Computers & Operations Research 174 (2025) 106894

Contents lists available at ScienceDirect

Computers and Operations Research

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

The min max multi-trip drone location arc routing problem

Teresa Corberán [a], Isaac Plana [b], José María Sanchis [c,*]

[a] Departament d'Estadística i Investigació Operativa, Universitat de València, Spain
[b] Departament de Matemàtiques per a l'Economia i l'Empresa, Universitat de València, Spain
[c] Institut Universitari de Matemàtica Pura i Aplicada, Universitat Politècnica de València, Spain

ARTICLE INFO

Keywords:
Drones
Location arc routing problem
Multi-trip
Length constraints
Matheuristic
Branch-and-cut
Polyhedral study

ABSTRACT

This paper studies the Min Max Multi-Trip drone Location Arc Routing Problem (MM-MT-dLARP), an arc routing problem that combines trucks and drones. We have a set of lines (usually curved) that have to be flown over by drones to perform a service (inspection, for example). There is a depot from which the trucks leave, each one carrying a drone, and a set of potential launching points where the truck can launch and pick up the drone. Drones have limited autonomy, but they can make several flights. We consider a min–max objective, in which the makespan, or time necessary to complete the service, must be minimized. Using aerial drones instead of ground vehicles allows to travel off the network: drones can enter a line through any of its points, service only a portion of that line and then exit through another of its points, without following the lines of the network. This allows for finding better solutions but also increases the difficulty of the problem. This issue can be addressed by digitizing the MM-MT-dLARP instances, approximating each line by a polygonal chain with a finite number of intermediate points, and requiring that drones can only enter and exit a line through those intermediate points. Thus, an instance of a discrete Min Max Multi-Trip Location Arc Routing Problem (MM-MT-LARP) is obtained. Here, an integer formulation for the MM-MT-LARP is proposed, some families of valid inequalities are proved to be facet-inducing of a relaxed polyhedron, and a branch-and-cut algorithm based on the strengthened formulation is developed. This algorithm has only been applied to small instances without intermediate points on the lines. In addition, we have developed a matheuristic algorithm for the MM-MT-dLARP that combines a construction phase, four local search procedures integrated into a Variable Neighborhood Descent (VND) algorithm, and a set of rules for selecting intermediate points to improve the solutions. We present the results obtained on a set of randomly generated instances involving up to 6 launching points and 88 original lines.

1. Introduction

The use of drones (unmanned aerial vehicles) for delivery or inspection tasks is becoming more common, since they offer several advantages compared to traditional means of transportation, being able to reach places that are difficult to access or dangerous for people or conventional vehicles. Research in the design of routes for drones has so far focused on two types of problems: those related to the delivery of goods, usually modeled as node routing problems (see, for example, the surveys by Macrina et al. (2020) and Rojas Viloria et al. (2021)) and those dedicated to the inspection or mapping of surfaces or networks, modeled as arc routing problems (see, for example,Campbell et al. (2018)). Lately, some papers have appeared that perform both types of tasks, surveillance of lines and delivery of goods, simultaneously (Campbell et al., 2023; Corberán et al., 2024).

The specific characteristics of drones make arc routing problems with drones quite different from arc routing problems with traditional terrestrial vehicles. While terrestrial vehicles are restricted to traveling through the links of the network (roads, streets,...), drones can travel off the network directly between any two points. Moreover, drones can enter and exit a link at any point, allowing for shorter routes than with traditional vehicles. However, this implies that the problem is a continuous optimization one and, therefore, quite more difficult to solve. Arc Routing Problems with drones were studied for the first time by Campbell et al. (2018), where each continuous line is given by a polygonal chain, with a large (but finite) number of points that can be used by drones to enter and leave the line. A heuristic for the same problem is proposed in Xie et al. (2024) and tested on the same instances. The same discretization of the lines is considered in Campbell et al. (2021, 2022) to solve a multi-drone version of the Rural Postman Problem, in which the service of the lines is performed by a fleet of drones with limited autonomy. A more advanced type

* Corresponding author.
E-mail addresses: tecorfa@alumni.uv.es (T. Corberán), isaac.plana@uv.es (I. Plana), jmsanchis@mat.upv.es (J.M. Sanchis).

https://doi.org/10.1016/j.cor.2024.106894
Received 18 July 2024; Received in revised form 30 October 2024; Accepted 31 October 2024
Available online 9 November 2024

of drone that can make deliveries and map surfaces in the same flight, called a multi-purpose drone, is considered in Campbell et al. (2023), where the authors present a formulation for the problem and propose a branch-and-cut procedure and a matheuristic algorithm. A stochastic arc-inventory routing problem applied to traffic monitoring with drones is studied by Chow (2016), who proposes an approximate dynamic programming algorithm. The capacitated and inventory arc routing problems are combined by Li et al. (2018) in a mixed integer programming formulation for solving a drone scheduling problem in traffic monitoring.

Given that drones have limited flight autonomy, for some applications it is necessary to use other types of support ground vehicles that can serve as a point for launching and retrieving the drones, as well as recharging or exchanging batteries. The coordination of ground vehicles with drones has been widely discussed in the literature for node routing problems (see, for example, Chung et al. (2020)). However, few works have addressed this problem from the point of view of arc routing problems. Amorosi et al. consider a problem in which there is a mothership that can move freely in a two-dimensional space. In these works, a single mothership and one (Amorosi et al., 2021, 2024) or several drones (Amorosi et al., 2023), are considered, which must travel a certain percentage of the length of a set of lines. Xu et al. (2023) study a problem in which a ground vehicle, which moves on a directed network, carries several drones that can be launched to perform patrolling tasks on some arcs. They propose a formulation and a metaheuristic algorithm for its solution. Luo et al. (2019) studied a patrolling problem with one ground vehicle and one drone, in which both vertices and arcs of an urban road network had to be visited. They propose a heuristic algorithm with a neighborhood search strategy.

Arc routing problems with several depots have not been given much attention in the literature. Fernández and Rodríguez-Pereira (2017) study the Multi-Depot Rural Postman Problem (MD-RPP), in which there is a fleet of vehicles, each one based on a different depot, that has to collectively traverse a subset of required edges. They consider two different objective functions, one minimizing the total cost and another one the length of the longest route. The same problem is addressed by Fernández et al. (2018), who perform a worst-case comparison between the optimal values of the MD-RPP and the RPP, propose an alternative formulation of the problem and implement a branch-and-cut algorithm. Another multi-depot arc routing problem that has been studied is the Multi-Depot Capacitated Arc Routing Problem (MD-CARP). In this problem, there is a demand associated with each required edge and a maximum capacity for each vehicle. For this problem, Amberg et al. (2000) propose a tabu search algorithm that transforms the problem into a multiple center capacitated minimum spanning tree problem. Muyldermans et al. (2003) present heuristic algorithms based on clustering the required arcs into districts. Wøhlk (2004) develops a mathematical model for the MD-CARP that also takes into account the fixed costs of the vehicles and designs a heuristic algorithm for its solution. Xing et al. (2010) and Hu et al. (2013) propose metaheuristic algorithms for the problem. Krushinsky and Van Woensel (2015) study an asymmetric version of the MD-CARP and provide a formulation and an exact algorithm. As far as we know, the only work dealing with a multi-depot arc routing considering the use of drones is the one by Corberán et al. (2024), who study the Multi-Depot Drone General Routing Problem with duration and capacity constraints. In that problem, a fleet of drones, each one based on a different depot, have to jointly traverse a set of required lines and deliver some commodities in a set of required vertices, without exceeding the capacity or the maximum flight time of the drones. A branch-and-cut algorithm and two matheuristics are proposed.

Problems that combine the design of routes servicing some arcs and the decision of where to locate the depots from which these routes start are called Location Arc Routing Problems (LARP). These problems were first described in Ghiani (1998), where location and allocation arc routing problems are introduced. Doulabi and Seifi (2013) consider a LARP with vehicle capacity constraints, present two mixed integer formulations to obtain lower bounds, and propose a simulated annealing metaheuristic. Lopes et al. (2014) propose some heuristic algorithms for solving the LARP with capacity constraints. More recently, Fernández et al. (2019) studied several families of LARPs and proposed exact solution procedures for them. For more information on LARPs, we refer the reader to the recent survey by Albareda-Sambola and Rodríguez-Pereira (2019).

There are real applications in which a set of lines that form a network (power lines, pipelines, roads, etc.) must be monitored or inspected. As we have mentioned before, drones equipped with video cameras are increasingly being used for these jobs, which only have to fly over the lines and retransmit or record the captured images. If the network to be inspected is very large, such as electrical, pipe, or road networks, this work must be carried out by several drones, since a single drone does not have a sufficiently large range. In addition, drones must be supported by ground vehicles that bring them closer to the network. These ground vehicles need an operational point close to the infrastructure for parking and performing the necessary tasks for launching, handling and retrieving the drone. In addition, the vehicle can carry a set of batteries so that, if the drone does not have sufficient autonomy to perform the entire task, it can return to the vehicle, have its battery replaced by a new one, and then fly again to continue inspecting the lines. Therefore we need to select an operational point for each vehicle and design the flights that the drone will perform from this point so that every line of the network is supervised by one drone. The goal is to complete the work as soon as possible, so we need to take into account the time used by each vehicle to get to the assigned point and back plus the flight time of its drone, and then minimize the longest time among all the vehicles.

Consider a set of (usually curved) lines that must be serviced while traversed by drones, each with an associated service cost, a point called the depot where there are P trucks, each one carrying a drone, and a set D of $D \geq P$ points called the launching (and landing) points. Each truck has to travel to one of the launching points to launch and then pick up the drone from there, which can perform several flights. Two different trucks cannot go to the same launching point. We assume that we know the cost (in distance or time) of a truck traveling from the depot to each one of the launching points, we know the cost (in distance or time) of traveling while servicing with a drone each line, and that the cost (in distance or time) for a drone to deadhead between any two points is proportional to the Euclidean distance. The cost of each drone flight cannot be greater than a certain constant L (related to the drone's flight range). The goal is to determine to which launching point each truck has to go, and find, for each truck, a set of some drone routes starting and ending at this launching point and with costs no greater than L in such a way that they jointly traverse all the given lines completely and that the maximum cost of a truck plus all the corresponding flights of its drone is minimum. This problem will be called *Min Max Multi-trip drone Location Arc Routing Problem, MM-MT-dLARP*.

Given that drones can travel off the network and go directly between any two points, not necessarily the endpoints of the required lines, this is a continuous optimization problem in which the shape of the lines to service must be taken into account. To deal with this problem, the instances are digitized, that is, each line to be serviced is described as a polygonal chain, with a sequence of points given by their coordinates and assuming that drones can only enter and exit a line through these points. Each segment in the polygonal chain has a service cost that is proportional to the service cost of the line so that the sum of the costs of the segments that approximate a line is equal to the service cost of the line. The depot and the D launching points are also given by their coordinates. We assume that drones can fly directly between any two points, with a deadheading cost proportional to the Euclidean distance computed from their coordinates. Hence, we have an instance of an ARP (studied in Section 2).

The main contributions of this paper are the following. We introduce a new problem, the MM-MT-dLARP, that considers drones and terrestrial support vehicles and combines multi-depot, location and arc routing problems. We propose a matheuristic algorithm for its solution. Furthermore, we present a formulation for its discrete version, the *Min Max Multi-Trip Location Arc Routing Problem, MM-MT-LARP*, and introduce several families of inequalities that strengthen the formulation. We use this polyhedral study to develop a branch-and-cut algorithm. A new set of benchmark instances for the MM-MT-LARP has been generated and extensive computational experiments have been conducted to assess the performance of the proposed methods.

The remainder of this paper is structured as follows. In Section 2, we describe the MM-MT-LARP, propose a formulation and conduct the polyhedral study. The branch-and-cut algorithm for its exact solution is detailed in Section 3 while Section 4 is devoted to the description of the matheuristic algorithm for the MM-MT-dLARP. The computational experiments are reported in Section 5 and some conclusions and future research lines are discussed in Section 6.

2. The Min Max Multi-Trip Location Arc Routing problem

The MM-MT-LARP is defined on an undirected graph $G = (V, E)$. The set of edges E contains a subset E_R of required edges (corresponding to the segments of the polygonal chains). The set of vertices V contains the subset V_R with the vertices incident with edges in E_R, a subset D with the vertices corresponding to the launching points, and a vertex 0 that represents the depot (and can also be a launching point). The set of non-required edges $E_{NR} \subset E$ includes an edge between each pair of vertices in V. Note that each $e \in E_R$ has a parallel non-required edge, which we denote as e'. E'_{NR} represents the set of non-required edges parallel to a required one, while $E''_{NR} = E_{NR} \setminus E'_{NR}$.

Each $e \in E_R$ has a service cost $c^s_e > 0$ and each non-required edge $e \in E_{NR}$ has a deadheading cost $c_e > 0$. We assume that $c^s_e \geq c_{e'}$ for each required edge e and its corresponding parallel non-required edge e', and that the deadheading costs satisfy the triangle inequality. For each launching point $d \in D$, $c_{0d} \geq 0$ is the cost for a truck to travel from depot 0 to point d. There is a fixed number P of trucks, and the goal is to select P launching points, one for each truck, and find, for each one of these launching points, a set of drone flights (or drone tours, i.e., closed walks starting and ending at the launching point) with cost no greater than a given constant L in such a way that all the tours jointly traverse all the required edges, and that the maximum of the costs calculated as the sum of the cost of the travel truck plus the costs of all its corresponding tours, is minimum. The launching points could be considered as some facilities to be opened if a truck goes to this point to launch its drone. The cost $2c_{0d}$ can be understood as a fixed cost of opening the point d considered as a facility. In this way, we do not need the depot to be a given vertex in the graph and we assume $V = V_R \cup D$. Therefore, our problem can be considered as a *Location Arc Routing Problem*. Hence, we will call the problem the *Min Max Multi-trip Location Arc Routing Problem, MM-MT-LARP*.

Although in the definition of the problem the number of flights that each drone performs is not limited, to formulate the problem we need to define this number. We will call K the maximum number of flights that a drone can perform from its launching point. The value of K for a given instance is obtained from the solutions reached with the heuristic algorithm presented in Section 4. We want to note that the instance may have a better solution with more than K flights. Since there are $D = |D|$ launching points and each single drone can perform up to K flights, a feasible solution of the MM-MT-LARP on G can have up to DK drone tours, although only PK drone tours can be non-empty in a feasible solution (up to K flights for each of the P trucks). Let a *MM-MT-LARP solution* denote any set of DK tours on graph G (some maybe empty), satisfying that

- there are up to K tours starting and ending at each launching point $d \in D$,
- there are at most P launching points that have some of their flights not empty, and therefore, there are at least $D–P$ launching points that have all their flights empty,
- they jointly service (traverse) all the required edges, and
- the length of each one of these DK tours is less than or equal to L.

These MM-MT-LARP solutions can be expressed with the following binary variables (in the absence of ambiguity, we will use the notation $dk \in D \times K$ instead of $(d, k) \in D \times K$ for simplicity):

- For each edge $e \in E$ and for each tour $dk \in D \times K$, variable x^{dk}_e takes the value 1 if e is traversed by tour k starting at the launching point d, and 0 otherwise.
- For each edge $e \in E_{NR}$ and for each tour $dk \in D \times K$, variable y^{dk}_e takes the value 1 if e is traversed twice by tour k starting at the launching point d, and 0 otherwise.

Variable x^{dk}_e represents the first traversal of edge e and y^{dk}_e the second one. Note that the required edges do not need variable y^{dk}_e since a second traversal, if needed, will always be done through the parallel non-required edges, which is cheaper. Hence, associated with each MM-MT-LARP solution we can consider:

- DK incidence vectors $(x^{dk}, y^{dk}) \in \mathbb{Z}^{2|E_{NR}|+|E_R|}$, one for each drone tour dk, where variables x^{dk}_e take the value 1 if edge e is traversed once, variables y^{dk}_e take the value 1 if edge e is traversed twice, and
- DK support graphs $(V, E^{(x^{dk}, y^{dk})})$, one for each drone tour dk, where $E^{(x^{dk}, y^{dk})}$ contains one copy of edge $e \in E$ for each variable $x^{dk}_e = 1$ or $y^{dk}_e = 1$.

Note that the support graphs are even and connected. Conversely, any even and connected subgraph of G corresponds to a tour on G. In fact, an incidence vector or a subgraph may correspond to several different closed walks, but all of them have the same cost and they can be easily computed (with the Hierholzer algorithm (1873) (Hierholzer, 1873), for example). Hence, and for the sake of simplicity, we will let "tour on G" denote the closed walk, its incidence vector, and its corresponding support graph.

2.1. A formulation

The variables x^{dk}, y^{dk} defined above are enough to represent an MM-MT-LARP solution: We can check, for each dk, if (x^{dk}, y^{dk}) forms a tour starting and ending at d, we can see if there are exactly P launching points with, at most, K non-empty tours starting at it, we can check if the length of each tour is less than or equal to L, and we can calculate the total cost of the route (truck plus drone flights) associated with each launching point. However, to formulate the MM-MT-LARP problem we need, or at least are convenient, some additional z^d variables to indicate which launching point is open to receive a truck:

- For each $d \in D$, variable z^d takes the value 1 if the launching point d is used by a truck, and 0 otherwise.

These z variables are "superfluous" in the sense that if we know the x, y variables associated with an MM-MT-LARP solution, we know the corresponding variables z. But they are useful to formulate the problem, as we will see next.

We use the following notation. Given two subsets of vertices $S, S' \subseteq V$, $(S : S')$ denotes the edge set with one endpoint in S and the other one in S'. Given a subset $S \subseteq V$, let us denote $\delta(S) = (S : V \setminus S)$ and $E(S) = (S : S)$. For simplicity, when $S = \{i\}$, $i \in V$, we write $\delta(i)$ instead of $\delta(\{i\})$. For any subset $F \subseteq E = E_R \cup E_{NR}$, we denote $F_R = F \cap E_R$ and $F_{NR} = F \cap E_{NR}$, and, for simplicity, we write $\delta_R(S)$ instead of $\delta(S)_R$. Moreover, given a vector x indexed on the edges set E and a subset of edges $F \subset E$, $x(F)$ denotes $\sum_{e \in F} x_e$. Given a vector y indexed on the set E_{NR} and a subset of edges $F \subset E$, $y(F)$ denotes $\sum_{e \in F_{NR}} y_e$. Finally, $(x + y)(F)$ and $(x - y)(F)$ represent $x(F) + y(F)$ and

$x(F) - y(F)$, respectively. Note that $(x+y)(F)$ includes variables x_e for $e \in F$, and variables y_e for $e \in F_{NR}$.

The MM-MT-LARP can be formulated as follows:

$$\text{Minimize} \quad t \tag{1}$$

s.t.

$$\sum_{k\in\mathcal{K}}\sum_{e\in E_R} c_e^s x_e^{dk} + \sum_{k\in\mathcal{K}}\sum_{e\in E_{NR}} c_e\left(x_e^{dk}+y_e^{dk}\right) + 2c_{0d}z^d \le t, \quad \forall d \in \mathcal{D} \tag{2}$$

$$\left(x^{dk}+y^{dk}\right)\left(\delta(v)\right) \equiv 0 \pmod 2, \quad \forall dk \in \mathcal{D}\times\mathcal{K},\ \forall v \in V \tag{3}$$

$$\left(x^{dk}+y^{dk}\right)\left(\delta(S)\right) \ge 2x_f^{dk}, \quad \forall dk \in \mathcal{D}\times\mathcal{K}, \ \forall S \subseteq V\backslash\{d\},\ \forall f \in E(S) \tag{4}$$

$$\sum_{k\in\mathcal{K}}\sum_{d\in\mathcal{D}} x_e^{dk} \ge 1, \quad \forall e \in E_R \tag{5}$$

$$x_e^{dk} \ge y_e^{dk}, \quad \forall e \in E_{NR},\ \forall dk \in \mathcal{D}\times\mathcal{K} \tag{6}$$

$$\sum_{e\in E_R} c_e^s x_e^{dk} + \sum_{e\in E_{NR}} c_e(x_e^{dk}+y_e^{dk}) \le Lz^d, \quad \forall dk \in \mathcal{D}\times\mathcal{K} \tag{7}$$

$$\sum_{d\in\mathcal{D}} z^d \le P \tag{8}$$

$$x_e^{dk} \in \{0,1\}, \quad \forall e \in E,\ \forall dk \in \mathcal{D}\times\mathcal{K} \tag{9}$$

$$y_e^{dk} \in \{0,1\}, \quad \forall e \in E_{NR},\ \forall dk \in \mathcal{D}\times\mathcal{K} \tag{10}$$

$$z^d \in \{0,1\}, \quad \forall d \in \mathcal{D}. \tag{11}$$

The objective function (1) minimizes t, which is the maximum of the costs calculated as the sum of the cost of the truck travel plus the costs of all its corresponding drone flights, defined by constraints (2). Constraints (3) force that the number of times a drone flight visits a vertex is even, possibly zero. Constraints (4) avoid subtours and ensure that each tour is connected and connected to its corresponding launching point and are called *connectivity inequalities*. Constraints (5) force each required edge to be serviced at least once. Note that these inequalities could be replaced by equalities, $\sum_{k\in\mathcal{K}}\sum_{d\in D} x_e^{dk} = 1$, but we need the inequalities to do the polyhedral study. Nevertheless, in the branch-and-cut algorithm, we use the equalities to strengthen the formulation. Constraints (6) guarantee that a second traversal of an edge by a drone can only occur when it has been traversed previously by this drone. Constraints (7) guarantee that the cost of each tour corresponding to a given launching point does not exceed L if this point is used by a truck and that the tour is empty if that launching point is not used by a truck. Constraint (8) ensures that at most P locations are used (one for each truck). Constraints (9), (10), and (11) are the binary conditions for the variables. Note that the z variables are used to compute the total cost associated with each truck (2), to ensure that at most P launching points are opened (8) and that if a launching point is not open then all the tours that start at it are zero while, if it is open, then the length of the tours that start at it are of length less than or equal to L (7).

For each drone, there are three variables to represent the traversals between the two endpoints i and j of a required edge e. The reason is that we need to distinguish between traversing e while serving it (with a cost c_e^s) or deadheading e' once or twice (with a cost $c_{e'} \le c_e^s$). Although in all the optimal solutions the three variables will never be non-zero simultaneously (see Proposition 1), we need the three variables to state the length of a route in the problem formulation (constraints (2) and (7)).

Proposition 1. *For each $e \in E_R$ and its parallel edge $e' \in E'_{NR}$, for each tour $dk \in \mathcal{D}\times\mathcal{K}$, the inequality $x_e^{dk} + y_{e'}^{dk} \le 1$ is satisfied by any optimal solution of the MM-MT-LARP on G (if $c_{e'} > 0$).*

Proof. If a solution satisfies $x_e^{dk} + y_{e'}^{dk} = 2$, necessarily satisfies $x_{e'}^{dk} = 1$ (because $x_{e'}^{dk} \ge y_{e'}^{dk}$ holds) and the drone travels three times between i and j, so we can remove two copies and get a better feasible solution. ♦

2.2. Polyhedral study

To study the polyhedron associated with the solutions of the MM-MT-LARP, we have to ignore the restriction that limits the length of the drone tours to L. Let *MT-LARP(G)* denote the polytope defined as the convex hull of all the incidence vectors of MT-LARP solutions on G, i.e., MM-MT-LARP solutions in G without considering the mentioned restriction on the length of the drone tours, or any set of DK tours on graph G (closed walks, maybe empty),

$$\left(x^{1,1}, y^{1,1}, x^{1,2}, y^{1,2}, \ldots, x^{1,K}, y^{1,K}, x^{2,1}, y^{2,1} \ldots, x^{D,K}, y^{D,K}, z\right) \in \mathbb{Z}^{DK(2|E_{NR}|+|E_R|)+D},$$

satisfying that

- each tour $\left(x^{d,k}, y^{d,k}\right)$ starts and ends at launching point $d \in D$,
- there are **at most** P launching points that have some of their flights not empty, and **at most** P z^d variables taking the value 1,
- there are **at least** $D-P$ launching points have all their flights empty, and **at least** $D-P$ z^d variables taking the value 0, and
- they jointly service (traverse) all the required edges.

In this section we are going to use some results about the K-vehicles Rural Postman Problem (K-RPP) polyhedron studied in Campbell et al. (2022). In that paper, all vertices of the graph G are assumed to be incident with required edges. However, it can be seen that the results on the polyhedron are also valid for graphs in which there are "isolated" vertices, that is, vertices that are not incident with any required edge. This is necessary in this section since if in the graph G we declare a launching point as the depot to define a K-RPP instance, the other launching points may be "isolated" vertices.

2.2.1. The length-constrained K-Drones RPP and K-RPP

The Length-Constrained K-Drones Rural Postman Problem (LC K-DRPP), presented in Campbell et al. (2018) and studied in Campbell et al. (2022), is defined as follows: "Given a set of lines, each one with an associated service cost, and a point called the depot (vertex 1, for example), assuming that the cost of deadheading between any two points is the Euclidean distance, and given a constant L, find a set of drone routes starting and ending at the depot and with lengths no greater than L such that they jointly traverse all the given lines completely with minimum total cost". With the previous notation, except that now variables x_e^k, y_e^k have a superindex k referring the drone, $k \in \{1,2,\ldots,K\}$, the discrete version of the LC K-DRPP, called LC K-RPP, is formulated in Campbell et al. (2022) as follows:

$$\text{Minimize} \quad \sum_{k=1}^{K}\left(\sum_{e\in E_R} c_e^s x_e^k + \sum_{e\in E_{NR}} c_e\left(x_e^k + y_e^k\right)\right)$$

s.t.

$$\left(x^k+y^k\right)\left(\delta(i)\right) \equiv 0 \pmod 2, \ \forall i \in V, \quad \forall k = 1,\ldots,K \tag{12}$$

$$\left(x^k+y^k\right)\left(\delta(S)\right) \ge 2x_f^k, \ \forall S \subseteq V\backslash\{1\}, \forall f \in E(S), \quad \forall k = 1,\ldots,K \tag{13}$$

$$\sum_{e\in E_R} c_e^s x_e^k + \sum_{e\in E_{NR}} c_e\left(x_e^k + y_e^k\right) \le L, \ \forall k = 1,\ldots,K \tag{14}$$

$$\sum_{k=1}^{K} x_e^k \ge 1, \ \forall e \in E_R \tag{15}$$

$$x_e^k \ge y_e^k, \ \forall e \in E_{NR}, \quad \forall k = 1,\ldots,K \tag{16}$$

$$x_e^k \in \{0,1\},\ \forall e \in E_R,\quad \forall k = 1,\dots,K \tag{17}$$

$$x_e^k, y_e^k \in \{0,1\},\ \forall e \in E_{NR},\quad \forall k = 1,\dots,K. \tag{18}$$

If we remove constraints (14), we have the K-RPP. In what follows, we recall some results presented in Campbell et al. (2022) for the K-RPP with $K \geq 2$ on graph $G = (V,E) = (V_R, E_R \cup E'_{NR} \cup E''_{NR})$. Let K-RPP(G) denote the polytope defined as the convex hull of the vectors $(x^1, y^1, x^2, y^2, \dots, x^K, y^K) \in \mathbb{Z}^{K(2|E_{NR}|+|E_R|)}$ corresponding to K-RPP solutions. The proofs of the following two propositions can be found in Campbell et al. (2022):

Proposition 2. *If (V, E_{NR}) is a 3-edge connected graph, then K-RPP(G) is a full-dimensional polyhedron, i.e., dim(K-RPP(G))= $K(2|E_{NR}|+|E_R|)$.*

Proposition 3. *The following inequalities are facet-inducing of K-RPP(G):*

1. *Inequality $y_e^k \geq 0$, for each edge $e \in E_{NR}$, and for each drone $k \in \{1,2,\dots,K\}$.*
2. *Inequality $x_e^k \leq 1$, for each edge $e \in E_{NR}$, and for each drone $k \in \{1,2,\dots,K\}$.*
3. *Inequality (16) $x_e^k \geq y_e^k$, for each edge $e \in E_{NR}$, and for each drone $k \in \{1,2,\dots,K\}$, if graph $(V, E_{NR} \setminus \{e\})$ is 3-edge connected.*
4. *Inequality $x_e^k \leq 1$, for each edge $e \in E_R$, and for each drone $k \in \{1,2,\dots,K\}$.*
5. *Inequality $x_e^k \geq 0$, for each edge $e \in E_R$, and for each drone $k \in \{1,2,\dots,K\}$.*
6. *Inequalities (15), $\sum_{k=1}^K x_e^k \geq 1$, for each edge $e \in E_R$.*
7. *Connectivity inequalities (13), if subgraph $(S, E_{NR}(S))$ is 3-edge connected and either $|V \setminus S| = 1$ or subgraph $(V \setminus S, E_{NR}(V \setminus S))$ is 3-edge connected.*

Furthermore, other inequalities such as parity or p-connectivity inequalities are also studied in Campbell et al. (2022) and proved that, under mild conditions, they also are facet inducing of K-RPP(G). These inequalities are adapted to the MT-LARP in Section 2.3.

2.2.2. The polytope of solutions

Proposition 4. *If (V, E_{NR}) is a 3-edge connected graph, then MT-LARP(G) is a full-dimensional polyhedron, i.e., dim(MT-LARP(G))= $DK(2|E_{NR}|+|E_R|)+D$, for $D \geq P \geq 2$.*

Proof. Consider the K-RPP defined on graph G considering a given launching point d as the depot and looking for K tours that jointly traverses all the required edges $e \in E_R$. Since (V, E_{NR}) is 3-edge connected, from Proposition 2 we have that dim(K-RPP(G))= $K(2|E_{NR}|+|E_R|) = m$ (full-dimensional polyhedron). Therefore, there are $m+1$ affinely independent, and also m linearly independent K-RPP solutions formed with K tours that jointly traverse all the required edges.

For each launching point d, let $\bar{A}^d$ be the incidence matrix of the corresponding $m+1$ K-RPP affinely independent solutions as rows indexed by the variables x, y as columns. This matrix $\bar{A}^d$ has $m+1$ rows and we can assume, without loss of generality, that,

- after removing the first row we obtain a matrix, say A^d with rank m, and that
- after subtracting the first row of $\bar{A}^d$ from the rows of A^d we obtain a matrix, say M^d, also with rank m.

If we assign one of these K-RPP solutions to one launching point d, the vector zero to the remaining launching points $d' \neq d$, $z^d = 1$, and $z^{d'} = 0$, we obtain a feasible solution for the MT-LARP. For example, for $D = 3$ launching points and $P = 2$ trucks we can build in this way the three first block rows of the matrix in Fig. 1a, where each row corresponds to a feasible MT-LARP solution and the columns labeled (x^{d*}, y^{d*}) correspond to the flights departing from the launching point d. It can be seen that similar solutions could be built with all values for $D \geq P \geq 2$. Furthermore, if f_1 and m_2 are the first row of matrices $\bar{A}^1$ and A^2, respectively, the last three rows of the matrix in Fig. 1a, with a single drone but two launching points open, are also feasible solutions. Again, it can be seen that similar solutions could be built with all values for $D \geq P \geq 2$, where $P \geq 2$ is needed to open two launching points.

This matrix has $Dm+1+D$ rows representing MT-LARP solutions. If we subtract the first row from all the others, we subtract the first row of block two from the last row, and remove the first row, we obtain the matrix in Fig. 1b, which has its $Dm+D$ rows linearly independent. Hence, we have $Dm+1+D$ affinely independent MT-LARP solutions and the dimension of the polyhedron is $Dm+D = DK(2|E_{NR}|+|E_R|)+D$. ♦

In the following, we will assume that $D \geq P \geq 2$ and that (V, E_{NR}) is a 3-edge connected graph and thus MT-LARP(G) is a full-dimensional polytope. Therefore, every facet of the polyhedron is induced by a unique inequality (except scalar multiples).

Proposition 5. *Inequality $y_e^{dk} \geq 0$, for each edge $e \in E_{NR}$, and for each flight $dk \in D \times \mathcal{K}$, is facet-inducing of MT-LARP(G).*

Proof. Without loss of generality, let us suppose that $d = 1$ and $k = 1$. Consider the K-RPP defined on graph G considering the launching point $d = 1$ as the depot and looking for K tours that jointly traverses all the required edges $e \in E_R$. Since (V, E_{NR}) is 3-edge connected, inequality $y_e^1 \geq 0$ is facet inducing of K-RPP(G) (Proposition 3) and there are $m = K(2|E_{NR}|+|E_R|)$ affinely independent K-RPP solutions formed with K tours that jointly traverse all the required edges and satisfy $y_e^1 = 0$. Given that vector zero is in the affine hull of the vectors satisfying $y_e^1 = 0$, there are $m-1$ linearly independent K-RPP solutions formed with K tours that jointly traverse all the required edges and satisfy $y_e^1 = 0$.

Let B^1 be the incidence matrix of these $m-1$ K-RPP linearly independent solutions as rows indexed by the variables x, y as columns. This matrix has rank $m-1$. For the remaining launching points $d \neq 1$ let A^d be the incidence matrices in Proposition 4. If we assign one of the K-RPP solutions in B^1 to the launching point $d = 1$ and the vector zero to the remaining launching points, we obtain a feasible solution for the MT-LARP in G satisfying $y_e^{11} = 0$. If we assign the vector zero to the launching point $d = 1$, one of the K-RPP solutions in A^d to a point $d \neq 1$ and zero to the remaining launching points, we obtain also a feasible solution for the MT-LARP in G satisfying $y_e^{11} = 0$. By expressing these solutions as the rows of a matrix we obtain the three first row blocks of the matrix in Fig. 2 for $D = 3$ launching points and $P = 2$ trucks, where z variables have been defined accordingly. Furthermore, if f_1, m_2 and m_3 are the first row of matrices B^1, A^2 and A^3, respectively, the last three rows of the matrix in Fig. 2 are also feasible solutions. It can be seen that such a construction can be done with all $D \geq P \geq 2$ values.

This matrix has $Dm+D-1$ rows representing MT-LARP solutions satisfying $y_e^{11} = 0$. The rank of this matrix is $Dm+D-1$ and, given that vector zero is in the affine hull of the vectors satisfying $y_e^{11} = 0$, we have $Dm+D$ affinely independent MT-LARP solutions satisfying $y_e^{11} = 0$ and the inequality $y_e^{11} \geq 0$ is facet inducing of MT-LARP(G). ♦

Note that inequality $y_e^{dk} \leq 1$, for each edge $e \in E_{NR}$, is dominated by inequality (6), $x_e^{dk} \geq y_e^{dk}$, and, hence, it does not induce a facet. Similarly, we have that inequality $x_e^{dk} \geq 0$, for each edge $e \in E_{NR}$, does not induce a facet. On the other hand, inequalities $x_e^{dk} \leq 1$ and $z^d \geq 0$ are dominated by inequality $x_e^{dk} \leq z^d$.

Proposition 6. *Inequality $x_e^{dk} \geq y_e^{dk}$, for each edge $e \in E_{NR}$, and $x_e^{dk} \geq 0$, for each edge $e \in E_R$, and for each flight $dk \in D \times \mathcal{K}$, are facet-inducing of MT-LARP(G).*

(x^{1*}, y^{1*})	(x^{2*}, y^{2*})	(x^{3*}, y^{3*})	z
$\bar{A}^1$	0	0	1 0 0 1 0 0 1 0 0
0	A^2	0	0 1 0 0 1 0 0 1 0
0	0	A^3	0 0 1 0 0 1 0 0 1
f_1	0	0	1 0 1
f_1	0	0	1 1 0
0	m_2	0	1 1 0

(x^{1*}, y^{1*})	(x^{2*}, y^{2*})	(x^{3*}, y^{3*})	z
M^1	0	0	0 0 0 0 0 0 0 0 0
$-f_1$ $-f_1$ $-f_1$	A^2	0	-1 1 0 -1 1 0 -1 1 0
$-f_1$ $-f_1$ $-f_1$	0	A^3	-1 0 1 -1 0 1 -1 0 1
0	0	0	0 0 1
0	0	0	0 1 0
0	0	0	1 0 0

Fig. 1. Matrices appearing in the proof of Proposition 4 for $D = 3$ and $P = 2$.

(x^{1*}, y^{1*})	(x^{2*}, y^{2*})	(x^{3*}, y^{3*})	z
B^1	0	0	1 0 0 1 0 0 1 0 0
0	A^2	0	0 1 0 0 1 0 0 1 0
0	0	A^3	0 0 1 0 0 1 0 0 1
f_1	0	0	1 0 1
0	m_2	0	1 1 0
0	0	m_3	0 1 1

Fig. 2. Matrix appearing in the proof of Proposition 5 for $D=3$ and $P=2$.

(x^{1*}, y^{1*})	(x^{2*}, y^{2*})	(x^{3*}, y^{3*})	z
B^1	0	0	1 0 0 1 0 0 1 0 0
0	B^2	0	0 1 0 0 1 0 0 1 0
0	0	B^3	0 0 1 0 0 1 0 0 1
f_1	0	0	1 1 0
0	f_2	0	0 1 1
0	0	f_3	1 0 1

Fig. 3. Matrix appearing in the proof of Proposition 7 for $D=3$ and $P=2$.

Proof. The proof is similar to that of Proposition 5 and is omitted here for the sake of brevity. ♦

Proposition 7. *Inequality (5), $\sum_{k\in\mathcal{K}}\sum_{d\in D} x_e^{dk} \geq 1$, for each edge $e \in E_R$, is facet-inducing of MT-LARP(G).*

Proof. Consider a given launching point d and the K-RPP defined on graph G considering the launching point d as the depot and looking for K tours that jointly traverses all the required edges $e \in E_R$. Since (V, E_{NR}) is 3-edge connected, inequality $\sum_{k=1}^{K} x_e^k \geq 1$ is facet inducing of K-RPP(G) (see Campbell et al., 2022 or Proposition 3) and there are $m = K(2|E_{NR}| + |E_R|)$ affinely independent K-RPP solutions formed with K tours that jointly traverse all the required edges and satisfy $\sum_{k=1}^{K} x_e^k = 1$. Furthermore, as vector zero is not in the affine hull of the vectors satisfying $\sum_{k=1}^{K} x_e^k = 1$, there are m linearly independent K-RPP solutions that satisfy $\sum_{k=1}^{K} x_e^k = 1$. Let B^d be the incidence matrix of the above m K-RPP linearly independent solutions as rows indexed by the variables x, y as columns, with rank m.

If we assign one of the K-RPP solutions in B^d to the launching point d and the vector zero to the remaining launching points, we obtain a feasible solution for the MT-LARP in G satisfying $\sum_{k\in\mathcal{K}}\sum_{d\in D} x_e^{dk} = 1$. By defining the z variables accordingly and expressing these solutions as the rows of a matrix we obtain a matrix similar to that in Fig. 3 for $D = 3$ and $P = 2$, where each f_i is the first row of matrix B^i. This matrix has $Dm+D$ rows representing MT-LARP solutions satisfying $\sum_{k\in\mathcal{K}}\sum_{d\in D} x_e^{dk} = 1$. The rank of this matrix is $Dm + D$ and, given that vector zero is not in the affine hull of the solutions satisfying $\sum_{k\in\mathcal{K}}\sum_{d\in D} x_e^{dk} = 1$, we have $Dm + D$ affinely independent MT-LARP solutions satisfying $\sum_{k\in\mathcal{K}}\sum_{d\in D} x_e^{dk} = 1$. Therefore, inequality (5) is facet inducing of MT-LARP(G). ♦

Proposition 8. *Inequality $z^d \leq 1$, for each $d \in D$, is facet-inducing of MT-LARP(G).*

(x^{1*}, y^{1*})	(x^{2*}, y^{2*})	(x^{3*}, y^{3*})	z
$\bar{A}^1$	0	0	1 1 0 1 1 0 1 1 0
0	A^2	0	1 1 0 1 1 0 1 1 0
0	0	A^3	1 0 1 1 0 1 1 0 1
f_1	0	0	1 0 1
0	m_2	0	0 1 1

Fig. 4. Matrix appearing in the proof of Proposition 9 for $D=3$ and $P=2$.

Proof. Without loss of generality, let us suppose that $d = 1$. All the solutions constructed in Proposition 4 but changing all the entries 0 in the column corresponding to z^1 by 1, are also MT-LARP solutions satisfying $z^1 = 1$ (see the matrix in Fig. 1a). Removing the last row, since it is now the same as the first row of block rows two, and proceeding as in Proposition 4, we obtain that the remaining $Dm + D$ rows are affinely independent, and all satisfy that $z^1 = 1$. Therefore, the inequality $z^1 \leq 1$ induces a facet of MT-LARP(G). ♦

Proposition 9. *Inequality (8), $\sum_{d\in D} z^d \leq P$, is facet-inducing of MT-LARP(G) if, and only if, $P < D$.*

Proof. If $P = D$ then $\sum_{d\in D} z^d \leq P = D$ is the sum of the inequalities $z^d \leq 1$ for all d.

If $P < D$, the proof is similar to that of Proposition 4 and we omit the details for brevity. Just note that the solutions built in Proposition 4 but changing the entries corresponding to the variables z as stated in the matrix in Fig. 4 are MT-LARP solutions that satisfy $\sum_{d\in D} z^d = P$. Proceeding as in Proposition 4, we obtain that these $Dm + D$ rows are affinely independent, and the inequality $\sum_{d\in D} z^d \leq P$ is facet inducing of MT-LARP(G). ♦

Proposition 10. *Connectivity inequalities (4), $\left(x^{dk} + y^{dk}\right)\left(\delta(S)\right) \geq 2x^{dk}_f$, $\forall dk \in \mathcal{D}\times\mathcal{K}$, $\forall S \subseteq V\backslash\{d\}$, and $\forall f \in E(S)$, are facet-inducing of MT-LARP(G) if subgraph $(S, E_{NR}(S))$ is 3-edge connected and either $|V\backslash S| = 1$ or subgraph $(V\backslash S, E_{NR}(V\backslash S))$ is 3-edge connected.*

Proof. Without loss of generality, let us suppose that $d = 1$ and $k = 1$. Consider the K-RPP defined on graph G considering the launching point $d = 1$ as the depot and looking for K tours that jointly traverses all the required edges $e \in E_R$. Since (V, E_{NR}) is 3-edge connected, and subgraph $(S, E_{NR}(S))$ is 3-edge connected and either $|V\backslash S| = 1$ or subgraph $(V\backslash S, E_{NR}(V\backslash S))$ is 3-edge connected, inequality $x^1\left(\delta(S)\right) + y^1\left(\delta(S)\right) \geq 2x^1_f$ is facet inducing of K-RPP(G) (see Campbell et al., 2022 or Proposition 3) and there are $m = K(2|E_{NR}| + |E_R|)$ affinely independent K-RPP solutions formed with K tours that jointly traverse all the required edges and satisfy $x^1\left(\delta(S)\right) + y^1\left(\delta_{NR}(S)\right) = 2x^1_f$. Given that vector zero is in the affine hull of the vectors satisfying $x^1\left(\delta(S)\right) + y^1\left(\delta_{NR}(S)\right) = 2x^1_f$, there are $m - 1$ of such solutions linearly independent. Let B^1 be the incidence matrix of these $m - 1$ K-RPP linearly independent solutions as rows indexed by the variables x, y as columns. This matrix B^1 has rank $m - 1$. For the remaining launching points $d \neq 1$ let A^d be the incidence matrices in Proposition 4. The remaining of the proof is similar to that of Proposition 5. ♦

2.3. Other valid inequalities for the MM-MT-LARP

In this section we present two families of inequalities that are valid for the MM-MT-LARP and are also facet-inducing of MT-LARP(G) under mild conditions, the parity inequalities and the p-connectivity inequalities, as well as the max-length inequalities, which we do not know if they induce a facet of the polyhedron or not but are valid for the MM-MT-LARP and have proven to be very useful computationally.

2.3.1. Parity inequalities

Although constraints (3) could be linearized by adding some new variables, we consider the following *parity inequalities* instead:

$$(x^{dk} - y^{dk})(\delta(S)\backslash F) \geq (x^{dk} - y^{dk})(F) - |F| + 1, \quad \forall dk \in \mathcal{D}\times\mathcal{K}, \ \forall S \subset V, \forall F \subseteq \delta(S) : |F| \text{ is odd} \tag{19}$$

Proposition 11. *Parity inequalities (19), for all $dk \in \mathcal{D}\times\mathcal{K}$, for all $S \subset V$, and for all $F \subseteq \delta(S)$ such that $|F|$ is odd, are valid for MT-LARP(G).*

Proof. Let $(\bar{x}, \bar{y}, \bar{z})$ be an arbitrarious MT-LARP solution and let us see that it satisfies $(\bar{x}^{dk} - \bar{y}^{dk})(\delta(S)\backslash F) \geq (\bar{x}^{dk} - \bar{y}^{dk})(F) - |F| + 1$.

If $(\bar{x}, \bar{y}, \bar{z})$ satisfies $(\bar{x}^{dk} - \bar{y}^{dk})(F) \leq |F| - 1$, the inequality reduces to $(\bar{x}^{dk} - \bar{y}^{dk})(\delta(S)\backslash F) \geq 0$, which is obviously satisfied. Otherwise, the solution $(\bar{x}, \bar{y}, \bar{z})$ satisfies $(\bar{x}^{dk} - \bar{y}^{dk})(F) = |F|$, i.e., $\bar{x}^{dk}(F) = |F|$ and $\bar{y}^{dk}(F) = 0$, and the inequality reduces to $(\bar{x}^{dk} - \bar{y}^{dk})(\delta(S)\backslash F) \geq 1$. Since the drone tour dk must traverse the cut-set $\delta(S)$ an even number of times, and $|F|$ is odd, $\delta(S)\backslash F$ must be traversed an odd number of times. Hence, there is, at least an edge $e^* \in \delta(S)\backslash F$ such that $\bar{x}^{dk}_{e^*} - \bar{y}^{dk}_{e^*} = 1$ and we are done. ♦

Constraints (19) cut off infeasible solutions in which there is a cut-set with an odd number of edges traversed exactly once (these edges define the set F) and the other edges are traversed twice or none.

Proposition 12. *Parity inequalities (19), for all $dk \in \mathcal{D}\times\mathcal{K}$, for all $S \subset V$, and for all $F \subseteq \delta(S)$ such that $|F|$ is odd, are facet-inducing of MT-LARP(G) if subgraphs $(S, E_{NR}(S))$ and $(V\backslash S, E_{NR}(V\backslash S))$ are 3-edge connected.*

Proof. (a) Let us suppose that $|F| = 1$. Without loss of generality, let us assume that $d = 1$ and $k = 1$. Consider the K-RPP defined on graph G considering the launching point $d = 1$ as the depot and looking for K tours that jointly traverses all the required edges $e \in E_R$. Since subgraphs $(S, E_{NR}(S))$ and $(V \setminus S, E_{NR}(V \setminus S))$ are 3-edge connected, inequality $(x^1 - y^1)(\delta(S)\backslash F) \geq (x^1 - y^1)(F) - |F| + 1$ is facet inducing of K-RPP(G) (see Campbell et al., 2022) and there are $m = K(2|E_{NR}| + |E_R|)$ affinely independent K-RPP solutions formed with K tours that jointly traverse all the required edges and satisfy $(x^1 - y^1)(\delta(S)\backslash F) = (x^1 - y^1)(F) - |F| + 1$. Given that vector zero is in the affine hull of the vectors satisfying the inequality as equality (since $|F| = 1$), there are $m - 1$ linearly independent K-RPP solutions formed with K tours that jointly traverse all the required edges and satisfy $(x^1 - y^1)(\delta(S)\backslash F) = (x^1 - y^1)(F) - |F| + 1$. Let B^1 be the incidence matrix of these $m - 1$ K-RPP linearly independent solutions as rows indexed by the variables x, y as columns. This matrix B^1 has rank $m - 1$. The remaining of the proof is similar to that of Proposition 5 in this case.

(x^{1*}, y^{1*})	(x^{2*}, y^{2*})	(x^{3*}, y^{3*})	z
B^1	0	0	1 0 0 1 0 0 1 0 0
f_1	A^2	0	1 1 0 1 1 0 1 1 0
f_1	0	A^3	1 0 1 1 0 1 1 0 1
f_1	0	0	1 1 0
f_1	0	0	1 0 1
f_1	m_2	m_3	1 1 1

Fig. 5. Matrix appearing in the proof of Proposition 12 for $D=3$ and $P=2$.

(b) Let us suppose now that $|F| > 1$. In this case, the proof is similar to the previous one but using MT-LARP solutions built as those represented in the matrix in Fig. 5 (the rank of submatrix B^1 is m since vector zero is not in the affine hull of the vectors satisfying the inequality as equality). This matrix has $Dm + D$ rows representing MT-LARP solutions satisfying the inequality as equality and its rank is $Dm + D$. Therefore, since vector zero is not in the affine hull of the solutions satisfying the inequality as equality, the parity inequality with $|F| \geq 3$ is facet inducing of MT-LARP(G). ♦

Remark 1. *It can be seen that Proposition 12 also applies if one of the two shores S or $V \setminus S$ consists of only one vertex.*

2.3.2. *p-connectivity inequalities*

These constraints are based on those introduced in Corberán et al. (2013) for the maximum benefit CPP to cut off some fractional solutions in which several variables take value 0.5. For example, consider an MT-LARP instance with the vertex set V divided into three subsets S_0, S_1 and S_2 as shown in Fig. 6a, with a given launching point $d \in S_0$, represented by a triangle, and a fractional solution where, for a given k, we have $x_e^{dk} = y_e^{dk} = 0.5$ for the three edges depicted in the figure joining the sets S_i, $x_e^{dk} = y_e^{dk} = 0$ for the remaining edges joining the sets S_i, and $x_e^{dk} = 1$, $y_e^{dk} = 0$ for the two edges drawn in $E(S_1)$ and $E(S_2)$. It can be seen that this solution satisfies all the inequalities presented in previous sections. In particular, it satisfies as equality the connectivity inequality (4) associated with sets S_0, S_1 and S_2. The previous fractional solution is cut by the following inequality, valid for the MT-LARP (Proposition 13):

$$\left(x^{dk} + y^{dk}\right)\left(\delta(S_0)\right) + 2x^{dk}\left(S_1 : S_2\right) \geq 2x_{e_1}^{dk} + 2x_{e_2}^{dk}.$$

In general, let $\{S_0, \ldots, S_p\}$ be a partition of V, a launching point $d \in D$, and a flight $k \in \mathcal{K}$. Assume that $d \in S_i$, $i \in \{0, \ldots, p\}$, and select an edge $e_j \in E(S_j)$ for each $j \in \{0, \ldots, p\}\setminus\{i\}$ (see Fig. 6b). We will call *p-connectivity* inequality to:

$$\left(x^{dk} + y^{dk}\right)\left(\delta(S_0)\right) + 2\sum_{1\leq r<t\leq p} x^{dk}(S_r : S_t) \geq 2\sum_{j\in\{0,\ldots,p\}\setminus\{i\}} x_{e_j}^{dk} \qquad (20)$$

Proposition 13. *p-connectivity inequalities (20) are valid for MT-LARP(G).*

Proof. For the sake of simplicity we will assume that $d \in S_0$. Let $(\bar{x}, \bar{y}, \bar{z})$ be any MT-LARP solution and let us prove that it satisfies

$$(\bar{x}^{dk} + \bar{y}^{dk})(\delta(S_0)) + 2\sum_{1\leq r<t\leq p} \bar{x}^{dk}(S_r : S_t) \geq 2\sum_{j=1}^{p} \bar{x}_{e_j}^{dk}.$$

If $\bar{x}_{e_j}^{dk} = 0$ for some $j \in \{1, \ldots, p\}$, then we could consider a new $(p-1)$-connectivity configuration where sets S_j and S_{j+1} have been merged into a single vertex $S_j \cup S_{j+1}$ (or S_j and S_{j-1} if $j = p$). It can be proved that if its associated $(p-1)$-connectivity inequality is satisfied by $(\bar{x}^{dk}, \bar{y}^{dk})$, then the original p-connectivity inequality is also satisfied by $(\bar{x}^{dk}, \bar{y}^{dk})$. So, we can assume that $\bar{x}_{e_1}^{dk} = \cdots = \bar{x}_{e_p}^{dk} = 1$.

If $\bar{x}_e^{dk} = 1$ for some $e \in (S_r : S_t)$ with $1 \leq r < t \leq p$, then we can consider a new $(p-1)$-connectivity configuration where sets S_r and S_t have been merged into $S'_r = S_r \cup S_t$ and where $e'_r = e_r$. Again, it can be proved that if its associated $(p-1)$-connectivity inequality is satisfied by $(\bar{x}^{dk}, \bar{y}^{dk})$, then the original p-connectivity inequality is also satisfied by $(\bar{x}^{dk}, \bar{y}^{dk})$. Hence, we can assume that $\bar{x}^{dk}(S_r : S_t) = 0$ for each r, t, $1 \leq r < t \leq p$.

Therefore, since $d \in S_0$ and $\bar{x}_{e_i}^{dk} = 1$, $(\bar{x}^{dk} + \bar{y}^{dk})(S_0 : S_i) \geq 2$ for each $i = 1, \ldots, p$, and the inequality holds. ♦

Proposition 14. *p-connectivity inequalities (20) are facet-inducing for MT-LARP(G) if subgraphs $(S_j, E_{NR}(S_j))$, $\forall j = 0, \ldots, p$, are 3-edge connected, $|(S_0 : S_j)| \geq 2$, $\forall j = 1, \ldots, p$, the graph induced by $V \setminus S_0$ is connected, and all the edges $e_j \in E(S_j)$, $\forall j \in \{0, \ldots, p\}\setminus\{i\}$, are required edges.*

Proof. Without loss of generality, let us suppose that $d = 1$ and $k = 1$. Consider the K-RPP defined on graph G considering the launching point $d = 1$ as the depot and looking for K tours that jointly traverses all the required edges $e \in E_R$. Since subgraphs $(S_j, E_{NR}(S_j))$, $\forall j = 0, \ldots, p$, are 3-edge connected, $|(S_0 : S_j)| \geq 2$, $\forall j = 1, \ldots, p$, and the graph induced by $V \setminus S_0$ is connected, and all the edges $e_j \in E(S_j)$, $\forall j \in \{0, \ldots, p\}\setminus\{i\}$, are required edges, inequality

$$\left(x^1 + y^1\right)\left(\delta(S_0)\right) + 2\sum_{1\leq r<t\leq p} x^1(S_r : S_t) \geq 2\sum_{j\in\{0,\ldots,p\}\setminus\{i\}} x_{e_j}^1$$

is facet inducing of K-RPP(G) (see Campbell et al., 2022) and there are $m = K(2|E_{NR}| + |E_R|)$ affinely independent K-RPP solutions formed with K tours that jointly traverse all the required edges and satisfy the previous inequality as equality.

Given that vector zero is in the affine hull of the vectors satisfying the inequality as equality, there are $m - 1$ linearly independent K-RPP solutions formed with K tours that jointly traverse all the required edges and satisfy the inequality as equality. Let B^1 be the incidence matrix of these $m - 1$ K-RPP linearly independent solutions as rows indexed by the variables x, y as columns. This matrix B^1 has rank $m-1$. The remaining of the proof is similar to that of Proposition 5. ♦

2.3.3. *Max-length inequalities*

The inequalities presented in this section are based on the limit L to the cost of a drone flight. Consider an edge set $F \subseteq E_R$ such that the cost of the optimal tour (or a lower bound to it) starting and ending at a given launching point $\bar{d}$, and traversing all the edges in F, is greater than L. We have that a single flight k cannot service all the arcs in F from $\bar{d}$ and, hence, inequalities

$$x^{\bar{d}k}(F) \leq |F| - 1, \quad \forall k \in \mathcal{K}, \qquad (21)$$

called *disaggregate max-length* inequalities, are valid for the MM-MT-LARP.

On the other hand, under the same circumstances, for a given k, at least another flight different from $(\bar{d}, k)$ must enter any subgraph of G that contains all edges in F but does not contain any launching point

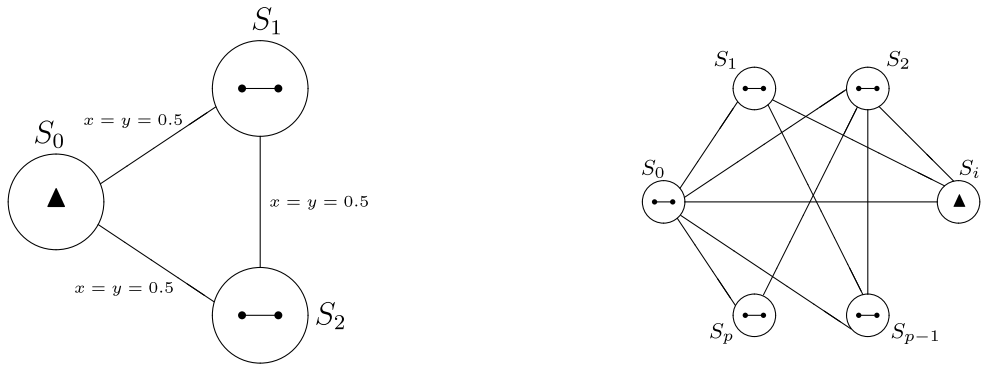

Fig. 6. p-connectivity inequalities.

$d \in D$. Hence, if $W \subset V \setminus D$ is a set of vertices containing the vertices incident with the edges in F, the following inequalities

$$\sum_{(d',k') \neq (\bar{d},k)} \left(x^{d'k'} + y^{d'k'}\right) \left(\delta(W)\right) \geq 2, \quad \forall k \in \mathcal{K}, \tag{22}$$

called *aggregate max-length* inequalities, are also valid for the MM-MT-LARP on G.

Remark 2. *If, in inequalities (22) the set W includes some launching points, the inequality could be changed to*

$$\sum_{\substack{(d',k') \neq (\bar{d},k) \\ d \notin W}} \left(x^{d'k'} + y^{d'k'}\right) \left(\delta(W)\right) \geq 2 - \sum_{d \in W \cap D} 2z^d, \quad \forall k \in \mathcal{K}. \tag{23}$$

2.4. Other inequalities that strengthen the formulation

The inequalities in this section are not valid for the MM-MT-LARP, because they are not satisfied by all the MM-MT-LARP solutions, but they can be used because there is at least one optimal solution that satisfies them. Furthermore, the following results allow us to remove some y variables from the formulation (see Note 3).

2.4.1. Maximum deadheading inequalities

For each $n = 1, 2, \ldots$, let V_R^n denote the set of vertices in V_R with required degree n (the number of required edges incident with the vertex). Note that the required degree of vertices in D is zero.

Proposition 15. *There is an optimal MM-MT-LARP solution in which each drone tour $dk \in D \times \mathcal{K}$, considered as a walk (sequence of edge traversals), satisfies:*

(a) It does not deadhead two edges (i,j), (j,k) consecutively.
(b) It does not visit the vertices in D, except the initial and final visit to d.
(c) It does not visit a vertex in V_R^n more than n times.
(d) It does not deadhead more than twice a cutset $\delta(i)$ for any vertex $i \in V$.

Proof. (a) Let us suppose a tour deadheads two edges (i,j), (j,k) consecutively. These two edges are non-required since each required edge is only traversed once when it is serviced. Recall that the non-required edge (i,k) exists because E_{NR} induces a complete graph, and that $c_{ik} \leq c_{ij} + c_{jk}$ due to the triangle inequality. The tour obtained after replacing the traversal of $(i,j),(j,k)$ with the traversal of (i,k) has a lower or equal cost and services the same required edges as the former tour.

(b) A drone tour starts and ends at a given launching point $d \in D$. If it visits d again, or another $d' \in D$, it deadheads two edges, which contradicts (a).

(c) Let us suppose a tour visits a given vertex $j \in V_R^n$ more than n times (each visit implies traversing two edges). Note that there are only n services associated with vertex j: the service of the n required edges incident with it. Therefore, all visits to the vertex j, except n of them, are performed by deadheading two edges consecutively, which contradicts (a).

(d) Let us suppose a tour traverses a cutset $\delta(i)$ three times in deadheading. From (a), we can assume that no two of these traversals are consecutive. Hence, each visit to i consists of traversing one required edge and one non-required one. Let us suppose that two visits are performed by traversing first the required edge and then the non-required one (if it is the other way round, the argument is analogous), and let (i,v_1) and (i,v_2) be these two non-required edges. Then, by removing these two traversals, changing the direction of traversal of the edges between v_1 and i and then adding the traversal of the non-required edge (v_1,v_2) we obtain a new drone tour with lower or equal cost, because $c_{v_1v_2} \leq c_{v_1i} + c_{iv_2}$. ♦

Corollary 1. *The following inequalities can be added to the formulation:*

$$(x^{dk} + y^{dk})(\delta(d)) \leq 2, \quad \forall dk \in D \times \mathcal{K} \tag{24}$$

$$x_e^{dk} = y_e^{dk} = 0, \quad \forall dk \in D \times \mathcal{K}, \quad \forall e \in \delta(d'), \ \forall d' \in D \setminus \{d\} \tag{25}$$

$$\sum_{e \in \delta_{NR}(i)} x_e^{dk} = x_{ij}^{dk}, \quad \forall i \in V_R^1, \ (i,j) \in E_R, \ \forall dk \in D \times \mathcal{K} \tag{26}$$

$$y_e^{dk} = 0, \quad \forall \ dk \in D \times \mathcal{K}, \quad \forall \ e = (i,j) \in E_{NR} \text{ with } i \in V_R^1 \tag{27}$$

$$\sum_{e \in \delta_{NR}(i)} \left(x_e^{dk} + y_e^{dk}\right) \leq \sum_{e \in \delta_R(i)} x_e^{dk}, \ \forall i \in V_R^n \ (n > 1), \ \forall dk \in D \times \mathcal{K} \tag{28}$$

$$\sum_{e \in \delta_{NR}(i)} \left(x_e^{dk} + y_e^{dk}\right) \leq 2, \ \forall i \in V, \ \forall dk \in D \times \mathcal{K}. \tag{29}$$

Proof. All these inequalities except for (26) can be trivially deduced from Proposition 15: Inequalities (24) and equalities (25) are deduced from (b). Note that inequalities (24) are not equalities because some drone tours could be empty. If $i \in V_R^1$ and $(i,j) \in E_R$, from (c) it is deduced that $\sum_{e \in \delta_{NR}(i)} \left(x_e^{dk} + y_e^{dk}\right) \leq x_{ij}^{dk}$. Since the number of edges incident with any vertex must be even, this inequality must be satisfied with equality. Moreover, given that the right-hand side is at most 1 and $y_e^{dk} \leq x_e^{dk}$, all the variables y_e^{dk} must be 0 and can be removed, thus obtaining (26) and (27). Inequalities (28) are deduced from (c). Inequalities (29) are deduced from (d). ♦

Remark 3. *The variables in (25) and (27) for each drone tour dk can be removed from the formulation, rather than set equal to zero.*

2.4.2. Symmetry-breaking inequalities

To avoid equivalent solutions produced by the symmetry among the different flights of each drone, the following symmetry-breaking inequalities are also added. Assume the required edges are ordered as $(e_1, e_2, \dots, e_m)$. For each launching point $d \in D$ we add the constraints

$$x_{e_i}^{dk} \le \sum_{j=k-1}^{i-1} x_{e_j}^{d(k-1)}, \quad \forall k = 2, \dots, K,\ i = k, \dots, m \tag{30}$$

$$x_{e_i}^{dk} = 0, \quad \forall k = i+1, \dots, K,\ i = 1, \dots, m-1 \tag{31}$$

Constraints (30) ensure that a flight dk can only service edge e_i if the previous flight $d(k-1)$ has serviced at least one of the previous edges in the ordering. Constraints (31) ensure that no edge before e_k in the ordering can be served by flight dk. Note that flights based on different launching points are not interchangeable, because they have to start and end at a different vertex. Therefore, we do not need symmetry-breaking constraints involving flights from different launching points.

3. Branch-and-cut algorithm for the MM-MT-LARP

Based on the formulation presented in Section 2, we have implemented a branch-and-cut algorithm for the MM-MT-LARP. The initial LP includes inequalities (2), (6), (7) and (8), inequalities (5) as equalities, and the bounds on the variables. It also includes inequalities in Proposition 1 for each $e \in E_R$, the constraints in Corollary 1, and the symmetry-breaking inequalities (30)–(31). Moreover, the variables considered in Note 3 are fixed to zero. Connectivity inequalities (4), and the previous parity (19), p-connectivity (20), and max-length (21)–(22) inequalities can be separated with the algorithms described in this section. The order and frequency in which these algorithms are applied will be determined by the computational experiments described in Section 5.

3.1. Separation algorithms

The separation algorithms are used at each iteration of the cutting-plane procedure to identify valid inequalities violated by the current LP solution, say $(\bar{x}, \bar{y}, \bar{z})$. In some of these procedures, we will use a parameter ε that takes values close to zero. In what follows we describe these separation procedures.

Connectivity inequalities (4)

Although connectivity inequalities (4) for each flight $dk \in \mathcal{D} \times \mathcal{K}$ can be exactly separated in polynomial time, this exact procedure is quite time-consuming and usually produces many violated inequalities that are very similar to each other. Therefore, we have decided not to use it and implement a heuristic separation algorithm instead. We compute the connected components of the graph induced by the edges $e \in E$ such that $\bar{x}_e^{dk} + \bar{y}_e^{dk} > \varepsilon$, plus the launching point d, if necessary. For each connected component and the edge f with maximum $\bar{x}_f^{dk}$, the corresponding inequality (4) is checked for violation.

Parity inequalities (19)

For each flight $dk \in D \times \mathcal{K}$ we consider all the subsets $S \subset V$ obtained by computing the connected components of the graph induced by the edges $e \in E_{NR}$ with $\bar{x}_e^{dk} - \bar{y}_e^{dk} > \varepsilon$ and the edges $e \in E_R$ with $\bar{x}_e^{dk} > \varepsilon$, plus the launching point d, if necessary. For each one of these subsets S, we apply a procedure similar to that described in Ghiani and Laporte (2000) for choosing the subset of edges $F \subseteq \delta(S)$ that produces the most violated parity inequality (19) associated with S. This procedure is as follows:

We first include in F each edge $e \in \delta_{NR}(S)$ with $\bar{x}_e^{dk} - \bar{y}_e^{dk} > 0.5$ and each $e \in \delta_R(S)$ with $\bar{x}_e^{dk} > 0.5$. If $|F|$ is odd, we are done. If $|F|$ is even, we consider the edge in F with the lowest value of $\bar{x}_e^{dk} - \bar{y}_e^{dk}$ (or $\bar{x}_e^{dk}$) and the edge in $\delta(S) \setminus F$ with the highest value of $\bar{x}_e^{dk} - \bar{y}_e^{dk}$ (or $\bar{x}_e^{dk}$), and we choose either to remove the first one from F or to add the second one to F. Finally, the parity inequality corresponding to sets S and F is checked for violation.

In addition to the previous algorithm, we try a second one consisting of applying the Ghiani and Laporte procedure to all sets S containing one single vertex. Note that, by applying this algorithm, we guarantee that the integer solutions found will satisfy the parity conditions.

p-connectivity inequalities (20)

To separate these inequalities, we have devised the following heuristic algorithm. We first compute the connected components of the graph induced by the required edges with $\bar{x}_e^{dk} > 0$. Let us call C^i to the vertex sets of these components. We iteratively shrink two sets C^i and C^j if $(\bar{x}^{dk} + \bar{y}^{dk})(C^i : C^j) \ge 2$, while possible. For each resulting set of vertices C^i we proceed as follows. For each C^i that contains more than one vertex or the launching point d, let us denote $S_0 = C^i$ and the remaining sets C^j as $S_1, \dots, S_p$. For each set S_i containing one single vertex, we merge it with the more convenient set S_j. Then, for each set S_i not containing d, we choose the edge $e_i \in E(S_i)$ with maximum value $\bar{x}_{e_i}^{dk}$. While $p > 2$, we try to merge pairs of sets S_i and S_j according to the following rules (to allow a simpler notation, we will assume that for the set S_i containing d, $\bar{x}_{e_i}^{dk} = 1$):

- If $i, j \neq 0$ and $\bar{x}^{dk}(S_i : S_j) \ge \min\{\bar{x}_{e_i}^{dk}, \bar{x}_{e_j}^{dk}\}$, we merge S_i and S_j. Notice that if we shrink S_i and S_j, the LHS of the inequality (20) decreases by $2\bar{x}^{dk}(S_i : S_j)$ while the RHS decreases by $2\min\{\bar{x}_{e_i}^{dk}, \bar{x}_{e_j}^{dk}\}$.
- If $j = 0$ (or $i = 0$) and $\left(\bar{x}^{dk} + \bar{y}^{dk}\right)(S_0 : S_i) + \sum_{r \neq 0,i} \left(\bar{x}^{dk} - \bar{y}^{dk}\right)(S_i : S_r) \ge 2\min\{\bar{x}_{e_i}^{dk}, \bar{x}_{e_0}^{dk}\}$ we make $S_0 := S_0 \cup S_i$. Note that if we shrink S_0 and S_i, the LHS of the inequality (20) decreases by $\left(\bar{x}^{dk} + \bar{y}^{dk}\right)(S_0 : S_i)$ since these variables disappear, and by $\sum_{r \neq 0,i} \left(\bar{x}^{dk} - \bar{y}^{dk}\right)(S_i : S_r)$ since each $2\bar{x}^{dk}(S_i : S_r)$ becomes $(\bar{x}^{dk} + \bar{y}^{dk})(S_i : S_r)$, while the RHS decreases by $2\min\{\bar{x}_{e_i}^{dk}, \bar{x}_{e_0}^{dk}\}$.

The p-connectivity inequality (20) associated with the resulting sets $S_0, S_1, \dots, S_p$ is checked for violation. If no violated p-connectivity inequality has been generated, we repeat the previous process starting from the connected components of the graph induced by the edges (required or not) with $\bar{x}_e^{dk} > 0.5$.

Max-length inequalities (21) –(22)

Max-length inequalities are separated by using a heuristic procedure. This procedure first looks for violated inequalities (21). Given a set of edges F, let $\text{GRP}^d(F)$ be a lower bound of the cost of the GRP instance defined in G by the edges in F as required edges and the launching point d as a required vertex. For each flight $dk \in D \times \mathcal{K}$, let $\{e_1, e_2, \dots, e_m\}$ be a set of required edges such that $\bar{x}_{e_1}^{dk} \ge \bar{x}_{e_2}^{dk} \ge \dots \ge \bar{x}_{e_m}^{dk} \ge 0.5$. We define $F = \{e_1, e_2, \dots, e_f\}$, where f is the maximal number such that $\bar{x}^k(F) > |F| - 1 + \varepsilon$. We check if $\text{GRP}^d(F)$ is greater than L and, therefore, the corresponding inequality (21) is violated. Otherwise, for each edge $\overline{e} \in \{e_{f+1}, \dots, e_m\}$, we iteratively consider the set $\overline{F} = F \cup \{\overline{e}\}$ and check if $\text{GRP}^d(\overline{F})$ is greater than L. For each subset F (or $\overline{F}$) whose corresponding inequality (21) is violated, we check the max-length inequality (22) corresponding to the cut-set $\delta(W)$, where W is the set of vertices incident with the edges in F (or $\overline{F}$).

Given a set of edges F, the value of $\text{GRP}^d(F)$ is computed by solving the corresponding GRP with the branch-and-cut algorithm described in Corberán et al. (2007). To minimize the number of GRPs solved, two lists containing the already studied sets F of edges are managed. Note that we do not separate max-length inequalities (23), since in our instances, the launching points are not incident with required edges, and thus these inequalities can be reduced to inequalities (22) by removing the launching points from set W.

3.2. The cutting-plane algorithm

The cutting-plane algorithm is applied at each node of the search tree. It applies, at each iteration and for each flight $dk \in \mathcal{D} \times \mathcal{K}$, all the separation heuristic algorithms described in Section 3.1:

- The heuristic for connectivity inequalities (4) with $\varepsilon = 0, 0.25, 0.5$, where the value of ε is increased only if no violated inequalities are found with the previous value.
- The exact parity separation procedure for single vertices and the heuristic procedure for parity inequalities (19) with $\varepsilon = 0, 0.25, 0.5$.
- The heuristic for identifying violated max-length inequalities (21)–(22) with $\varepsilon = 0.5$.
- The heuristic separation algorithm for p-connectivity inequalities (20).

The order and frequency in which these separation algorithms are applied are established from the computational study carried out in Section 5.2.

4. Matheuristic algorithm for the MM-MT-dLARP

In this section, we introduce a matheuristic algorithm proposed for the approximate solution of the MM-MT-dLARP. Each instance of MM-MT-dLARP has required lines and launching points (including the depot). Each line in an MM-MT-dLARP instance is represented by a polygonal chain with a large number of points and segments, offering a close approximation to the original line. Our first simplification of the problem is to restrict the drone entry and exit to the line through these points, thus obtaining the MM-MT-LARP, a discrete optimization problem. Note that if the number of points describing each line is large, given that the non-required edges induce a complete graph, the size of the MM-MT-LARP instances makes them very hard to solve. Our proposed approach is to address this complexity by working with smaller MM-MT-LARP instances, not considering all points. We select only a small but significant subset of intermediate points of each line. The resulting segments connecting these points form the required edges of the MM-MT-LARP instance, with service costs proportional to the lengths of the segments in such a way the sum of these costs equals the service cost of the original line. At the beginning of the algorithm, no intermediate points are considered, and each line is represented by a single segment. Thereafter, some intermediate points are added and removed following the rules described in Section 4.3.

The algorithm we present combines a construction phase, four local search procedures integrated into a Variable Neighborhood Descent (VND) algorithm for improving the solutions, and a set of guidelines for selecting intermediate points.

4.1. Construction phase

To find an initial set of MM-MT-LARP solutions, we proceed iteratively as follows. We select randomly P launching points from the set D. Then, each required edge is assigned randomly to one of the P launching points with some probabilities that are inversely proportional to the distance from the edge to each launching point. Once the edges are assigned to the launching points, for each launching point d we construct a giant tour T^d, starting and ending at d and traversing all the edges assigned to it, following the Nearest Neighbor strategy. Starting at d, the tour goes to its closest vertex incident with an assigned edge e. After traversing edge e, from the other endpoint of e, the tour goes to the closest vertex incident with an assigned edge not traversed yet. This process is repeated until all the assigned edges are traversed and the tour ends at d.

Each tour T^d determines an ordered sequence of edges but does not necessarily satisfy the time limitation L for the flights. Therefore, we split each T^d into feasible flights in the following way. Let $\{e_1, e_2, \dots\}$ be the sequence of required edges in the order determined by the giant tour T^d. The first flight starts at d and follows the giant tour while the time limitation of the flight is satisfied, taking into account the time of the way back to the launching point. Let e_i be the last required edge of T^d traversed by the previous flight. Then, a new flight starts at d and goes to e_{i+1}. It continues following T^d until adding one more edge would make the flight infeasible. This procedure is repeated until all edges assigned to d are traversed by a flight. After applying this procedure to all the giant tours T^d, we have an MM-MT-LARP solution. We iteratively repeat this process until we have a given number of different MM-MT-LARP solutions or a given computing time is reached. Specifically, if 300 different solutions have been generated in less than 30 s, the algorithm stops. Otherwise, it stops when it reaches 20 different solutions or after 600 s, whatever happens first.

4.2. Local search procedures

To improve a given MM-MT-LARP solution, we have implemented the following local search procedures that iteratively explore different ways of exchanging required edges either within the same flight or between different flights:

(*i*) *Destroy and repair move*. For the launching point d with the largest makespan, this second method randomly selects n required edges among all the flights of d, where n is a random number between 1 and $n_{max} = 5$, and removes them from their flights. Then, in the same order they were removed, we evaluate the cost of inserting them in any flight and position. If the new solution obtained is not better than the starting one (the makespan has not been reduced), or if it is not possible to insert an edge in any flight without exceeding L, the changes are discarded and a new iteration is executed. This procedure stops when $it_{max} = 30$ consecutive iterations that do not improve the current solution are performed.

(*ii*) *Intraroute move*. This first procedure is applied to each flight of each launching point d to try to reduce its cost. Iteratively, each required edge is moved to the position and traversed in the direction of that flight which minimizes its total cost until no improvements can be found.

The following two methods swap sequences of at most ℓ_{max} consecutive required edges between two flights by applying a *first improvement* strategy.

(*iii*) *0 to ℓ exchange*. This procedure starts with $\ell = 1$ and studies the removal of all the possible sequences of ℓ edges from a flight based on the launching point d with the largest makespan, and their insertion into another flight and position. If the solution is improved, ℓ is reset to 1, otherwise, it is incremented by one unit. The procedure stops when ℓ reaches $\ell_{max} = 3$.

(*iv*) *ℓ_1 to ℓ_2 exchange*. As in the previous move, chains of ℓ_1 consecutive edges of flights from the launching point d with the largest makespan are selected, but this move tries to swap these chains with another chain of ℓ_2 consecutive edges from another flight. We begin with $\ell_1 = \ell_2 = 1$ and then increase ℓ_2 one by one until ℓ_2 reaches $\ell_{max} = 3$ (without improvement). We then increase ℓ_1 and reset ℓ_2 to 1. If an improvement is achieved, both parameters are reset to 1, and we stop when $\ell_1 = \ell_2 = \ell_{max}$.

To improve the MM-MT-LARP solutions, we have embedded the above improvement procedures into a Variable Neighborhood Descent (VND) algorithm (see Hansen and Mladenović, 2001). Once the VND procedure has been performed, an optimization procedure is applied to each flight from the launching point d with the largest makespan. For each flight, we define a GRP instance on the graph induced in G by the required edges and its launching point. This GRP instance is solved to optimality with the branch-and-cut algorithm proposed in Corberán et al. (2007). If after this procedure the launching point with the largest makespan has changed, it is applied again to the flights of the new one.

4.3. "Splitting" phase: the selection of intermediate points

For each of the 10 best solutions obtained so far, and in order to take advantage of the fact that drones can enter and leave a line through any intermediate point, we try to obtain better solutions by "splitting" some required edges by adding intermediate points to them. These intermediate points are selected as follows: for each required edge, the midpoint (equidistant to the two endpoints of the edge) is added, creating a new instance where each original edge becomes now two required edges. Note that each previous solution can be trivially converted to a solution in this new instance, where the two halves of each edge are traversed consecutively. The VND procedure is now applied to this solution to try to find a better solution that uses these new intermediate points to enter or leave the lines. If the new solution uses some of these intermediate points, we construct a new instance as follows. For those points that are used to enter or leave a required edge, we add two new midpoints on their incident required edges. Those intermediate points that are not used are removed. We apply the VND again to the solution in this new instance. This procedure consisting of adding new intermediate points and applying the VND is repeated until no improvement is observed.

5. Computational experiments

The computational experiments have been executed on an i7-9700F CPU at 3 GHz. All the algorithms have been coded in C++ and the branch and cut uses Cplex 22.1 Concert Technology, with Cplex cuts turned off, and a time limit of one hour.

5.1. Instances

Since we do not have instances based on real-life data, we have randomly generated some MM-MT-dLARP instances in a similar way as the one presented in Campbell et al. (2018) for the DARP. For the generation of these MM-MT-dLARP instances, we consider a total of 17 different grids with sizes ranging between 4×4 and 10×10. For each grid size, we have generated D random points, $3 \leq D \leq 6$, that will act as potential launching points. For each combination of grid size and number of launching points, two graphs have been generated with different values for parameters p (probability for an edge to be declared required) and $curvature$ (measure of how curved an edge is). The first one with $p = 0.3$ and $curvature = 0.4$ and the second one with $p = 0.4$ and $curvature = 0.3$. For each one of these 136 graphs, generated with a combination of those parameters, we generate several MM-MT-dLARP instances, one for each value of P, with P ranging from 2 to $D - 1$. Hence, we obtain a total of 340 instances. The characteristics of these instances are shown in Table 1, organized into three classes (small, medium, and large) according to the number of vertices of the initial grid mentioned before.

Each one of the 34 rows in Table 1 corresponds to a set of 4 different graphs, one for each value of $D \in \{3, 4, 5, 6\}$, and a set of 10 instances, one for each value of P, $2 \leq P \leq D - 1$. Columns 2 and 3 report the average numbers of original lines and vertices, respectively, of the 4 graphs. The last column shows the average number of total vertices, including the intermediate points in each line. The digits at the end of the name in the first column indicate the grid size and whether the instances have been generated with version 1 or 2 of the combination of parameters p and $curvature$. Thus, for example, the 10 instances grouped as "MM-MT-dLARP6_8_2" correspond to 4 graphs generated with grid size 6×8 and version 2 ($p = 0.4$, $curvature = 0.3$). The 136 graphs and the results presented in this section for the corresponding 340 instances can be found in http://www.uv.es/plani/instancias.html.

For each of these 136 graphs, we must choose a value for L such that the instance is not infeasible. For each required edge, we calculate the cost of flying to it from its closest launching point, servicing it, and going back to the launching point. Let L_{min} be the maximum of such values for all the required edges. Note that a value for L lower than L_{min} would produce an infeasible instance. Now we set $L = 1.5 \times L_{min}$. It may still be possible that some instance turns out to be infeasible with the chosen L. In that case, we increase L by 10%.

Table 1
Characteristics of the 340 MM-MT-dLARP instances.

Instance name	Average original lines	Average original vertices	Average total vertices
MM-MT-dLARP4_4_1	9.0	14.2	149.4
MM-MT-dLARP4_4_2	10.4	17.2	172.6
MM-MT-dLARP5_5_1	16.6	23.2	271.8
MM-MT-dLARP5_5_2	21.8	26.4	353.8
MM-MT-dLARP5_6_1	17.2	24.4	281.6
MM-MT-dLARP5_6_2	20.0	27.0	326.8
MM-MT-dLARP6_6_1	21.4	29.4	349.6
MM-MT-dLARP6_6_2	28.4	34.4	460.0
MM-MT-dLARP5_8_1	25.2	33.6	411.0
MM-MT-dLARP5_8_2	33.8	37.6	545.0
MM-MT-dLARP6_8_1	29.6	40.6	484.0
MM-MT-dLARP6_8_2	48.6	47.2	776.4
MM-MT-dLARP7_7_1	30.6	42.0	501.2
MM-MT-dLARP7_7_2	50.4	48.6	804.4
MM-MT-dLARP5_10_1	30.8	43.6	506.6
MM-MT-dLARP5_10_2	50.0	47.8	797.0
MM-MT-dLARP6_10_1	40.6	53.2	662.2
MM-MT-dLARP6_10_2	52.8	56.0	848.8
MM-MT-dLARP7_9_1	44.0	56.4	716.2
MM-MT-dLARP7_9_2	58.6	59.4	938.0
MM-MT-dLARP8_8_1	45.6	55.8	739.0
MM-MT-dLARP8_8_2	61.8	62.2	989.6
MM-MT-dLARP7_10_1	50.2	63.0	815.8
MM-MT-dLARP7_10_2	63.6	64.4	1019.2
MM-MT-dLARP8_9_1	52.4	64.8	849.8
MM-MT-dLARP8_9_2	66.8	66.2	1068.4
MM-MT-dLARP8_10_1	53.6	65.6	869.6
MM-MT-dLARP8_10_2	78.0	74.8	1243.6
MM-MT-dLARP9_9_1	53.6	68.4	873.6
MM-MT-dLARP9_9_2	74.8	70.8	1191.2
MM-MT-dLARP9_10_1	59.6	74.0	969.2
MM-MT-dLARP9_10_2	75.0	79.0	1202.2
MM-MT-dLARP10_10_1	66.8	82.0	1086.0
MM-MT-dLARP10_10_2	87.4	90.2	1400.4

5.2. Choosing the best separation strategy

To assess the contribution of each family of cuts and determine when each separation algorithm should be applied, we have run the branch-and-cut algorithm on a subset of 10 instances chosen randomly among the set of small instances, using different configurations of the cutting-plane procedure. We start with a *base* configuration that uses the separation algorithms for connectivity inequalities and for parity inequalities for sets S containing one single vertex, but only when the solution is integer, to guarantee that it is a feasible solution. The results obtained with this *base* configuration are shown in the first row of Table 2, providing 7 out of 10 optimal solutions on an average time of 1417.2 s, and an average gap (calculated as $100\frac{UB-LB}{UB}$, where LB and UB are the final lower and upper bound, respectively) of 2.61%. The four entries zero in this row means that the separation algorithms of the corresponding columns, labeled *Parity heu* (heuristic for parity inequalities), *Parity 1v* (heuristic for parity inequalities with sets S containing one single vertex), *p-conn* (heuristic for p-connectivity inequalities), and *Max-length* (heuristic for max-length inequalities), are not used in this *base* configuration.

Then we iteratively test the effect of adding only one of these four separation algorithms to the *base* configuration, executing it at all the iterations of the cutting-plane procedure when the solution is fractional and on every node of the search tree. This is shown in rows

Table 2
Cutting-plane configurations 1.

		Parity heu.	Parity 1v.	p-conn.	Max-length	# Opt	Time (s)	%Gap
Round 1	Base	0	0	0	0	7	1417.2	2.61
		1	0	0	0	7	1410.3	1.41
		0	1	0	0	7	1380.8	2.07
		0	0	1	0	7	1440.4	3.55
		0	**0**	**0**	**1**	**7**	**1288.3**	**0.57**
Round 2	Base	0	0	0	1	7	1288.3	0.57
		1	0	0	1	8	1414.8	0.63
		0	**1**	**0**	**1**	**9**	**1339.9**	**0.15**
		0	0	1	1	7	1380.7	2.17
Round 3	Base	**0**	**1**	**0**	**1**	**9**	**1339.9**	**0.15**
		1	1	0	1	8	1287.0	0.72
		0	1	1	1	7	1296.1	1.47

Table 3
Cutting-plane configurations 2.

		Parity heu.	Parity 1v.	p-conn.	Max-length	# Opt	Time (s)	%Gap
Round 4	Base	0	1	0	1	9	1339.9	0.15
		2	1	0	1	8	1259.8	0.74
		0	**2**	**0**	**1**	**9**	**1193.7**	**0.15**
		0	1	2	0	7	1325.5	0.78
		0	1	0	2	8	1374.2	1.05
Round 5	Base	0	2	0	1	9	1193.7	0.15
		2	2	0	1	8	1153.5	0.47
		0	**2**	**2**	**1**	**9**	**1160.2**	**0.15**
		0	2	0	2	7	1289.5	1.98
Round 6	Base	**0**	**2**	**2**	**1**	**9**	**1160.2**	**0.15**
		2	2	2	1	8	1160.1	0.64
		0	2	2	2	7	1445.1	0.67

2 to 5 of Table 2, where an entry 1 in these rows means that the separation algorithm corresponding to this column is used. Note that the best results are obtained with the version in row 5, corresponding to adding the *Max-length* separation algorithm. Hence, this algorithm is incorporated into the cutting-plane procedure of the branch and cut and this new configuration is referred to as the *base* version for the next round of runs. In subsequent rounds, we incorporate one more of the remaining separation algorithms and choose the best one, until no improvement in the overall results is observed. The results can be seen in Table 2, where each row corresponds to a configuration of the cutting-plane procedure. In Round 2 we obtain that the version with *algorithm 1v* improves the base version, and this algorithm is incorporated to define the *base* configuration for Round 3. In Round 3 we observe that the incorporation of neither of the two remaining separation algorithms produces improvements in the overall results, and hence the process ends: The best configuration for the cutting-plane procedure is the one using the *Max-length* and *Parity 1v* separation algorithms, providing 9 out of 10 optimal solutions on an average time of 1339.9 s.

Additionally, we have studied the effect of using each separation procedure only at the root node of the search tree. Starting with the best version of the previous phase as the base configuration, we check the effect of using each one of the separation algorithms only at the root node of the branch and cut. This is expressed in the rows of Table 3 with a value of 2 in the corresponding entry. Note that this means that if a given separation algorithm is not in the base configuration then it is incorporated into it and used only at the root node, while if a given separation algorithm is already in the base configuration (invoked at all the nodes), then it is restricted to be used only at the root node. The results in Table 3 show that the best version of the cutting-plane procedure is the one that separates parity inequalities for sets S with one vertex and p-connectivity inequalities only at the root node, and max-length inequalities at every node. The number of optimal solutions found is still 9, but the average time has decreased to 1160.2 s. Therefore, the remaining experiments have been carried out using this cutting-plane configuration.

5.3. Results

We have run the B&C on the MM-MT-LARP instances with each line represented by a single segment, without any intermediate point, and with a service cost equal to the cost of the original line. We call these instances MM-MT-LARP(0). Furthermore, we are showing only the branch-and-cut results obtained on the MM-MT-LARP(0) instances corresponding to the "small" graphs, since the algorithm was not able to solve any instance with "medium" or "large" graphs. The heuristic has been tested on all the instances, starting with MM-MT-LARP(0) and then iteratively adding some intermediate points that are "promising" to improve the current solution, as explained in Section 4.3.

The results obtained with the B&C algorithm with a time limit of one hour for the "small" MM-MT-LARP(0) instances are reported in Table 4, grouped by grid size. Each row corresponds to the 20 instances associated with the grid size presented in the first column. The column "#opt" reports the number of instances out of 20 in which the B&C has reached the optimal solution. Column "%Gap" shows the average percentage gaps between the value of the optimal solution (or the best upper bound found) and the final lower bound. The column "Time" reports the average total time, in seconds, used by the B&C on all the 20 instances, while the column "Time opt" shows the average total time in the instances in which the optimal solution was reached. The number of instances out of 20 for which an upper bound (feasible solution) has been found by the B&C is shown in the last column "# feasible". As it can be seen in Table 4, the B&C obtains good results in the smallest instances, those from grids with up to 30 nodes, in which the optimal solution is obtained in more than half of the instances, a feasible solution is obtained for all of them, and the average gap in these 60 MM-MT-LARP(0) instances is low, 2.43%. In the remaining "small" instances, the performance of the B&C algorithm is quite poor: in the 20 instances that come from the 5×8 grid (last row) after one hour of computation only 1 optimal solution has been reached, and the average gap is greater than 20%. These results seem to imply that the MM-MT-LARP problem is too hard to be solved with a branch-and-cut algorithm.

Table 4
Branch-and-cut results on the "small" MM-MT-LARP(0) instances.

Size	#opt	%Gap	Time (s)	Time opt (s)	#feasible
4 × 4	20	0	8.3	8.3	20
5 × 5	13	5.09	1662.4	619.1	20
5 × 6	15	2.21	1702.1	1069.5	20
6 × 6	8	15.38	2843.3	1708.1	19
5 × 8	1	22.29	3467.2	941.5	19
Total	57	9.00	1936.7	681.8	98

The previous execution with the exact B&C algorithm also serves to evaluate the performance of our heuristic algorithm. Tables 5 and 6 report some results obtained with the set of "small" MM-MT-LARP(0) instances to compare both algorithms. Again the instances are grouped by grid size and each row corresponds to 20 instances. The results in Table 5 correspond to the set of all the 100 "small" instances, 20 instances associated with each grid size. Column "LB" contains the average lower bound at the end of the B&C while column "Z_heur" shows the average cost of the best solution found by the heuristic. The number of instances (of those in which the B&C found the optimal solution) for which the solution provided by the heuristic is also optimal is shown in the column "#opt". Column "%Gap LB" reports the average percentage gap between the cost of the heuristic solution and the final lower bound at the end of the B&C for all the instances, while column "%Gap LB opt" reports the same data but only for the instances for which the optimum is known. Finally, the two last columns show the total times, in seconds, of the B&C and the heuristic algorithm, respectively. The performance of the heuristic algorithm in these "small" instances is very good, obtaining the optimal solution in 39 of the 57 instances for which the optimal solution is known. In addition, the average gap between the cost of the heuristic solution and the optimal cost (or LB) in these 57 instances is very small, 0.66%. Note that the same average gap in all 100 instances is much larger, 10.81%, which means that the LB of the B&C in the instances that it has not been able to solve is quite poor. We think that if these lower bounds were closer to the optimal value, the "%Gap LB" would be quite good in these 100 "small" instances. In summary, the heuristic algorithm is capable of finding in 54 s on average a solution quite close to the solution provided by the B&C algorithm after 1123 s.

To compare the cost of the heuristic solutions with the B&C upper bounds, the data in Table 6 refer only to the 98 out of 100 "small" instances for which the B&C has obtained a feasible solution, and therefore an upper bound (column "# feasible"). Column "UB" contains the average upper bound at the end of the B&C algorithm and column "Z_heur" shows the average cost of the best solutions found by the heuristic. Column "%Gap UB" reports the average percentage gap between the values in the two previous columns and the last two columns show the average total times, in seconds. Note that in the 60 smallest instances "%Gap UB" is positive (although small), because many of the upper bounds are optimal. In the 38 largest instances (last two rows), "%Gap UB" is negative on average (although again small), which means that the heuristic algorithm has been able to find better solutions in 53 s than the B&C in 1106 s on average. In particular, the heuristic has found better solutions in 9 out of 19 6 × 6 instances and in 14 out of 19 5 × 8 instances.

The previous results of the heuristic algorithm refer only to the MM-MT-LARP(0) instances, that is, to the instances without considering any intermediate point of any original line. This corresponds to the first two parts of the heuristic algorithm: the construction and the local search phases (see Section 4). Once the solutions for the MM-MT-LARP(0) instances are obtained, the algorithm applies the splitting phase, an iterative process of insertion (and possible subsequent removing) of some intermediate points of the original lines to look for better solutions (see Section 4.3). Table 7 shows the results obtained after this splitting phase compared with the results on the MM-MT-LARP(0) instances, grouped as small, medium and large. Again, each row in Table 7 corresponds to the 20 instances associated with the grid size. Column "Z_0" shows the average cost of the best solution found by the heuristic on the MM-MT-LARP(0) instances while column "Z_splits" shows the average cost after the splitting phase. Column "Imp" reports the average percentage improvement in the cost of the solutions obtained by the algorithm with the splitting phase with respect to the algorithm without the splitting phase. The last two columns show the average total times, in seconds, of the heuristic algorithm without and with the splitting phase, respectively. Note that the splitting phase reduces the cost of heuristic solutions by an average of 1.2% in a negligible time, less than half a second on average. We want to point out that the improvement of 1.2% on average obtained after the splitting phase is due to a fine readjustment of the entry and exit points of the lines which would not improve further even if more time was given.

6. Conclusions

This paper addresses the Min-Max Multi-Trip drone Location Arc Routing Problem (MM-MT-dLARP), which integrates the use of trucks and drones for arc routing tasks. This continuous optimization problem is transformed into a discrete one if we allow the drone to enter and exit a line only through a finite set of intermediate points. For the discrete problem, we have introduced an integer formulation, as well as a set of valid inequalities that have been shown to be facet-inducing of a relaxed polyhedron. A branch-and-cut algorithm has been developed and tested on small instances without intermediate points. Additionally, we have designed a matheuristic algorithm, incorporating a construction phase, four local search procedures within a Variable Neighborhood Descent (VND) framework, and rules for selecting intermediate points to enhance solutions. The computational experiments, conducted on a set of randomly generated instances with up to 6 launching points and 88 lines, show that this is a difficult problem for which only small instances can be expected to be solved exactly and that the proposed matheuristic is capable of providing high-quality solutions of the continuous problem that take advantage of the ability of drones to travel between any pair of points on the network, in reasonable computing times.

CRediT authorship contribution statement

Teresa Corberán: Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Isaac Plana:** Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **José María Sanchis:** Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization.

Acknowledgments

This work was supported by grant PID2021-122344NB-I00 funded by MCIN/AEI, Spain/10.13039/501100011033 and "ERDF A way of making Europe".

Data availability

Data will be made available on request.

Table 5
Comparison between the algorithms in all the "small" instances.

Size	LB	Z_heur	#opt	%Gap LB	%Gap LB opt.	Time B&C (s)	Time Heur (s)
4 × 4	1404.03	1404.30	19 of 20	0.03	0.03	584.4	20.0
5 × 5	2334.89	2497.38	7 of 13	8.47	1.83	1020.6	52.9
5 × 6	2554.60	2638.73	8 of 15	4.19	1.07	1036.1	49.3
6 × 6	2934.91	3372.89	4 of 8	17.08	0.37	1456.4	68.4
5 × 8	3134.00	3861.71	1 of 1	24.30	0.00	1516.6	80.8
Total	2472.48	2755.00	39 of 57	10.81	0.66	1122.8	54.3

Table 6
Comparison between the algorithms in "small" instances with known UB.

Size	# feasible	UB	Z_heur	%Gap UB	Time B&C (s)	Time Heur (s)
4 × 4	20	1404.03	1404.30	0.03	584.4	20.0
5 × 5	20	2451.58	2497.38	1.90	1020.6	52.9
5 × 6	20	2604.11	2638.73	1.63	1036.1	49.3
6 × 6	19	3436.97	3297.13	−3.50	1460.9	66.8
5 × 8	19	3953.80	3623.58	−4.75	1461.7	78.4
Total	98	2751.22	2676.55	−0.87	1105.6	53.1

Table 7
Comparison of the results obtained with the heuristic without and with splitting phase.

Size	Z_0	Z_splits	Imp	Time_0 (s)	Time_splits (s)
4 × 4	1404.30	1391.11	0.9	20.03	20.15
5 × 5	2497.38	2438.61	2.2	52.87	53.07
5 × 6	2638.73	2606.43	1.4	49.32	49.51
6 × 6	3372.89	3339.48	0.9	68.36	68.63
5 × 8	3861.71	3802.61	1.4	80.76	81.04
5 × 10	5316.40	5270.04	1.1	113.68	114.07
6 × 8	4732.53	4665.33	1.3	110.95	111.30
6 × 10	5979.72	5886.23	1.8	135.54	136.01
7 × 7	5206.11	5152.52	1.1	120.46	120.82
7 × 9	6426.66	6378.90	0.8	153.78	154.29
8 × 8	6643.02	6569.25	1.2	165.19	165.73
7 × 10	7198.36	7149.53	0.7	146.79	147.42
8 × 9	7482.77	7423.68	0.9	175.15	175.81
8 × 10	8995.63	8866.12	1.5	246.75	247.59
9 × 9	8406.65	8321.23	1.2	301.02	301.78
9 × 10	8610.68	8531.67	0.9	256.02	256.82
10 × 10	9879.02	9767.08	1.2	353.89	354.94
Total	5803.09	5738.81	1.2	150.03	150.53

References

Albareda-Sambola, M., Rodríguez-Pereira, J., 2019. Location-routing and location-arc routing. In: Laporte, G., Nickel, S., Saldanha da Gama, F. (Eds.), Location Science. Springer, Cham.

Amberg, A., Domschke, W., Voß, S., 2000. Multiple center capacitated arc routing problems: A tabu search algorithm using capacitated trees. European J. Oper. Res. 124, 360–376.

Amorosi, L., Puerto, J., Valverde, C., 2021. Coordinating drones with mothership vehicles: The mothership and drone routing problem with graphs. Comput. Oper. Res. 136, 105445.

Amorosi, L., Puerto, J., Valverde, C., 2023. A multiple-drone arc routing and mothership coordination problem. Comput. Oper. Res. 159, 106322.

Amorosi, L., Puerto, J., Valverde, C., 2024. An extended model of coordination of an all-terrain vehicle and a multivisit drone. Int. Trans. Oper. Res. 31 (2), 780–806.

Campbell, J.F., Corberán, Á., Plana, I., Sanchis, J.M., 2018. Drone arc routing problems. Networks 72, 543–559.

Campbell, J.F., Corberán, Á., Plana, I., Sanchis, J.M., Segura, P., 2021. Solving the length-constrained K-drones rural postman problem. European J. Oper. Res. 292, 60–72.

Campbell, J.F., Corberán, Á., Plana, I., Sanchis, J.M., Segura, P., 2022. Polyhedral analysis and a new algorithm for the length-constrained K-drones rural postman problem. Comput. Optim. Appl. 83, 67–109.

Campbell, J.F., Corberán, Á., Plana, I., Sanchis, J.M., Segura, P., 2023. The multi-purpose K-drones general routing problem. Networks 82 (4), 437–458.

Chow, J.Y.J., 2016. Dynamic UAV-based traffic monitoring under uncertainty as a stochastic arc-inventory routing policy. Int. J. Transp. Sci. Technol. 5, 167–185.

Chung, S.H., Sah, B., Lee, J., 2020. Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions. Comput. Oper. Res. 123, 105004.

Corberán, Á., Plana, I., Rodríguez-Chía, A.M., Sanchis, J.M., 2013. A branch-and-cut algorithm for the maximum benefit Chinese postman problem. Math. Program. 141, 21–48.

Corberán, Á., Plana, I., Sanchis, J.M., 2007. A branch & cut algorithm for the windy general routing problem and special cases. Networks 49, 245–257.

Corberán, T., Plana, I., Sanchis, J.M., Segura, P., 2024. The multi-depot drone general routing problem with duration and capacity constraints. Int. Trans. Oper. Res..

Doulabi, S.H.H., Seifi, A., 2013. Lower and upper bounds for location-arc routing problems with vehicle capacity constraints. European J. Oper. Res. 224 (1), 189–208.

Fernández, E., Laporte, G., Rodríguez-Pereira, J., 2018. A branch-and-cut algorithm for the multidepot rural postman problem. Transp. Sci. 52, 229–496.

Fernández, E., Laporte, G., Rodríguez-Pereira, J., 2019. Exact solution of several families of location-arc routing problems. Transp. Sci. 53 (5), 1313–1333.

Fernández, E., Rodríguez-Pereira, J., 2017. Multi-depot rural postman problems. TOP 25, 340–372.

Ghiani, G., 1998. Arc Routing, Allocation-Arc Routing and Location-Arc Routing Problems (Ph.D. thesis). Università degli Studi di Napoli Federico II, Italy.

Ghiani, G., Laporte, G., 2000. A branch-and-cut algorithm for the undirected rural postman problem. Math. Program. 87, 467–481.

Hansen, P., Mladenović, N., 2001. Variable neighborhood search: Principles and applications. European J. Oper. Res. 130, 449–467.

Hierholzer, C., 1873. Über die möglichkeit, einen linienzug ohne wiederholung und ohne unterbrechung zu umfahren. Math. Ann. 6, 30–32.

Hu, H., Liu, T., Zhao, N., Zhou, Y., Min, D., 2013. A hybrid genetic algorithm with perturbation for the multi-depot capacitated arc routing problem. J. Appl. Sci. 13, 32–39.

Krushinsky, D., Van Woensel, T., 2015. An approach to the asymmetric multi-depot capacitated arc routing problem. European J. Oper. Res. 244, 100–109.

Li, M., Zhen, L., Wang, S., Lv, W., Qu, X., 2018. Unmanned aerial vehicle scheduling problem for traffic monitoring. Comput. Ind. Eng. 122, 15–23.

Lopes, R.B., Plastria, F., Ferreira, C., Sousa Santos, B., 2014. Location-arc routing problem: Heuristic approaches and test instances. Comput. Oper. Res. 43, 309–317.

Luo, H., Zhang, P., Wang, J., Wang, G., Meng, F., 2019. Traffic patrolling routing problem with drones in an urban road system. Sensors 19, 5164.

Macrina, G., Di Puglia Pugliese, L., Guerriero, F., Laporte, G., 2020. Drone-aided routing: A literature review. Transp. Res. C 120, 102762.

Muyldermans, L., Cattrysse, D., Van Oudheusden, D., 2003. District design for arc-routing applications. J. Oper. Res. Soc. 54, 1209–1221.

Rojas Viloria, D., Solano-Charris, E.L., Muñoz-Villamizar, A., Montoya-Torres, J.R., 2021. Unmanned aerial vehicles/drones in vehicle routing problems: a literature review. Int. Trans. Oper. Res. 28, 1626–1657.

Wøhlk, S., 2004. Contributions to Arc Routing (Ph.D. dissertation). University of Southern Denmark, Odense, Denmark, (2004).

Xie, A., Miyagawa, K., Wu, W., Yagiura, M., 2024. A heuristic algorithm for the drone rural postman problem. J. Ind. Manag. Optim. 20 (5), 1951–1966.

Xing, L., Rohlfshagen, P., Chen, Y., Yao, X., 2010. An evolutionary approach to the multidepot capacitated arc routing problem. IEEE Trans. Evol. Comput. 14, 356–374.

Xu, B., Zhao, K., Luo, Q., Wu, G., Pedrycz, W., 2023. A GV-drone arc routing approach for urban traffic patrol by coordinating a ground vehicle and multiple drones. Swarm Evol. Comput. 77.