurance constraints for the ‘wait’ case.

Tournament crossing constraints (42) 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 sortie launches defined by \( \overline{g}_{ij} > 0 \) and \( \overline{g}_{lm} > 0 \) and such that there is no node \( k \in P \setminus \{i\} \) with \( \overline{g}_{ik} > 0 \). In this case the second sortie starts before the first sortie is terminated and the following “tournament” crossing sorties elimination holds:

\[ \text{(TCS2)} \sum_{h=1}^{P-1} \sum_{j=h+1}^{|P|} x_{v(h)v(j)} + \sum_{(i,j) \in A, j \notin P} \overline{g}_{ij} + \sum_{(l,j) \in A, j \notin P} \overline{g}_{lj} \leq |P| \quad P \in P \]  

where \( P = \{v(1), v(2), \ldots, v(q)\} \) with \( v(1) = i, v(q) = l \), and \( P \) now defines the set of all the paths with the described characteristics.

Tournament backward constraints 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 return identified by \( \overline{g}_{ki} > 0 \) and such that \( k \notin P \) and a sortie launch identified by \( \overline{g}_{ij} > 0 \) and such that \( k \notin P \), that, together determine an infeasible solution. Let \( B \) now denote the set of all truck paths \( \{v(1), \ldots, v(m), \ldots, v(q)\} \) with \( v(1) = 0 \), \( v(m) = i \), and \( v(q) = j \). The tournament version of the backward sorties elimination constraints is:

\[ \text{(TBS2)} \sum_{h=1}^{P-1} \sum_{l=h+1}^{|P|} x_{v(h)v(l)} + \sum_{(l,k) \in A, l \notin P} \overline{g}_{lk} + \sum_{(k,l) \in A, l \in P} \overline{g}_{kl} \leq |P| \quad P \in B, k \notin P \]  

\( \copyright \) Springer
We finally need to impose the following constraints to avoid infeasibilities:

\[
\begin{align*}
\vec{g}_{ij} + \vec{g}_{ji} & \leq 1 \quad (i, j) \in A \\
\vec{g}_{ij} + \vec{g}_{ji} & \leq 1 \quad (i, j) \in A
\end{align*}
\] (74) (75)

Formulation **DMN2** is thus composed by (3)–(5), (14), (15), (27), (28), (30), (50)–(69), (72)–(75) plus (70) and (71) for the ‘no-wait’ case, or their straightforward modification for the ‘wait’ case.

### 7 Computational experiments

To test the above models we have implemented them to run on an Intel Core i3-2100 CPU, with 3.10 GHz and 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.

Formulation **MC** was solved directly by CPLEX, while **DMN** and **DMN2** required a branch-and-cut implementation, since they include exponentially many constraints. To separate these constraints we considered the residual graph \(G' = (N, A')\) obtained from \(G\) (see Sect. 4) by selecting only the arcs associated with a non zero variable \((x, y, \vec{x}_{ij} \text{ and } \vec{y}_{ij})\) in the continuous relaxation of the model. For the crossing sorties elimination (CSEC, TSC, and TSC2) constraints and for the backward sorties elimination (BSEC, TBS, and TBS2) constraints we simply explore the graph starting from depot 0, until a truck path violating one of the constraints is identified, if any. The overall procedure has a time complexity \(O(|A'|)\) when separating integer solutions. While when separating fractional solutions the complexity can be higher, since we separate them exactly. To separate the subtour elimination constraints (SEC) we use the standard approach which requires to solve at most \(O(n^2)\) max flow problems on the residual graph.

To facilitate the solution of the problem, we give all the algorithms an initial heuristic solution computed as follows. We first find a heuristic (TSP) solution in which all the customers are served by the truck. We build this solution with a greedy constructive algorithm followed by a local search improvement using as moves to define neighboring solutions the relocation of one customer and the swap of two customers. Next we examine all the possible sorties and we select the one, if any, that improves the solution. We repeat the search for improving sorties (and update of the solution) until no one is found.

We based our tests on the 72 benchmark instances provided by Murray and Chu [36]. In each of these, 10 customers are randomly distributed across an eight-mile square region, while the depot is located in four different positions. The endurance of the drone was chosen to be either 20 or 40 min, while the speed of the drone was selected to be 15, 25, 35 miles/h based on Euclidean distances. The truck speed was assumed to be 25 miles/h and based on Manhattan distances. The launch and rendezvous times are set to 1 min. In the first 24 instances it has been set a ratio \(|C'|/|C| = 90\%\), while for the remaining ones it has been set to 80\%. We tested the formulations on each instance for both the ‘no-wait’ case (in which the drone is not allowed to wait
at a customer), and the ‘wait’ case, adopted in [36], in which waiting is allowed, but not considered in the computation of the battery endurance (see Sect. 1). Overall 144 instances are available in our test bed.

The benchmark instances make use of symmetric time matrices; however, the proposed models can as well accommodate asymmetric instances. In this sense we believe that testing our formulations on asymmetric instances could be an interesting insight. We thus generated asymmetric instances starting from the Murray and Chu [36] benchmark instances; in particular, we made the truck travel matrix asymmetric. To do so, for each arc \((i, j) \in A\), we randomly generated a number \(\zeta_{ij} \in [-10\%, 10\%]\) and modified the truck time matrix such that:

\[
\tau_{ij, asymm} = (1 + \zeta_{ij}) \cdot \tau_{ij}.
\]

The interested reader can find all the results, solutions, and newly generated instances on our web site www.or.unimore.it following the online resources link.

### 7.1 Tests

Table 2 presents the performances of the lower bounds for different formulations. The entries of the table are the average percentage gaps over the instances referred in each row, computed as \(100 \cdot (\text{opt} - \text{LB})/\text{opt}\), where ‘opt’ is the value of the optimal solution obtained giving to formulation DMN2 enough time to solve all instances to proven optimality, and LB is a lower bound to the optimal solution. The first column, labeled ‘E’, shows the battery endurance value, while column labeled ‘speed’ refers to the drone speed. Column ‘MC’ displays the lower bound gap obtained by the Murray–Chu formulation, modified as in Sect. 4.2. All the other columns give the gap computed with the lower bound obtained at the root node. Columns labeled ‘newF’ (new objective function) refer to formulation MC modified by adopting the objective function introduced in Sect. 5. Columns labeled ‘newF2’ refer to formulation newF modified by substituting the three-indexed variables \(y\) with the two-indexed variables \(g\) and \(g\) as in Sect. 6. Finally, ‘DMN’ and ‘DMN2’ report on the lower bound gaps of our complete formulations.

Formulation MC clearly gives bad results, in terms of lower bound gap, providing a gap close to 80% for all classes of instances. The adoption of the new objective function greatly improves the behavior, reducing the gap drastically to about 22%. The use of two-indexed variables and complete formulations further improve the gap. We observe that instances in which the drone is allowed to wait at customers have slightly smaller gaps.

Table 3 gives the same data as in Table 2, but grouped by depot position. Column labeled ‘dep’ refers to the four depot positions, with respect to the square in which the customers are generated. 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, on the bottom border and below the bottom border at a distance which is equal to the distance of the barycenter from this border. One can see that the worst bound gaps are obtained when the depot is in position ‘b’ and improve when the depot is far from the barycenter (positions ‘c’ and ‘d’). The best gaps are obtained when the depot is outside the square, that is far
Table 2  Lower bounds at the root node, by speed class

| E speed | No-wait | Wait |
|---------|---------|------|
|         | MC      | newF | newF2 | DMN  | DMN2 | MC    | newF | newF2 | DMN  | DMN2 |
| 20 15   | 83.31   | 14.38| 13.71 | 9.35 | 8.46 | 83.42 | 13.83| 13.71 | 9.35 | 8.46 |
| 25      | 81.99   | 26.60| 25.30 | 23.10| 18.57| 81.39 | 23.99| 21.79 | 20.25| 15.36|
| 35      | 79.72   | 19.47| 19.14 | 16.12| 11.41| 79.64 | 19.21| 18.56 | 15.99| 11.12|
| 40 15   | 82.86   | 29.95| 29.27 | 26.53| 22.90| 82.78 | 29.77| 29.30 | 27.16| 23.06|
| 25      | 80.68   | 23.58| 21.94 | 20.05| 16.06| 80.68 | 23.58| 22.55 | 20.28| 15.80|
| 35      | 79.66   | 20.93| 19.85 | 18.22| 12.60| 79.65 | 20.79| 19.61 | 17.82| 12.49|
| All     | 81.37   | 22.49| 21.53 | 18.89| 15.00| 81.26 | 21.86| 20.76 | 18.36| 14.11|

Table 3  Lower bounds at the root node, by depot location

| E dep | No-wait | Wait |
|-------|---------|------|
|       | MC      | newF | newF2 | DMN  | DMN2 | MC    | newF | newF2 | DMN  | DMN2 |
| 20 a  | 89.87   | 20.97| 20.24 | 15.88| 13.79| 89.27 | 19.10| 17.17 | 13.80| 11.21|
| b     | 88.08   | 25.19| 24.25 | 21.57| 16.28| 88.14 | 24.16| 22.81 | 19.86| 14.72|
| c     | 82.06   | 19.03| 17.86 | 14.53| 11.10| 81.94 | 18.46| 17.67 | 14.91| 9.95 |
| d     | 66.69   | 15.41| 15.18 | 12.78| 10.08| 66.18 | 14.31| 13.11 | 11.31| 8.55 |
| 40 a  | 90.08   | 26.65| 25.03 | 23.00| 18.71| 90.07 | 26.46| 24.60 | 23.07| 18.84|
| b     | 87.41   | 29.92| 28.72 | 26.86| 21.08| 87.41 | 29.92| 28.72 | 27.55| 20.75|
| c     | 81.30   | 23.49| 21.88 | 20.16| 16.01| 81.29 | 23.46| 23.42 | 19.87| 15.79|
| d     | 65.46   | 19.21| 19.11 | 16.38| 12.95| 65.37 | 19.01| 18.53 | 16.51| 13.08|

Fig. 2  Comparison of lower bound gaps at the root node for the proposed methods
from all customers. In this case there is a sort of ‘offset’ distance that must be covered by any route to reach the customers, and this improves the bound.

Figure 2 provides a graphical representation of the lower bound gaps for the ‘no-wait’ instances. One can see that the lower bound gap at the root node is improved largely by the proposed methods with respect to the original model with no difference based on endurance and speed class. The reader can notice that each of the improvement features applied provides a relevant contribution to the results.

Table 4 shows the computing times in seconds needed for solving the different formulations at the root node. The reported result is the average computed among all the instances, because there is no evident variation for different instance settings such as endurance, depot location, or speed. The main result one can note is that DMN2 is

| Table 4 Computing times (seconds) at the root node for the different formulations |
|---------------------------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| MC | newF | newF2 | DMN | DMN2 |
| Wait | 1.10 | 1.88 | 0.33 | 1.23 | 0.49 |
| No-wait | 1.21 | 1.84 | 0.33 | 1.47 | 0.45 |

| Table 5 Exact solutions for ‘wait’ instances, by speed class |
|---------------------------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| E | sp | gap% | opt | nodes | gap% | opt | time | nodes | gap% | opt | time | nodes |
| 20 | 15 | 83.65 | 0 | 180356.6 | 0.00 | 12 | 10.2 | 379.9 | 0.00 | 12 | 6.2 | 340.4 |
| 25 | 82.07 | 0 | 159165.3 | 8.25 | 11 | 797.1 | 17056.3 | 1.75 | 11 | 503.1 | 19934.1 |
| 35 | 80.94 | 0 | 142095.8 | 11.53 | 9 | 1342.2 | 24671.7 | 8.37 | 10 | 920.5 | 35803.4 |
| 40 | 15 | 83.70 | 0 | 155622.6 | 10.34 | 2 | 3313.3 | 54078.5 | 10.87 | 9 | 2247.2 | 92228.7 |
| 25 | 81.85 | 0 | 144169.7 | 11.25 | 4 | 2533.3 | 46797.5 | 10.34 | 7 | 2033.7 | 83021.6 |
| 35 | 80.61 | 0 | 139364.6 | 10.40 | 8 | 1433.6 | 25229.2 | 8.04 | 10 | 1020.6 | 44946.3 |
| All | 82.14 | 0 | 153462.4 | 10.24 | 46 | 1571.7 | 28035.5 | 9.15 | 59 | 1121.9 | 46045.8 |

| Table 6 Exact solutions for ‘wait’ instances, by depot location |
|---------------------------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| E | dep | gap% | opt | nodes | gap% | opt | time | nodes | gap% | opt | time | nodes |
| 20 | a | 90.29 | 0 | 161330.4 | 11.84 | 6 | 1496.6 | 26211.3 | 6.17 | 6 | 1293.5 | 41864.8 |
| b | 88.41 | 0 | 162479.0 | 7.34 | 8 | 1034.0 | 22432.4 | 0.00 | 9 | 559.4 | 29790.4 |
| c | 82.93 | 0 | 157777.2 | 0.00 | 9 | 155.0 | 3771.9 | 0.00 | 9 | 29.2 | 1703.1 |
| d | 67.25 | 0 | 160570.2 | 0.00 | 9 | 180.7 | 3728.2 | 0.00 | 9 | 24.2 | 1412.2 |
| 40 | a | 90.92 | 0 | 145540.9 | 5.48 | 2 | 2937.7 | 49395.2 | 12.57 | 3 | 2662.4 | 86080.9 |
| b | 88.24 | 0 | 145526.4 | 10.68 | 3 | 2555.8 | 46533.9 | 7.67 | 6 | 2081.0 | 86932.6 |
| c | 82.56 | 0 | 151308.9 | 7.31 | 5 | 2033.6 | 35907.6 | 0.00 | 9 | 1064.3 | 53117.3 |
| d | 66.48 | 0 | 143166.2 | 6.38 | 4 | 2180.0 | 36303.6 | 1.96 | 8 | 1261.1 | 67464.7 |
not only the formulation providing better results, but is also very fast at the root node. No relevant differences can be noticed when comparing ‘wait’ and ‘no-wait’ cases.

In Tables 5 and 6 we report on the performances of the formulations in finding the proven optimal solution for the ‘wait’ instances. The values in the table are averaged over the 12 (resp. 9) instances of each class of speed (resp. depot location). The overall results are computed over the 72 instances. In the columns labeled ‘gap%’, ‘opt’, ‘time’, and ‘nodes’, we display, respectively: the percentage gap computed as $100 \cdot (UB - LB)/UB$, being $UB$ the upper bound (averaged only on the instances not solved to optimality), the number of proven optimal solutions, the average computing time in CPU seconds, and the average number of branch-decision-nodes explored. Formulation MC, in one hour of CPU time, was not able to solve any instance given the poor performances of the lower bound, so we do not show the CPU time for this formulation. For formulations DMN and DMN2 we set a time limit of one hour.

Looking at Table 5 we observe that the instances with endurance 20 are much easier than those with endurance 40: in terms of obtained optimal solutions and computing times. This is due to the fact that the number of possible drone sorties is much less in the first case, thus reducing the possible choices of the algorithm. The two-indexed

| $E$ | $sp$ | DMN | DMN2 |
|-----|------|-----|------|
|     |      | gap%| opt | time | nodes | gap%| opt | time | nodes |
| 20  | 15   | 0.00| 12  | 10.6 | 356.9 | 0.00| 12  | 5.3  | 315.3 |
| 25  | 5.36 | 11  | 734.4 | 15527.4 | 4.49| 11  | 508.6 | 21356.8 |
| 35  | 6.53 | 8   | 1268.8 | 20926.7 | 8.53| 10  | 923.4 | 33466.1 |
| 40  | 15   | 10.76| 2   | 3342.9 | 53484.3 | 12.02| 8   | 2258.1 | 97451.6 |
|     | 10.12| 4   | 2571.6 | 40030.3 | 7.32| 5   | 2294.2 | 99665.8 |
| 25  | 10.41| 10  | 1233.0 | 23973.8 | 7.36| 10  | 971.2 | 44651.3 |
| All | 10.02| 46  | 1526.9 | 25716.6 | 8.48| 55  | 1160.1 | 49484.5 |

| $E$ | dep | DMN | DMN2 |
|-----|-----|-----|------|
|     |     | gap%| opt | time | nodes | gap%| opt | time | nodes |
| 20  | a   | 8.94| 6   | 1430.1 | 24584.7 | 7.18| 6   | 1275.5 | 40677.6 |
|     | b   | 2.33| 7   | 1074.60 | 19742.7 | 0.00| 9   | 569.9 | 28377.3 |
|     | c   | 0.00| 9   | 108.5 | 2962.0 | 0.00| 9   | 39.6  | 2517.2 |
|     | d   | 0.00| 9   | 71.9 | 1792.0 | 0.00| 9   | 31.4 | 1945.4 |
| 40  | a   | 15.63| 2   | 3007.2 | 49944.3 | 11.78| 2   | 2816.5 | 97932.2 |
|     | b   | 13.03| 5   | 2320.8 | 41996.6 | 8.80| 6   | 2058.6 | 94786.1 |
|     | c   | 8.09| 5   | 2109.7 | 29472.3 | 2.51| 8   | 1210.4 | 60611.7 |
|     | d   | 5.03| 4   | 2092.2 | 35238.0 | 1.36| 7   | 1279.1 | 69028.3 |

Table 7 Exact solutions for ‘wait’ asymmetric instances, by speed class

Table 8 Exact solutions for ‘wait’ asymmetric instances, by depot location
formulation DMN2 performs better than DMN, since it solves 13 more instances, the unsolved instances have smaller gaps, and also the computing time is smaller.

For the ‘no-wait’ instances the formulations exhibit a very similar behavior that is not reported here (the interested reader can find the related results at www.or.unimore.it).

Looking at Table 6 we can observe that instances of class ‘a’, in which the depot is centrally located, are more difficult to solve than those of the other classes. Class ‘d’, which gave the best results for the root node lower bounds, has, instead, the same difficulty of classes ‘b’ and ‘c’.

Similarly to the previously presented ones, Tables 7 and 8 show, with respect to the drone speed and the depot location, the behavior of formulations DMN and DMN2 on the asymmetric instances. One can see that the behavior is very similar to the symmetric case: DMN2 converges faster than DMN, endurance is an important factor, making the instances with smaller endurance easier to solve, and the instances characterized with a depot location ‘a’ are harder to solve. For sake of brevity, we reported only the ‘wait’ case, but we report the ‘no-wait’ results on our web-site; however, results of the two cases are similar.

Table 9 depicts the difference between the ‘wait’ and ‘no-wait’ solutions with respect to the speed class and the depot location. The entries of Table 9a and 9b, are the endurance, the speed (depot location, respectively), the gap% computed as $100 \cdot \frac{opt^n - opt^w}{opt^n}$ averaged only when the two solutions differ, being $opt^n$
New formulations for the FSTSP

Fig. 3 Comparison between the ‘no-wait’ (a) and ‘wait’ (b) solution of an instance with $E = 20$, depot location ‘d’, and ‘speed’ = 25 miles/h. The solid, dashed, lines represent the truck route, sorties, respectively. The square indicates the depot while the circles indicate the customers.

and $opt^w$ the optimal solution of the ‘no-wait’ and ‘wait’ case, respectively. The last column shows the number of occurrences of a difference between the two optimal solutions. One can note that a more restrictive endurance make the allowance of waiting at customers without flying more important. This appears to be more relevant than the speed class. On the other hand, depot position ‘a’ (centrally located) and ‘d’ (positioned out of the square) have a strong impact on the solution when waiting is allowed. Moreover, based on extensive computational tests, we can state that the ‘wait’ model is only slightly easier to be solved than the ‘no-wait’ one.

Figure 3 shows the difference between a ‘no-wait’ and a ‘wait’ solution. For this particular instance, the gap between the values of the two optimal solutions is around 9%. Note that one more sortie can be performed in the ‘wait’ solution. In particular, allowing to wait at customer 2, the drone can save battery while the truck travels path (0,8,1), whose length would exceed battery endurance if the drone had to wait while flying. Similarly, the ‘wait’ solution can let the drone wait at customer 3, while the truck runs on the path (7,4,10,9), whose time would exceed the battery endurance. In this case the ‘no-wait’ solution performs a shorter sortie, instead; however, this difference does not improve the solution, but it shows, nevertheless, that the instances of the FSTSP can have more than one optimal solution, with the same truck route, and different sorties. This last consideration is one of the reasons why these problems are hard to solve also for small instances and it further justifies the increasing interest of researchers.
Table 10 Comparison between DMN2 and Yurek and Ozmutlu methods for 12 and 13 customers instances

|          | $|C| = 12$ |          | $|C| = 13$ |
|----------|-----------|----------|-----------|
|          | #opt  | Time    | #opt  | Time    | #opt  | Time    |
| DMN2     |       |         | Yurek and Ozmutlu |       |         | DMN2     |       |         |
| Uniform  | 1     | 52.23   | 10     | 760.20   | 2     | 265.91   |
| Centered | 2     | 22.37   | 1      | 2591.00  | 5     | 1014.11  |
| Clustered| 6     | 248.33  | 1      | 694.00   | 4     | 1867.26  |

We finally compare the most promising of our methods with Yurek and Ozmutlu [56], which represents the current state-of-the-art in the literature. The largest instances solved by Yurek and Ozmutlu [56] include 12 customers, have a truck speed of 40 km/h, a drone speed of 56 km/h, and endurance of 20 min, while the other main characteristics are as in Murray and Chu [36]. They separate their instances in uniformly distributed, centered, and clustered ones. For each of these types they present 10 instances. In order to compare our method we generated 10 instances for each type using the same characteristics. In Table 10 we show results for DMN2 and the Yurek and Ozmutlu’s method averaged on the 10 instances for each type, with 12 customers and the drone speed at 56 km/h. Columns ‘#opt’ display the number of optima for each method having a one hour time limit, columns ‘time’ show the computing time in seconds computed only on the instances solved to optimality. For uniform instances with 12 customers, Yurek and Ozmutlu provide better results than DMN2; however, centered and clustered instances are hard to solve by their method. On the other hand, DMN2 provides better results for centered and clustered instances, solving 2 and 6 out of 10 instances to optimality, with respect to 1 each by Yurek and Ozmutlu, in shorter computing times.

Yurek and Ozmutlu [56] also test their method on uniform instances with a drone speed of 40 km/h, comparable with the one of the truck. We thus tested DMN2 on the uniform instances with $|C| = 12$ by setting the drone speed to 40 km/h. Their method performs slightly better than our in terms of optimal solutions, they solve 7 out of 10 with respect to 4 out of 10 instances, underlining once more that DMN2 is more competitive with centered and clusters instances; however, the instances that we can solve are solved with less than a half of their computing times. This shows that our method provides better results for instances having slower drones in case of uniform distributed customers. They test their method on asymmetric instances as well, showing that, in such a case, they need three times the computing times necessary to solve the symmetric instances. Our method appears to be more robust, providing similar results in both cases.

Yurek and Ozmutlu [56] state that their method cannot solve instances with 13 customers, we then decided to test DMN2 on instances with 13 customers, generating 10 instances for each type, as done for the 12 customer instances. One can see in Table 10 that we could solve 11 instances out of 30, confirming that the uniform instances are the hardest to solve, with 2 out of 10 solved to optimality, in comparison to 5 and 4
for the centered and clustered, respectively. This shows that our method can improve on the method proposed in the literature.

8 Conclusions

This paper considered one of the most promising topics in parcel deliveries nowadays, the combined use of traditional vehicles and drones. We focused on one particular problem called the Flying Sidekick Traveling Salesman Problem, in which a drone and a truck are coupled and synchronized, and must serve a set of customers. We provided improvements on the literature by proposing a three and a two-indexed formulations for which we proposed a novel objective function that could increase remarkably the lower bounds, and a set of good inequalities to be separated in a branch-and-cut fashion that provided an important contribution to obtain better solutions in a faster way. Our method outperforms the previously proposed ones, being the two-indexed formulation the preferable one: 59 out of the 72 benchmark instances could be solved to optimality with an average percentage gap between the best lower and upper bound of 1.70% and within an average computing time of less than 20 min. We evaluated two versions of the problem: one in which the drone is allowed to wait at customers to save battery and another in which this is prohibited. Both are hard to solve even when considering small sized instances. The version in which waiting is allowed is only slightly easier to be solved than the other one: larger sets of feasible solutions must be treated by the algorithm, but feasible solutions are easier to find and possibly with a smaller solution value. Among the several features considered, the drone endurance is the one that has the stronger influence on the convergence of the algorithm, a smaller endurance allows the algorithms to have better performances, while reducing the number of feasible sorties. To conclude, our ideas could be easily applied to similar problems: this could suggest a direction of future research considering analogous problems, with different constraints. Another possible direction of research could be the development of metaheuristic algorithms for solving those problems.

References

1. 7-Eleven, Flirtey make first FAA-approved drone delivery to home. (2016). https://www.gpsworld.com/7-eleven-flirtey-make-first-faa-approved-drone-delivery-to-home/. Accessed 22 Mar 2019
2. Aerial Id Card renewal: UAE to use drones for government services (2014). https://www.reuters.com/article/us-emirates-drones/aerial-id-card-renewal-uae-to-use-drones-for-government-services-idUSBREA19O6E20140210. Accessed 22 Mar 2019
3. Alibaba’s drones deliver packages to islands. http://www.chinadaily.com.cn/business/2017-11/07/content_34230012.htm. Accessed 22 Mar 2019
4. Amazon Prime Air. https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011. Accessed 22 Mar 2019
5. Ambulance drone. https://www.tudelft.nl/en/ide/research/research-labs/applied-labs/ambulance-drone/. Accessed 22 Mar 2019
6. The defibrillator drone that can beat ambulance times. https://www.bbc.com/news/av/technology-40360164/the-defibrillator-drone-that-can-beat-ambulance-times. Accessed 22 Mar 2019
7. Dpdgroup drone delivers parcels using regular commercial line (2016). https://www.suasnews.com/2016/12/dpdgroup-drone-delivers-parcels-using-regular-commercial-line/. Accessed 22 Mar 2019
8. Drone delivery: DHL, ‘parcelcopter’ flies to German isle (2014). https://www.reuters.com/article/us-deutsche-post-drones/drone-delivery-dhl-parcelcopter-flies-to-german-isle-idUSKCN0HJ1ED20140924. Accessed 22 Mar 2019
9. How UPS and Zipline are using drones to save lives in remote Rwandan clinics. https://www.freenterprise.com/how-ups-and-zipline-are-using-drones-to-save-lives/. Accessed 22 Mar 2019
10. JD.com to build 150 drone launch facilities in China by 2020 (2017). https://aircargoworld.com/allposts/jd-com-to-build-150-drone-launch-facilities-in-china-by-2020-video/. Accessed 22 Mar 2019
11. JD.com drone delivery program takes flight in rural China (2016). http://corporate.jd.com/whatIsNewDetail?contentCode=6lhXLeeSAFLjLLlyuZatDA. Accessed 22 Mar 2019
12. JD’s first large-scale UAV debuts in Shaanxi (2018). http://www.chinadaily.com.cn/a/201811/20/WS5bf3af9ca310efff303289e31.html. Accessed 22 Mar 2019
13. Parcelcopter: HL’s drone (2018). https://discover.dhl.com/business/business-ethics/parcelcopter-drone-technology. Accessed 22 Mar 2019
14. UPS drivers may tag team deliveries with drones. https://money.cnn.com/2017/02/21/technology/ups-drone-delivery/index.html. Accessed 22 Mar 2019
15. UPS tests residential delivery via drone. https://www.youtube.com/watch?v=xx9_6OyjJrQ. Accessed 22 Mar 2019
16. Vans & drones in Zurich (2017). https://www.mercedes-benz.com/en/mercedes-benz/vehicles/transporter/vans-drones-in-zurich/. Accessed 22 Mar 2019
17. Wing - transforming the way goods are transported. https://x.company/projects/wing/. Accessed 22 Mar 2019
18. Zookal will deliver textbooks using drones in Australia next year. https://edition.cnn.com/2013/10/18/tech/innovation/zookal-will-deliver-textbooks-using-drones/index.html. Accessed 22 Mar 2019
19. Agatz, N., Bouman, P., Schmidt, M.: Optimization approaches for the traveling salesman problem with drone. Transp. Sci. 52(4), 965–981 (2018)
20. Boone, N., Sathyan, A., Cohen, K.: Enhanced approaches to solving the multiple traveling salesman problem. In: Proceedings of the AIAA Infotech @ Aerospace Conference, p. 0889 (2015). https://doi.org/10.2514/6.2015-0889
21. Bortoff, S.A.: Path planning for uav’s. In: Proceedings of the 2000, American Control Conference, IEEE, vol. 1, no. 6, pp. 364–368 (2000)
22. Bouman, P., Agatz, N., Schmidt, M.: Dynamic programming approaches for the traveling salesman problem with drone. Networks 72(4), 528–542 (2018)
23. Boysen, N., Briskorn, D., Fedtke, S., Schwerdfeger, S.: Drone delivery fru trucks: drone scheduling for given truck routes. Networks 72(4), 506–527 (2018)
24. Campbell, J.F., Corberán, Á., Plana, I., Sanchis, J.M.: Drone arc routing problems. Networks 72(4), 543–559 (2018)
25. Chao, I.M.: A tabu search method for the truck and trailer routing problem. Comput. Oper. Res. 29(1), 33–51 (2002)
26. Current, J.R., Schilling, D.A.: The covering salesman problem. Transp. Sci. 23(3), 208–213 (1989)
27. Daknama, R., Kraus, E.: Vehicle routing with drones. arXiv preprint arXiv:1705.00431 (2017)
28. Dorling, K., Heinrichs, J., Messier, G.G., Magierowski, S.: Vehicle routing problems for drone delivery. IEEE Trans. Syst. Man Cybern. Syst. 47(1), 70–85 (2017)
29. DrexI, M.: Synchronization in vehicle routing—a survey of vrps with multiple synchronization constraints. Transp. Sci. 46(3), 297–316 (2012)
30. Ferrandez, S.M., Harbison, T., Weber, T., Sturges, R., Rich, R.: Optimization of a truck-drone in tandem delivery network using k-means and genetic algorithm. J. Ind. Eng. Manag. 9(2), 374 (2016)
31. Ha, Q.M., Deville, Y., Pham, Q.D., Hà, M.H.: Heuristic methods for the traveling salesman problem with drone. Comput. Sci. arXiv:1509.08764v1 (2015)
32. Kashuba, S., Novikov, V., Lysenko, O., Alekseeva, I.: Optimization of uav path for wireless sensor network data gathering. In: 2015 IEEE International Conference on Actual Problems of Unmanned Aerial Vehicles Developments (APUAVD), IEEE, pp. 280–283 (2015). https://doi.org/10.1109/APUAVD.2015.7346621
33. Lin, C.: A vehicle routing problem with pickup and delivery time windows, and coordination of transportable resources. Comput. Oper. Res. 38(11), 1596–1609 (2011)
34. Mbiadou Saleu, R.G., Deroussi, L., Feillet, D., Grangeon, N., Quilliot, A.: An iterative two-step heuristic for the parallel drone scheduling traveling salesman problem. Networks 72(4), 459–474 (2018)
35. Miller, C.E., Tucker, A.W., Zemlin, R.A.: Integer programming formulation of traveling salesman problems. J. ACM (JACM) 7(4), 326–329 (1960)
36. Murray, C.C., Chu, A.G.: The flying sidekick traveling salesman problem: optimization of drone-assisted parcel delivery. Transp. Res. Part C Emerg. Technol. 54, 86–109 (2015)
37. O’Rourke, K.P., Bailey, T.G., Hill, R., Carlton, W.B.: Dynamic routing of unmanned aerial vehicles using reactive tabu search. Tech. rep. Air Force Inst. of Tech. Wright-Patterson AFB Oh (1999)
38. Otto, A., Agatz, N., Campbell, J., Golden, B., Pesch, E.: Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: a survey. Networks 72(4), 411–458 (2018)
39. Poikonen, S., Wang, X., Golden, B.: The vehicle routing problem with drones: extended models and connections. Networks 70(1), 34–43 (2017)
40. Ponza, A.: Optimization of drone-assisted parcel delivery. Master’s Thesis (2016)
41. Pugliese, L.D.P., Guerriero, F.: Last-mile deliveries by using drones and classical vehicles. In: International Conference on Optimization and Decision Science, pp. 557–565. Springer, Berlin (2017). https://doi.org/10.1007/978-3-319-67308-0_56
42. Pugliese, L.D.P., Guerriero, F., Zorbas, D., Razafindralambo, T.: Modelling the mobile target covering problem using flying drones. Optim. Lett. 10(5), 1021–1052 (2016)
43. Richards, A., Bellingham, J., Tillerson, M., How, J.: Coordination and control of multiple uavs. In: AIAA Guidance, Navigation, and Control Conference, Monterey, CA (2002). https://doi.org/10.2514/6.2002-4588
44. Roberge, V., Tarbouchi, M., Labonté, G.: Comparison of parallel genetic algorithm and particle swarm optimization for real-time uav path planning. IEEE Trans. Ind. Inf. 9(1), 132–141 (2013). https://doi.org/10.1109/TII.2012.2198665
45. Ryan, J.L.: Embedding a reactive tabu search heuristic in unmanned aerial vehicle simulations. Tech. rep, Air Force Inst. of Tech. Wright-Patterson AFB Oh School of Engineering (1998)
46. Ryan, J.L., Bailey, T.G., Moore, J.T., Carlton, W.B.: Reactive tabu search in unmanned aerial reconnaissance simulations. In: Proceedings of the 30th Conference on Winter Simulation, IEEE Computer Society Press, pp. 873–880 (1998). https://doi.org/10.1109/WSC.1998.745084
47. Savuran, H., Karakaya, M.: Route optimization method for unmanned air vehicle launched from a carrier. Lect. Notes Softw. Eng. 3(4), 279 (2015)
48. Scott, J., Scott, C.: Drone delivery models for healthcare. In: Proceedings of the 50th Hawaii International Conference on System Sciences (2017). https://doi.org/10.24251/HICSS.2017.399
49. Shang, B., Wu, C., Hu, Y., Yang, J.: An algorithm of visual reconnaissance path planning for uavs in complex spaces. J. Comput. Inf. Syst. 8(1), 1–8 (2014)
50. Shavarani, S.M., Nejad, M.G., Rismanchian, F., Izbirak, G.: Application of hierarchical facility location problem for optimization of a drone delivery system: a case study of amazon prime air in the city of San Francisco. Int. J. Adv. Manuf. Technol. 95(9–12), 3141–3153 (2018)
51. Shuttleworth, R., Golden, B.L., Smith, S., Wasil, E.: Advances in meter reading: Heuristic solution of the close enough traveling salesmen problem over a street network. In: The Vehicle Routing Problem: Latest Advances and New Challenges, pp. 487–501. Golden B., Raghavan S., Wasil E. (eds.) vol. 43. Springer, Berlin (2008)
52. Sisson, M.R.: Applying tabu heuristic to wind influenced, minimum risk and maximum expected coverage routes. Tech. rep, Air Force Inst. of Tech. Wright-Patterson AFB Oh School of Engineering (1997)
53. Tseng, C.M., Chau, C.K., Elbassioni, K., Khonji, M.: Flight tour planning with recharging optimization for battery-operated autonomous drones. arXiv preprint arXiv:1703.10049 (2017)
54. Ulmer, M.W., Thomas, B.W.: Same-day delivery with heterogeneous fleets of drones and vehicles. Networks 72(4), 475–505 (2018)
55. Wang, X., Poikonen, S., Golden, B.: The vehicle routing problem with drones: several worst-case results. Optim. Lett. 11(4), 679–697 (2017)
56. Yurek, E.E., Ozmutlu, H.C.: A decomposition-based iterative optimization algorithm for traveling salesman problem with drone. Transp. Res. Part C Emerg. Technol. 91, 249–262 (2018)
57. Zheng, C., Li, L., Xu, F., Sun, F., Ding, M.: Evolutionary route planner for unmanned aerial vehicles. IEEE Trans. Robot. 21(4), 609–620 (2005)
58. Zorbas, D., Razafindralambo, T., Guerriero, F., et al.: Energy efficient mobile target tracking using flying drones. Proc. Comput. Sci. 19, 80–87 (2013)

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.