Coordinating drones with mothership vehicles: The multitarget drone routing problem

Lavinia Amorosi[a], Justo Puerto[b], Carlos Valverde[b,*]

[a] *Department of Statistical Sciences, Sapienza University of Rome, Italy*
[b] *Department of Statistical Sciences and Operational Research, University of Seville, Spain*

Abstract

This paper considers the optimization problems that arise to coordinate a tandem between a mothership vehicle and a drone. The drone is supported by the mothership to perform its operations. They consist on visiting one or more points or traversing (partially) polygonals with given lengths to deliver some service or to perform some surveillance/inspection activities. The mothership works as a mobile recharging station for the drone and it moves freely on a continuous space so that the launching and retrieving points of the drone can be anywhere and must be determined. The goal is to minimize the overall weighted distance traveled by the mothership and the drone, while satisfying some requirements in terms of percentages of visits to these polygonals. In all cases, we develop exact formulations resorting to mixed integer second order cone programs that are compared on a testbed of instances to assess their performance. We also develop heuristic algorithms that provide reasonable solutions in short computing time. Extensive computational experiments show the usefulness of our methods in different scenarios.

Keywords: Routing; Networks, Logistics; Drones; Mixed integer conic programming

1. Introduction

More and more frequently we hear about new technologies, as robots, self-driving vehicles and drones, and their use to replace humans in some activities, especially the most repetitive or dangerous ones, [7], or to create infrastructures and service networks alternative to traditional ones (see for example [6] and [1]). In this context, many management and coordination problems arise, which can also be addressed and solved by means of optimization models. The variety of problems and applications in this area has already led to a wide scientific production and to the development of extensions of existing combinatorial optimization models (see for example [11]) or to the formalization of new classes of problems, also by resorting to non-linear programming, as in [2] and [3]. In [5] and [16, 10] the authors study exact methods for the Traveling Salesman Problem with one or multiple drones. An interesting literature review on drone routing problems is presented in [12]. In particular, the use of drone technology in various sectors is a well-studied topic that continues to receive growing interest (see, e.g. [10, 9, 18, 17, 13]). Indeed, in a context where the urgency of sustainable solutions with low environmental impact is growing, this technology represents a valid alternative to the use of traditional means of transport. Furthermore, drones can also reach areas that are difficult to be reached by people and in a faster and safer way. Examples of this can be found both in the parcel delivery and in many inspection and monitoring activities also in post-disaster contexts (see [14] and [8] for extensive surveys). In this work, we refer to these latter activities, resorting to the use of one drone supported by a mothership vehicle working as a mobile recharging station to manage its limited endurance. Such a system requires coordination and synchronization of the vehicles involved. In [2] the authors studied the Mothership Drone Routing Problem with Graphs (AMDRPG) where one drone is supported by a mothership and they formulated the coordination problem in order to visit a set of target graphs by minimizing the total distance traveled by both vehicles. In [3] an extension of this problem with multiple drones, called the Mothership and Multiple Drones Routing Problem with Graphs (MMDRPG) has been investigated. In this paper we face the case in which one drone, supported by a mothership vehicle, must visit a set of targets but, differently with respect to the previously mentioned works, in each mission, the drone can visit more

*Corresponding author
Email address: cvalverde@us.es (Carlos Valverde)

than one target or a portion of it. We assume that the mothership can move freely in the continuous space and we consider two possible shapes of the target to be visited: (i) the targets are points or (ii) the targets are polygonal chains. This is an important contribution of this work as compared with recent papers in the literature where launching and retrieving points are forced to be nodes of a given network and targets are always points [5, 10, 9]. We mathematically formulate the problem as a Mixed Integer Non-Linear Programming (MINLP) model for which we also derive valid inequalities to reinforce it. Moreover, in order to deal with larger size instances, we design alternative matheuristic procedures. Extensive computational experiments are performed reporting the usefulness of our exact and heuristic methods to solve the problem.

The rest of this paper is structured as follows: Section 2 describes the problem details and the proposed mathematical programming formulation. Section 3 discusses a strengthening of the formulation also by resorting to valid inequalities. Section 4 presents (alternative) matheuristic procedures in order to deal with larger size instances of the problem. Section 5 reports the experimental results obtained by testing the model on different instances of points and polygonal chains and the comparison with the ones provided by the matheuristic algorithms in order to evaluate their usefulness. Finally Section 6 concludes the paper.

2. Problem description and valid formulation

2.1. Problem description

In the Multitarget Mothership and Drone Routing Problem (Multitarget-MDRP), there exists one drone that has to be coordinated with one mothership (that plays the role of base vehicle) to complete a number of operations consisting on visiting some targets. All these targets must be visited by the drone before finishing the complete tour. Both vehicles start at a known location, denoted *orig*, then they depart to perform all the operations and, once all the targets are visited, they must return together to a final location, called *dest*. We refer to an operation as the sequence of launching a drone from the mothership, visiting one or more targets and returning back to the mothership. The shape of the targets that are considered in this paper are points and polygonal chains. A similar analysis allows to extend the models from points to convex sets as well as from polygonal chains to general graphs (see [2] and [3]). Nevertheless, for the sake of simplicity and to improve the readability of this paper, we restrict ourselves to the above mentioned cases that already capture the essence of the problem. The operation of visiting a point consists on getting to it and coming back, whereas for polygonal chains the drone has to traverse a given percentage of their lengths for considering a successful visit. We also assume that the mothership and the drone travel at constant velocities v_M and v_D, respectively, although it can be extended to more general cases where these velocities can be modelled as a time-dependent function. Moreover, the drone has a limited time N (endurance) to complete each operation and return back to the base vehicle to recharge batteries. We assume that the drone and the base vehicle movements follow straight lines on a continuous space. This implies that Euclidean distance is used to measure displacements.

The set of targets $\mathcal{T}$ to be visited permit to model real situations like goods delivery or roads or wire inspection. Figure 1 shows an example of the problem framework, where the black squares represent the origin and the destination of the mothership tour.

Moreover, at each operation the drone must be launched from the base vehicle (the launching points have to be determined) and it must be retrieved when its battery needs to be recharged (the retrieving points also have to be determined). Nonetheless, this does not imply that the tandem must reach at a rendezvous location at the same time: the base vehicle may wait for the drone at the rendezvous location. Furthermore, it is supposed that the cost produced by the drone's trip is lower as compared to the one incurred by the base vehicle. Therefore, the goal is to minimize the weighted total distance traveled by the mothership and the drone. Some works assume that this cost is negligible in comparison with the mothership ([2]). The reader may note that the extension not including the distances traveled by the drone in the objective function is straightforward by setting the corresponding weight to zero.

The goal of the Multitarget-MDRP is to find the launching and rendezvous points of the drone satisfying the visit requirements for the targets in $\mathcal{T}$ and minimizing the weighted length of the paths traveled by the mothership and the drone.

2.2. Mixed Integer Non Linear Programming Formulations

The purpose of this section is to present a Mixed Integer Non-Linear Programming formulation for the Multitarget-MDRP that can be applied to solve instances of this problem.

Problem Parameters		
P. Name	**Range**	**Description**
$orig$	$\mathbb{R}^2$	Coordinates of the point defining the origin of the mothership path (or tour).
$dest$	$\mathbb{R}^2$	Coordinates of the point defining the destination of the mothership path (or tour).
$\mathcal{A}$	$\{1,\ldots,\lvert\mathcal{A}\rvert\}$	Set of points.
$\mathcal{P}$	$\{1,\ldots,\lvert\mathcal{P}\rvert\}$	Set of polygonal chains.
$\mathcal{T}=\mathcal{A}\cup\mathcal{P}$	$\{1,\ldots,\lvert\mathcal{T}\rvert\}$	Set of targets.
$a=(a(x_1),a(x_2))$	$\mathbb{R}^2$	Coordinates of the point $a\in\mathcal{A}$.
$p=(V_p,S_p)$	$\mathbb{R}^2$	Set of breakpoints and segments of each polygonal chain $p\in\mathcal{P}$.
$V=\mathcal{A}\cup\left(\bigcup_{p\in\mathcal{P}}V_p\right)$	$\mathbb{R}^2$	Set of target points and set of breakpoints of polygonal targets.
$\mathcal{L}(p)$	$\mathbb{R}_+$	Total length of the polygonal chain $p\in\mathcal{P}$.
$\mathcal{L}(s_p)$	$\mathbb{R}_+$	Length of the segment s_p of the polygonal chain $p\in\mathcal{P}$.
A^{v_p}	$\mathbb{R}^2$	Coordinates of the point v_p of the polygonal $p\in\mathcal{P}$.
α^p	$[0,1]$	Percentage of the polygonal $p\in\mathcal{P}$ that must be visited.
v_D	$\mathbb{R}_+$	Drone speed.
v_M	$\mathbb{R}_+$	Mothership speed.
N	$\mathbb{R}_+$	Drone endurance.
M	$\mathbb{R}_+$	Big-M constant.
m	$\mathbb{R}_+$	Small-M constant.

Table 1: Nomenclature for the Multitarget-MDRP

In Subsection 2.1, we mention that the mothership can move without any restriction in a continuous space that for simplicity is supposed to be $\mathbb{R}^2$. Although it is possible to measure distances with any l_p-norm, $1\leq p\leq\infty$ (see [4]), for the sake of presentation, in this work the distances are measured by the Euclidean norm ($p=2$).

First of all, we introduce the parameters or initial data that formally describe the problem and that are summarized in Table 1.

To represent the movement of the drone within a polygonal $p\in\mathcal{P}$, we proceed to introduce some notation related to p. Let $p=(V_p,S_p)$ be a polygonal chain in $\mathcal{P}$ whose total length is denoted by $\mathcal{L}(p)$. Here, V_p denotes the set of vertices and S_p denotes the set of segments connecting pairs of consecutive vertices whose cardinality is $|S_p|$. Let $s_p=\overline{v_p(v+1)_p}$ be the segment s of the polygonal $p\in\mathcal{P}$ and let $\mathcal{L}(s_p)$ be its length. Since we need to refer to interior points of the segment s_p, these continuum of points is parameterized by the two endpoints $A^{v_p}=(A^{v_p}(x_1),A^{v_p}(x_2))$ and $A^{(v+1)_p}=(A^{(v+1)_p}(x_1),A^{(v+1)_p}(x_2))$ of the segment: $x\in[A^{v_p},A^{(v+1)_p}]$ if and only if $\exists\ \gamma\in[0,1]$ such that $x=\gamma A^{v_p}+(1-\gamma)A^{(v+1)_p}$. Hence, the length of the segment s_p is $\mathcal{L}(s_p)=\|A^{v_p}-A^{(v+1)_p}\|$.

Next, we need to determine the placement of the entry point, R^p, on the polygonal chain p introducing the following inequalities for each $p\in\mathcal{P}$:

$$R^p\in p\Longleftrightarrow\begin{cases}\gamma_R^{1_p} &\leq \mu_R^{1_p},\\ \gamma_R^{s_p} &\leq \mu_R^{s_p-1}+\mu_R^{s_p}, \quad s_p\in S_p\setminus\{1\},\\ \gamma_R^{|V_p|_p} &\leq \mu_R^{|S_p|_p},\\ \displaystyle\sum_{s_p\in S_p}\mu_R^{s_p} &= 1,\\ \displaystyle\sum_{v_p\in V_p}\gamma_R^{v_p} &= 1,\\ R^p &= \displaystyle\sum_{v_p\in V_p}\gamma_R^{v_p}A^{v_p}.\end{cases}\tag{$\mathcal{P}$-C}$$

The first, second and third inequalities link $\mu_R^{s_p}$ and $\gamma_R^{s_p}$ variables: they state that the variable $\gamma_R^{s_p}$ that gives the representation of a point R^p on the line segment s_p is active (non-null) only if this line segment is chosen to enter in polygonal p, i.e., $\mu_R^{s_p}=1$. The fourth equation sets that only one line segment is

Table 2: Summary of binary variables used in the mathematical programming model

Binary Decision Variables			
Name	**Set**	**Domain**	**Description**
$\mu_R^{s_p}$	$s_p \in S_p : p \in \mathcal{P}$	Binary	1, if the entry point on the polygonal chain is located in the line segment s_p, 0, otherwise.
$\mu_L^{s_p}$	$s_p \in S_p : p \in \mathcal{P}$	Binary	1, if the exit point on the polygonal chain is located in the line segment s_p, 0, otherwise.
$z_{\mathcal{P}}^{s_p s'_p}$	$s_p, s'_p \in S_p : p \in \mathcal{P}$	Binary	1, if the entry and exit points are located in the line segments s_p and s'_p, respectively, 0, otherwise.
u^{to}	$t \in \mathcal{T}, o \in \mathcal{O}$	Binary	1, if the drone starts the operation o in the target t, 0, otherwise.
v^{to}	$t \in \mathcal{T}, o \in \mathcal{O}$	Binary	1, if the drone finishes the operation o in the target t, 0, otherwise.
δ^{to}	$t \in \mathcal{T}, o \in \mathcal{O}$	Binary	1, if the drone visits the target t in the operation o, 0, otherwise.
$y^{tt'o}$	$t, t' \in \mathcal{T}, o \in \mathcal{O}$	Binary	1, if the drone goes from the target t to the target t' in the operation o, 0, otherwise.

Table 3: Summary of continuous variables used in the mathematical programming model

Continuous Decision Variables			
Name	**Set**	**Domain**	**Description**
R^t	$t \in \mathcal{T}$	$\mathbb{R}^2$	Coordinates representing the entry point on the target t. If the target is a point, it coincides with L^t.
$\gamma_R^{v_p}$	$v_p \in V_p : p \in \mathcal{P}$	$[0, 1]$	Parameterization of the entry point R^p on the segment $s_p = \overline{v_p(v+1)_p}$ of the polygonal chain.
L^t	$t \in \mathcal{T}$	$\mathbb{R}^2$	Coordinates representing the exit point on each target. If the target is a point, it coincides with R^t.
$\gamma_L^{v_p}$	$v_p \in V_p : p \in \mathcal{P}$	$[0, 1]$	Parameterization of the exit point L^p on the segment $s_p = \overline{v_p(v+1)_p}$ of the polygonal chain.
x_L^o	$o \in \mathcal{O}$	$\mathbb{R}^2$	Coordinates representing the point where the mothership launches the drone at operation o.
x_R^o	$o \in \mathcal{O}$	$\mathbb{R}^2$	Coordinates representing the point where the mothership retrieves the drone at operation o.
d_L^{to}	$t \in \mathcal{T}, o \in \mathcal{O}$	$\mathbb{R}_+$	Distance travelled by the drone from the launching point x_L^o on the mothership to the first target point R^t associated to the operation o.
$d_{\text{out}}^{tt'}$	$t, t' \in \mathcal{T}$	$\mathbb{R}_+$	Distance travelled by the drone from the launching point L^t on one target to the rendezvous point $R^{t'}$ on another one.
d_R^{to}	$t \in \mathcal{T}, o \in \mathcal{O}$	$\mathbb{R}_+$	Distance travelled by the drone from the launching point of the last visited target L^t to the rendezvous point x_R^o associated to the operation o.
d_{LR}^o	$o \in \mathcal{O}$	$\mathbb{R}_+$	Distance travelled by the drone from the launching point x_L^o to the rendezvous point x_R^o at operation o.
d_{RL}^o	$o \in \mathcal{O} \setminus \{\lvert\mathcal{O}\rvert\}$	$\mathbb{R}_+$	Distance travelled by the mothership and the drone from the rendezvous point x_R^o at operation o to the launching point x_L^{o+1} at the operation $o+1$.
$d_{\mathcal{P}}^{s_p s'_p}$	$s_p, s'_p \in S_p : p \in \mathcal{P}$	$\mathbb{R}_+$	Distance travelled by the drone from the exit point on the segment s_p on one polygonal to the entry point on the segment s'_p of the same polygonal.
d_{in}^t	$t \in \mathcal{T}$	$\mathbb{R}_+$	Distance travelled by the drone from the rendezvous point R^t to the launching point L^t on the same target. If the target is a point, this distance is 0.

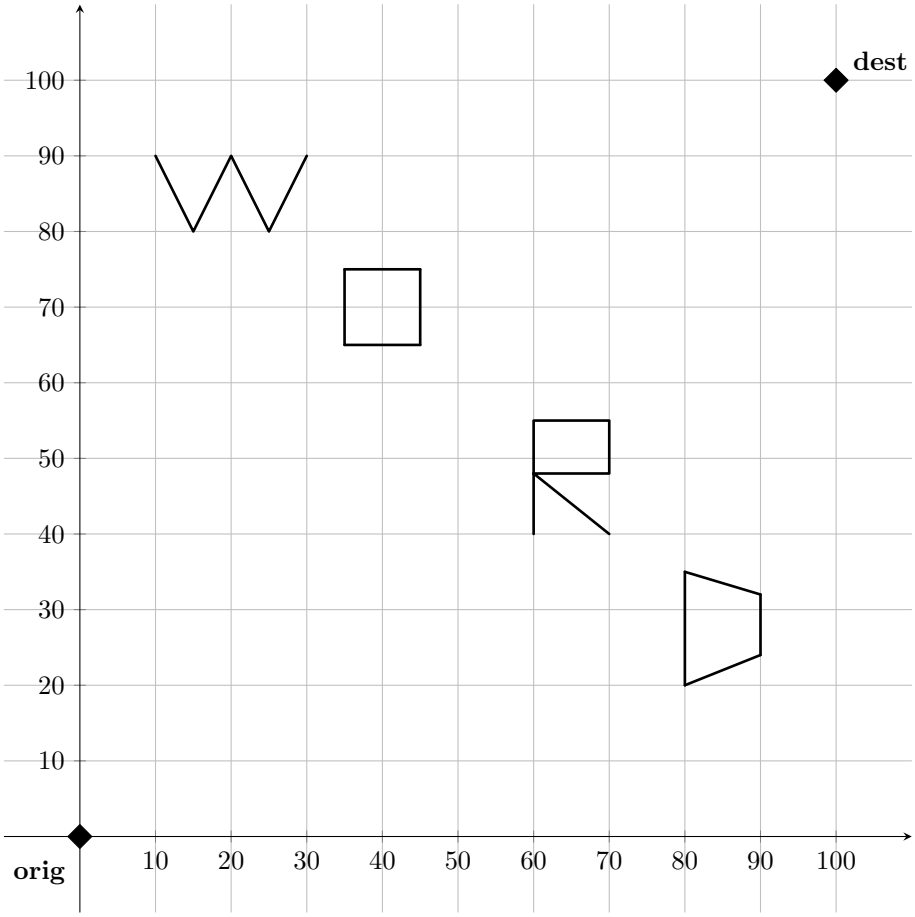

Figure 1: Example of problem instance with polygonal targets.

chosen for entering each polygonal chain. Finally, the fifth and sixth equations set the representation of R^p as a convex combination of the extreme points of the selected line segment. In the same way, we can model the location of the exit point, L^p, by using the variables L^p, $\mu_L^{s_p}$ and $\gamma_L^{s_p}$ explained in Table 2 and Table 3, respectively.

In our approach to model the Multitarget-MDRP we use operations identified with the order in which the different elements in the problem are visited. Let us denote by $\mathcal{O}$ the set of operations that the mothership and the drone have to carry out. An operation $o \in \mathcal{O}$ is referred to as the one in which the mothership launches a drone from a taking-off location, denoted by x_L^o and later it takes it back on a rendezvous location x_R^o. Each operation consists on the drone visit to one or more targets in $\mathcal{T}$ with the required constraints. At this point, it is relevant to note that the pair of locations x_L^o and x_R^o must be selected in the plane where the mothership is presumed to move. Observe that $|\mathcal{O}| \leq |\mathcal{T}|$ because of the assumption that at least one target is visited for each operation.

Figure 1 shows an example of problem instance with 4 polygonal targets. The point *orig* from where the mothership together with the drone must start their tour, is located in the origin, while the destination point *dest* is the point (100,100).

To include the definition of the paths followed by the drone in our mathematical programming formulation we need to make decisions to choose:

- The optimal assignment of the targets to each operation o.
- The optimal order to visit the targets for its corresponding operation o.

We can model the route followed by the drone using the binary variables u^{to}, $y^{tt'o}$, δ^{to} and v^{to} defined in Table 2.

$$\sum_{t\in\mathcal{T}} u^{to} \leq 1, \qquad \forall o \in \mathcal{O}, \tag{1}$$

$$\sum_{t\in\mathcal{T}} v^{to} \leq 1, \qquad \forall o \in \mathcal{O}, \tag{2}$$

$$\sum_{o\in\mathcal{O}} \delta^{to} = 1, \qquad \forall t \in \mathcal{T}, \tag{3}$$

$$\delta^{to} - u^{to} = \sum_{t'\neq t} y^{t'to}, \qquad \forall t \in \mathcal{T},\ \forall o \in \mathcal{O}, \tag{4}$$

$$\delta^{to} - v^{to} = \sum_{t'\neq t} y^{tt'o}, \qquad \forall t \in \mathcal{T},\ \forall o \in \mathcal{O}, \tag{5}$$

$$\sum_{t,t'\in S} y^{tt'o} \leq |S| - 1, \qquad \forall S \subsetneq \mathcal{T},\ \forall o \in \mathcal{O}. \tag{6}$$

Constraints (1) and (2) state that for each operation the drone only can enter and exit, respectively, by one target. Constraints (3) ensure that every target will be visited in some operation. Constraints (4) assure that if target t is visited by the drone for the operation o, one of two alternative conditions must take place: either t is the first target for the operation o or target t is visited by the drone after visiting another target t' for the operation o. Similarly, constraints (5) state that if the target t for the operation o is visited by the drone, either t is the last target of the operation, or the drone must move to another target t' of the operation o after visiting target t. Finally, inequalities (6) are the subtour elimination constraints applied to each operation. Note that, the complete family of SEC constraints can not be included to the model when we implement this formulation, because there is a exponential number of them and it can induce a memory problem on-the-shelf solvers. This problem can be solved by performing a row generation procedure that includes the constraints whenever they are needed by a separation oracle. To detect these SEC inequalities, as usual, we look for disconnected components in the current solution. Among them, we choose the shortest subtour found in the solution to be added as a lazy constraint to the model.

To take into account the different distances among the decision variables of the model, we need to set the continuous variables d_L^{to}, $d_{\text{out}}^{tt'}$, d_{in}^{t}, d_R^{to}, d_{RL}^{o} and d_{LR}^{o}, defined in Table 3. This can be done by means of the following constraints:

$$\|x_L^o - R^t\| \leq d_L^{to}, \qquad \forall t \in \mathcal{T}, \forall o \in \mathcal{O}, \tag{DIST$_1$}$$

$$\|R^t - L^{t'}\| \leq d_{\text{out}}^{tt'}, \qquad \forall t, t' \in \mathcal{T}, \tag{DIST$_2$}$$

$$\|L^t - x_R^o\| \leq d_R^{to}, \qquad \forall t \in \mathcal{T}, \forall o \in \mathcal{O}, \tag{DIST$_3$}$$

$$\|x_L^o - x_R^o\| \leq d_{LR}^{o}, \qquad \forall o \in \mathcal{O}. \tag{DIST$_4$}$$

$$\|x_R^o - x_L^{o+1}\| \leq d_{RL}^{o}, \qquad \forall o \in \mathcal{O} \setminus \{|\mathcal{O}|\}. \tag{DIST$_5$}$$

Note that d_{in}^{t} is zero when the target is a point. However, to compute the distance inside the polygonal, we need to set the following expressions for each $p \in \mathcal{P}$:

$$d_{\mathcal{P}}^{s_p s'_p} = \begin{cases} |\gamma_R^{s_p} - \gamma_L^{s'_p}|\mathcal{L}(s_p), & \text{if } s_p = s'_p, \\ (1-\gamma_L^{s_p})\mathcal{L}(s_p) + \sum_{s''=s_p+1}^{s'_p-1} \mathcal{L}(s'') + \gamma_R^{s'_p}\mathcal{L}(s'_p), & \text{if } s_p < s'_p, \\ \gamma_L^{s_p}\mathcal{L}(s_p) + \sum_{s''=s'_p+1}^{s_p-1} \mathcal{L}(s'') + (1-\gamma_R^{s'_p})\mathcal{L}(s'_p), & \text{if } s_p > s'_p. \end{cases} \tag{$d_{\mathcal{P}}$}$$

This expression needs to define a binary variable $z_{\mathcal{P}}$ that determines whether $d_{\mathcal{P}}$ is defined by the first, the second or the third expression in the formula:

$$z_{\mathcal{P}}^{s_p s'_p} = \mu_R^{s_p} \mu_L^{s'_p}.$$

Finally, we can compute the total distance between the launching and the rendezvous points in each polygonal $p \in \mathcal{P}$:

$$d^p_{\text{in}} = \sum_{s_p, s'_p \in S_p} d^{s_p s'_p}_{\mathcal{P}} z^{s_p s'_p}_{\mathcal{P}}. \tag{DIST$_6$}$$

In [2, 15] it is discussed the idea of traversing a polygonal chain and the authors consider two different modes: 1) visiting a percentage of the total length of the graph and 2) traversing a given percentage of the length of each one of its segments. From an application point of view, the reader may understand these operations as reliability inspections so that testing a given percentage of the target suffices to certify a correct operation. Looking at the difficulty of these operation modes, the computational results reported in that work suggest that there is not a meaningful difference in terms of difficulty induced by the considered mode of visit. Hence, for the sake of simplicity, we only consider a simpler form of the first case where the drone, once it enters the polygonal chain, has to traverse the entire required percentage before leaving the target. In other words, no preemption is allowed. To include this requirement, we only need to impose that

$$d^p_{\text{in}} \geq \alpha^p \mathcal{L}(p). \tag{$\alpha - \mathcal{P}$}$$

There exists a special case of the above condition, when all the segments of the polygonal chain have the same length, that enables a simplified representation. We refer the interested reader to the Appendix 6 for further details of these constraints.

The coordination between the drone and the mothership must ensure that the time spent by the drone to do the operation o is less than or equal to the time that the mothership needs to move from the launching point to the retrieving point during this operation. To this end, we include the following coordination constraint for each operation $o \in \mathcal{O}$:

$$\frac{1}{v_D}\left(\sum_{t \in \mathcal{T}} u^{to} d^{to}_L + \sum_{t,t' \in \mathcal{T}} y^{tt'o} d^{tt'}_{\text{out}} + \sum_{t \in \mathcal{T}} \delta^{to} d^t_{\text{in}} + \sum_{t \in \mathcal{T}} v^{to} d^{to}_R\right) \leq \frac{d^o_{LR}}{v_M}. \tag{DCW}$$

Eventually, we have to impose that the tour of the mothership, together with the drone, starts from the origin *orig* and ends at the destination *dest*. This is ensured by including the following constraints:

$$x^0_L = orig, \tag{ORIG$_1$}$$
$$x^0_R = orig, \tag{ORIG$_2$}$$
$$x^{|\mathcal{O}|+1}_L = dest, \tag{DEST$_1$}$$
$$x^{|\mathcal{O}|+1}_R = dest. \tag{DEST$_2$}$$

Observe that, one of the addends of the objective function of this problem minimizes the right-hand-side of (DCW). Thus, this constraint will become an equality and thus, it is able to model the time capacity (endurance) restriction for a particular operation $o \in \mathcal{O}$ by limiting the space traveled by the mothership for this operation:

$$d^o_{LR} \leq N. \tag{Capacity}$$

The goal of the Multitarget-MDRP is to find a feasible solution that minimizes the total weighted distance traveled by the mothership and the drone. The following formulation, that includes all the constraints explained before, gives an exact model for this problem:

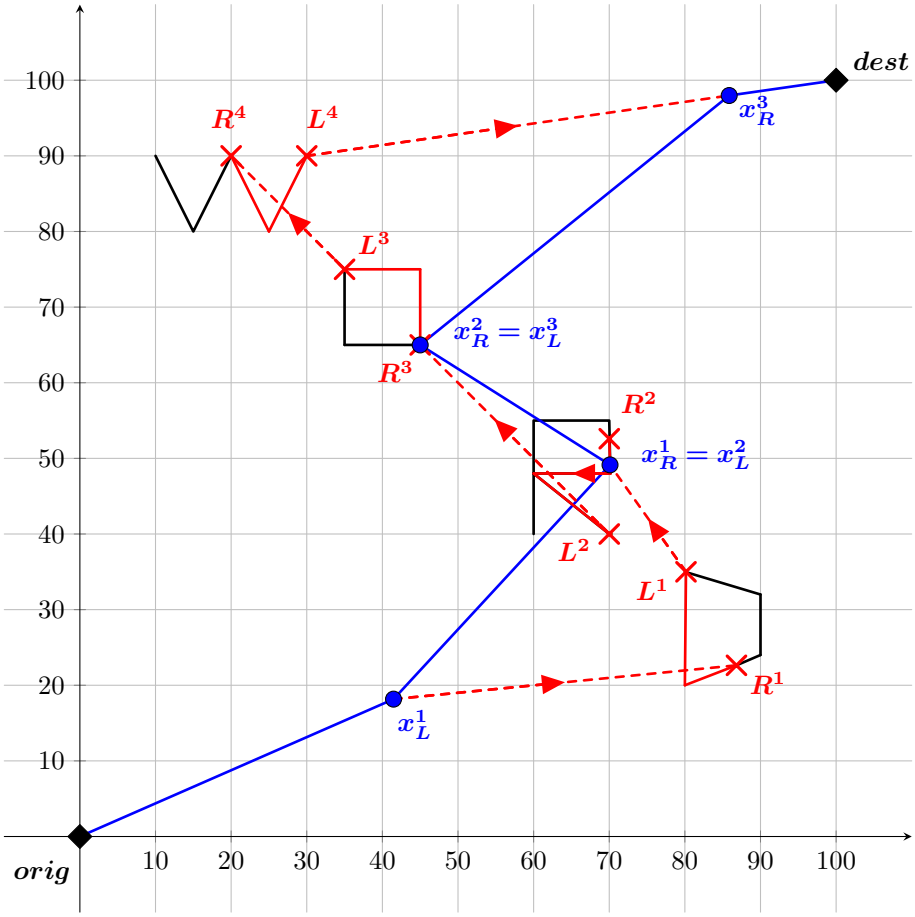

Figure 2: Optimal solution obtained for the data of the example

$$\min \quad \beta_D \left(\sum_{t\in\mathcal{T}} \sum_{o\in\mathcal{O}} u^{to} d_L^{to} + \sum_{t\in\mathcal{T}} \sum_{o\in\mathcal{O}} \delta^{to} d_{\text{in}}^{t} + \sum_{t\neq t'\in\mathcal{T}} \sum_{o\in\mathcal{O}} y^{tt'o} d_{\text{out}}^{tt'} + \sum_{t\in\mathcal{T}} \sum_{o\in\mathcal{O}} v^{to} d_R^{to} \right) + \tag{Multitarget-MDRP}$$

$$+ \beta_M \left(\sum_{o\in\mathcal{O}} d_{LR}^{o} + \sum_{o\in\mathcal{O}\setminus\{|\mathcal{O}|\}} d_{RL}^{o} \right)$$

s.t. (1) – (6),
($\mathcal{P}$-C), ($\alpha - \mathcal{P}$),
(DCW),
(Capacity),
($d_{\mathcal{P}}$),
(DIST_1) – (DIST_6),
(ORIG_1) – (DEST_2).

The objective function describes the weighted distances traveled by the drone and the mothership, respectively. Constraints (1)-(6) model the path made by the drone; ($\mathcal{P}$-C) and ($\alpha - \mathcal{P}$) describe the location of the rendezvous and launching points and the requirement of visiting an α percentage for the polygonal chain $p \in \mathcal{P}$, respectively. Constraint ($d_{\mathcal{P}}$) computes the distance between each pair of segments in the polygonal chain. Constraints (DIST_1)-(DIST_6) set the variables d_L^{to}, $d_{\text{out}}^{tt'}$, d_R^{to} , d_{LR}^{o}, d_{RL}^{o} and d_{in}^{t}, defined in Table 3, which set the Euclidean distances required in the model.

Note that, to handle the bilinear terms that appear in the objective function and in the (DCW)

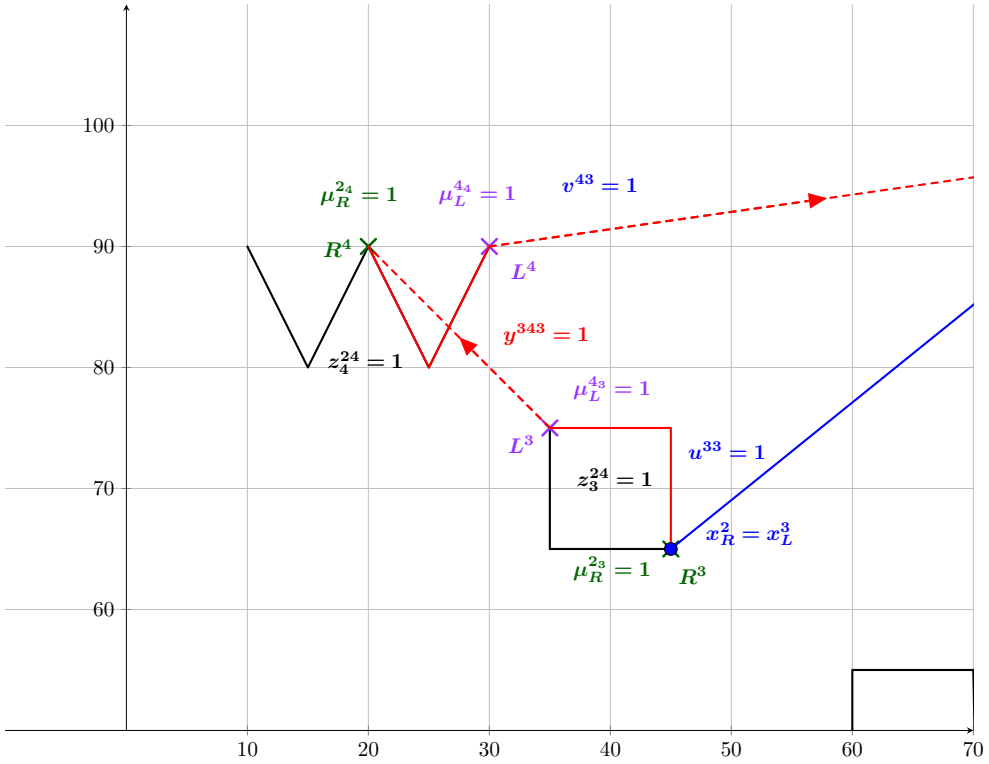

Figure 3: Binary decision variables associated to each target

constraint, we make use of the McCormick's envelopes to linearize these terms by including variables $q \geq 0$ representing the products and introducing the following constraints:

$$
\begin{aligned}
& q \leq Mz, \\
& q \leq d, \\
& q \geq mz, \\
& q \geq d - M(1 - z),
\end{aligned}
$$

where m and M are, respectively, the lower and upper bounds of the distance variable d. These bounds will be tightened for each bilinear term in Section 3.

Figure 2 shows an example of solution obtained by means of the exact method solving the formulation. We run the model on the same example of Figure 1 and we obtained the optimal solution consisting in the mothership tour, represented by the blue polygonal chain starting from the origin *orig*, ending at the destination *dest*, and with drone movements represented with the dotted red segments. The mothership launches the drone for its first operation from x_L^1. The drone flies to the retrieving point R^1 for visiting a portion (50%) of the first target. It leaves the first target from point L^1 and meets the mothership at point x_R^1. This point is also the launching point from where the drone starts its second operation for visiting the second target. From that point it flies to point R^2 and traverses the portion of the second target from R^2 to L^2. Then, the drone flies to point x_R^2 where it meets again the mothership. From this point the drone starts its last operation for visiting the third and the fourth targets. Indeed, after visiting the third target from point R^3 to point L^3, it directly flies to point R^4 of the last target. It visits the required portion of this last target from point R^4 to L^4 and then meets the mothership at point x_R^3. The mothership and the drone complete their service at the destination *dest*.

Figure 3 shows a zoom on the last two targets of the example of solution reported in Figure 2. We can visualize in detail the values of the different binary variables introduced in the formulation to define the order of visit of the targets, the launching and retrieving points and thus the mothership and drone tours. In particular, variable $u^{33} = 1$ indicates that the third operation of the drone starts in the third target from point R^3. This point is located on the line segment 2 of the polygonal target 3, $\mu_R^{23} = 1$, and it is also the retrieving point x_R^2 of the second operation. The visit on the third target ends at the launching point L^3 that is located on the line segment 4 of the polygonal target 3, $\mu_R^{43} = 1$. Indeed, the visit of target 3 starts in segment 2 and ends in segment 4, $z_3^{24} = 1$. From this point, always during the

third operation, the drone directly flies to the last target, that is target 4, $y^{343} = 1$. The visit of the target 4 starts from the point R^4, located on the line segment 2, $\mu_R^{24} = 1$, and ends at the launching point L^4, located on the line segment 4, $\mu_R^{44} = 1$. Eventually, the third and last operation of the drone finishes on target 4, $v^{43} = 1$.

3. Strengthening the formulation of Multitarget-MDRP

3.1. Pre-processing

In this subsection we explore the nature of the problem to fix a priori some variables and to increase the efficiency of the model. Particularly, the following proposition allows to fix $y^{tt'o}$ binary variables to zero.

Proposition 1. *Let $t, t' \in \mathcal{T}$ be two targets. Let $d_{min}(t, t')$ denote the minimum distance between them and $length(t)$, the length of the target t. If t, t' verify that*

$$\alpha^t \, length(t) + d_{min}(t, t') + \alpha^{t'} \, length(t') > \frac{v_D}{v_M} N, \tag{7}$$

then the drone cannot visit in a single operation t and t'.

Proof. Let assume that the drone can go from t to t' in the operation o. Then it must satisfy (Capacity) and (DCW) constraints:

$$\sum_{t \in \mathcal{T}} u^{to} d_L^{to} + \sum_{t,t' \in \mathcal{T}} y^{tt'o} d_{\text{out}}^{tt'} + \sum_{t \in \mathcal{T}} \delta^{to} d_{\text{in}}^{t} + \sum_{t \in \mathcal{T}} v^{to} d_R^{to} \leq \frac{v_D}{v_M} N.$$

Since $d_L^{to}, d_R^{to} \geq 0$, the left hand side of this inequality can be bounded from below by the left hand side of (7):

$$\alpha^t \, length(t) + d_{\min}(t, t') + \alpha^{t'} \, length(t') \leq \sum_{t \in \mathcal{T}} u^{to} d_L^{to} + \sum_{t,t' \in \mathcal{T}} y^{tt'o} d_{\text{out}}^{tt'} + \sum_{t \in \mathcal{T}} \delta^{to} d_{\text{in}}^{t} + \sum_{t \in \mathcal{T}} v^{to} d_R^{to},$$

which is impossible because each side is lower (resp. upper) bounded by $\frac{v_D}{v_M} N$.

□

3.2. Valid inequalities for the Multitarget-MDRP

In this subsection we introduce some valid inequalities for Multitarget-MDRP that strengthen the formulation presented in the Subsection 2.2. In addition, the constraint that coordinates the movement of the drone and the mothership, namely (DCW), and the objective function of the model hold products of binary and continuous variables. Each of these products generates big-M constants that must be tightened, when they are linearized. The present section provides some bounds for these constants.

For this problem, we are assuming that the drone endurance suffices to visit more than one target in the same operation because, otherwise, the problem is similar to (AMDRPG) that has been already considered in [2]. Hence, provided that there exists an operation in which the drone visits two or more targets, the mothership does not need to perform $|\mathcal{O}|$ different operations. This idea can be used to compactify all the operations made by the drone on the first operations, avoiding void tasks in $\mathcal{O}$.

Let β^o be a binary variable that attains the value one if the entire set of targets is visited when the operation o starts, and zero, otherwise. Observe that, if the drone has traversed all the targets before the operation o then they are also traversed before to start the operation $o + 1$. Therefore, β variables must fulfill the following constraints:

$$\beta^o \leq \beta^{o+1}, \text{ for all } o = 1, \ldots, |\mathcal{O}| - 1. \tag{Monotonicity}$$

Let k^o represent the number of targets that are visited at the operation o. δ variables can be used to compute this number because δ^{to} attains the value one when the target t is visited in the operation o. Thus:

$$k^o = \sum_{t \in \mathcal{T}} \delta^{to}.$$

Thus, if β^o is one, the full set $\mathcal{T}$ must have been visited before the operation o:

$$\sum_{o'=1}^{o-1} k^{o'} \geq |\mathcal{T}|\beta^o, \tag{VI-1}$$

where $|\mathcal{T}|$ stands for the cardinality of $\mathcal{T}$.

To reduce the space of feasible solutions, it is possible to assume, without loss of generality, that it is not allowed to have an operation o without any visit if the drone still has to visit some targets. To enforce that, we can set the following constraints:

$$k^o \geq 1 - \beta^o. \tag{VI-2}$$

Following the idea given in the Proposition 1 of the previous subsection, we can set some valid inequalities that indicate that the drone cannot visit a subset $S \subset \mathcal{T}$ of targets because of the (Capacity) constraint. Let $\mathcal{S}$ be the collection of subsets of $\mathcal{T}$ that do not verify (Capacity), then:

$$\sum_{t \in S} \delta^{to} \leq |S| - 1, \qquad \forall S \in \mathcal{S}, \quad \forall o \in \mathcal{O}. \tag{VI-3}$$

We can construct $\mathcal{S}$ by fixing the δ variables and partially solving (Multitarget-MDRP) for each subset of $\mathcal{T}$, which is a very expensive computation. However, it is sufficient to find minimal subsets by monotonicity.

´

The different models that we have proposed include in one way or another big-M constants. We have defined different big-M constants along this work. In order to strengthen the formulations we provide tight upper and lower bounds for those constants.

Big M constants bounding the distance from the launching / rendezvous point on the path followed by the mothership to the rendezvous / launching point on the target $t \in \mathcal{T}$

To linearize the first addend of the objective function in Multitarget-MDRP, we set the auxiliar non-negative continuous variables q_L^{to} (resp. q_R^{to}) to model the product by inserting the following inequalities:

$$\begin{aligned} q_L^{to} &\geq m_L^{to} u^{to}, \\ q_L^{to} &\leq d_L^{to} - M_L^{to}(1 - u^{to}). \end{aligned}$$

The best upper bound M_L^{to} or M_R^{to} is the full diameter of the data, that is, the maximum distance between every pair of vertices of the targets in $\mathcal{T}$, i.e., every point that must be determined is inside the circle whose diametrically opposite points are explained below.

$$M_L^{to} = \max_{\{v,v' \in V\}} \|v - v'\| = M_R^{to}.$$

Conversely, the minimum distance in this problem can be zero. This bound is attainable whenever the launching or the rendezvous points of the mothership is the same that the rendezvous or launching point on a given target.

Bounds on the big M constants for the distance from the launching to the rendezvous points on the operation $o \in \mathcal{O}$.

When the drone travels in the operation o, it has to go from one target t to another target t' depending on the order given by $y^{tt'o}$. This fact produces another product of variables linearized by the following constraints:

$$\begin{aligned} q^{tt'o} &\geq m^{tt'} y^{tt'o}, \\ q^{tt'o} &\leq d_{\text{out}}^{tt'} - M^{tt'}(1 - y^{tt'o}). \end{aligned}$$

The evaluation of the bounds appearing in these constraints presents three cases depending on the structure of the targets:

- If both targets t, t' are points, then the distance is fixed and we can set

$$M^{tt'} = \|R^t - R^{t'}\| = m^{tt'}.$$

- If one target t is a point and the other t' is a polygonal chain, we can compute the minimum distance as a minimum distance point-to-set problem:

$$\begin{aligned} m^{tt'} = \min \quad & \|R^t - x\| \\ \text{s.a.} \quad & x \text{ verifies } (\mathcal{P}\text{-C}). \end{aligned}$$

  On the other hand, the maximum distance between these targets can be obtained by taking the maximum of the distance between the point t and the breakpoints of the polygonal chain t':

$$M^{tt'} = \max_{v \in V_{t'}} \|v - R^t\|.$$

- If both targets t, t' are polygonal chains, it is also possible compute exactly the minimum distance:

$$\begin{aligned} m^{tt'} = \min \quad & \|x' - x\| \\ \text{s.a.} \quad & x, x' \text{ verifies } (\mathcal{P}\text{-C}). \end{aligned}$$

  On the other hand, to estimate the maximum distance we can repeat the procedure described in the previous case, but now for each breakpoint of the first polygonal chain. Then, taking the maximum of the maximum distances for each breakpoint we get an upper bound for $M^{tt`}$:

$$M^{tt'} = \max_{v \in V_t, v' \in V_{t'}} \|v - v'\|.$$

4. A Matheuristic for the Multitarget-MDRP

This section is devoted to present our matheuristic approach to provide good feasible solutions of the Multitarget-MDRP. Our motivation comes from the fact that the exact method based on the mathematical programming formulation presented in the previous section can be highly time demanding. Alternatively, the matheuristic provides good quality solution in limited computing times.

Assuming that the drone has enough endurance to visit every target, the basic idea of the algorithm is to associate each target to one operation by solving a crossing postman problem with neighbors (XPPN) (see [15]) for the targets including *orig* and *dest*. The motivation of this approach comes from the results in [15] which show that the XPPN is easily solvable for medium-size instances provided that the neighbors are points or polygonal chains. In the following, we present the pseudo-code of this algorithm:

STEP 1 (Order of visit the targets)
Compute the order of visit by solving the XPPN for the targets of the problem including *orig* as the first point in the tour and *dest* as the last one and associate each target $t \in \mathcal{T}$ to one operation $o \in \mathcal{O}$ in the given order.

STEP 2 (Solution of the Multitarget-MDRP model by fixing an initial partial solution)
Set the values of the binary variables u^{to}, v^{to}, δ^{to} and $y^{tt'o}$ provided by the solution of STEP 1 and solve the resulting Multitarget-MDRP model to obtain a complete feasible solution.

It is possible to refine the previous algorithm, by slightly modifying STEP 2. Indeed, after STEP 1, starting from the first visited target, according with the order provided by the XPPN solution, we can iteratively add the next target to the same operation, if the drone endurance allows it. In this way the number of operations can be reduced and a better initial partial solution can be provided to start STEP 2.

Figure 4 shows the solution obtained, by means of the matheuristic, for the same example of Figure 1. In particular, the upper subfigure reports the solution after STEP 2 in its original form. We can observe that the solution consists in the mothership tour represented in blue and in four operations of the drone. Indeed, the drone first visits the 50% of the target with label 4 ($u^{41} = v^{41} = 1$), then flies to meet the mothership at the retrieve point x_R^1 and from there starts its second operation to visit a 50% of the target with label 3 ($u^{32} = v^{32} = 1$). After that, the drone flies to the retrieve point x_R^2, it visits another 50% of the target with label 2 and then, from the launch point x_L^4 it starts its last operation to visit also another 50% of the the target with label 1 ($u^{14} = v^{14} = 1$). Then, it flies to the last retrieve point x_R^4 and moves to the destination point together with the mothership. In the lower subfigure of Figure 4 we can observe the solution obtained with the modified version of STEP 2. Differently from

the upper one, the number of drone operations is equal to two. Indeed, thanks to the refinement of STEP 2, the drone endurance permits to visit the two targets with label 4 and 3 in the first operation ($u^{41} = v^{31} = 1$ and $y^{431} = 1$). Similarly, the drone can also visit two targets, namely 2 and 1, in its second operation ($u^{22} = v^{12} = 1$ and $y^{212} = 1$).
In terms of objective function values, in this example, the refinement of STEP 2 does not provide an improved solution. Indeed, its value is equal to 888.01 without refinement and equal to 920.4 with the refinement of STEP 2. However, the length of the mothership tour, when STEP 2 is implemented in its original form, is equal to 189.72, while with the refinement of STEP 2 it is equal to 180.39. Moreover, we point out that the total time associated with the mothership tour is shorter in the solution without refinement of STEP 2. This is due to the different number of stops performed by the mothership in the two solutions. Indeed, in the solution obtained by the refinement, the number of mothership stops is 4 instead of 5 and, in some of them, the mothership waits for the drone. Summing up, in this example the refinement of STEP 2 generates a solution with a shorter tour of the mothership but with a weighted sum of the distances travelled by both, drone and mothership, that is worse than the one obtained without refinement. In general, depending on the instance and the weighting factor of the two terms in the objective function, the refinement of STEP 2 can provide better solutions. For this reason in the implementation we compared the solutions obtained with and without this refinement, and we select the best one to be provided as initial solution for the exact model.

5. Experimental results

In this section we discuss the experimental results obtained testing the formulation presented in Section 2.2 and the matheuristic procedure proposed in Section 4 on a testbed of instances.
In particular, we consider instances of two typologies: the first one in which the targets to be visited are represented by points randomly located in a square of side 100 units, and the second one where the targets are represented by polygonals.
In this latter case, we set the cardinality of the set of points of the polygonal we want to build equal to 4 (this is for guaranteeing that the drone endurance is sufficient for traversing the whole polygonal) and we generate a random value r in the interval $(5, 10)$. This value is used to generate a point $P = (P_x, P_y)$ inside the sub-square $[r, 100 - r]^2$.
Next, sequentially, we generate a random angle $\alpha \in (0, 2\pi)$ and from the current point P, we generate a new point $Q = (P_x + r * cos(\alpha), P_y + r * sin(\alpha))$. If point Q belongs to the square $[r, 100 - r]^2$, then we connect point Q to point P with a segment. Otherwise, we update α by adding $\pi/6$ to it until we obtain a point $Q^{'} = (P_x + r * cos(\alpha), P_y + r * sin(\alpha))$ that belongs to the square $[r, 100 - r]^2$. Then, the same procedure is repeated to generate the remaining break points of the polygonal.
For each of the two typologies, we generate instances of increasing size (number of targets) ranging between 5 and 15. For each size, we consider values of the drone endurance in the set $\{30, 40, 50, 60, 70\}$. For each combination of size and endurance value, we generate 5 instances. We run the formulation on these instances by adopting two different commercial solvers, Cplex and Gurobi, setting a time limit of 2 hours. Table 4 shows the results of this comparison. Specifically, for each typology (Type 1 for points, Type 2 for polygonals), for each size and each endurance value, we report the number of instances for which the solvers are able to find at least a feasible solution of the problem within the time limit (# f.i.), the average gap (Gap) and the average solution time in seconds (Time).

From Table 4 we can observe that there is a significant difference in the solvers performances. Indeed, Gurobi is not able to find a feasible solution, within the time limit, for any of the instances of Type 1 for the largest size instances ($|\mathcal{T}| \in \{14, 15\}$) and for instances of Type 2 from size 8 and drone endurance equal to 40. Moreover, for instances of Type 1 and sizes 12 and 13 and for instances of Type 2 and sizes 6, 7 and 8, it is able to find a feasible solution only for a subset of them, as reported by the counter # f.i.
On the contrary, Cplex is always able to find feasible solutions for all instances of both Types and for all sizes.

This behaviour of the two solvers seems to be explainable by the different internal implementation for handling the lazy constraints in the formulations during the solution process. Indeed, by analyzing the .mps files generated by both solvers, we found out that Gurobi embeds the lazy constraints directly before to start the branch and bound procedure, while Cplex adds them during the solution procedure when they are actually needed. This difference leads Gurobi to load and solve a more complex formulation since the beginning of the solution procedure. For this reason it is not able to find a feasible solution in 2 hours for most of the largest instances.

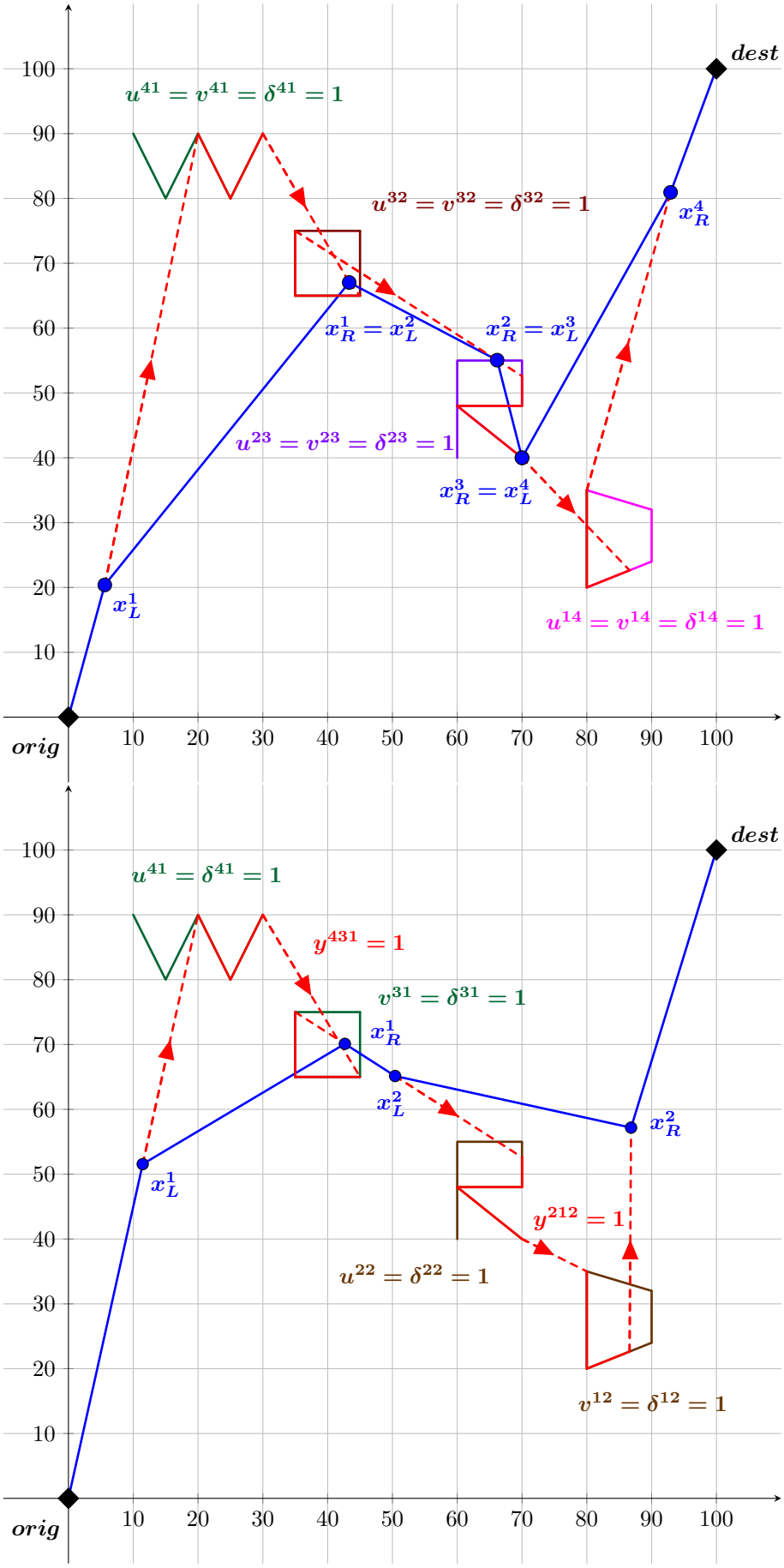

Figure 4: Solution provided by the matheuristic.

Table 4: Comparison between Cplex and Gurobi

$\|\mathcal{T}\|$	N	Type 1						Type 2					
		Cplex			Gurobi			Cplex			Gurobi		
		# f.i.	Gap	Time	# f.i.	Gap	Time	# f.i.	Gap	Time	# f.i.	Gap	Time
5	30	5	0	5,33	5	0	4.99	5	0	13,41	5	0	196.26
	40	5	0	5,08	5	0	3.13	5	0	10,9	5	0	322.56
	50	5	0	4,73	5	0	4.86	5	0	13,05	5	0	216.28
	60	5	0	5,27	5	0	3.48	5	0	19,07	5	0	339.01
	70	5	0	5,27	5	0	4.42	5	0	16,36	5	0	276.53
6	30	5	0	21,95	5	0	262.57	5	0	148.69	5	0.09	4250.68
	40	5	0	30,5	5	0	106.7	5	0	238.77	5	0.14	5196.74
	50	5	0	34,08	5	0	193.8	5	0	193.8	5	0.09	5333.45
	60	5	0	33,39	5	0	138.46	5	0	185.92	4	0.25	6602.12
	70	5	0	36,3	5	0	198.59	5	0	264.39	4	0.46	7200.52
7	30	5	0	229,75	5	0.11	4729.31	5	0	3063.24	2	0.62	7200.23
	40	5	0	268,47	5	0.08	4060.19	5	0,06	3420.54	2	0.44	7200.11
	50	5	0	255,35	5	0.23	6592.4	5	0,05	4034.04	3	0.58	7200.17
	60	5	0	348,62	5	0.29	5879.46	5	0	4787.98	1	0.55	7200.17
	70	5	0	301,47	5	0.27	6258.75	5	0	3986.24	0	*	*
8	30	5	0	2265,56	5	0.73	7200.15	5	0,42	7200.02	1	0.73	7200.68
	40	5	0	2820,11	5	0.61	7200.21	5	0,48	7200.02	0	*	*
	50	5	0	2877,69	5	0.65	7200.09	5	0,51	7200.02	0	*	*
	60	5	0	3159,62	5	0.76	7200.25	5	0,44	7200.02	0	*	*
	70	5	0	3188,16	5	0.72	7200.13	5	0,48	7200.02	0	*	*
9	30	5	0,36	7200,03	5	0.86	7200.19	5	0,68	7200.03	0	*	*
	40	5	0,35	7200,02	5	0.89	7200.23	5	0,66	7200.02	0	*	*
	50	5	0,37	7200,1	5	0.83	7200.2	5	0,67	7200.02	0	*	*
	60	5	0,36	7200,1	5	0.83	7200.18	5	0,66	7200.02	0	*	*
	70	5	0,33	7200,02	5	0.87	7200.24	5	0,64	7200.02	0	*	*
10	30	5	0,5	7200,42	5	0.92	7200.35	5	0,69	7200.03	0	*	*
	40	5	0,48	7200,62	5	0.91	7200.19	5	0,7	7200.04	0	*	*
	50	5	0,53	7200,47	5	0.92	7200.2	5	0,7	7200.04	0	*	*
	60	5	0,51	7200,94	5	0.92	7200.2	5	0,69	7200.14	0	*	*
	70	5	0,47	7201,19	5	0.92	7200.23	5	0,71	7200.04	0	*	*
11	30	5	0,61	7200,35	5	0.94	7200.48	5	0,71	7200.03	0	*	*
	40	5	0,65	7200,2	5	0.95	7200.18	5	0,7	7200.07	0	*	*
	50	5	0,65	7200,42	5	0.93	7200.24	5	0,69	7200.03	0	*	*
	60	5	0,64	7200,54	5	0.94	7200.28	5	0,71	7200.04	0	*	*
	70	5	0,6	7201,07	5	0.93	7200.23	5	0,72	7200.03	0	*	*
12	30	5	0,76	7200,59	5	0.96	7200.24	5	0,72	7200.04	0	*	*
	40	5	0,75	7200,64	5	0.96	7200.29	5	0,72	7200.04	0	*	*
	50	5	0,72	7200,84	5	0.95	7200.28	5	0,72	7200.04	0	*	*
	60	5	0,74	7200,92	5	0.96	7200.22	5	0,71	7200.05	0	*	*
	70	5	0,71	7200,96	4	0.96	7200.25	5	0,71	7200.06	0	*	*
13	30	5	0,85	7200,04	2	0.97	7200.31	5	0,76	7200.07	0	*	*
	40	5	0,84	7200,06	2	0.96	7200.32	5	0,75	7200.04	0	*	*
	50	5	0,82	7200,13	1	0.96	7200.24	5	0,75	7200.04	0	*	*
	60	5	0,82	7200,08	2	0.97	7200.32	5	0,74	7200.05	0	*	*
	70	5	0,83	7200,3	3	0.97	7200.4	5	0,75	7200.06	0	*	*
14	30	5	0,91	7200,26	0	*	*	5	0,73	7200.06	0	*	*
	40	5	0,9	7200,04	0	*	*	5	0,72	7200.07	0	*	*
	50	5	0,89	7200,28	0	*	*	5	0,72	7200.06	0	*	*
	60	5	0,89	7200,23	0	*	*	5	0,73	7200.08	0	*	*
	70	5	0,91	7200,22	0	*	*	5	0,71	7200.05	0	*	*
15	30	5	0,92	7200,07	0	*	*	5	0,72	7200.12	0	*	*
	40	5	0,9	7200,05	0	*	*	5	0,7	7200.08	0	*	*
	50	5	0,91	7200,25	0	*	*	5	0,72	7200.19	0	*	*
	60	5	0,92	7200,19	0	*	*	5	0,73	7200.41	0	*	*
	70	5	0,93	7200,22	0	*	*	5	0,71	7200.08	0	*	*

Regarding the number of instances solved to optimality, Cplex can solve instances up to size 8 of Type 1 within the time limit. In particular, it is able to solve, in average, in less than 6 seconds instances of size 5, in at most 36 seconds instances of size 6, in less than 6 minutes instances of size 7 and in at most 53 minutes instances of size 8. We can observe that the average solution time, increases by an order of magnitude, with respect to the instance size, up to size 8 and for larger size instances it reaches the time limit of 2 hours. Also considering the comparison between Cplex and Gurobi, we can observe a difference of an order of magnitude in the solution times for Type 1 instances up to size 7 and for Type 2 instances up to size 6.

The average gap associated with the solutions provided by both Cplex and Gurobi, ranges between 0.05 and 0.93 and it increases with the instances size. The instances of Type 2 seem to be the most difficult to solve. Indeed, depending on the instances size, the solution time is one or two orders of magnitude

greater than that for instances of Type 1. However, from instances of size 12, we can observe that the average gap associated with instances of Type 1, solved by means of Cplex, is always greater than the one associated with instances of Type 2.

Table 5: Exact solution via Cplex with and without initialization

$\|\mathcal{T}\|$	N	Type 1				Type 2			
		Gap (i)	Time_h	Time_f	Gap (wi)	Gap (i)	Time_h	Time_f	Gap (wi)
5	30	0	0,29	4,04	0	0	1.42	16.83	0
	40	0	0,21	3,88	0	0	1.47	17.17	0
	50	0	0,25	4,05	0	0	1.45	20.31	0
	60	0	0,32	4,72	0	0	1.42	20.75	0
	70	0	0,25	4,13	0	0	1.46	25.54	0
6	30	0	0,2	18,66	0	0	2.41	286.12	0
	40	0	0,17	17,92	0	0	2.55	394.29	0
	50	0	0,12	19,35	0	0	2.35	383.94	0
	60	0	0,16	19,72	0	0	2.24	322.92	0
	70	0	0,18	23,35	0	0	2.69	413.62	0
7	30	0	0,21	201,92	0	0	4.25	2806.77	0
	40	0	0,46	204,39	0	0.03	4.07	4651.65	0.06
	50	0	0,35	213,41	0	0.15	3.87	5409.26	0.05
	60	0	0,46	208,24	0	0.12	4.58	6281.02	0
	70	0	0,39	243,37	0	0.06	4.48	6103.81	0
8	30	0	0,7	1619,37	0	0.5	7.61	7200.03	0.42
	40	0	0,47	2333,97	0	0.52	7.65	7200.03	0.48
	50	0	0,48	2020,92	0	0.5	7.45	7200.03	0.51
	60	0	0,63	2484,49	0	0.47	7.02	7200.15	0.44
	70	0	0,62	2724,14	0	0.49	7.8	7200.09	0.48
9	30	0,27	0,94	7200,15	0,36	0.65	13.64	7200.06	0.68
	40	0,33	0,91	7200,71	0,35	0.66	12.31	7200.03	0.66
	50	0,29	0,9	7200,37	0,37	0.66	14.88	7200.04	0.67
	60	0,36	0,72	7200,4	0,36	0.66	14.63	7200.03	0.66
	70	0,31	0,68	7200,55	0,33	0.63	13.37	7200.03	0.64
10	30	0,45	1,43	7200,99	0,5	0.66	22.58	7200.02	0.69
	40	0,48	1,68	7201,77	0,48	0.64	20.89	7200.02	0.7
	50	0,44	1,61	7201,67	0,53	0.68	21.31	7200.16	0.7
	60	0,49	1,74	7201,77	0,51	0.66	22.1	7200.03	0.69
	70	0,46	1,8	7201,99	0,47	0.67	23.62	7200.03	0.71
11	30	0,54	1,8	7201,87	0,61	0.67	34.72	7200.03	0.71
	40	0,55	2,1	7201,78	0,65	0.67	34.9	7200.12	0.7
	50	0,54	2,12	7201,68	0,65	0.68	34.31	7200.13	0.69
	60	0,58	2	7201,54	0,64	0.68	34.9	7200.07	0.71
	70	0,56	2,11	7201,75	0,6	0.68	35.02	7200.03	0.72
12	30	0,66	3,14	7201,16	0,76	0.68	51.52	7200.18	0.72
	40	0,67	3,43	7201,26	0,75	0.68	53.85	7200.15	0.72
	50	0,74	3,04	7201,15	0,72	0.69	53.02	7200.08	0.72
	60	0,72	2,97	7201,35	0,74	0.68	53.83	7200.07	0.71
	70	0,71	3,12	7201,67	0,71	0.68	58.29	7200.04	0.71
13	30	0,81	5,42	7200,86	0,85	0.72	78.7	7200.1	0.76
	40	0,8	4,3	7201,14	0,84	0.72	83.12	7200.03	0.75
	50	0,83	5,46	7201,09	0,82	0.72	79.63	7200.08	0.75
	60	0,81	4,16	7201,16	0,82	0.72	86.79	7200.08	0.74
	70	0,82	4,28	7201,1	0,83	0.72	83.36	7200.04	0.75
14	30	0,85	6,75	7201,03	0,91	0.67	107.31	7200.16	0.73
	40	0,84	5,88	7201,02	0,9	0.67	110.38	7200.04	0.72
	50	0,87	6,07	7201,29	0,89	0.67	107.14	7200.15	0.72
	60	0,85	5,71	7201,15	0,89	0.67	112.05	7200.04	0.73
	70	0,84	6,4	7200,97	0,91	0.67	111.62	7200.06	0.71
15	30	0,87	7,1	7202,06	0,92	0.64	141.05	7200.27	0.72
	40	0,87	7	7201,2	0,9	0.64	146.34	7200.27	0.7
	50	0,88	9,47	7200,87	0,91	0.63	132.54	7200.26	0.72
	60	0,88	6,31	7201,39	0,92	0.63	144.69	7200.27	0.73
	70	0,88	7,15	7201,33	0,93	0.64	140.05	7200.21	0.71

Table 5 summarizes the comparison between the exact solution of the formulation via Cplex, with and without initialization with the solution found by the matheuristic algorithm. For each typology of instances and for each size and drone endurance, we report the average gap with initialization (Gap (i)), the average running time of the matheuristic (Time_h), the average running time of the formulation (Time_f) and the average gap without initialization (Gap (wi)). We can observe that the matheuristic provides a solution of the problem in less than 1 second for instances of Type 1 up to 9 targets and in less than 10 seconds for instances of Type 1 up to 15 targets. The polygonals instances (Type 2) are more challenging to be solved with solution times of the matheuristic ranging between 1.42 seconds and 2.4 minutes. However, comparing these times with those required by exact solution of the formulation, we

can conclude that the matheuristic algorithm is a very good alternative to the exact solver. Moreover, we can observe that when the solution provided by the matheuristic algorithm is used to initialize the solver for the formulation, in most of the cases the final average gap slightly decreases after the time limit is reached. The most significant reduction in the average gap can be observed for the largest size instances of Type 2 for which the average gap with initialization is up to 13% lower than the one without initialization. As regards the computation times for the exact solution method, we can also observe that if the solution provided by the matheuristic algorithm is used to initialize the solver, the convergence to the optimal solution for instances of Type 1 up to 8 targets is significantly improved.

Table 6: Exact solution via Gurobi with and without initialization

$\|\mathcal{T}\|$	N	Type 1				Type 2			
		Gap (i)	Time_h	Time_f	Gap (wi)	Gap (i)	Time_h	Time_f	Gap (wi)
5	30	0	0.01	2.15	0	0	0.37	71.63	0
	40	0	0.01	1.94	0	0	0.34	63.95	0
	50	0	0.01	2.11	0	0	0.35	75.82	0
	60	0	0.01	1.59	0	0	0.35	103.98	0
	70	0	0.01	1.76	0	0	0.35	141.09	0
6	30	0	0.01	33.6	0	0	0.64	642.4	0.09
	40	0	0.01	29.34	0	0	0.69	1131.4	0.14
	50	0	0.01	66.03	0	0	0.68	1362	0.09
	60	0	0.01	73.52	0	0	0.68	2161.66	0.25
	70	0	0.01	49.35	0	0	0.73	2011.79	0.46
7	30	0	0.02	1113.79	0.11	0.29	1.07	6680.17	0.62
	40	0	0.02	585.02	0.08	0.36	1.01	6969.07	0.44
	50	0	0.02	1193.87	0.23	0.35	1.04	6708.26	0.58
	60	0	0.02	1474.48	0.29	0.32	1.09	6871.41	0.55
	70	0	0.02	1046.2	0.27	0.42	1.06	7200.19	*
8	30	0.09	0.02	4446.46	0.73	0.63	3.03	7200.19	0.73
	40	0.18	0.02	5660.75	0.61	0.62	1.34	7200.16	*
	50	0.28	0.02	6581.53	0.65	0.62	1.36	7200.19	*
	60	0.32	0.02	6112.57	0.76	0.63	1.45	7200.25	*
	70	0.21	0.02	5863.52	0.72	0.62	1.42	7200.14	*
9	30	0.55	0.04	7200.16	0.86	0.72	3.28	7200.14	*
	40	0.49	0.04	7200.15	0.89	0.71	3.24	7200.12	*
	50	0.61	0.03	7200.12	0.83	0.7	3.16	7200.13	*
	60	0.55	0.03	7200.16	0.83	0.7	3.27	7200.11	*
	70	0.44	0.03	7200.21	0.87	0.71	3.26	7200.11	*
10	30	0.66	0.06	7200.23	0.92	0.7	8.27	7200.15	*
	40	0.65	0.06	7200.21	0.91	0.69	8.19	7200.17	*
	50	0.64	0.05	7200.19	0.92	0.69	8.21	7200.15	*
	60	0.62	0.06	7200.22	0.92	0.68	8.13	7200.15	*
	70	0.68	0.07	7200.18	0.92	0.7	8.3	7200.13	*
11	30	0.7	0.11	7200.37	0.94	0.68	11.7	7200.2	*
	40	0.73	0.11	7200.2	0.95	0.68	11.71	7200.1	*
	50	0.69	0.11	7200.21	0.93	0.67	11.71	7200.15	*
	60	0.71	0.11	7200.21	0.94	0.67	11.65	7200.16	*
	70	0.72	0.11	7200.2	0.93	0.67	11.68	7200.14	*
12	30	0.83	0.25	7200.22	0.96	0.68	15.83	7200.09	*
	40	0.83	0.25	7200.27	0.96	0.68	15.76	7200.14	*
	50	0.85	0.26	7200.19	0.95	0.68	23.28	7201.3	*
	60	0.84	0.25	7200.28	0.96	0.68	15.92	7201.16	*
	70	0.87	0.31	7200.22	0.96	0.69	15.85	7201.7	*
13	30	0.92	0.8	7200.29	0.97	0.71	26.34	7200.23	*
	40	0.89	0.85	7200.5	0.96	0.71	26.41	7200.24	*
	50	0.91	0.82	7200.72	0.96	0.71	26.41	7200.4	*
	60	0.89	0.8	7200.37	0.97	0.71	26.43	7200.26	*
	70	0.92	0.84	7200.41	0.97	0.71	26.33	7200.22	*
14	30	0.95	2.95	7200.7	*	0.67	52.07	7280.27	*
	40	0.95	2.87	7200.81	*	0.67	61.06	7200.33	*
	50	0.95	2.87	7200.77	*	0.67	78.34	7200.34	*
	60	0.96	2.91	7200.87	*	0.67	61.55	7200.28	*
	70	0.93	2.92	7201.08	*	0.67	52.78	7200.37	*
15	30	0.99	8.71	7200.4	*	0.62	99.44	7200.55	*
	40	1	8.54	7200.61	*	0.62	84.45	7200.54	*
	50	1	8.43	7200.36	*	0.62	124.48	7201.41	*
	60	1	8.35	7200.5	*	0.63	139.61	7200.8	*
	70	1	8.48	7200.27	*	0.63	109.46	7200.33	*

Similarly, Table 6 reports the comparison between the exact solution of the formulation via Gurobi, with and without initialization with the solution found by the matheuristic algorithm. In this case, the initialization permits to overcome the problem met by the Gurobi solver that, as also reported in Table 4, was not able to find a feasible solution for instances of Type 1 from size 14 and for instances of Type 2

from size 8. In particular, we can observe that for instances of Type 1 and from size 9, the initialization with the matheuristic solution improves more than with respect to same process in Cplex in terms of average gap.

In Figure 5 and Figure 6 we show the boxplots representing the gap values for the different instance sizes, distinguishing between the two typologies (Type 1: point or Type 2: polygonal targets) respectively by using Gurobi and Cplex. On the left of both figures we can observe the gap values obtained by solving the formulation without initialization, while on the right, the ones obtained by providing to the solvers the initial solution generated by the matheuristic algorithm. Specifically, we can visualize that the gap values for instances with point targets (Type 1) increase with the problem size in the exact solution both without and with initialization and with both solvers. However, as already observed in Table 4, Gurobi solver without initialization is not able to solve point instances of size 14 and 15. As regards the polygonal instances (Type 2), we can observe a different behaviour with respect to the gap, which increases with the instances size up to 9 targets. From instances of size 10 its gap, excluding outliers, always ranges in the interval [0.6, 0.8]. This can be still observed also by initializing the exact solution of the formulation even if, as already mentioned, in this latter case the gap values decrease and in most of the cases they belong to the interval [0.6,0.7], always excluding outliers. Thus, the polygonal instances are more difficult than the point instances for sizes up to 10. For bigger size instances, the gap value for polygonal instances consistently belongs to a fixed interval, while the gap for point instances increases with the size.

In order to further investigate this behaviour, we run the mathematical programming formulation with Cplex on one of the biggest size instances with 15 targets and endurance equal to 40, for both types of targets (Type 1 and 2) and setting a time limit of 24 hours. Figure 7 reports the objective function value and the lower bound value obtained over time, respectively, for the point and the polygonal targets versions of this instance. We can observe that the initial value of the lower bound for the point targets instance is equal to 0 while the one for the polygonal targets instance is equal to 331. This difference is due to the different structure of the targets. Indeed, for the polygonal targets the formulation includes the constraint imposing the visit of a given minimum percentage of each polygonal. This implies that the first value of the lower bound is greater than 0. Thus, while the solution process for the point targets instance starts from a gap equal to 100%, the one for the polygonal targets instance starts from a gap equal to 80%, as we can also observe in Figure 7. This different initial gap allows the solver to slowly improve it in the first case but not in the second one. Actually, only after around 15 hours of cpu, we can observe that the two gaps (for the instance of Type 1 and 2) are equal and then the value of the gap for the point targets instance continues to decrease, whereas the one related to the polygonal targets instance does not change in the remaining time. This is explained because the lower bound of the point targets instance improves over time, see Figure 7, but it does not change at all for the polygonal targets instance. This behaviour shows that the instances of Type 2 (polygonal targets) are still more difficult to be solved than the instances of Type 1 (point targets). However, the different structure of the targets provides an initial advantage in the lower bound of the polygonal target instances, due to the constraint related to the minimum percentage of each polygonal to be visited, that can be exploited by the solver for the biggest sizes.

6. Concluding remarks

This papers has analyzed a coordination problem that arises between a mothership vehicle and a drone that is allowed to visit more than one target per operation syncronizing their displacements to minimize travel distances. The mothership can move freely in a continuous space so that the drone can be launched and recovered at any point that is convenient to optimize the problem goal. We present an exact model that can be adapted to deal with pointwise and graph-like targets. This model is a mixed integer second order cone problem and it can be solved by most of current nowadays solvers. Our computational results show that the considered problem is rather hard and only small to medium size problems can be solved to optimality. For that reason, we also developed a matheuristic algorithm as a fast alternative to exact methods which provides acceptable feasible solutions in short computing time.

An interesting problem related with the one in this paper is the the coordination of the operations of one or several motherships with several drones each one of them allowed to visit more than one target per operation. This is a realistic, challenging problem that models actual situations in drone's delivery situations but although it is very interesting and deserves to be studied, it is beyond the scope of this paper and will be the topic of a follow up paper.

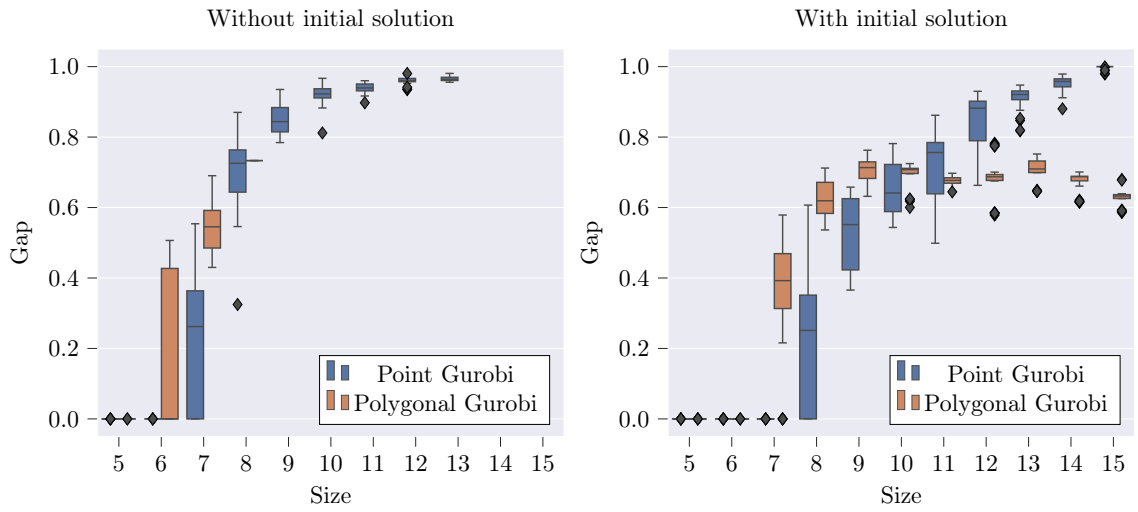

Figure 5: Final gap with and without initialization by using Gurobi.

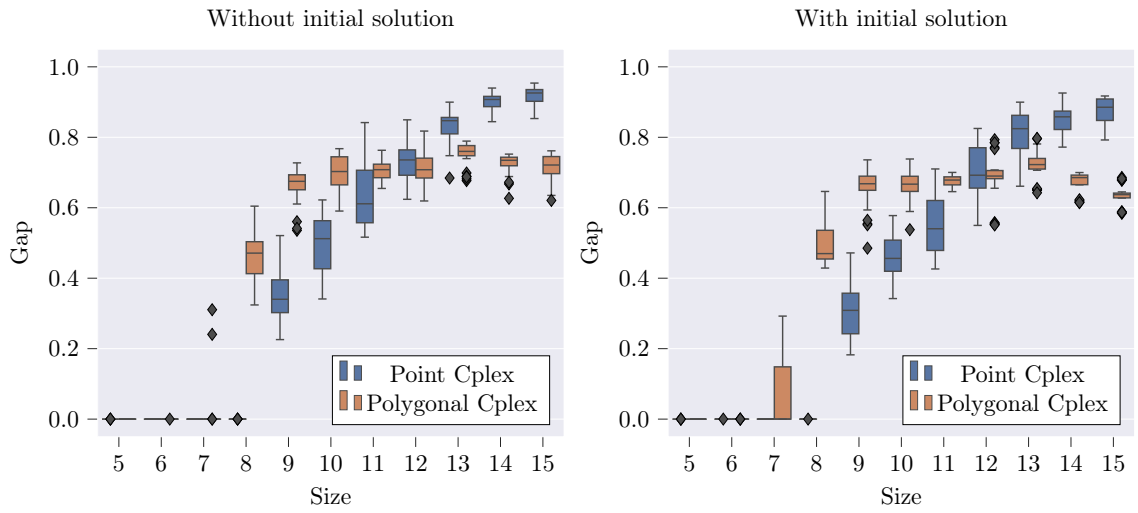

Figure 6: Final gap with and without initialization by using Cplex.

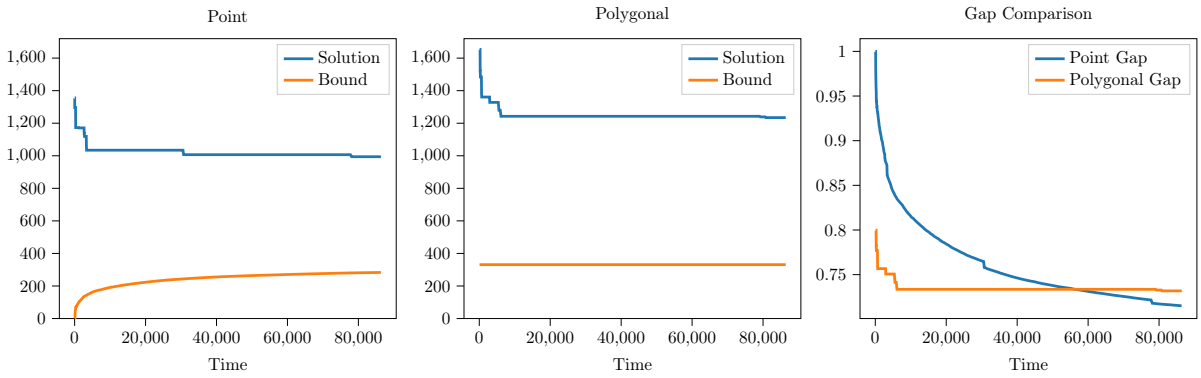

Figure 7: Evolution of objective function and lower bound values of one instance with 15 pointwise targets and drone endurance equal to 40 in 24 hours of CPU time.

Acknowledgements

This research has been partially supported by the Agencia Estatal de Investigación (AEI) and the European Regional Development's funds (ERDF): PID2020-114594GB-C21; Regional Government of Andalusia: projects CEI-3-FQM331, FEDER-US-1256951, and P18-FR-1422; Fundación BBVA: project NetmeetData (Ayudas Fundación BBVA a equipos de investigación científica 2019) and University of Rome, Sapienza grant number RM11916B7F962975.

References

[1] Amorosi, L., Chiaraviglio, L., and Galan-Jimenez, J. (2019). Optimal energy management of uav-based cellular networks powered by solar panels and batteries: Formulation and solutions. IEEE Access, Vol. 7:53698–53717.

[2] Amorosi, L., Puerto, J., and Valverde, C. (2021a). Coordinating drones with mothership vehicles: The mothership and drone routing problem with Graphs. Computers and Operations Research, 136.

[3] Amorosi, L., Puerto, J., and Valverde, C. (2021b). Coordinating drones with mothership vehicles: The mothership and multiple drones routing problem with Graphs. http://arxiv.org/abs/2109.01447.

[4] Blanco, V., Fernández, E., and Puerto, J. (2017). Minimum spanning trees with neighborhoods: Mathematical programming formulations and solution methods. European Journal of Operational Research, 262(3):863–878.

[5] Cavani, S., Iori, M., and Roberti, R. (2021). Exact methods for the traveling salesman problem with multiple drones. Transportation Research Part C: Emerging Technologies, 130.

[6] Chiaraviglio, L., Amorosi, L., Blefari-Melazzi, N., Dell'olmo, P., Lo Mastro, A., Natalino, C., and Monti, P. (2019). Minimum Cost Design of Cellular Networks in Rural Areas with UAVs, Optical Rings, Solar Panels, and Batteries. IEEE Transactions on Green Communications and Networking, 3(4):901–918.

[7] Chui, M., Manyika, J., and Miremadi, M. (2016). Where machines could replace humans-and where they can't (yet). McKinsey Q., 7:1–6.

[8] Chung, S. H., Sah, B., and Lee, J. (2020). Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions. Computers & Operations Research, 123:105004.

[9] Coindreau, M., Gallay, O., and Zufferey, N. (2021). Parcel delivery cost minimization with time window constraints using trucks and drones. Networks, On line: DOI: 10.1002/net.22019.

[10] Dell'Amico, M., Montemanni, R., and Novellani, S. (2021). Modeling the flying sidekick traveling salesman problem with multiple drones. Networks, On line: DOI: 10.1002/net.22022.

[11] Di Puglia Pugliese, L. and Guerriero, F. (2017). Last-Mile Deliveries by Using Drones and Classical Vehicles BT - Optimization and Decision Science: Methodologies and Applications. pages 557–565, Cham. Springer International Publishing.

[12] Macrina, G., Di Puglia Pugliese, L., Guerriero, F., and Laporte, G. (2020). Drone-aided routing: A literature review. Transportation Research Part C: Emerging Technologies, 120.

[13] Mbiadou Saleu, R. G., Deroussi, L., Feillet, D., Grangeon, N., and Quilliot, A. (2021). The parallel drone scheduling problem with multiple drones and vehicles. European Journal of Operational Research.

[14] Otto, A., Agatz, N., Campbell, J., Golden, B., and Pesch, E. (2018). Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey. . Networks, Vol. 72:1–48.

[15] Puerto, J. and Valverde, C. (2021). Routing for unmanned aerial vehicles: Touring dimensional sets. European Journal of Operational Research.

[16] Roberti, R. and Ruthmair, M. (2021). Exact methods for the traveling salesman problem with drone. Transportation Science, 55:315–335.

[17] Vidal, T., Laporte, G., and Matl, P. (2020). A concise guide to existing and emerging vehicle routing problem variants. European Journal of Operational Research, 286(2):401–416.

[18] Wang, K., Pesch, E., Kress, D., Fridman, I., and Boysen, N. (2022). The piggyback transportation problem: Transporting drones launched from a flying warehouse. European Journal of Operational Research, 296(2):504–519.

Appendix: The special case of polygonal chains with equal-length segments

This appendix shows a simplification of the constraints modelling traversing a given percentage of the polygonal targets provided that all their segments are of the same length. In that case, we can model the entry point R^p and the exit point L^p associated with the polygonal chain $p \in \mathcal{P}$ by means of a parameter $\rho^p \in [1, |V_p|]$ (resp. $\lambda^p \in [1, |V_p|]$) that determines the absolute position in p.

To compute the value of these parameters, we can link them with the variables that model the location of the points inside the polygonal by including the following inequalities for each $s_p \in S_p$:

$$\begin{aligned}
\rho^p - s_p &\geq \gamma_R^{s_p} - |V_p|(1 - \mu_R^{s_p}),\\
\rho^p - s_p &\leq \gamma_R^{s_p} + |V_p|(1 - \mu_R^{s_p}),\\
\lambda^p - s_p &\geq \gamma_L^{s_p} - |V_p|(1 - \mu_L^{s_p}),\\
\lambda^p - s_p &\leq \gamma_L^{s_p} + |V_p|(1 - \mu_L^{s_p}).
\end{aligned}$$

The first and second inequalities determine the upper and lower limits for the parameterization of each segment of p. If $\mu_R^{s_p} = 0$ the inequalities are always fulfilled and there is no entry point in the s_p-th segment of the polygonal. Conversely, if $\mu_R^{s_p} = 1$ then $\rho^p = \gamma_R^{s_p} + s_p$ meaning that the corresponding entry or exit point is in the s_p-th segment of the polygonal and its value is equals to the number of segments plus the part of the segment s_p that has been already traversed. The same idea is applied in the third and fourth inequalities for the λ^p parameter.

Finally, we can model the condition of traversing a percentage α^p of the polygonal p by the standard trick for the absolute value constraint:

$$|\rho^p - \lambda^p| \geq \alpha^p |S_p| \iff \begin{cases} \rho^p - \lambda^p &= \nu^p_{\max} - \nu^p_{\min} \\ \nu^p_{\max} &\leq 1 - \text{entry}^p, \\ \nu^p_{\min} &\leq \text{entry}^p, \\ \nu^p_{\max} + \nu^p_{\min} &\geq \alpha^p |S_p|. \end{cases} \qquad (\alpha - \mathcal{P})$$

Here $\nu^p_{\max}$ and $\nu^p_{\min}$ are auxiliary variables used for linearizing the absolute value and the binary variable entry^p represents the traveling direction in the polygonal chain p.