Coordinating drones with mothership vehicles: The mothership and multiple-drone routing problem with graphs

Lavinia Amorosi[1,], Justo Puerto[1,], Carlos Valverde[1,]

[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 fleet of drones. Each drone can be launched from the mothership to perform a task. After completing their tasks, the drones return to the mothership to recharge their batteries and to be ready for a new task. Tasks consist of (partially) visiting graphs with a given length to deliver some services or to perform a surveillance/inspection activity. The goal is to minimize the overall time of traveling carried out by the mothership (makespan) while satisfying some requirements in terms of fractions of visits to the target graphs. 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 a matheuristic algorithm that provides reasonable solutions. Computational experiments show the usefulness of our methodology in different scenarios.

Keywords: Arc Routing Problems, Networks, Drones, Conic Programming

1. Introduction

In recent years, the growth of potential business opportunities related to the use of drone technology has motivated the appearance of an interesting body of methodological literature on the optimization of the use of such technology. Examples may be found of that in many different sectors, like telecommunications where drones can be adopted in place of traditional infrastructures to provide connectivity (see, for example [?], [?], [?], [?], and [?]), or to temporarily deal with the damage caused by a disaster ([?], [?]), deliveries (see, for example [?] , [?], [?], [?] and [?]), also in emergency ([?]), inspection ([?]) and other contexts. The reader is referred to the recent surveys [?], [?] and [?] for further details. Among the different aspects that can be considered, the focus of this article due to its relationship with the development in this paper, is on the design, coordination and optimization of the combined routes of drones with a base vehicle. After the initial paper [?], where a combined model of truck and drone is considered, the work [?] also considers another model where trucks and drones are dispatched as orders are placed and analyzes the effect of different policies for either the truck or the drone. Other papers, such as, for instance, [?], [?], [?], [?] and [?], have also considered hybrid truck-and-drone models in order to mitigate the limited delivery range of drones. In [?] the authors advance on the coordination problem considering the *Mothership and drone routing problem* where these two vehicles are used to design a route that visits a number of points allowing the truck to launch and recover the drone in a continuous space. More recently, in [?], the authors consider the *k-Multi-Visit drone routing problem* where a truck that acts as a mobile depot only allowed to stop at a predefined set of points, launches drones that can deliver more than one package to their designated destination points. Many of these papers make assumptions that the set of allowable locations to launch/retrieve a drone are fixed and known a priori, the operations performed by the drone consist of delivering to a single point and the coordination is between a truck and a single drone. These assumptions may be appropriate in some frameworks, but, in other cases, it would be better to relax them.
In particular, only a few papers in the literature focus on drone operations consisting of traversing graphs rather than visiting single points. Later the authors in [?] introduce the *Drone Rural Postman Problem* (DRPP). These authors present a solution algorithm based on the approximation of curves in the plane by polygonal chains that iteratively increases the number of points in the polygonal chain where the

*Equally contributing authors

UAV can enter or leave. Thus, they solve the problem as a discrete optimization problem trying to better define the curve by increasing the number of points. The authors also consider the case in which the drone has limited endurance and thus it cannot serve all the lines. To deal with the latter case, they assume to have a fleet of drones and the problem consists of finding a set of routes, each of limited length.
In [?] this problem was defined as the *Length Constrained K-drones Rural Postman Problem*, a continuous optimization problem where a fleet of homogeneous drones has to jointly service (traverse) a set of (curved or straight) lines of a network. The authors design and implement a branch-and-cut algorithm for its solution and a matheuristic algorithm capable of providing good solutions for large-scale instances of the problem.
Scanning the literature of arc routing problems involving hybrid systems consisting of one vehicle and one or multiple drones, the number of contributions is rather limited.
In [?] the authors study the path planning problem of a system composed of a ground robot and one drone in precision agriculture and solve it by applying orienteering algorithms. Moreover, paper [?] studies the problem of path planning for systems consisting of a carrier vehicle and a carried one to visit a set of target points and assuming that the carrier vehicle moves in continuous space.
To the best of our knowledge, paper [?] is the only one that deals with the coordination of a mothership with one drone to visit targets represented by graphs. The authors made different assumptions on the route followed by the mothership: it can move on a continuous framework (the Euclidean plane), ii) on a connected piecewise linear polygonal chain, or iii) on a general graph. In all cases, the authors develop exact formulations resorting to mixed integer second-order cone programs and propose a matheuristic algorithm capable of obtaining high quality solutions in short computing time.
The set of target graphs to be visited permits one to model real situations like monitoring or inspection activities on fractions of networks (roads or wires) where traditional vehicles cannot arrive, due to, for example, the presence of narrow streets, or because of a natural disaster or a terrorist attack that caused damage to the network. In all these cases, the inspection or monitoring of the drone consists of traversing edges of the network to perform a reconnaissance activity. For this reason, we model the targets to be visited by the drone as graphs. The action of visiting a graph can be of two different types: 1) traversing a given fraction of the length of each one of its edges or 2) visiting a fraction of the total length of the network. Other kinds of inspection activities, like, for example, video surveillance of urban areas of big cities, can also be modeled by adopting the formulations presented in this paper. In this context, the request of visiting only a given fraction of the target graphs (e.g., borders of a neighborhood) can be due to the necessity of "covering" different areas in a limited amount of time. Another example that we can mention is traffic flow monitoring. In this case, to verify whether traffic progression is not disrupted, only by inspecting a fraction of the edge provides valuable information.
In this paper we deal with an extension of the problem studied in [?] for which we propose a novel truck-and-multi-drone coordination model. We consider a system where a base vehicle (mothership) travels in continuous space and has to support the launch/retrieval of a number of drones that must visit graphs. Our contribution over the existing literature is to extend the coordination beyond a single drone to the more cumbersome case of several drones and operations to traversing graphs rather than visiting single points. In particular, we focus on two different versions. In the first, called *complete overlapping model*, operations consisting of the launching and retrieving of a set of drones are done sequentially so that no two consecutive launches are possible without retrieving the previously launched drones (See Figure ?? left). The second version, called *partial overlapping model*, allows consecutive launching or retrieving actions so that the visits of several drones to their target graphs are allowed to partially overlap over time (See Figure ?? right). Note that the partial overlapping variant is an extension of the complete overlapping one. Indeed, in both versions of the problem, the situation is included in which one drone is launched and retrieved before another different drone is launched to visit another target graph. For both problem variants, we present mathematical programming formulations, valid inequalities to strengthen them and ad hoc matheuristics to deal with large sized instances of the problem.

The work is structured as follows: Section ?? provides a detailed description of the problem under consideration. Section ?? develops valid mixed integer nonlinear programming formulations for the two versions of the problem considered. Here, we also show the relationship between the two models and prove that the second model is a relaxation of the first. Section ?? provides some valid inequalities that strengthen the formulations and derives upper and lower bounds on the bigM constants introduced in the proposed formulations. Section ?? presents details of the matheuristic algorithms designed to handle large sized instances. In Section ?? we report the results obtained testing the formulations and

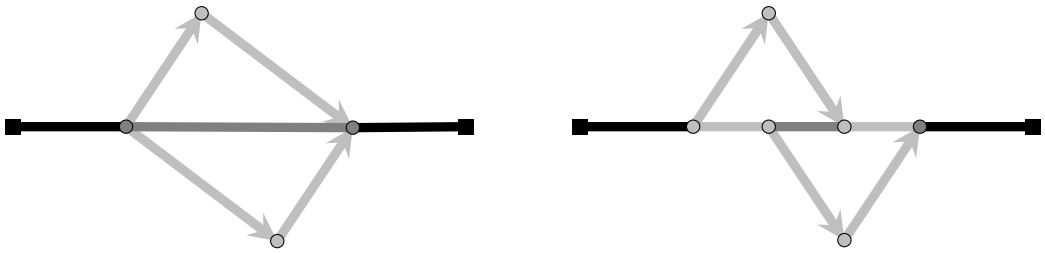


Figure 1: Models with complete and partial overlapping.

the matheuristic algorithm on different classes of planar graphs to assess their effectiveness. In Section **??**, we introduce an illustrative example applying the coordination models to the Courtyards Festival of Cordoba. Section **??** concludes the paper. Finally, for the sake of completeness, we have included an appendix with an extension of the model of complete overlapping where the drones are not assumed to be homogeneous.

2. Problem description

In the All Terrain Mothership and Multiple-Drone Routing Problem with Graphs (AMMDRPG), there is one mothership (the base vehicle) and a fleet of homogeneous drones $\mathcal{D}$ that have to coordinate between each other and with the mothership to perform a number of operations consisting of visiting given fractions of the length of a set of graphs $\mathcal{G}$. The mothership and the drones travel at constant velocities v_M and v_D, respectively. We also assume that it is not necessary for the mothership to be stopped to launch and retrieve the drones and that the time spent by the mothership to launch and retrieve them is negligible. Moreover, it is assumed that each drone has a limited flying autonomy endurance N_D, so that once it is launched, it must complete the operation and return to the base vehicle to recharge its batteries before the time limit. In addition, the base vehicle moves freely on the continuous space. This assumption can model the case where the base vehicle is a helicopter or a boat, so that there are no obstacles or restrictions to its movement. Nowadays, this type of system consisting of a boat and a fleet of drones is used, for example, by coast guards to perform surveillance activities to identify immigrants that need help in the sea (see [**?**]). The mothership starts at a known location, denoted *origin* where the whole system is ready to depart. Once all the operations are finished, the mothership and the drones must return together to a final location, called *dest*. Moreover, we assume, without loss of generality, that the drone endurance does not allow it to visit all target graphs in a single trip starting from the origin and finishing at the destination. Otherwise, the problem becomes trivial and no coordination is required.
In this problem, it is assumed that each graph must be visited by one drone: once the drone is assigned to this action, it goes to visit the graph and has to complete the entire action of traversing this target before returning to the base. We assume that the time spent by the drones to visit the graph must be lower than or equal to the time that the mothership needs to move from the launching point to the retrieving point. Note also that every drone in the fleet cannot be launched from the same base vehicle location to carry out all the tasks because of its limited endurance. In addition, it is supposed that the costs induced by the drones' trips are negligible compared to those incurred by the base vehicle. Therefore, the goal is to minimize the makespan which in this case coincides with the overall time traveled by the mothership. In spite of that, the reader may note that from a theoretical point of view, the extension to include in the objective function the times traveled by the drones is straightforward and does not increase the complexity of the models or formulations. The goal of the AMMDRPG is to find the launching and rendezvous points of the fleet of drones $\mathcal{D}$ satisfying the visit requirements for the graphs in $\mathcal{G}$ and minimizing the makespan (total time traveled by the mothership).

3. Mixed-Integer Non-Linear Programming Formulations

In this section, we present a mixed-integer nonlinear programming (MINLP) formulation for the AMMDRPG that can be used to solve medium-size instances of this problem. As mentioned in Section **??**, we assume that the mothership is allowed to move freely in a continuous space that for the sake of presentation we assume to be $\mathbb{R}^2$. Here, the distances are measured by the Euclidean norm, $\|\cdot\|_2$, although

this assumption can be extended to any l_p norm, $1 \leq p \leq \infty$ (see [?]). In the following, we introduce the parameters or input data that formally describe the problem and that are summarized in Table ??.

Problem Parameters
origin: coordinates of the point defining the origin of the mothership path (or tour).
dest: coordinates of the point defining the destination of the mothership path (or tour).
$\mathcal{G}$: set of the target graphs.
$g = (V_g, E_g)$: set of nodes and edges of each target graph $g \in \mathcal{G}$.
$\mathcal{L}(e_g)$: length of edge e of graph $g \in \mathcal{G}$.
$\mathcal{L}(g) = \sum_{e_g \in E_g} \mathcal{L}(e_g)$: total length of the graph $g \in \mathcal{G}$.
B^{e_g}, C^{e_g}: coordinates of the endpoints of edge e of graph $g \in \mathcal{G}$.
α^{e_g}: fraction of length of edge e of graph $g \in \mathcal{G}$ that must be visited. It ranges from 0 to 1.
α^g: fraction of length of graph $g \in \mathcal{G}$ that must be visited. It ranges from 0 to 1.
v_M: mothership speed.
$\lvert\mathcal{D}\rvert$: number of drones.
v_D: drone speed.
N_D: drone endurance.
$\mathcal{O}$: set of drone operations to perform visits to the target graphs. $\mathcal{O} = \{1, \ldots, \lvert\mathcal{O}\rvert\}$.
M: big-M constant.

Table 1: Nomenclature for AMMDRPG-CO

In the following, we describe all the constraints required to formulate the AMMDRPG model. Table ?? summarizes the set of decision variables that appear in that formulation.

Visits to graphs

To represent the movement of the drone within a graph $g \in \mathcal{G}$, we proceed to introduce some notations related to g. Let $g = (V_g, E_g)$ be a graph in $\mathcal{G}$ whose total length is denoted by $\mathcal{L}(g)$. Here, V_g denotes the set of nodes and E_g denotes the set of edges connecting pairs of nodes. Let e_g be the edge e of the graph $g \in G$ and let $\mathcal{L}(e_g)$ be its length. Each edge e_g is parameterized by its endpoints $B^{e_g} = (B^{e_g}(x_1), B^{e_g}(x_2))$ and $C^{e_g} = (C^{e_g}(x_1), C^{e_g}(x_2))$ and we can compute its length $\mathcal{L}(e_g) = \|C^{e_g} - B^{e_g}\|$. For each edge e_g an indicator binary variable μ^{e_g} is associated, that is one if the drone visits the segment e_g. Moreover, we define the entry and exit points $R^{e_g} = (B^{e_g}, C^{e_g}, \rho^{e_g})$ and $L^{e_g} = (B^{e_g}, C^{e_g}, \lambda^{e_g})$ that determine the fraction of the edge visited by the drone. The coordinates of the points R^{e_g} and L^{e_g} are given, respectively by

$$R^{e_g} = \rho^{e_g} B^{e_g} + (1 - \rho^{e_g}) C^{e_g} \quad \text{and} \quad L^{e_g} = \lambda^{e_g} B^{e_g} + (1 - \lambda^{e_g}) C^{e_g},$$

where $\rho^{e_g} \in [0, 1]$ and $\lambda^{e_g} \in [0, 1]$ are variables to determine the position of the points on the segment. As discussed in Section ??, we consider two modes of visit to the target graphs $g \in \mathcal{G}$:

- Visiting a fraction α^{e_g} of each edge e_g which can be modeled by using the following constraints:

$$|\lambda^{e_g} - \rho^{e_g}| \geq \alpha^{e_g}, \quad \forall e_g \in E_g. \tag{α-E}$$

  These inequalities state that the difference between the parameterizations of the entry and exit points associated with each edge e_g must be higher than the fraction of the length of e_g required to be traversed.

- Visiting a fraction α^g of the total length of the graph:

$$\sum_{e_g \in E_g} \mu^{e_g} |\lambda^{e_g} - \rho^{e_g}| \mathcal{L}(e_g) \geq \alpha^g \mathcal{L}(g). \tag{α-G}$$

  This constraint ensures that the sum of the fractions of length of those edges chosen to be crossed must be higher than the fraction of length of g required to be traversed.

In both cases, the corresponding constraints are nonlinear. To linearize them, we need to introduce a binary variable entry^{e_g} that determines the traveling direction on the edge e_g as well as the definition of

Binary and Integer Decision Variables
$\mu^{e_g} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G})$: equal to 1 if edge e of graph g (or a fraction of it) is visited by the drone, 0 otherwise.
$\text{entry}^{e_g} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G})$: auxiliary binary variable used for linearizing expressions.
$u^{e_g o} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O}$: equal to 1 if one drone enters graph g through the edge e_g at operation o, 0 otherwise.
$z^{e_g e_g^{'}} \in \{0,1\},\ \forall e_g, e_g' \in E_g\ (g \in \mathcal{G})$: equal to 1 if one drone goes from e_g to $e_g^{'}$, 0 otherwise.
$v^{e_g o} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O}$: equal to 1 if one drone exits graph g by e_g at operation o, 0 otherwise.
Continuous Decision Variables
$s^{e_g} \in [0, \lvert E_g \rvert - 1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: continuous non-negative variable representing the order of visits to the edge e of graph g.
$\rho^{e_g} \in [0,1]$ and $\lambda^{e_g} \in [0,1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: defining the entry and exit points on e_g.
$\nu_{\min}^{e_g}$ and $\nu_{\max}^{e_g} \in [0,1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: auxiliary variables used for linearizing expressions.
$x_L^o \in \mathbb{R}^2,\ \forall o \in \mathcal{O}$: coordinates representing the point where the mothership launches the drones at operation o.
$x_R^o \in \mathbb{R}^2,\ \forall o \in \mathcal{O}$: coordinates representing the point where the mothership retrieves the drones at operation o.
$R^{e_g} \in \mathbb{R}^2,\ \forall e_g \in E_g\ (g \in \mathcal{G})$: coordinates representing the entry point on edge e_g of graph g.
$L^{e_g} \in \mathbb{R}^2,\ \forall e_g \in E_g\ (g \in \mathcal{G})$: coordinates representing the exit point on edge e_g of graph g.
$d_{origin} \geq 0$: distance from the origin $origin$ to the first launching point x_L^1.
$d_L^{e_g o} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O}$: representing the distance traveled by one drone from the launching point x_L^o on the mothership at operation o to the first visiting point R^{e_g} on e_g.
$p_L^{e_g o} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O}$: auxiliary variable used for modeling the product of $d_L^{e_g o}$ and $u^{e_g o}$.
$d^{e_g} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G})$: representing the distance traveled by the drone from the rendezvous point R^{e_g} to the launching point L^{e_g} on e_g.
$p^{e_g} \in [0,1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: auxiliary variable used for modeling the product of μ^{e_g} and $\lvert \lambda^{e_g} - \rho^{e_g} \rvert$.
$d^{e_g e_g^{'}} \geq 0,\ \forall e_g, e_g' \in E_g\ (g \in \mathcal{G})$: representing the distance traveled by the drone from the launching point L^{e_g} on e_g to the rendezvous point $R^{e_g^{'}}$ on e_g'.
$p^{e_g e_g'} \geq 0,\ \forall e_g, e_g' \in E_g\ (g \in \mathcal{G})$: auxiliary variable used for modeling the product of $d^{e_g e_g'}$ and $z^{e_g e_g'}$.
$d_R^{e_g o} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O}$: representing the distance traveled by one drone from the last visiting point L^{e_g} on e_g to the rendezvous point x_R^o on the mothership at operation o.
$p_R^{e_g o} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O}$: auxiliary variable used for modeling the product of $d_R^{e_g o}$ and $v^{e_g o}$.
$d_{LR}^o \geq 0,\ \forall o \in \mathcal{O}$: representing the distance traveled by the mothership from the launching point x_L^o to the rendezvous point x_R^o at operation o.
$d_{RL}^o \geq 0,\ \forall o \in \mathcal{O} \setminus \lvert \mathcal{O} \rvert$: representing the distance traveled by the mothership from the rendezvous point x_R^o at operation o to the launching point $x_L^{(o+1)}$ at operation $o+1$.
$d_{dest} \geq 0$: distance from the last retrieving point $x_R^{\lvert \mathcal{O} \rvert}$ to the destination $dest$.
$time_D^o \geq 0,\ \forall o \in \mathcal{O}$: maximum time spent by a drone during operation o.
$time_M^o \geq 0,\ \forall o \in \mathcal{O}$: time spent by the mothership to go from the launching point x_L^o to the retrieving point x_R^o of operation o.
$time_M \geq 0$: total time spent by the mothership to go from the origin to the destination (makespan).

Table 2: Decision Variables for AMMDRPG-CO

the auxiliary variables $\nu_{\min}^{e_g}$ and $\nu_{\max}^{e_g}$ of the access and exit points on that segment. Then, for each edge e_g, the absolute value constraint (**??**) can be represented by:

$$|\rho^{e_g} - \lambda^{e_g}| \geq \alpha^{e_g} \iff \begin{cases} \rho^{e_g} - \lambda^{e_g} & = & \nu_{\max}^{e_g} - \nu_{\min}^{e_g}, \\ \nu_{\max}^{e_g} & \leq & 1 - \text{entry}^{e_g}, \\ \nu_{\min}^{e_g} & \leq & \text{entry}^{e_g}, \\ \nu_{\min}^{e_g},\ \nu_{\max}^{e_g} & \geq & 0, \\ \nu_{\max}^{e_g} + \nu_{\min}^{e_g} & \geq & \alpha^{e_g}. \end{cases} \qquad (\alpha\text{-E})$$

The first four inequalities model the standard trick of the linearization of the absolute value. The last constraint ensures that the value of the linear expression of the absolute value is higher than the required

fraction α^{e_g}. Similarly, (??) can be linearized as follows:

$$\sum_{e_g \in E_g} \mu^{e_g} |\rho^{e_g} - \lambda^{e_g}| \mathcal{L}(e_g) \geq \alpha^g \mathcal{L}(g). \iff \begin{cases} \rho^{e_g} - \lambda^{e_g} & = & \nu^{e_g}_{\max} - \nu^{e_g}_{\min}, \\ \nu^{e_g}_{\max} & \leq & 1 - \text{entry}^{e_g}, \\ \nu^{e_g}_{\min} & \leq & \text{entry}^{e_g}, \\ \nu^{e_g}_{\min},\, \nu^{e_g}_{\max} & \geq & 0, \\ p^{e_g} & \leq & \nu^{e_g}_{\max} + \nu^{e_g}_{\min}, \\ p^{e_g} & \leq & \mu^{e_g}, \\ p^{e_g} & \geq & \nu^{e_g}_{\max} + \nu^{e_g}_{\min} + \mu^{e_g} - 1, \\ \sum_{e_g \in E_g} p^{e_g} \mathcal{L}(e_g) & \geq & \alpha^g \mathcal{L}(g), \end{cases} \tag{α-G}$$

where p^{e_g} is the auxiliary variable that represents the product of the binary variable μ^{e_g} and the absolute value difference $|\rho^{e_g} - \lambda^{e_g}|$. The first four inequalities linearize the absolute value expression again. The following three constraints model the product of the expression of the absolute value and the binary variable μ^{e_g}. The last inequality ensures that the fraction of the length of those edges chosen to be crossed must be higher than the fraction of the length of g required to be traversed.

Elimination of subtours

To represent the actual routes of the drones over the target graph, subtours cannot be allowed. The reader may note that the subtour elimination constraints are needed to avoid the presence of disconnected paths on the edges of the graph. To prevent the existence of subtours within each graph $g \in \mathcal{G}$ that the drone must visit, one can include, among others, either the compact formulation that uses the Miller-Tucker-Zemlin constraints (MTZ) or the subtour elimination constraints (SEC).
For the MTZ formulation, we use the continuous variables s^{e_g}, defined in Table ??, that state the order to visit the edge e_g and set the following constraints for each $g \in \mathcal{G}$:

$$s^{e_g} - s^{e'_g} + |E_g| z^{e_g e'_g} \leq |E_g| - 1, \qquad \forall e_g \neq e'_g \in E_g, \tag{MTZ$_1$}$$

$$0 \leq s^{e_g} \leq |E_g| - 1, \qquad \forall e_g \in E_g. \tag{MTZ$_2$}$$

Alternatively, we can also use the family of subtour elimination constraints for each $g \in \mathcal{G}$:

$$\sum_{e_g, e'_g \in S} z_g^{e_g e'_g} \leq |S| - 1, \quad \forall S \subset E_g. \tag{SEC}$$

Since there is an exponential number of SEC constraints, when we implement this formulation, we need to perform a row generation procedure including constraints whenever they are required by a separation oracle. To find SEC inequalities, as usual, we search 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.

3.1. AMMDRPG *with complete overlapping*

To model this problem, we use operations identified with the order in which the different target graphs in the problem are visited. Let us denote by $\mathcal{O}$ the set of operations that the mothership and the fleet of drones have to carry out. These operations are visits to different graphs in $\mathcal{G}$ with the required constraints. An *operation* $o \in \mathcal{O}$ is referred to as the actions in which the mothership launches some drones from a taking-off location, denoted by x^o_L and later it takes them back at a rendezvous location x^o_R. Here, it is important to realize that both locations x^o_L and x^o_R must be determined in the continuous space in which the mothership is assumed to move. Note that $|\mathcal{O}| \leq |\mathcal{G}|$, since it is assumed that, for each operation, at least one drone must be launched. For each operation $o \in \mathcal{O}$, each of the drones launched from the mothership must follow a path starting from and returning to the mothership, while visiting the required edges of one of the graphs $g \in \mathcal{G}$. According to the notation introduced above, we write this generic path in the following form:

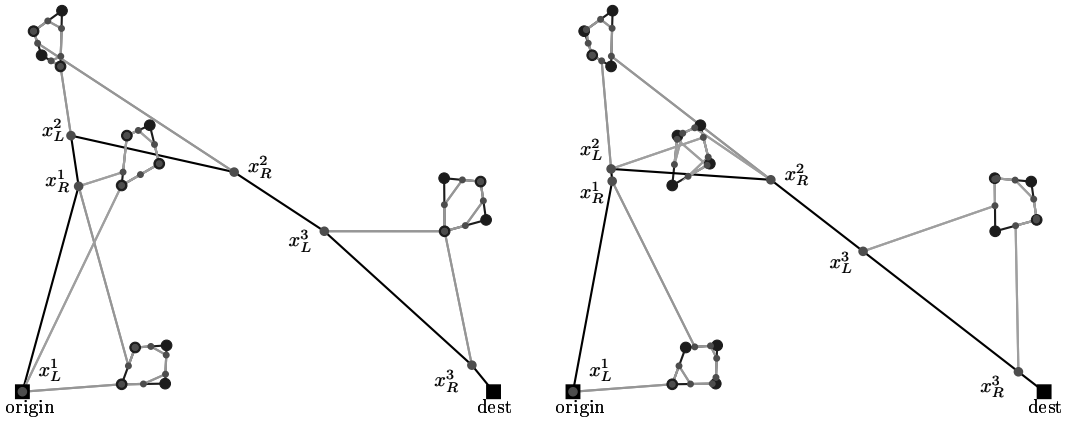


Figure 2: Example of feasible solution (a) and optimal solution (b) for a problem instance with 4 graphs and 2 drones.

$$x_L^o \rightarrow R^{e_g} \rightarrow L^{e_g} \rightarrow \ldots \rightarrow R^{e_g'} \rightarrow L^{e_g'} \rightarrow \ldots \rightarrow R^{e_g''} \rightarrow L^{e_g''} \rightarrow x_R^o.$$

Figure **??** shows an example of the notation over a configuration with four target graphs that have four nodes and four edges. Here, it is supposed that the number of available drones is equal to two. In particular, figure **??**(a) represents a feasible solution to the problem for this configuration. The mothership, whose path is represented in black, begins at its starting point *origin* which coincides with the first launching point x_L^1 where two drones are launched to visit two graphs. There, each drone follows a route (represented by gray paths) that ensures the coverage of one half of the length of each edge of the graph. The smaller gray dots on the visited graphs are the intermediate points R^{e_g} and L^{e_g} used by the drones in their visit to the edges of the different graphs. After finishing the visit to the first two graphs, the drones return to the point x_R^1. The mothership moves from this point to the second launching point x_L^2 from where only one drone is launched to visit the third graph. Once this graph has been visited, the drone returns to the mothership at the rendezvous point x_R^2. Finally, the mothership moves to the point x_L^3 from where one drone is launched for the last visit to the fourth graph. Then, the drone is retrieved by the mothership at the point x_R^3 and then the mothership ends its route at the destination point *dest*. Figure **??**(b) represents the optimal solution for the same instance of the problem. We can observe that, in this case, from the first launching point x_L^1 only one drone is launched, while from the second x_L^2 two drones are launched to visit the second and the third graph. The different position in space of this latter point, with respect to the feasible solution reported in figure **??**(a) whose makespan is 158.36, ensures that the makespan of the optimal solution equals 152.39, which is shorter.
To include the definition of these paths in our mathematical programming formulation, we need to make decisions to choose:

- The optimal assignment of drones for visiting graphs in a given operation o.
- The order to visit the edges of each graph in its corresponding operation.

Drone Constraints
We model the route that the drone follows by using the binary variables $u^{e_g o}$, $z^{e_g e_g'}$ and $v^{e_g o}$ defined in Table **??**.

$$\sum_{g \in \mathcal{G}} \sum_{e_g \in E_g} u^{e_g o} \leq |\mathcal{D}|, \qquad \forall o \in \mathcal{O}, \tag{1}$$

$$\sum_{g \in \mathcal{G}} \sum_{e_g \in E_g} v^{e_g o} \leq |\mathcal{D}|, \qquad \forall o \in \mathcal{O}, \tag{2}$$

$$\sum_{e_g \in E_g} \sum_{o \in \mathcal{O}} u^{e_g o} = 1, \qquad \forall g \in \mathcal{G}, \tag{3}$$

$$\sum_{e_g \in E_g} \sum_{o \in \mathcal{O}} v^{e_g o} = 1, \qquad \forall g \in \mathcal{G}, \tag{4}$$

$$\sum_{e_g \in E_g} u^{e_g o} = \sum_{e_g \in E_g} v^{e_g o}, \qquad \forall g \in \mathcal{G}, \forall o \in \mathcal{O}, \tag{5}$$

$$\sum_{o \in \mathcal{O}} u^{e_g o} + \sum_{e'_g \in E_g} z_g^{e'_g e_g} = \mu^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}, \tag{6}$$

$$\sum_{o \in \mathcal{O}} v^{e_g o} + \sum_{e'_g \in E_g} z_g^{e_g e'_g} = \mu^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}. \tag{7}$$

Inequalities (??) and (??) state that it is not possible to use a number of drones larger than the available one at each operation o. Constraints (??) and (??) ensure that each graph is visited at an operation o by a drone. Equations (??) ensure that the action of entering and exiting the graph g occurs in the same operation o. Constraints (??) state that if an edge e of graph g is visited by a drone, one of two alternative situations must occur: either e is the first edge of graph g visited by the drone at operation o, or edge e is visited by the drone after visiting another edge e' of graph g. Similarly, constraints (??) state that if an edge e of graph g is visited by a drone, either e is the last edge of graph g visited by the drone at operation o, or the drone must move to another edge e' of graph g after visiting edge e.

Distance and Time Constraints

The goal of the AMMDRPG-CO is to find a feasible solution that minimizes the makespan. To account for the different distances between the decision variables of the model we need to set the continuous variables $d_L^{e_g o}$, d^{e_g}, $d^{e_g e'_g}$, $d_R^{e_g o}$, d_{origin}, d_{RL}^o, d_{LR}^o and d_{dest} defined in Table ??. This can be done by means of the following constraints:

$$\|x_L^o - R^{e_g}\| \leq d_L^{e_g o}, \qquad \forall e_g \in E_g : g \in \mathcal{G},\ \forall o \in \mathcal{O}, \qquad \text{(Drone DIST}_1\text{-CO)}$$

$$\|R^{e_g} - L^{e_g}\| \leq d^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}, \qquad \text{(Drone DIST}_2\text{-CO)}$$

$$\|R^{e_g} - L^{e'_g}\| \leq d^{e_g e'_g}, \qquad \forall e_g \neq e'_g \in E_g : g \in \mathcal{G}, \qquad \text{(Drone DIST}_3\text{-CO)}$$

$$\|L^{e_g} - x_R^o\| \leq d_R^{e_o}, \qquad \forall e_g \in E_g : g \in \mathcal{G},\ \forall o \in \mathcal{O}. \qquad \text{(Drone DIST}_4\text{-CO)}$$

$$\|origin - x_L^1\| \leq d_{origin}, \qquad \text{(Mothership DIST}_1\text{-CO)}$$

$$\|x_L^o - x_R^o\| \leq d_{LR}^o, \qquad \forall o \in \mathcal{O}, \qquad \text{(Mothership DIST}_2\text{-CO)}$$

$$\|x_R^o - x_L^{o+1}\| \leq d_{RL}^o, \qquad \forall o \in \mathcal{O} : o < |\mathcal{O}|, \qquad \text{(Mothership DIST}_3\text{-CO)}$$

$$\|x_R^{|\mathcal{O}|} - dest\| \leq d_{dest}. \qquad \text{(Mothership DIST}_4\text{-CO)}$$

All variables modeling the distances covered by drones, namely $d_L^{e_g o}$, d^{e_g}, $d^{e'_g e_g}$ and $d_R^{e_g o}$, as well as those modeling the distance traveled by the mothership, namely d_{origin}, d_{LR}^o, d_{RL}^o and d_{dest}, are defined in Table ??.

In order to compute the maximum time spent by a drone to visit a graph $g \in \mathcal{G}$ associated with operation $o, \forall o \in \mathcal{O}$, we introduce the following constraints:

$$time_D^o \geq \frac{1}{v_D}\left(\sum_{e_g \in E_g} u^{e_g o} d_L^{e_g o} + \sum_{e_g, e'_g \in E_g} z^{e_g e'_g} d^{e_g e'_g} + \sum_{e_g \in E_g} \mu^{e_g} d^{e_g} + \sum_{e_g \in E_g} v^{e_g o} d_R^{e_g o}\right) - N_D\left(1 - \sum_{e_g \in E_g} u^{e_g o}\right). \qquad \text{(Time}_D^o\text{)}$$

The first addend within the brackets accounts for the time spent by the drone to go from the launching point x_L^o to the first retrieving point in the graph R^{e_g}. The second addend considers the time consumed by the drone to go from edge e_g to e'_g in graph g. The third one computes the time required to traverse the required edges in g. The fourth one measures the time to travel from the last launching point $L^{e''_g}$ to the retrieving point x_R^o. Note that, in the special case where all edges must be visited, the third sum of the right-hand side of the (??) constraint, reduces to $\sum_{e_g \in E_g} d^{e_g}$ by setting all the μ^{e_g} variables equal to one. The endurance term in constraint (??) ensures that the constraint becomes active only when a graph g is visited during operation o. The reader may observe that the endurance constraint (??) restricts the time spent by the drone to perform operation o to be less than the endurance N_D. Therefore, the constant N_D can be taken as the bigM term in the (??) constraint. Note that, to deal

with the bilinear terms of the (??) constraint, we use McCormick's envelope to linearize them by adding variables $p \geq 0$ representing the products and introducing the following constraints:

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

where m and M are, respectively, the lower and upper bounds of the distance variable d. These bounds will be adjusted for each bilinear term in Section ??.
The constraint (??) defines the time spent by the mothership to go from the launching point x_L^o to the retrieving point x_R^o associated with operation o.

$$time_M^o = \frac{d_{LR}^o}{v_M}, \quad \forall o \in \mathcal{O}. \tag{Time$_M^o$}$$

Thus, the overall time spent by the mothership to move from the origin to the destination (makespan) can be expressed as follows:

$$time_M = \frac{1}{v_M}(d_{origin} + \sum_{o \in \mathcal{O}} d_{LR}^o + \sum_{o \in \mathcal{O}: o < |\mathcal{O}|} d_{RL}^o + d_{dest}). \tag{Time$_M$}$$

Coordination and Endurance Constraints
The coordination between the drones and the mothership must ensure that the maximum time $time_D^o$ spent by a drone to visit a graph g at 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 operation o. To this end, we need to define the following coordination constraint for each operation $o \in \mathcal{O}$:

$$time_D^o \leq time_M^o. \tag{DCW-CO}$$

We can model the time endurance constraint for a particular operation $o \in \mathcal{O}$ by limiting the time traveled by the drone for this operation o:

$$time_D^o \leq N_D. \tag{Endurance-CO}$$

AMMDRPG-Complete Overlapping Formulation
Putting together all the constraints introduced hitherto, the following formulation minimizes the makespan, ensuring the coordination with the fleet of drones while guaranteeing the required coverage of the target graphs.

$$\begin{aligned} \min \quad & time_M && \text{(AMMDRPG-Complete Overlapping)} \\ \text{s.t.} \quad & (??) \text{ or } (??), \\ & (??) - (??) \text{ or } (??), \\ & (??) - (??), \\ & (??) - (??), \\ & (??) - (??), \\ & (??), (??), (??), \\ & (??), (??). \end{aligned}$$

The objective function accounts for the makespan. Constraints (??)-(??) model the route followed by the drones, (??) - (??) or (??) ensure that the displacement of a drone assigned to the target graph $g \in \mathcal{G}$ is a route, (??) or (??) define what is required in each visit to a target graph. Constraints (??)-(??) set the variables $d_L^{e_g o}$, d^{e_g}, $d^{e_g e'_g}$, $d_R^{e_g o}$. The mothership distances d_{RL}^o and d_{LR}^o, are defined by means of constraints (??)-(??). Constraints (??), (??) and (??) define the times traveled by the drones and the mothership. Finally, constraints (??)-(??) guarantee that the coordination and drone endurance are satisfied.

Binary and Integer Decision Variables
$\mu^{e_g} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G})$: equal to 1 if edge e of graph g (or a portion of it) is visited by the drone, 0 otherwise.
$\text{entry}^{e_g} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G})$: auxiliary binary variable used for linearizing expressions.
$u^{e_g t} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall t \in \mathcal{T}$: equal to 1 if the visit of graph g starts in stage t from edge e_g, 0 otherwise.
$z^{e_g e_g'} \in \{0,1\},\ \forall e_g, e_g' \in E_g\ (g \in \mathcal{G})$: equal to 1 if the drone goes from e_g to e_g', 0 otherwise.
$\gamma^{gt} \in \{0,1\},\ \forall g \in \mathcal{G},\ \forall t \in \mathcal{T}$: equal to 1 if the operation of visiting graph g continues when stage t occurs, 0 otherwise.
$v^{e_g t} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall t \in \mathcal{T}$: equal to 1 if the the visit of graph g ends in stage t on edge e_g, 0 otherwise.
$y^t_{LL} \in \{0,1\},\ \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert$: equal to 1 if the mothership moves from a launching point to a launching point between stage t and stage $t+1$, 0 otherwise.
$y^t_{LR} \in \{0,1\},\ \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert$: equal to 1 if the mothership moves from a launching point to a retrieving point between stage t and stage $t+1$, 0 otherwise.
$y^t_{RL} \in \{0,1\},\ \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert$: equal to 1 if the mothership moves from a retrieving point to a launching point between stage t and stage $t+1$, 0 otherwise.
$y^t_{RR} \in \{0,1\},\ \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert$: equal to 1 if the mothership moves from a retrieving point to a retrieving point between stage t and stage $t+1$, 0 otherwise.
$\mathcal{K}(t) \in \{0,1,2,\ldots,\lvert\mathcal{D}\rvert\},\ \forall t \in \mathcal{T}$: integer non-negative variable representing the number of available drones at stage t.
Continuous Decision Variables
$s^{e_g} \in [0, \lvert E_g\rvert - 1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: continuous non-negative variable representing the order of visit of the edge e of graph g.
$x^t_L \in \mathbb{R}^2,\ \forall t \in \mathcal{T}$: coordinates representing the launching point visited by the mothership at stage t.
$x^t_R \in \mathbb{R}^2,\ \forall t \in \mathcal{T}$: coordinates representing the retrieving point visited by the mothership at stage t.
$R^{e_g} \in \mathbb{R}^2,\ \forall e_g \in E_g\ (g \in \mathcal{G})$: coordinates representing the entry point on edge e_g of graph g.
$L^{e_g} \in \mathbb{R}^2,\ \forall e_g \in E_g\ (g \in \mathcal{G})$: coordinates representing the exit point on edge e_g of graph g.
$d_L^{e_g t} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall t \in \mathcal{T}$: representing the distance traveled by the drone from the launching point x^t_L on the mothership at stage t to the first visiting point R^{e_g} on e_g.
$d^{e_g} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G})$: representing the distance traveled by the drone from the rendezvous point R^{e_g} to the launching point L^{e_g} on e_g.
$d^{e_g e_g'} \geq 0,\ \forall e_g, e_g' \in E_g\ (g \in \mathcal{G})$: representing the distance traveled by the drone from the launching point L^{e_g} on e_g to the rendezvous point $R^{e_g'}$ on e_g'.
$d_R^{e_g t} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall t \in \mathcal{T}$: representing the distance traveled by the drone from the last visiting point L^{e_g} on e_g to the rendezvous point x^o_R on the mothership at stage t.
$d_{origin} \geq 0$: distance from the origin *origin* to the first launching point x^1_L.
$d^t_{LL} \geq 0,\ \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert$: distance from the launching point x^t_L to the launching point x^{t+1}_L.
$d^t_{LR} \geq 0,\ \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert$: distance from the launching point x^t_L to the retrieving point x^{t+1}_R.
$d^t_{RL} \geq 0,\ \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert$: distance from the retrieving point x^t_R to the launching point x^{t+1}_L.
$d^t_{RR} \geq 0,\ \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert$: distance from the retrieving point x^t_R to the retrieving point x^{t+1}_R.
$d_{dest} \geq 0$: distance from the last retrieving point $x^{\lvert\mathcal{T}\rvert}_R$ to the destination *dest*.
$d^g_{LR} \geq 0,\ \forall g \in \mathcal{G}$: representing the distance traveled by the mothership from the launching point x^t_L to the rendezvous point $x^{t'}_R$ associated with graph g for some $t, t' \in \mathcal{T}$.
$time^g_M \geq 0,\ \forall g \in \mathcal{G}$: time spent by the mothership while graph g is visited by a drone.
$time^g_D \geq 0,\ \forall g \in \mathcal{G}$: time spent by a drone to visit graph g.
$time_M \geq 0$: total time spent by the mothership to go from the origin to the destination (makespan).

Table 3: Decision Variables for AMMDRPG-PO

3.2. The AMMDRPG with partial overlapping

In the AMMDRPG-CO version of the problem, we assume that every drone is launched and retrieved in the same operation. In this subsection, we show how this assumption can be relaxed. We consider a variant of the model presented in Section **??**, in which we assume that the mothership can retrieve one drone in a different phase from that in which it has been launched. That is, the mothership can move to another point to launch a new drone without having retrieved all the drones that were launched previously. In the following formulation, we use the concept of *stage* to refer to the action of launching or receiving a drone by the mothership. Each graph must be visited by a drone so that each operation gives rise to two stages: one when the drone is launched and another once the same drone has been retrieved by the mothership. We denote by $\mathcal{T}$ the set of stages. It is clear that $|\mathcal{T}| = 2|\mathcal{G}|$. Using the concept of stage we can substitute the set of operations with the set of stages to model the coordination between drones and mothership in the partial overlapping version of the problem. Indeed, in this case, differently from the complete overlapping version of the problem, the launch of a drone it is not necessarily followed

by its retrieval but, for example, by the launch of a different drone to visit another target graph, as shown in Figure ??. We can notice that when the fleet of drones consists of only one drone, the two versions of the problem coincide. Table ?? summarizes all the variables used in our formulation for the AMMDRPG-PO model.

Drone Constraints

Similarly to the complete overlapping version of the problem, we model the route followed by the drone by using the binary variables $u^{e_g t}$, $v^{e_g t}$ and $z^{e_g e'_g}$. However, in this case, the variables $u^{e_g t}$ and $v^{e_g t}$ are associated with stage t and because of the problem assumptions, we need to introduce the additional binary variables γ^{gt} . Thus, the following constraints model the route followed by the drone while it is operating in a graph $g \in \mathcal{G}$:

$$\sum_{t\in\mathcal{T}}\sum_{e_g\in E_g} u^{e_g t} = 1, \qquad \forall g \in \mathcal{G}, \tag{8}$$

$$\sum_{t\in\mathcal{T}}\sum_{e_g\in E_g} v^{e_g t} = 1, \qquad \forall g \in \mathcal{G}, \tag{9}$$

$$\sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t} \leq \mathcal{K}(t), \qquad \forall t \in \mathcal{T}, \tag{10}$$

$$\sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} (u^{e_g t} + v^{e_g t}) \leq 1, \qquad \forall t \in \mathcal{T}, \tag{11}$$

$$\sum_{e_g\in E_g} u^{e_g t} \leq \sum_{e_g\in E_g}\sum_{t'\in\mathcal{T}:t'>t} v^{e_g t'}, \qquad \forall g \in \mathcal{G},\ \forall t \in \mathcal{T}, \tag{12}$$

$$\sum_{t\in\mathcal{T}} u^{e_g t} + \sum_{e'_g\in E_g} z^{e'_g e_g} = \mu^{e_g}, \qquad \forall e_g \in E_g :\ g \in \mathcal{G}, \tag{13}$$

$$\sum_{t\in\mathcal{T}} v^{e_g t} + \sum_{e'_g\in E_g} z^{e_g e'_g} = \mu^{e_g}, \qquad \forall e_g \in E_g :\ g \in \mathcal{G}, \tag{14}$$

$$\gamma^{gt} \geq \sum_{e_g\in E_g} u^{e_g t}, \qquad \forall g \in \mathcal{G},\ \forall t \in \mathcal{T}, \tag{15}$$

$$\gamma^{g(t+1)} \geq \gamma^{gt} - \sum_{e_g\in E_g} v^{e_g (t+1)}, \qquad \forall g \in \mathcal{G},\ \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{16}$$

$$\sum_{t'\in\mathcal{T}:t'<t} \gamma^{gt'} \leq (t-1)(1 - \sum_{e_g\in E_g} u^{e_g t}), \qquad \forall g \in \mathcal{G},\ \forall t \in \mathcal{T}, \tag{17}$$

$$\sum_{t'\in\mathcal{T}:t'\geq t} \gamma^{gt'} \leq (|\mathcal{T}| - t + 1)\,(1 - \sum_{e_g\in E_g} v^{e_g t}), \qquad \forall g \in \mathcal{G},\ \forall t \in \mathcal{T}, \tag{18}$$

$$\mathcal{K}(1) = |\mathcal{D}|, \tag{19}$$

$$\mathcal{K}(t+1) = \mathcal{K}(t) + \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g t} - \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|. \tag{20}$$

Constraints (??) and (??) ensure that a launching and a retrieving point are associated with each graph g. Constraints (??) allow the mothership to launch a drone in stage t only if a drone is available when stage t occurs. Constraints (??) guarantee that a launching or a retrieving occurs in each stage $t \in \mathcal{T}$. Constraints (??) indicate that the retrieving stage associated with graph g happens after the launching stage associated with the same graph g. Equations (??) state that if an edge e of graph g is visited by a drone, either e is the first edge of graph g visited by the drone at stage t, or edge e is visited by the drone after visiting another edge e' of graph g. Similarly, constraints (??) state that if an edge e of graph g is visited by a drone, either e is the last edge of graph g visited by the drone at stage t, or the drone must move to another edge e' of graph g after visiting edge e. Constraints (??) ensure that the operation associated with graph g starts when the drone is launched during stage t. Inequalities (??) state that the drone is still operating in graph g for successive stages until it is retrieved in stage t. Constraints (??) ensure that the drone is not operating in g until the stage of launching occurs. Constraints (??) guarantee that the drone finishes operating in graph g when the retrieving stage happens. Finally, constraints (??) and (??) model the number of available drones at stage t.

Mothership Constraints
This subsection models all possible sequences of stages in terms of launching and retrieving that can be followed by the mothership: launching-launching, launching-retrieving, retrieving-launching and retrieving-retrieving.

$$y_{LL}^1 + y_{LR}^1 = 1, \tag{21}$$
$$y_{LL}^{t+1} + y_{LR}^{t+1} \geq y_{RL}^t + y_{LL}^t, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{22}$$
$$y_{RR}^{t+1} + y_{RL}^{t+1} \geq y_{LR}^t + y_{RR}^t, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{23}$$
$$y_{LR}^{|\mathcal{T}|-1} + y_{RR}^{|\mathcal{T}|-1} = 1. \tag{24}$$

Constraints (??) state that at stage 1 the mothership must depart from the launching point x_L^1. Constraints (??) (resp. (??)) ensure that if the mothership goes to the launching (resp. retrieving) point x_L^{t+1} (resp. x_R^{t+1}) then in the next stage it must depart from x_L^{t+1} (resp. x_R^{t+1}). Constraint (??) guarantees that the path followed by the mothership finishes in the retrieving point $x_R^{|\mathcal{T}|}$.

Distance and Time Constraints
This subsection considers the second-order cone constraints that model the distances covered by the drones and the mothership:

$$\|x_L^t - R^{e_g}\| \leq d_L^{e_g t}, \qquad \forall e_g \in E_g : g \in \mathcal{G},\ \forall t \in \mathcal{T}, \qquad \text{(Drone DIST}_1\text{)}$$
$$\|R^{e_g} - L^{e_g}\| \leq d^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}, \qquad \text{(Drone DIST}_2\text{)}$$
$$\|R^{e_g} - L^{e'_g}\| \leq d^{e_g e'_g}, \qquad \forall e_g \neq e'_g \in E_g : g \in \mathcal{G}, \qquad \text{(Drone DIST}_3\text{)}$$
$$\|L^{e_g} - x_R^t\| \leq d_R^{e_g t}, \qquad \forall e_g \in E_g : g \in \mathcal{G},\ \forall t \in \mathcal{T}. \qquad \text{(Drone DIST}_4\text{)}$$

$$\|origin - x_L^1\| \leq d_{origin}, \qquad \text{(Mothership DIST}_1\text{)}$$
$$\|x_L^t - x_L^{t+1}\| \leq d_{LL}^t, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Mothership DIST}_2\text{)}$$
$$\|x_L^t - x_R^{t+1}\| \leq d_{LR}^t, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Mothership DIST}_3\text{)}$$
$$\|x_R^t - x_L^{t+1}\| \leq d_{RL}^t, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Mothership DIST}_4\text{)}$$
$$\|x_R^t - x_R^{t+1}\| \leq d_{RR}^t, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Mothership DIST}_5\text{)}$$
$$\|x_R^{|\mathcal{T}|} - dest\| \leq d_{dest}. \qquad \text{(Mothership DIST}_6\text{)}$$

All the variables modeling the distances covered by drones, namely, $d_L^{e_g t}$, d^{e_g}, $d^{e'_g e_g}$ and $d_R^{e_g t}$, as well as those modeling the distance traveled by the mothership, namely, d_{origin}, d_{LL}^t, d_{LR}^t, d_{RL}^t, d_{RR}^t and d_{dest}, are defined in Table ??. The time spent by the drone to perform the operation of visiting graph g is given by:

$$time_D^g = \frac{1}{v_D}\left(\sum_{t\in\mathcal{T}}\sum_{e_g\in E_g} u^{e_g t} d_L^{e_g t} + \sum_{e_g, e'_g \in E_g} z^{e_g e'_g} d^{e_g e'_g} + \sum_{e_g \in E_g} \mu^{e_g} d^{e_g} + \sum_{t\in\mathcal{T}}\sum_{e_g\in E_g} v^{e_g t} d_R^{e_g t}\right). \qquad \text{(Time}_D^g\text{)}$$

The time spent by the mothership while the drone is operating in graph g is given by:

$$time_M^g = \frac{1}{v_M} d_{LR}^g = \frac{1}{v_M} \sum_{t\in\mathcal{T}: t<|\mathcal{T}|} (\|x_L^t - x_L^{t+1}\| y_{LL}^t + \|x_L^t - x_R^{t+1}\| y_{LR}^t + \|x_R^t - x_L^{t+1}\| y_{RL}^t + \|x_R^t - x_R^{t+1}\| y_{RR}^t)\gamma^{gt},\ \forall g \in \mathcal{G}. \qquad \text{(Time}_M^g\text{)}$$

Finally, the overall time spent by the mothership (makespan) can be described as follows:

$$time_M = \frac{1}{v_M}\left(d_{origin} + \sum_{t\in\mathcal{T}: t<|\mathcal{T}|} \left(\|x_L^t - x_L^{t+1}\| y_{LL}^t + \|x_L^t - x_R^{t+1}\| y_{LR}^t + \|x_R^t - x_L^{t+1}\| y_{RL}^t + \|x_R^t - x_R^{t+1}\| y_{RR}^t\right) + d_{dest}\right). \qquad \text{(Time}_M\text{)}$$

Coordination and Endurance Constraints

Once having defined the time spent by the drone to visit graph g and the time spent by the mothership while the drone is visiting this graph g, we can model the coordination constraint simply as:

$$time_D^g \leq time_M^g, \quad \forall g \in \mathcal{G}. \tag{DCW-PO}$$

In addition, the time spent by the drone to operate in graph g must not exceed its endurance:

$$time_D^g \leq N_D \tag{Endurance-PO}$$

Linearization Constraints

This subsection is devoted to linearizing the relationship between the decision variables that model the route of the mothership and the drones. The relationship of these variables is given by the the following non-linear expressions:

$$\begin{aligned}
y_{LL}^t &= \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t} \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g (t+1)}, && \forall t \in \mathcal{T} : t < |\mathcal{T}|,\\
y_{LR}^t &= \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t} \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g (t+1)}, && \forall t \in \mathcal{T} : t < |\mathcal{T}|,\\
y_{RL}^t &= \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g t} \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g (t+1)}, && \forall t \in \mathcal{T} : t < |\mathcal{T}|,\\
y_{RR}^t &= \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g t} \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g (t+1)} && \forall t \in \mathcal{T} : t < |\mathcal{T}|.
\end{aligned}$$

The products above can be linearized, respectively, by means of the following constraints:

$$y_{LL}^t \leq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{25}$$

$$y_{LL}^t \leq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g (t+1)}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{26}$$

$$y_{LL}^t \geq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t} + \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g (t+1)} - 1, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{27}$$

$$y_{LR}^t \leq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{28}$$

$$y_{LR}^t \leq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g (t+1)}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{29}$$

$$y_{LR}^t \geq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t} + \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g (t+1)} - 1, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{30}$$

$$y_{RL}^t \leq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g t}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{31}$$

$$y_{RL}^t \leq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g (t+1)}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{32}$$

$$y_{RL}^t \geq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g t} + \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g (t+1)} - 1, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{33}$$

$$y_{RR}^t \leq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g t}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{34}$$

$$y_{RR}^t \leq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g (t+1)}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \tag{35}$$

$$y_{RR}^t \geq \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g t} + \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g (t+1)} - 1, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|. \tag{36}$$

AMMDRPG-Partial Overlapping Formulation

Hence, the formulation of the AMMDRPG with partial overlapping operations is:

$$\begin{aligned} \min \quad & time_M && \text{(AMMDRPG-Partial Overlapping)} \\ \text{s.t.} \quad & (??) \text{ or } (??), \\ & (??) - (??) \text{ or } (??), \\ & (??) - (??), \\ & (??) - (??), \\ & (??) - (??), \\ & (??) - (??), \\ & (??) - (??), \\ & (??), (??), (??), \\ & (??), (??). \end{aligned}$$

3.3. Relationship between problem variants

In this section, we present two results that link the two models presented above. Note that the only difference between the solutions of these models is that, for the partial overlapping case, the mothership can launch a second drone sequentially before retrieving the ones that were launched previously. Figure ?? shows a solution that is not possible for the model with complete overlapping. Indeed, we can see that a first drone is launched at x_L^1 to visit P_1 that is retrieved at x_R^1. However, the mothership has launched another drone at x_L^2 that goes to visit P_2 before having retrieved the first drone. Clearly, this solution does not satisfy the assumptions in the complete overlapping model.

Theorem 3.1. *Let X_{CO} be the feasible set of the* AMMDRPG *with complete overlapping operations and let X_{PO} be the feasible set of the* AMMDRPG *with partial overlapping operations, then:*

$$X_{CO} \subsetneq X_{PO}.$$

Proof. To prove the theorem, first we show that a feasible solution $\bar{\omega} \in X_{CO}$ is also feasible for the AMMDRPG-PO. We can notice that all the discrete decision variables of the AMMDRPG-PO model can be directly obtained once the $\hat{u}^{e_g t}$ and $\hat{v}^{e_g t}$ variables are set via the constraints (??)-(??). Thus, we can limit ourselves to show how their values can be derived from those of $\bar{\omega}$ to obtain a feasible solution $\hat{\omega} \in X_{PO}$. We consider $\bar{u}^{e_g o}$ and $\bar{v}^{e_g o}$ equal to 1. Let $\bar{\mathcal{O}} = \{o \in \mathcal{O} : \bar{u}^{e_g o} = 1\}$. Let $\bar{\mathcal{G}}(o)$ be the set of graphs visited in operation $o \in \bar{\mathcal{O}}$. We can compute for each $o \in \bar{\mathcal{O}}$ the corresponding set $\mathcal{T}(o)$, that is the set of stages defining the launching and retrieving actions that occur in operation o. More in detail, we can identify the first element $t(o)$ of this set as follows:

$$\begin{aligned} t(0) &= 1; \\ t(o+1) &= t(o) + \sum_{g \in \bar{\mathcal{G}}(o)} \sum_{e_g \in E_g} (\bar{u}^{e_g o} + \bar{v}^{e_g o}). \end{aligned}$$

Given its first element $t(o)$, we can split $\mathcal{T}(o)$ into two subsets of indexes $\mathcal{T}_u(o)$ and $\mathcal{T}_v(o)$ as follows:

$$\begin{aligned} \mathcal{T}_u(o) &= \{t \in \mathcal{T} : t(o) \leq t \leq t(o) + |\bar{\mathcal{G}}(o)| - 1\}, \\ \mathcal{T}_v(o) &= \{t \in \mathcal{T} : t(o) \leq t - |\bar{\mathcal{G}}(o)| \leq t(o) + |\bar{\mathcal{G}}(o)| - 1\}. \end{aligned}$$

Since the cardinality of set $\mathcal{T}_u(o)$ is equal to the cardinality of set $\bar{\mathcal{G}}(o)$, we can define a bijective function $\bar{\varphi}_{u(o)} : \mathcal{T}_u(o) \to \bar{\mathcal{G}}(o)$ and similarly we can define a bijective function $\bar{\varphi}_{v(o)} : \mathcal{T}_v(o) \to \bar{\mathcal{G}}(o)$. These functions define an assignment between graphs and stages. Note that any assignment defined by functions $\bar{\varphi}_{u(o)}$ and $\bar{\varphi}_{v(o)}$ is feasible.

By means of these two functions, we can set the values of the $\hat{u}^{e_g t}$ and $\hat{v}^{e_g t}$ variables. Indeed, by resorting to the Graph of the functions $\bar{\varphi}_{u(o)}$ and $\bar{\varphi}_{v(o)}$ we can define respectively the $\hat{u}^{e_g t}$ and $\hat{v}^{e_g t}$ variables that must be equal to 1:

$$\begin{aligned} \hat{u}^{e_g t} = 1, \quad & (t, g) \in \text{Graph}(\bar{\varphi}_{u(o)}) \wedge (\bar{u}^{e_g o} = 1) \\ \hat{v}^{e_g t} = 1, \quad & (t, g) \in \text{Graph}(\bar{\varphi}_{v(o)}) \wedge (\bar{v}^{e_g o} = 1) \end{aligned}$$

The remaining $\hat{u}^{e_g t}$ and $\hat{v}^{e_g t}$ variables are set equal to 0. To show that the binary variables $\hat{u}^{e_g t}$ and $\hat{v}^{e_g t}$ are feasible for the AMMDRPG-PO model, one can easily check that they satisfy constraints (??)-(??). Moreover, it is easy to show that the mothership constraints are also satisfied by the $\hat{y}^t = (\hat{y}^t_{LL}, \hat{y}^t_{LR}, \hat{y}^t_{RL}, \hat{y}^t_{RR})$ variables induced by the $\hat{u}^{e_g t}$ and $\hat{v}^{e_g t}$ variables. As regards continuous variables, they can be directly derived from the setting of the $\hat{x}^t_L$ and $\hat{x}^t_R$ variables that can be obtained as follows:

$$\hat{x}^t_L = \bar{x}^o_L, \quad \forall t \in \mathcal{T}_u(o) : o \in \bar{\mathcal{O}},$$
$$\hat{x}^t_R = \bar{x}^o_R, \quad \forall t \in \mathcal{T}_v(o) : o \in \bar{\mathcal{O}}.$$

We can notice that the distances between two consecutive launching or two consecutive retrieving points are equal to 0 by definition of the $\hat{x}^t_L$ and $\hat{x}^t_R$ variables. Consequently, the time $\widehat{time}^g_M$ spent by the mothership while the drone is visiting graph $g \in \bar{\mathcal{G}}(o) : o \in \bar{\mathcal{O}}$ is equal to $\overline{time}^o_M$.
To complete the proof, it is sufficient to notice that, on the contrary, there exist feasible solutions of the AMMDRPG-PO characterized by partial overlaps between operations as shown, for example, in Figure ??, that are not feasible for the AMMDRPG-CO. □

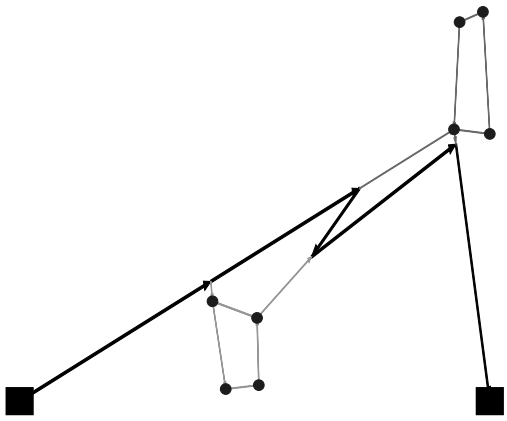


Figure 3: Feasible solution with partial overlapping that is not feasible for the complete overlapping model.

To present our next result, wlog, we restrict ourselves to the degenerate case where graphs reduce to points. The reader may note that it is possible to reduce the visit of graphs to the visit of points by assuming that the drone is stopped in the point which is at the same time as the one required to traverse the edges of the graph. We simplify the proof considering a generic solution between two consecutive target points.

Theorem 3.2. *Let x^1_L, x^2_L (resp. x^1_R, x^2_R) be the launching (resp. rendezvous) points associated with the visit of the target points P_1 and P_2. If there exist two points x_L and x_R verifying*

$$\begin{cases} \dfrac{\|x_L - x_R\|}{v_M} & \leq & \dfrac{\|x_L - P_1\| + \|P_1 - x_R\|}{v_D}, \\ \dfrac{\|x_L - x_R\|}{v_M} & \leq & \dfrac{\|x_L - P_2\| + \|P_2 - x_R\|}{v_D}, \\ \dfrac{\|x_L - x_R\|}{v_M} & \leq & N_D, \\ \|x_L - x_R\| & \leq & \|x^1_L - x^2_L\| + \|x^2_L - x^1_R\| + \|x^1_R - x^2_R\|, \end{cases}$$

then the contribution of this partial route to the optimal objective value will be the same in both models.

Proof. Note that in the configuration considered, the order of visits to the points P_1 and P_2 is fixed and then, the binary variables in the model are fixed in this case. Thus, the only differences that the two models can have are the location of the launching and rendezvous points. Hence, the only constraints that are involved are those related to these points. These are the conditions in the statement: The first two are the (??) inequalities. The third is the (??) constraint and the last ensures that the distance traveled by the mothership in the complete overlapping model is smaller than or equal to the distance assumed in the partial overlapping solution described in the statement. Therefore, the conclusion follows. □

Note that this result states sufficient conditions to obtain the same solution for the two models.

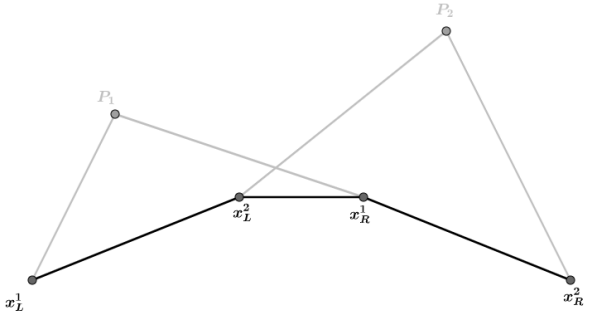


Figure 4: The mothership launches two drones sequentially

4. Strengthening the formulations

In this section, we present some valid inequalities for (??) that reinforce the formulation given in Subsection ??. Moreover, the (??) and (??) constraints have products of binary and continuous variables that, when they are linearized, produce bigM constants that have to be tightened. This section also provides some bounds for these constants whenever possible.

4.1. Valid inequalities for the ??

In this problem, we assume that the fleet has more than one drone since otherwise the problem reduces to the *All Terrain Mothership and Drone routing problem with graphs* that was already studied in [?]. Therefore, if there exists an operation in which more than one drone is launched, the mothership does not need to perform $|\mathcal{G}|$ different operations. Hence, most likely the model does not need to deal with those operations that are numbered at the end. By exploiting this idea, it is possible to concentrate all drone activities on the first operations, avoiding empty operations in $\mathcal{O}$. Let β^o be a binary variable that assumes the value of 1 if all the target graphs are visited when the operation o begins, and zero, otherwise. Note that, if all the graphs have already been visited before operation o then they are also completed before operation $o+1$. Hence, β variables must satisfy the following constraints:

$$\beta^o \leq \beta^{o+1}, \text{ for all } o = 1, \ldots, |\mathcal{G}| - 1. \qquad \text{(Monotonicity)}$$

Let k^o denote the number of graphs that are visited in operation o. This number can be computed using the u variables since $u^{e_g o}$ takes the value of 1 if graph g is visited in operation o. Thus:

$$k^o = \sum_{g \in \mathcal{G}} \sum_{e_g \in E_g} u^{e_g o}.$$

Hence, if β^o equals one, the entire set of graphs in $\mathcal{G}$ must have been visited before operation o:

$$\sum_{o'=1}^{o-1} k^{o'} \geq |\mathcal{G}|\beta^o, \quad \forall o \in \mathcal{O}, \qquad \text{(VI-1)}$$

where $|\mathcal{G}|$ denotes the number of graphs of $\mathcal{G}$.
To reduce the space of feasible solutions, we can assume without loss of generality that it is not permitted to have an operation o without any visiting graphs if some of them are still to be visited. This can be enforced by the following constraints:

$$k^o \geq 1 - \beta^o, \quad \forall o \in \mathcal{O}. \qquad \text{(VI-2)}$$

The model that we have proposed includes bigM constants. We have defined different bigM constants throughout this work. To strengthen the formulations, we provide tight upper and lower bounds for those constants. In this section, we present some results that adjust them for each of the models. The reader may note that the same bounds can be used for both models. Therefore, wlog, we focus on the bigM constants that appear in (??).

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 graph $g \in \mathcal{G}$
To linearize the first addend in (??), we define the auxiliary non-negative continuous variables $p_L^{e_g o}$ (resp. $p_R^{e_g o}$) and we model the product by including the following constraints:

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

Note that, among all graph nodes and the origin and destination points, it is possible to identify the pair of points at the maximum distance. From this pair of points, we can build a circle whose diameter is the segment joining them. Hence, because we are minimizing the distance traveled by the mothership, every launching or rendezvous point is inside this circle and the best upper bound $M_L^{e_g o}$ or $M_R^{e_g o}$ can be described as:

$$M_R^{e_g o} = \max_{\{v \in V_g \cup \{\text{origin,dest}\}, v' \in V_{g'} \cup \{\text{origin,dest}\}: g, g' \in \mathcal{G}\}} \|v - v'\| = M_L^{e_g o}.$$

On the other hand, the minimum distance in this case can be zero. This bound is attainable whenever the launching or rendezvous points of the mothership are the same that the rendezvous or launching point on the target graph $g \in \mathcal{G}$.

Bounds on the bigM constants for the distance from the launching to the rendezvous points on the target graph $g \in \mathcal{G}$.
When the drone visits a graph g, it has to go from one edge e_g to another edge e'_g depending on the order given by $z^{e_g e'_g}$. This fact produces a product of variables linearized by the following constraints:

$$\begin{aligned} & p^{e_g e'_g} \leq M^{e_g e'_g} z^{e_g e'_g}, \\ & p^{e_g e'_g} \leq d^{e_g e'_g}, \\ & p^{e_g e'_g} \geq m^{e_g e'_g} d^{e_g e'_g}, \\ & p^{e_g e'_g} \geq d^{e_g e'_g} - M^{e_g e'_g}(1 - z^{e_g e'_g}). \end{aligned}$$

Since we are taking into account the distance between two edges $e_g = (B^{e_g}, C^{e_g})$, $e'_g = (B^{e'_g}, C^{e'_g}) \in E_g$, the maximum distance between their vertices gives us the upper bound:

$$M^{e_g e'_g} = \max\{\|B^{e_g} - C^{e'_g}\|, \|B^{e_g} - B^{e'_g}\|, \|C^{e_g} - B^{e'_g}\|, \|C^{e_g} - C^{e'_g}\|\}.$$

We observe that the minimum distance between edges $m^{e_g e'_g}$ can easily be obtained computing the minimum distance between two edges, which results in a simple second-order cone program.

5. A Matheuristic for the Mothership-Drone Routing Problem with Graphs

This section is devoted to presenting our matheuristic approach to address the solution of the AMMDRPG. Our motivation comes from the fact that the exact solution of the models presented in Section ?? is highly time demanding. Alternatively, the matheuristic provides a good quality solution in limited computing times.
The basic idea of the algorithm is to determine the route that a drone should perform to visit each graph $g \in \mathcal{G}$, and thus the entry and exit points L^{e_g} and $R^{e'_g}$ for each graph. Sequentially, a clustering procedure on the target graphs is applied to compute the route of the mothership via their reference points and the origin/destination points. The clustering procedure is based on a random selection of the initial target graphs and for this reason it is repeated a number of times to consider different cluster structures. At each iteration, the new clusters are evaluated by computing the cost of the route visiting their reference points and the origin/destination points. The route of minimum length, computed on the reference points of the cluster generated by this iterative procedure, is used to set the values of the binary variables $u^{e_g o}$ and $v^{e_g o}$, that determines the order of visits to the graphs. Finally, these variables are provided as an initial partial solution to the AMMDRPG-CO model to produce a complete feasible

Algorithm 1 Matheuristic algorithm for **??**

Data: $\mathcal{G}$, $|\mathcal{D}|$, N_D, v_D, $maxit$ (maximum number of iterations to perform the clustering procedure), $maxseed$ (maximum number of the clustering procedure repetitions)

STEP 1 (First entry and last exit points for each target graph)

For each target graph $g \in \mathcal{G}$, compute the route:
$L^{e_g} \leftarrow$ entry point on g closest to the origin
$R^{e_g'} \leftarrow$ exit point from g closest to the origin
$\mathcal{L}(e_g, e_g') \leftarrow$ route length

STEP 2 (Clustering procedure)

$it \leftarrow 1$
$nit \leftarrow 1$
For each target graph $g \in \mathcal{G}$: $K_g \leftarrow g$ ▷ one cluster for each target graph
while $nit < maxit$ **do**
  Select randomly two clusters K_i and K_j $(i < j)$
  if $|K_i \cup K_j| < |\mathcal{D}|$ **then**
    Search for point P satisfying the following endurance constraint:

$$\frac{d(P, R^{e_g}) + \mathcal{L}(e_g, e_g') + d(L^{e_g'}, P)}{v_D} \leq N_D, \quad \forall R^{e_g}, L^{e_g'} \in K_i, \ K_j. \tag{37}$$

    if $P \ \exists$ **then**
      $K_i \leftarrow K_i \cup K_j$
    end
  end
  $nit \leftarrow nit + 1$
end
$\mathcal{K} \leftarrow$ set of clusters

STEP 3 (Computation of Reference Points)

For each cluster $K_i \in \mathcal{K}$ compute a reference point P_i by solving the following minimization problem:

$$\min \sum_{K_i \in \mathcal{K}} (\|P_i - origin\| + \|P_i - dest\|) + \sum_{g \in K_i : K_i \in \mathcal{K}} (\|P_i - R^{e_g}\| + \|P_i - L^{e_g'}\|) + \sum_{K_i, K_j \in \mathcal{K} : i \neq j} \|P_i - P_j\|$$

subject to (**??**).

STEP 4 (Order of visits to the graphs: route via the reference points and the origin/destination points)

Compute the TSP of the mothership among the reference points P_i of the clusters

$\mathcal{L}(TSP) \leftarrow$ TSP length [This update is performed only if $\mathcal{L}(TSP)$ decreases with respect to the previous iteration $it - 1$]
$it \leftarrow it + 1$
if $it < maxseed$ **then**
  go to STEP 2
else
  go to STEP 5
end

STEP 5 (Solution of the AMMDRPG model by fixing an initial partial solution)

Set the initial values of the binary variables $u^{e_g o}$ and $v^{e_g o}$ and solve the model AMMDRPG to obtain a feasible solution.

Result: Feasible solution for **??**

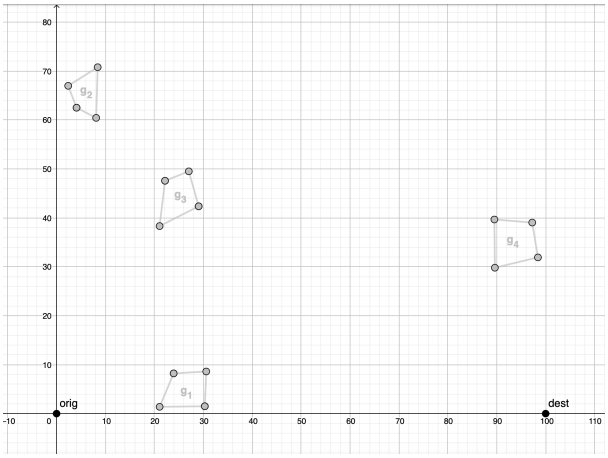


Figure 5: Illustrative example

solution.
Algorithm ?? reports the pseudocode of this algorithm.
Figure ?? shows an illustrative example consisting of four target planar graphs (g_1, g_2, g_3 and g_4) to be visited. We assume that their visits must be performed by a fleet of two drones supported by a mothership whose path starts from the origin $(0, 0)$ and ends at the destination point $(100, 0)$.
Figure ?? illustrates a zoom in on each single target graph, showing the tours generated by STEP 1 of the matheuristic procedure. A pair of points representing retrieving and launching points, together with an arrow pointing the direction followed by the drone according to the order in which the edges are visited, are depicted on each edge.
By applying STEP 2 to this illustrative example, we obtain three clusters, as shown in Figure ??(a). One cluster contains graphs g_1 and g_3 (in black), while graph g_2 and g_4 represent distinct clusters. The computation of the reference points of these clusters, according to STEP 3, produces the points P_1, P_2 and P_3, as shown in Figure ??(b).
STEP 4 of the matheuristic procedure generates the tour of the mothership along the origin point, P_1, P_2, P_3 and the destination point, as shown in Figure ??(a). This tour also returns the order in which the clusters are visited (and thus, also the order of visits to the target graphs) and this permits us to set the values of the variables $u^{e_g o}$ and $v^{e_g o}$ of the AMMDRPG model.
By providing the initial partial solution obtained from the values of the variables $u^{e_g o}$ and $v^{e_g o}$, STEP 5 solves the AMMDRPG model and returns the final feasible solution shown in Figure ??(b). From it, we can observe that the sequence of visits to the target graphs does not change with respect to that provided by STEP 4. The fleet of two drones first visits the graphs g_1 and g_3 starting from the launching point x_L^1. Then, both drones are retrieved by the mothership at point x_R^1. The mothership moves to the point x_L^2 where one drone is launched to visit graph g_2. Then the mothership reaches the point x_R^2 to retrieve the drone and from the same point it launches the other drone to visit graph g_4. Then, this drone is retrieved by the mothership at point x_R^3 before moving to the final destination point.
Focusing on each single target graph, Figure ?? shows the zoom in on the tours followed by the drones. For example, Figure ??[a] reports that performed by the drone that visits the graph g_1 (the gray segments) and the drone that visits the graph g_3 (the dotted gray segments). Both drones start from the mothership at point x_L^1 that is the *origin*. One drone first visits the segment $\overline{R_1^1 L_1^1}$ of the graph g_1, while the other starts the visit to graph g_3 by traversing the segment $\overline{R_1^3 L_1^3}$. From point L_1^1 the first drone moves to the second visited edge of the graph g_1 by traversing the segment $\overline{R_1^2 L_1^2}$. Then, it moves to the third visited edge of graph g_1, by flying over the segment $\overline{R_1^3 L_1^3}$. From point R_1^4 the drone starts the visit to the last edge of graph g_1 up to point L_1^4. Finally, the drone leaves graph g_1 at this latter point and is retrieved by the mothership at point x_R^1. Similarly, the second drone, that visits graph g_3, after traversing segment $\overline{R_3^1 L_3^1}$, moves to the second visited edge of the same graph and traverses segment $\overline{R_3^2 L_3^2}$. Then, it flies to the third visited edge, by traversing segment $\overline{R_3^3 L_3^3}$. Finally it moves to the last visited edge of the graph g_3, by flying over segment $\overline{R_3^4 L_3^4}$. The drone leaves graph g_3 at point L_3^4 and reaches the mothership at point x_R^1. Note that in this example the drones do not visit the full 100% of each graph edge, but only half of each of them.

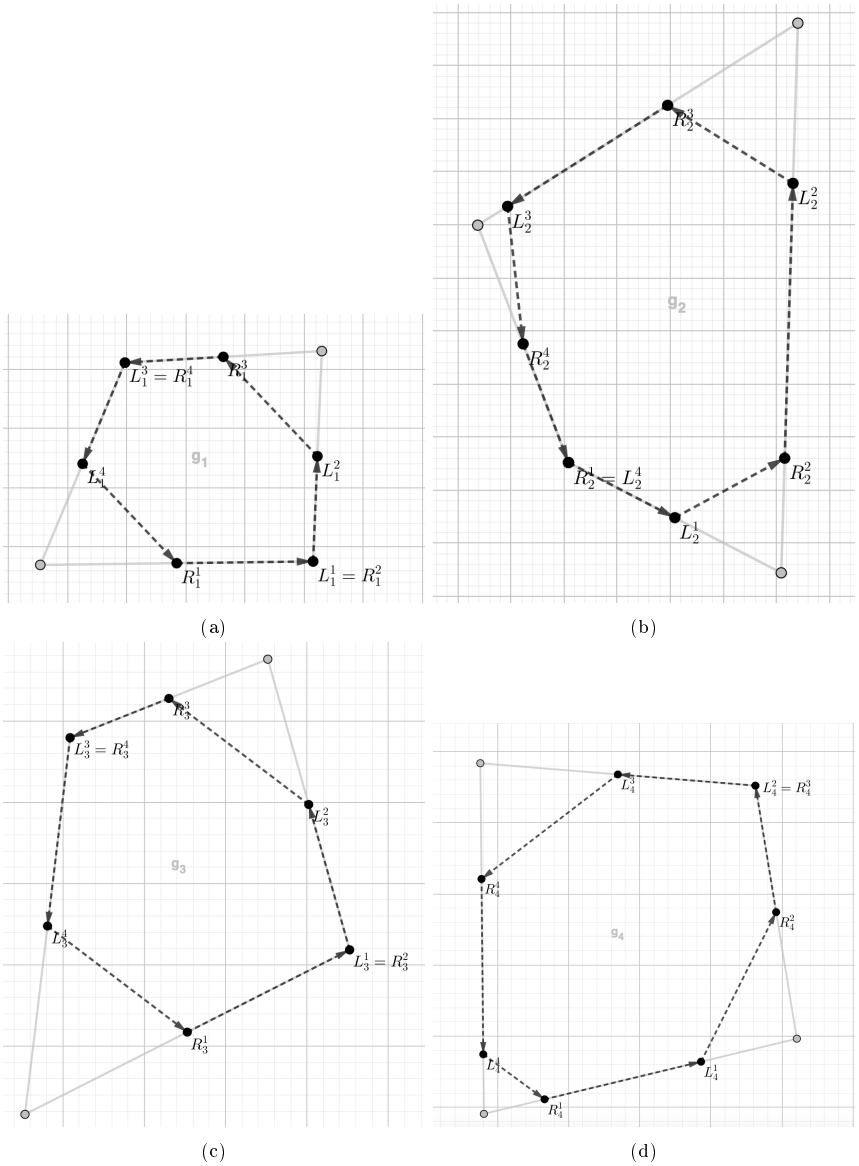


Figure 6: STEP 1 for the illustrative example

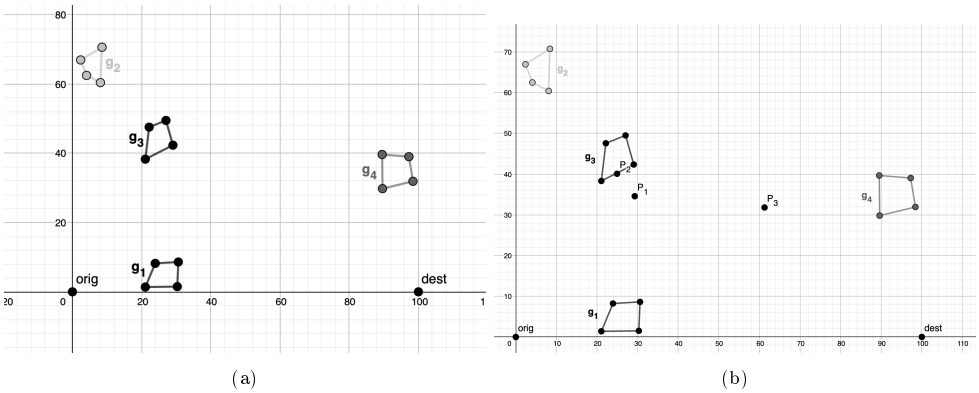
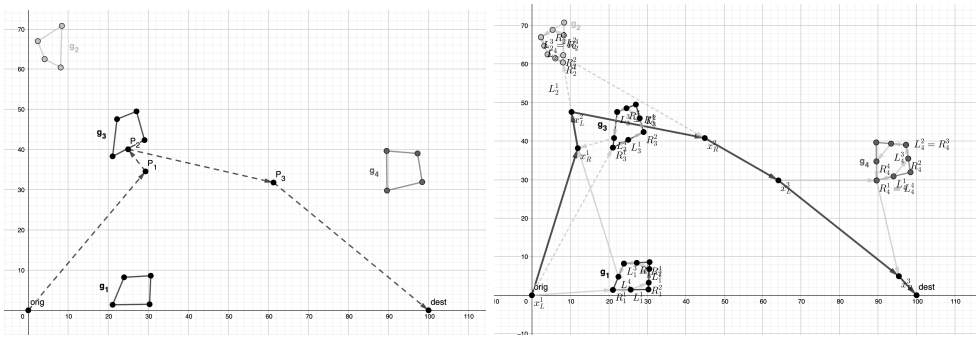


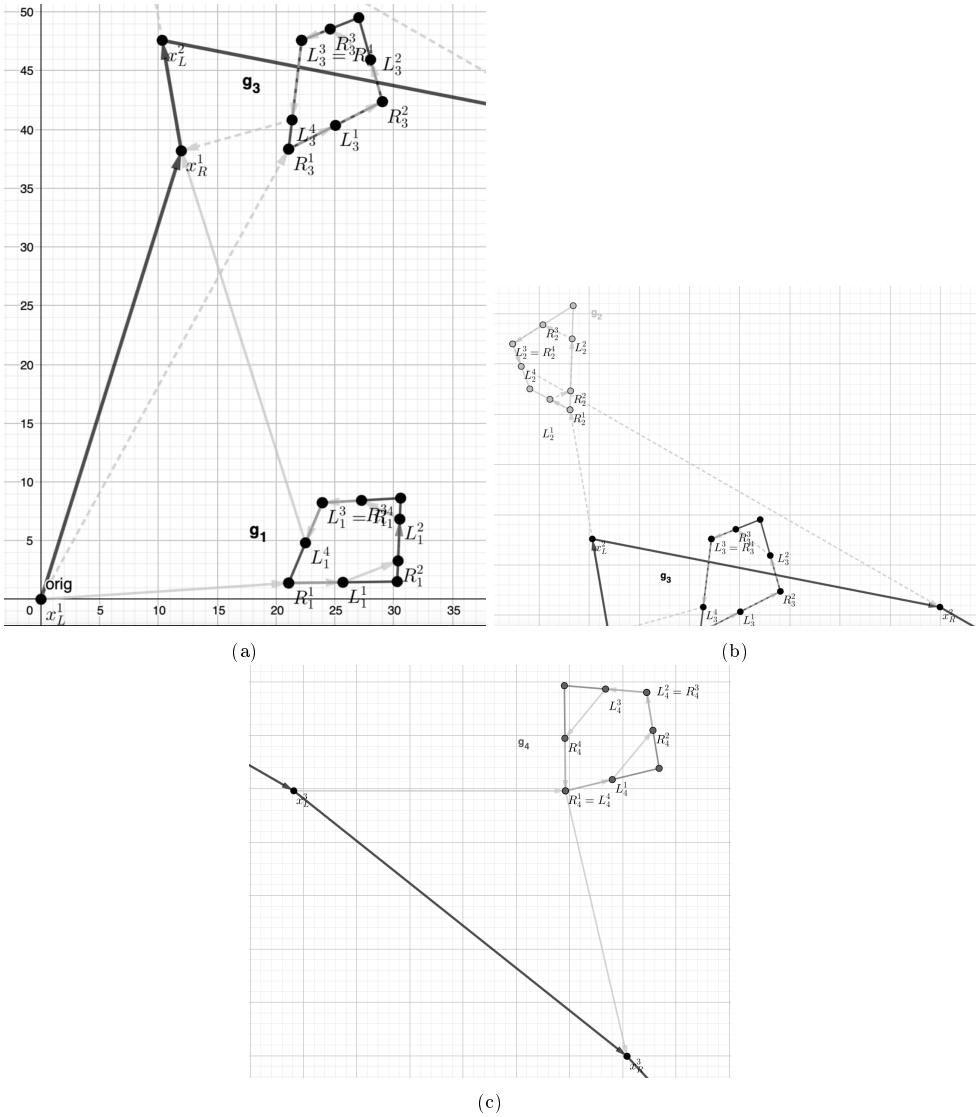
Figure 7: (a) STEP 2, (b) STEP 3 for the illustrative example

The reader may notice that the algorithm above can also be used to generate solutions for the partial overlapping model presented in Section **??** since any solution of the AMMDRPG-CO model is also feasible for the AMMDRPG-PO one as shown in Theorem **??**.



(a) (b)

Figure 8: (a) STEP 4, (b) STEP 5 for the illustrative example



(a) (b)

(c)

Figure 9: Zoom on the tour on each target graph provided by STEP 5

6. Experimental results

In this section, we discuss the experimental results obtained testing the formulations presented in Section ?? and the matheuristic procedure proposed in Section ?? on a testbed of instances. In particular, we consider instances like the ones used in [?], where the targets to be visited by the drones are represented by grid graphs. We generated one set of 5 instances with 5 graphs and another set of 5 instances with

10 graphs. More precisely, each instance is composed of 20% graphs with 4 nodes, 20% graphs of 6 nodes, 20% graphs of 8 nodes, 20% graphs of 10 nodes and and 20% graphs with 12 nodes. Moreover, we assume that the velocity of the drones is twice that of the mothership and that a random fraction of each target graph, or of each of its edges, must be visited by the fleet of drones. These fractions are uniformly randomly sampled in the interval $(0, 1)$.

We consider in our experiments that the number of drones varies between 1 and 3 and that the drone endurance (expressed as the maximum time that the drone can operate when it is fully recharged) ranges between 20 and 60. Note that the case of a single drone is also included in our experiments to compare the results and complexity of using one or more than one drone. The interested reader is referred to [**?**] to analyze the complexity in terms of gap of the model with a single drone. Table **??** reports a summary of the characteristics of our instances.

Table 4: Instance parameter values

$\lvert\mathcal{G}\rvert$	(5,10)
$\lvert\mathcal{D}\rvert$	(1,2,3)
$\lvert V_g\rvert$	(4,6,8,10,12)
N_D	(20,30,40,50,60)
fraction target (edge)	uniformly randomly sampled in $(0, 1)$.

We coded the matheuristic and the exact resolution of the model in Python 3.8.10. The mathematical programming formulation was implemented in Gurobi 9.1.2. All the tests were run on a machine AMD® Epyc 7402p with 24-core processor × 8. Table **??** reports the results obtained solving both variants of the AMMDRPG model on the instances previously described, by adopting the commercial solver Gurobi. We consider the exact solution both providing and not providing an initial solution computed by the matheuristic described in Section **??**. More precisely, the first row of Table **??** indicates the model variant and the second row reports the number of target graphs to be visited by the fleet of drones (5 or 10). From the third row, we split each column into three sub-columns. The first three sub-columns report respectively, the endurance of the drones, the size of the fleet of drones and an indication of the visit of a fraction of each edge (e) or a fraction of each target graph (g). From sub-column 4 to sub-column 15, we report, for each combination of the listed parameters characterizing the instances, respectively, the average gap without initialization by the solution provided by the matheuristic (wi), the average gap with initialization by the solution obtained by the matheuristic (i) and the solution time, in seconds, of the matheuristic (TimeH). The time limit of these experiments is set equal to 1 hour.
We can observe that the value of the average gap ranges between a minimum of 0.58 and a maximum of 1. This shows that the model is hard to solve even with small size instances. Moreover, we can see that for the complete overlapping version of the model, in most cases, the average gap associated with the variant of the model consisting of visiting a given fraction of each edge, is higher than that associated with the variant obligating to visit a given fraction of each target graph. Another thing that we can observe is that the average gap increases with the number of target graphs for both problem variants.
Moreover, the reader may note that the partial overlapping version of the problem is harder to solve than the complete overlapping version by looking at the values of the average gap. This is an expected behavior due to the fact that the feasible region of the partial overlapping variant contains the one associated with the complete overlapping variant, as proven in Theorem **??**. We can see that for both versions of the problem, by increasing the number of target graphs from 5 to 10, the exact method without initialization of the solution obtained with the matheuristic, becomes even harder. Indeed, the gray entries of the table mean that some instances could not find a feasible solution within the time limit (note that in the brackets we indicate the number of these instances). Furthermore, for the minimum level of endurance, the exact solution of the partial overlapping model without initialization provided by the matheuristic, does not find any solution within the time limit for instances with 10 graphs, 3 drones and a given fraction of each edge to be visited. The same can also be observed for the exact solution of the complete overlapping model without initialization provided by the matheuristic, for a level of endurance equal to 30, a fleet of 3 drones and a given fraction of each edge to be visited.
Considering the comparison with the exact method starting from the solution provided by the matheuristic, we can note that the values of the average gap are very close to those related to the exact solution method without initialization. Thus, the initialization does not speed up the convergence of the solver. However, we can see that the matheuristic is always able to find a feasible solution to the problem, even for the cases in which the solver is not.
Furthermore, the average solution times of the matheuristic range between a minimum of 4 seconds

to a maximum of 9.5 minutes. In particular, we can observe that, in most cases, for the complete overlapping version of the problem, the matheuristic running time is shorter for the instances where a fraction of the length of each graph is required to be visited. The same behavior can be noticed for the partial overlapping version of the problem for which the difference in terms of running time is even bigger. Indeed, when a given fraction of the length of each edge is required, STEP 1 of the matheuristic (computation of the TSP over the graph edges) takes more time. By increasing the number of target graphs from 5 to 10, the average solution times of the matheuristic increases for both model variants. Summing up, the results obtained show that the exact solution method given by solving the formulation is very challenging even for small size instances. However, exploiting it, the matheuristic is able to provide solutions for all instances quite quickly.

Table 5: Comparison between the partial and complete overlapping models

Model			Complete Overlapping						Partial Overlapping					
$\|\mathcal{G}\|$			5			10			5			10		
N_D	$\|\mathcal{D}\|$	α	Gap (wi)	Gap (i)	TimeH	Gap (wi)	Gap (i)	TimeH	Gap (wi)	Gap (i)	TimeH	Gap (wi)	Gap (i)	TimeH
20	1	g	0.78	0.79	6.01	0.91 (2)	0.86	177.69	0.65	0.63	16.53	1	1	215.59
		e	0.81	0.81	15.41	0.89 (2)	0.84	148.95	0.84	0.83	52.36	0.88 (3)	0.87	440.93
	2	g	0.81	0.87	5.76	0.96 (3)	0.96	139.24	0.97	0.96	13.91	1 (3)	1	76.77
		e	0.93	0.92	33.99	0.97 (3)	0.97	163.41	0.86 (2)	0.85	66.38	0.89 (4)	0.85	578.31
	3	g	0.88	0.89	4.83	0.95 (3)	0.94	67.76	0.97	0.97	17.87	1 (2)	1	18.88
		e	0.92	0.91	14.08	0.97 (2)	0.97	125.89	0.81 (3)	0.84	61.83	- (5)	0.82	237.33
30	1	g	0.71	0.7	9.66	0.82 (4)	0.82	87.4	0.77	0.75	15.43	1	1	39.83
		e	0.79	0.8	14.16	0.8 (4)	0.83	122.23	0.84	0.82	38.94	0.83 (4)	0.81	289.74
	2	g	0.82	0.82	4.98	0.95 (3)	0.92	174.64	0.97	0.96	12.94	1	1	45.37
		e	0.84	0.84	14.73	0.96 (3)	0.97	133.75	0.78	0.79	31.82	0.82	0.77	171.16
	3	g	0.82	0.81	4.63	0.93 (3)	0.95	105.54	0.96	0.96	16.22	1	1	33.95
		e	0.88	0.89	12.08	- (5)	0.97	127.78	0.83	0.82	35.38	0.79 (3)	0.8	213.06
40	1	g	0.68	0.68	5.79	0.81 (2)	0.82	93.21	0.73	0.71	11.46	1	1	48.85
		e	0.76	0.77	37.55	0.78 (4)	0.81	160.24	0.8	0.79	57.28	0.79 (1)	0.8	403.72
	2	g	0.72	0.66	5.14	0.91 (2)	0.92	131.26	0.96	0.95	11.48	1	1	35.71
		e	0.83	0.78	19.46	0.91 (2)	0.95	141.6	0.79	0.79	35.79	0.79 (1)	0.79	576.75
	3	g	0.61	0.62	3.91	0.91	0.91	115.48	0.95	0.95	15.13	1	1	17.98
		e	0.85	0.83	15.36	0.93	0.94	85.9	0.81	0.81	40.37	0.81 (1)	0.8	309.09
50	1	g	0.65	0.64	5.52	0.82 (3)	0.84	101.24	0.82	0.78	9.53	1	1	32.54
		e	0.74	0.73	16.63	0.81 (3)	0.83	118.67	0.78	0.77	58.95	0.82 (2)	0.82	311.02
	2	g	0.7	0.7	6.37	0.9 (1)	0.93	206.87	0.97	0.97	14.68	1	1	39.5
		e	0.67	0.73	12.07	0.92 (2)	0.93	168.57	0.77	0.77	36.46	0.8 (1)	0.81	265.16
	3	g	0.65	0.64	4.27	0.9 (1)	0.93	26.68	0.94	0.92	19.08	1	1	15.97
		e	0.74	0.74	12.95	0.9	0.94	90.14	0.8	0.79	40.77	0.76 (3)	0.79	195.68
60	1	g	0.69	0.7	5.58	0.8 (4)	0.81	83.02	0.78	0.76	11.18	1	1	36.78
		e	0.74	0.74	16.53	0.85 (2)	0.86	145.06	0.76	0.76	37.73	0.84 (2)	0.83	359.68
	2	g	0.67	0.72	4.09	0.94 (2)	0.94	81.69	0.95	0.94	13.33	1	1	17.04
		e	0.76	0.73	15.58	0.94 (2)	0.92	108.17	0.78	0.78	33.28	0.78	0.79	237.38
	3	g	0.58	0.53	7	0.89 (2)	0.9	60.99	0.91	0.94	20.15	1	1	33.93
		e	0.72	0.7	15.39	0.91 (2)	0.96	96.52	0.78	0.78	49.39	0.81	0.81	259.34

The boxplots in Figure ?? represent the percentage relative gap of the solution provided by the matheuristic for the complete overlapping version of the problem, with respect to that provided by the exact solution of the mathematical programming model within the time limit, with the initialization of the solution found by the matheuristic. Similarly, Figure ?? reports the same information for the partial overlapping version of the problem.

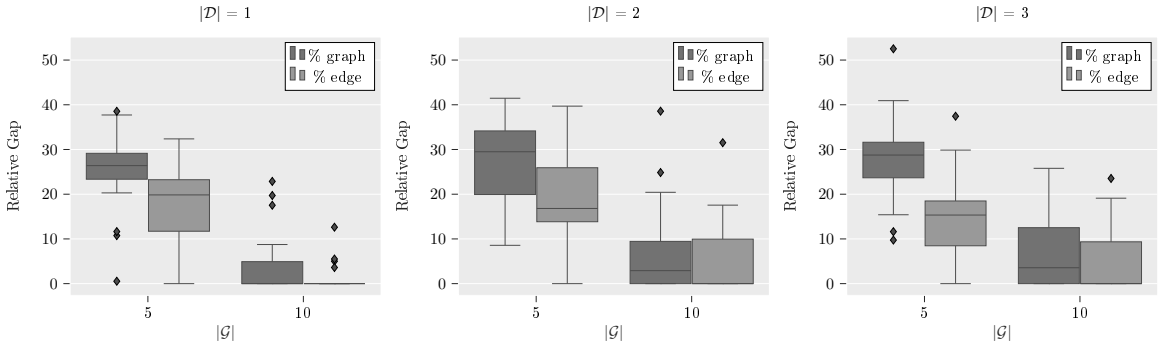


Figure 10: Relative gap boxplots for AMMDRPG-CO

From Figure ??, we can see that for the complete overlapping version of the problem, the relative gap of the solution provided by the matheuristic tends to be higher when a given fraction of each graph must be

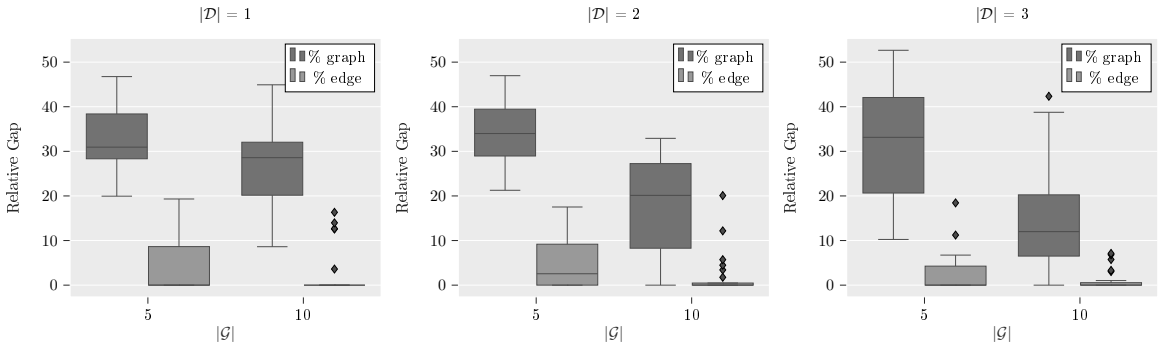


Figure 11: Relative gap boxplots for AMMDRPG-PO

visited, independently of the size of the fleet of drones. Moreover, its values decrease with the number of target graphs. A similar behavior can also be observed for the partial overlapping variant of the problem, from Figure **??**. In this latter case, we can notice a bigger difference between the values of the relative gap related to the case in which a given fraction of each graph must be visited, and that in which a given fraction of each edge must be visited. Indeed, in the first case the relative gap ranges between 0 and 50, while in the second case, between 0 and 20. Thus, we can conclude that the matheuristic provides very good quality solutions in short computing times, especially for the version of the problem in which a given fraction of each edge must be visited.

6.1. Comparing the solutions for different configurations of the problem

In this subsection, we compare the relationship between the number of available drones and their endurance and the objective function value obtained with the exact algorithm of the problem. In this experiment, we have generated a single instance with three target graphs for each combination of the parameters listed in Table **??**.

Table 6: Instance parameter values

$\|\mathcal{D}\|$	(1,2,3)
$\|V_g\|$	(4,6,8)
N^d	(10, 20,30,40,50,60)
fraction target (edge)	uniformly randomly sampled in $(0, 1)$.

Figure **??** reports the objective value varying these parameters. The darker color intensity, the smaller the objective value. As expected, our experiment confirms that both a greater number of drones and larger endurance reduce the makespan of the mothership route.

7. Case Study

In this section, we describe a realistic application of the system studied in this paper to perform surveillance operations. Considering the current COVID-19 restrictions, we focus on the problem of preventing and identifying possible concentrations of people during events such as popular or religious festivals. In particular, we consider the Courtyards Festival of Cordoba (`https://patios.cordoba.es/es/`). This is a social event that takes place every year in the city of Cordoba, Spain, during the first two weeks of May. Courtyard owners decorate their houses with many flowers trying to win the award that is offered by the Municipality. During this competition, a festival runs in parallel with a number of artistic performances along six different paths located in different areas of the city as shown in Figure **??**. In the pandemic context, to monitor the situation to avoid the concentration of people, we propose to apply a system consisting of one helicopter and a fleet of two drones. This kind of system has been tested successfully and has already been applied in the military field by the US Army to leave the helicopter at the edge of dangerous airspace and release drones, which will then penetrate enemy territory and send back intelligence, surveillance and reconnaissance information (see [**?**]). In our application, the reason to adopt a similar system is the possibility of simultaneously inspecting and in real time different paths, also reducing the risk of flying the helicopter over populated areas and the cost of moving the helicopter by minimizing the total length of its tour.

We run the models presented in Section **??** on this scenario starting from the initial solution provided by the matheuristic, where the 6 coloured paths reported in the map of Figure **??** represent the 6 target

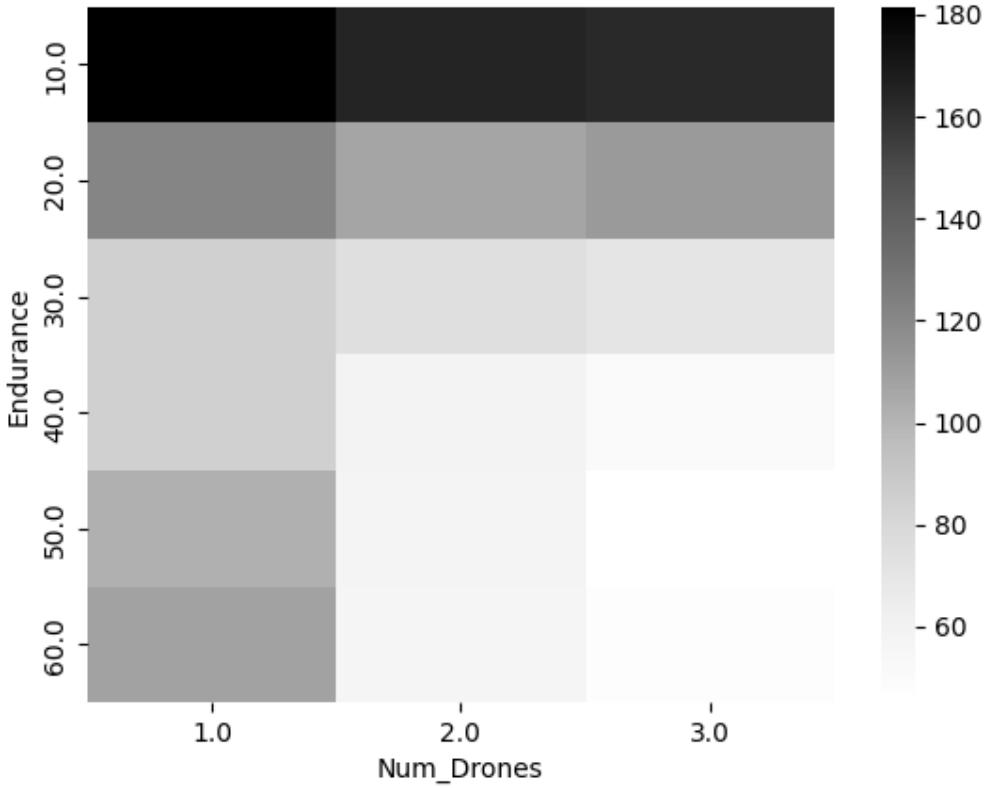


Figure 12: Heatmap of objective function values depending on number of drones and drone capacities. The darker the color intensity the greater the objective value.

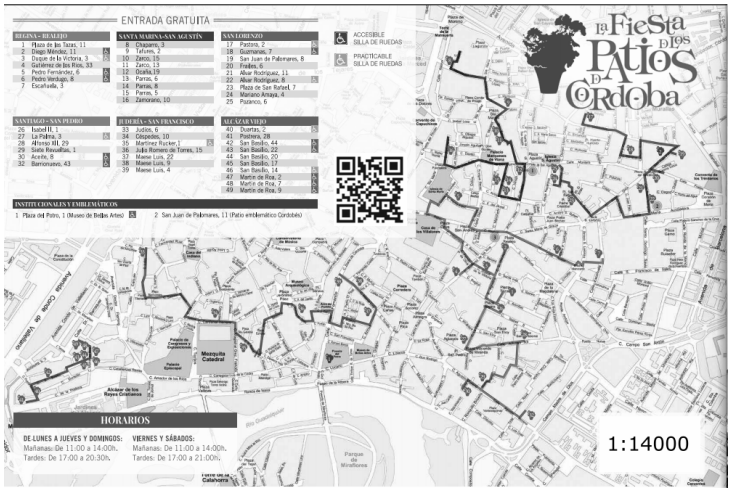


Figure 13: Map of the Courtyards Festival in Cordoba.

graphs to be visited, in this case inspected, by the fleet of drones. In addition, we suppose that the drones' speed is 43 km/h while that of the helicopter is 30 km/h aiming to minimize costs. Moreover, we assume that the fleet is composed of two drones with an endurance equal to 7.5 minutes, and we impose that each target graph must be fully visited (inspected). As we can see from Figure ?? and Figure ??, the origin of the mothership tour coincides with the destination and it is located in an area of the city where it is possible to suppose the take-off and landing of a helicopter. Figure ?? reports the tour followed by the helicopter in the solution of the complete overlapping version of the problem, after 4 hours of running time. We can observe that the helicopter, starting from the origin, flies to the

point x_R^1 that is the first rendezvous point, coinciding with the second launching point x_L^2. Then, it flies along the edge connecting x_R^1 with x_R^2, that is the second rendezvous point, which coincides with the third launching point x_L^3. Next, the helicopter flies to x_R^3 for retrieving the drone completing the third mission. From the same point the fourth and last mission starts and ends at point x_R^4, that is also the final destination of the helicopter tour.
Figure ?? also shows the tour (gray dotted paths) followed by the two drones to inspect the six paths. In particular, one drone, starts from the origin ($origin = x_L^1$) to visit the path of "Alcazar Viejo". It is retrieved by the helicopter at point x_R^1 and from the same point both drones, are launched to visit respectively the paths of "Juderia-San Francisco" and "Santa Maria-San Agustin". Both drones end their mission at point x_R^2. From this latter point they are launched to perform the visits to the paths of "San Lorenzo" and "Regina-Realejo". Then, they are both retrieved by the helicopter at point x_R^3 where only one drone starts its last mission to visit the path of "Santiago-San Pedro". In the meanwhile, the helicopter, containing the other drone, flies to point $x_R^4 = dest$ where it retrieves the other and ends its tour. The total time travelled by the helicopter is around 21 minutes. We can observe that in the drone tour on the "San Lorenzo" and "Regina-Realejo" graphs, there are two edges whose duplicate is represented with a dotted segment in Figure ??. They are associated with edges of the graph that are visited once, but traveled twice by the drone, in order to perform the inspection of the whole graph.
Figure ?? shows the mothership tour in the solution of the partial overlapping version of the problem, always obtained by setting a time limit of 4 hours. In this case, we can observe that the helicopter follows a different tour and that there are more launching and retrieving points due to the possibility of launching one drone before retrieving the other. From Figure ?? we can also see that, differently from the complete overlapping version, both drones start their first mission from the origin $origin = x_L^1 = x_L^2$. One drone visits the path of "Alcazar Viejo", while the other visits the path of "Santiago-San Pedro". The first is retrieved by the helicopter at point x_R^3 and is launched again from the point x_L^4. From this latter point this drone starts its second mission to visit the path of "Juderia-San Francisco". In the meantime, the helicopter flies to the point x_R^5 where the other drone is retrieved. From the same point $x_R^5 = x_L^6$ this latter drone is then launched to inspect the path of "Santa Maria-San Agustin". Both drones are retrieved by the helicopter at the point $x_R^7 = x_R^8$. From this latter point $x_R^8 = x_L^9$ one drone is launched to visit the path of "Regina-Realejo". Then, the helicopter flies to the point x_L^{10} from which the other drone starts its last visit to the path of "San Lorenzo". Finally, the helicopter flies to the destination *dest* and along its path, it retrieves the first drone at the point x_R^{11} and then the other at the point x_R^{12}. Also in this case, as in the solution of the complete overlapping version of the problem, we have one edge of the graph associated with the path of "Regina-Realejo" and one edge of the graph representing the path of "San Lorenzo", that are traversed twice represented with dotted segments in Figure ??. The total travel time of the helicopter is 19 minutes. It is slightly lower than that associated with the solution of the complete overlapping version of the problem. Thus, even if on this scenario we cannot observe big changes in terms of the objective function value, we can see how the different assumptions associated with the two versions of the problem can influence the structure of the solution, by producing a different schedule of the drone missions and a different location of the launching and rendezvous points.
All details of this case study, including map coordinates, .lp models and solutions can be found in [?].

8. Concluding remarks

This paper has analyzed the coordination problem that arises between a mothership vehicle and a fleet of drones that must coordinate their routes to minimize the makespan while visiting a set of targets modeled by graphs. We have presented exact mixed integer nonlinear programming formulations of the problem for its complete and partial overlapping versions. Moreover, we strengthen the models with some valid inequalities for them.
Our computational results show that the problem considered is very challenging to solve even on small and medium size instances. For that reason, additionally, we have proposed a matheuristic algorithm that provides good quality feasible solutions in a short computing time; so that it is a good alternative to the exact method. We report extensive computational experiments on randomly generated instances. Moreover, we present a case study related to inspection activities in the context of COVID-19 restrictions. We show the application of the system described in this paper in the framework of the Courtyards Festival in the city of Cordoba, by illustrating the solution obtained by adopting the problem formulation, in both versions of the model, and its solution by means of the initialization provided by the proposed matheuristic.

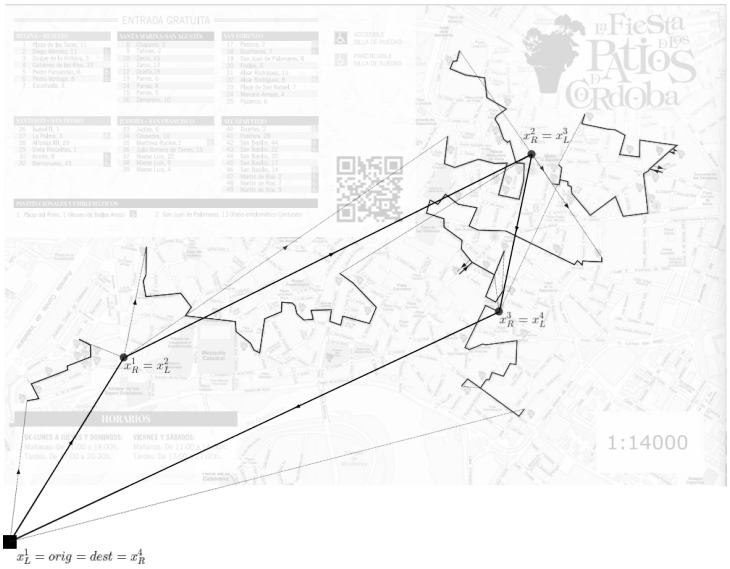


Figure 14: The complete solution (CO)

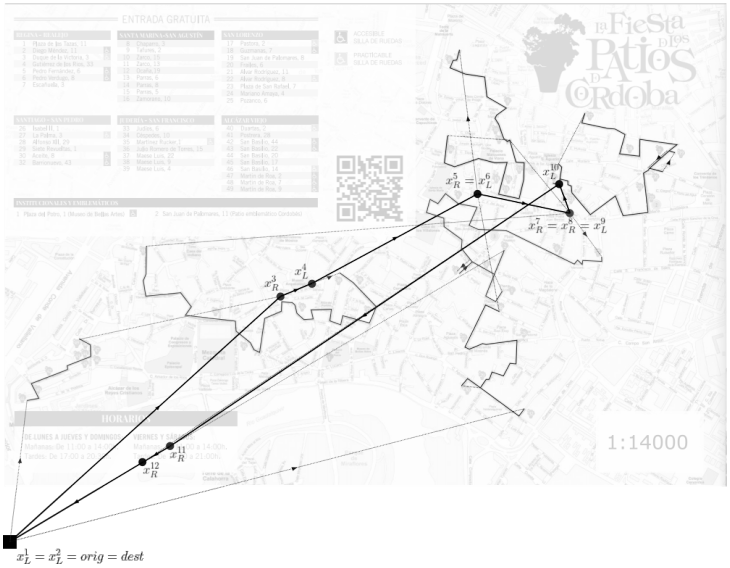


Figure 15: The complete solution (PO)

The formulation and algorithms proposed in this paper can be seen as a first building block to handle the coordination of systems composed of a base vehicle and a number of drones. Further research on this topic must focus on finding faster and more accurate algorithms able to solve larger size instances. Moreover, it is also challenging to model more complex operations allowing drones to visit more than one target per trip. Other extensions that may be considered can take into account that the time spent by the mothership to launch and retrieve the drones is not negligible as well as handling the speed of the mothership and drones as decision variables. These problems being very interesting are beyond the scope of the present paper and will be the focus of a follow up research line.

Acknowledgments

This research has been partially supported by the Spanish Ministry of Education and Science/FEDER grant number PID2020-114594GB02, projects Junta de Andalucia P18-FR-1422, FEDER-US-1256951, CEI-3-FQM331, *NetmeetData*: Ayudas Fundación BBVA a equipos de investigación científica 2019 and University of Rome, Sapienza grant number RM11916B7F962975.

9. Appendix

In this section, we report an extension of the MINLP formulation presented in Section **??**, to deal with the case of non-homogeneous fleets of drones. In the following, we introduce the parameters or input data that formally describe the problem and that are summarized in Table **??**. The formulation in this appendix is quite similar to the one in Section **??** but, because of the assumption of non-homogeneous drones, it needs to keep track of the drones used in each action. This implies including an extra index δ in most of the variables. For the sake of completeness, we have included the complete set of constraints of these formulations although some of them are similar to those in Section **??**. Table **??** summarizes all the decision variables used in this formulation.

Problem Parameters
origin: coordinates of the point defining the origin of the mothership path (or tour).
dest: coordinates of the point defining the destination of the mothership path (or tour).
$\mathcal{G}$: set of the target graphs.
$g = (V_g, E_g)$: set of nodes and edges of each target graph $g \in \mathcal{G}$.
$\mathcal{L}(e_g)$: length of edge e of graph $g \in \mathcal{G}$.
$\mathcal{L}(g) = \sum_{e_g \in E_g} \mathcal{L}(e_g)$: total length of the graph $g \in \mathcal{G}$.
B^{e_g}, C^{e_g}: coordinates of the endpoints of edge e of graph $g \in \mathcal{G}$.
α^{e_g}: fraction of edge e of graph $g \in \mathcal{G}$ that must be visited.
α^{g}: fraction of graph $g \in \mathcal{G}$ that must be visited.
v_M: mothership speed.
$\mathcal{D}$: set of drones.
v_δ: drone δ speed.
N_δ: drone δ endurance.
$\mathcal{O}$: set of drone operations to perform visits to the target graphs
M: big-M constant.

Table 7: Nomenclature for AMMDRPG with non-homogeneous fleet of drones

Binary and Integer Decision Variables
$\mu^{e_g} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G})$: equal to 1 if edge e of graph g (or a fraction of it) is visited by the drone, 0 otherwise.
$\text{entry}^{e_g} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G})$: auxiliary binary variable used for linearizing expressions.
$u^{e_g o \delta} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O},\ \forall \delta \in \mathcal{D}$: equal to 1 if the drone δ enters in graph g by the edge e_g at operation o, 0 otherwise.
$z^{e_g e_g'} \in \{0,1\},\ \forall e_g, e_g' \in E_g\ (g \in \mathcal{G})$: equal to 1 if the drone goes from e_g to e_g', 0 otherwise.
$v^{e_g o \delta} \in \{0,1\},\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O},\ \forall \delta \in \mathcal{D}$: equal to 1 if the drone δ exits from graph g by e_g at operation o, 0 otherwise.
Continuous Decision Variables
$s^{e_g} \in [0, \lvert E_g \rvert - 1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: continuous non negative variable representing the order of visit to the edge e of graph g.
$\rho^{e_g} \in [0,1]$ and $\lambda^{e_g} \in [0,1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: defining the entry and exit points on e_g.
$\nu_{\min}^{e_g}$ and $\nu_{\max}^{e_g} \in [0,1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: auxiliary variables used for linearizing expressions.
$p^{e_g} \in [0,1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: auxiliary variable used for modeling the product of μ^{e_g} and $\lvert \lambda^{e_g} - \rho^{e_g} \rvert$.
$x_L^o \in \mathbb{R}^2,\ \forall o \in \mathcal{O}$: coordinates representing the point where the mothership launches the drones at operation o.
$x_R^o \in \mathbb{R}^2,\ \forall o \in \mathcal{O}$: coordinates representing the point where the mothership retrieves the drones at operation o.
$R^{e_g} \in \mathbb{R}^2,\ \forall e_g \in E_g\ (g \in \mathcal{G})$: coordinates representing the entry point on edge e_g of graph g.
$L^{e_g} \in \mathbb{R}^2,\ \forall e_g \in E_g\ (g \in \mathcal{G})$: coordinates representing the exit point on edge e_g of graph g.
$d_L^{e_g o \delta} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O},\ \forall \delta \in \mathcal{D}$: representing the distance traveled by the drone δ from the launching point x_L^o on the mothership at operation o to the first visiting point R^{e_g} on e_g.
$p_L^{e_g o \delta} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O},\ \forall \delta \in \mathcal{D}$: auxiliary variable used for modeling the product of $d_L^{e_g o \delta}$ and $u^{e_g o \delta}$.
$d^{e_g} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G})$: representing the distance traveled by the drone from the rendezvous point R^{e_g} to the launching point L^{e_g} on e_g.
$d^{e_g e_g'} \geq 0,\ \forall e_g, e_g' \in E_g\ (g \in \mathcal{G})$: representing the distance traveled by the drone from the launching point L^{e_g} on e_g to the rendezvous point $R^{e_g'}$ on e_g'.
$p^{e_g e_g'} \geq 0,\ \forall e_g, e_g' \in E_g\ (g \in \mathcal{G})$: auxiliary variable used for modeling the product of $d^{e_g e_g'}$ and $z^{e_g e_g'}$.
$d_R^{e_g o \delta} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O},\ \forall \delta \in \mathcal{D}$: representing the distance traveled by the drone δ from the last visiting point L^{e_g} on e_g to the rendezvous point x_R^o on the mothership at operation o.
$p_R^{e_g o \delta} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O},\ \forall \delta \in \mathcal{D}$: auxiliary variable used for modeling the product of $d_R^{e_g o \delta}$ and $v^{e_g o \delta}$.
$d_{origin} \geq 0$: distance from the origin *origin* to the first launching point x_L^1.
$d_{LR}^o \geq 0,\ \forall o \in \mathcal{O}$: representing the distance traveled by the mothership from the launching point x_L^o to the rendezvous point x_R^o at operation o.
$d_{RL}^o \geq 0,\ \forall o \in \mathcal{O} \setminus \lvert \mathcal{O} \rvert$: representing the distance traveled by the mothership from the rendezvous point x_R^o at operation o to the launching point $x_L^{(o+1)}$ at operation $o+1$.
$d_{dest} \geq 0$: distance from the last retrieving point $x_R^{\lvert \mathcal{O} \rvert}$ to the destination *dest*.
$time_D^o \geq 0,\ \forall o \in \mathcal{O}$: maximum time spent by a drone during operation o.
$time_M^o \geq 0,\ \forall o \in \mathcal{O}$: time spent by the mothership to go from the launching point x_L^o to the retrieving point x_R^o of operation o.
$time_M \geq 0$: total time spent by the mothership to go from the origin to the destination (makespan).

Table 8: Decision Variables for AMMDRPG with non-homogeneous fleet of drones

Visits to graphs

As for the case of a homogeneous fleet of drones presented in Section **??**, to represent the movement of the drone within a graph $g \in \mathcal{G}$, we proceed to introduce some notations related to g. Let $g = (V_g, E_g)$ be a graph in $\mathcal{G}$ whose total length is denoted by $\mathcal{L}(g)$. Here, V_g denotes the set of nodes and E_g denotes the set of edges connecting pairs of nodes. Let e_g be the edge e of the graph $g \in G$ and let $\mathcal{L}(e_g)$ be its length.

Each edge e_g is parameterized by its endpoints $B^{e_g} = (B^{e_g}(x_1), B^{e_g}(x_2))$ and $C^{e_g} = (C^{e_g}(x_1), C^{e_g}(x_2))$ and we can compute its length $\mathcal{L}(e_g) = \|C^{e_g} - B^{e_g}\|$. For each edge e_g an indicator binary variable μ^{e_g} is associated that is one if the drone visits the segment e_g. Moreover, we define the entry and exit points $R^{e_g} = (B^{e_g}, C^{e_g}, \rho^{e_g})$ and $L^{e_g} = (B^{e_g}, C^{e_g}, \lambda^{e_g})$ that determine the fraction of the edge visited by the drone. The coordinates of the points R^{e_g} and L^{e_g} are given, respectively by

$$R^{e_g} = \rho^{e_g} B^{e_g} + (1 - \rho^{e_g}) C^{e_g} \quad \text{and} \quad L^{e_g} = \lambda^{e_g} B^{e_g} + (1 - \lambda^{e_g}) C^{e_g},$$

where $\rho^{e_g} \in [0, 1]$ and $\lambda^{e_g} \in [0, 1]$ are variables to determine the position of the points on the segment. As discussed in Section **??**, we consider two modes of visit to the target graphs $g \in \mathcal{G}$:

- Visiting a fraction α^{e_g} of each edge e_g which can be modeled by using the following constraints:

$$|\lambda^{e_g} - \rho^{e_g}| \geq \alpha^{e_g}, \quad \forall e_g \in E_g. \tag{α-E}$$

  These inequalities state that the difference between the parametrizations of the entry and exit points associated with each edge e_g must be higher than the fraction of the length of e_g required to be traversed.

- Visiting a fraction α^g of the total length of the graph:

$$\sum_{e_g \in E_g} \mu^{e_g} |\lambda^{e_g} - \rho^{e_g}| \mathcal{L}(e_g) \geq \alpha^g \mathcal{L}(g). \tag{α-G}$$

  This constraint ensures that the sum of the fractions of the length of those edges chosen to be crossed must be higher than the fraction of the length of g required to be traversed.

In both cases, the corresponding constraints are nonlinear. To linearize them, we need to introduce a binary variable entry^{e_g} that determines the traveling direction on the edge e_g as well as the definition of the auxiliary variables $\nu^{e_g}_{\min}$ and $\nu^{e_g}_{\max}$ of the access and exit points on that segment. Then, for each edge e_g, the absolute value constraint (**??**) can be represented by:

$$|\rho^{e_g} - \lambda^{e_g}| \geq \alpha^{e_g} \iff \begin{cases} \rho^{e_g} - \lambda^{e_g} & = & \nu^{e_g}_{\max} - \nu^{e_g}_{\min}, \\ \nu^{e_g}_{\max} & \leq & 1 - \text{entry}^{e_g}, \\ \nu^{e_g}_{\min} & \leq & \text{entry}^{e_g}, \\ \nu^{e_g}_{\min}, \nu^{e_g}_{\max} & \geq & 0, \\ \nu^{e_g}_{\max} + \nu^{e_g}_{\min} & \geq & \alpha^{e_g}. \end{cases} \tag{α-E}$$

The first four inequalities model the standard trick of the linearization of the absolute value. The last constraint ensures that the value of the linear expression of the absolute value is higher than the required fraction α^{e_g}. Similarly, (**??**) can be linearized as follows:

$$\sum_{e_g \in E_g} \mu^{e_g} |\rho^{e_g} - \lambda^{e_g}| \mathcal{L}(e_g) \geq \alpha^g \mathcal{L}(g). \iff \begin{cases} \rho^{e_g} - \lambda^{e_g} & = & \nu^{e_g}_{\max} - \nu^{e_g}_{\min}, \\ \nu^{e_g}_{\max} & \leq & 1 - \text{entry}^{e_g}, \\ \nu^{e_g}_{\min} & \leq & \text{entry}^{e_g}, \\ \nu^{e_g}_{\min}, \nu^{e_g}_{\max} & \geq & 0, \\ p^{e_g} & \leq & \nu^{e_g}_{\max} + \nu^{e_g}_{\min}, \\ p^{e_g} & \leq & \mu^{e_g}, \\ p^{e_g} & \geq & \nu^{e_g}_{\max} + \nu^{e_g}_{\min} + \mu^{e_g} - 1, \\ \sum_{e_g \in E_g} p^{e_g} \mathcal{L}(e_g) & \geq & \alpha^g \mathcal{L}(g), \end{cases} \tag{α-G}$$

where p^{e_g} is the auxiliary variable that represents the product of the binary variable μ^{e_g} and the absolute value difference $|\rho^{e_g} - \lambda^{e_g}|$. The first four inequalities again linearize the absolute value expression. The (second) following three constraints model the product of the expression of the absolute value and the binary variable μ^{e_g}. The last inequality ensures that the fraction of the length of those edges chosen to be crossed must be higher than the fraction of the length of g required to be traversed.

Elimination of subtours

As already presented in Section **??**, to prevent the existence of subtours within each graph $g \in \mathcal{G}$ that the drone must visit, one can include, among others, either the compact formulation that uses the Miller-Tucker-Zemlin constraints (MTZ) or the subtour elimination constraints (SEC).

For the MTZ formulation, we use the continuous variables s^{e_g}, defined in Table ??, that state the order to visit the edge e_g and set the following constraints for each $g \in \mathcal{G}$:

$$s^{e_g} - s^{e'_g} + |E_g| z^{e_g e'_g} \leq |E_g| - 1, \qquad \forall e_g \neq e'_g \in E_g, \qquad (\text{MTZ}_1)$$

$$0 \leq s^{e_g} \leq |E_g| - 1, \qquad \forall e_g \in E_g. \qquad (\text{MTZ}_2)$$

Alternatively, we can also use the family of subtour elimination constraints for each $g \in \mathcal{G}$:

$$\sum_{e_g, e'_g \in S} z_g^{e_g e'_g} \leq |S| - 1, \quad \forall S \subset E_g. \qquad (\text{SEC})$$

To find SEC inequalities, as usual, we search 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.

Drone constraints

To model this problem, as already described in Section ??, we adopt the concept of operation. Let us denote by $\mathcal{O}$ the set of operations that the mothership and the fleet of drones have to carry out. These operations are visits to the different graphs in $\mathcal{G}$ with the required constraints. An operation $o \in \mathcal{O}$ is referred to as the event in which the mothership launches some drones from a taking-off location, denoted by x_L^o and later it takes them back to a rendezvous location x_R^o.

For each operation $o \in \mathcal{O}$, each of the drones launched from the mothership must follow a path starting from and returning to the mothership, while visiting the required edges of g.

To include the definition of these paths in our mathematical programming formulation, we need to make decisions to choose:

- The optimal assignment of drones to visit graphs in a given operation o.
- The order to visit the edges of each graph in its corresponding operation.

We model the route that the drone follows by using the binary variables $u^{e_g o \delta}$, $z^{e_g e'_g}$ and $v^{e_g o \delta}$ defined in Table ??.

$$\sum_{g \in \mathcal{G}} \sum_{e_g \in E_g} u^{e_g o \delta} \leq 1, \qquad \forall o \in \mathcal{O}, \forall \delta \in \mathcal{D}. \qquad (38)$$

$$\sum_{g \in \mathcal{G}} \sum_{e_g \in E_g} v^{e_g o \delta} \leq 1, \qquad \forall o \in \mathcal{O}, \forall \delta \in \mathcal{D}. \qquad (39)$$

$$\sum_{e_g \in E_g} \sum_{o \in \mathcal{O}} \sum_{\delta \in \mathcal{D}} u^{e_g o \delta} = 1, \qquad \forall g \in \mathcal{G}, \qquad (40)$$

$$\sum_{e_g \in E_g} \sum_{o \in \mathcal{O}} \sum_{\delta \in \mathcal{D}} v^{e_g o \delta} = 1, \qquad \forall g \in \mathcal{G}, \qquad (41)$$

$$\sum_{e_g \in E_g} u^{e_g o \delta} = \sum_{e_g \in E_g} v^{e_g o \delta}, \qquad \forall g \in \mathcal{G}, \forall o \in \mathcal{O}, \forall \delta \in \mathcal{D}, \qquad (42)$$

$$\sum_{o \in \mathcal{O}} \sum_{\delta \in \mathcal{D}} u^{e_g o \delta} + \sum_{e'_g \in E_g} z_g^{e'_g e_g} = \mu^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}, \qquad (43)$$

$$\sum_{o \in \mathcal{O}} \sum_{\delta \in \mathcal{D}} v^{e_g o \delta} + \sum_{e'_g \in E_g} z_g^{e_g e'_g} = \mu^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}. \qquad (44)$$

Inequalities (??) and (??) state that a drone δ visits at most one graph g at operation o. Constraints (??) and (??) ensure that each graph is visited at some operation o by some drone δ. Equations (??) ensure that the operation of entering and exiting the graph g occurs in the same operation o and is done by the same drone δ. Constraints (??) state that if an edge e of graph g is visited by the drone δ, one of two alternative situations must occur: either e is the first edge of graph g visited by the drone δ at operation o, or edge e is visited by the drone δ after visiting another edge e' of graph g. Similarly, constraints (??) state that if an edge e of graph g is visited by the drone δ, either e is the last edge of graph g visited by the drone at operation o, or the drone δ must move to another edge e' of graph g after visiting edge e.

Distance and Time Constraints
The goal of the AMMDRPG (specify) is to find a feasible solution that minimizes the total time traveled by the mothership (makespan). To account for the different distances between the decision variables of the model, we need to set the continuous variables $d_L^{e_g o\delta}$, d^{e_g}, $d^{e_g e'_g}$, $d_R^{e_g o\delta}$, d_{origin}, d^o_{RL}, d^o_{LR} and d_{dest} defined in Table **??**. This can be done by means of the following constraints:

$$\|x_L^o - R^{e_g}\| \leq d_L^{e_g o\delta}, \qquad \forall e_g \in E_g : g \in \mathcal{G}, \forall o \in \mathcal{O}, \forall \delta \in \mathcal{D}, \qquad \text{(DIST}_1\text{-o)}$$
$$\|R^{e_g} - L^{e_g}\| \leq d^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}, \qquad \text{(DIST}_2\text{-o)}$$
$$\|R^{e_g} - L^{e'_g}\| \leq d^{e_g e'_g}, \qquad \forall e_g \neq e'_g \in E_g : g \in \mathcal{G}, \qquad \text{(DIST}_3\text{-o)}$$
$$\|L^{e_g} - x_R^o\| \leq d_R^{e_g o\delta}, \qquad \forall e_g : g \in \mathcal{G}, \forall o \in \mathcal{O}, \forall \delta \in \mathcal{D}, \qquad \text{(DIST}_4\text{-o)}$$
$$\|origin - x_L^1\| \leq d_{origin}, \qquad \text{(DIST}_5\text{-o)}$$
$$\|x_L^o - x_R^o\| \leq d^o_{LR}, \qquad \forall o \in \mathcal{O}. \qquad \text{(DIST}_6\text{-o)}$$
$$\|x_R^o - x_L^{o+1}\| \leq d^o_{RL}, \qquad \forall o \in \mathcal{O} : o < |\mathcal{O}|, \qquad \text{(DIST}_7\text{-o)}$$
$$\|x_R^{|\mathcal{O}|} - dest\| \leq d_{dest}, \qquad \text{(DIST}_8\text{-o)}$$

Thus, we can express the time spent by a drone $\delta \in \mathcal{D}$ to visit a graph $g \in \mathcal{G}$ during operation $o \in \mathcal{O}$ as follows:

$$time_\delta^o \geq \frac{1}{v_\delta}\left(\sum_{e_g \in E_g} u^{e_g o\delta} d_L^{e_g o\delta} + \sum_{e_g, e'_g \in E_g} z^{e_g e'_g} d^{e_g e'_g} + \sum_{e_g \in E_g} \mu^{e_g} d^{e_g} + \sum_{e_g \in E_g} v^{e_g o\delta} d_R^{e_g o\delta}\right) - N_\delta(1 - \sum_{e_g \in E_g} u^{e_g o\delta}) \tag{45}$$

The first addend within the brackets in the RHS of constraint (**??**), accounts for the time spent by drone δ to (depart) go from the launching point x_L^o to the first retrieving point in the graph R^{e_g}. The second addend considers the time consumed by the drone to go from edge e_g to e'_g in the graph g. The third computes the time required for traversing the required edges in g. The fourth measures the time to travel from the last launching point $L^{e''_g}$ to the retrieving point x_R^o. The bigM term ensures that the constraint becomes active only when a graph g is visited during operation o by drone δ. The reader may observe that the endurance constraint (**??**) restricts the time spent by the drone to perform the operation o to be less than its endurance N_δ. Hence, it is possible to take N_δ as a bigM constant in (**??**).

In order to compute the maximum time spent by a drone to visit a graph $g \in \mathcal{G}$ associated with operation $o \quad \forall o \in \mathcal{O}$, we introduce the following constraints:

$$time_D^o \geq time_\delta^o \ \forall \delta \in \mathcal{D} \tag{46}$$

Constraints (**??**) defines the time spent by the mothership to go from the launching point x_L^o to the retrieving point x_R^o associated with operation o.

$$time_M^o = \frac{d^o_{LR}}{v_M} \ \forall o \in \mathcal{O} \tag{47}$$

Thus, the overall time spent by the mothership to move from the origin to the destination (makespan) can be expressed as follows:

$$time_M = \frac{1}{v_M}(d_{origin} + \sum_{o \in \mathcal{O}}(d^o_{LR} + d^o_{RL}) + d_{dest}) \tag{48}$$

Coordination and Endurance Constraints
The coordination between the drones and the mothership must ensure that the maximum time $time_D^o$ spent by a drone to visit a graph g at 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 operation o. To this end, we need to define the following coordination constraint for each operation $o \in \mathcal{O}$:

$$time_D^o \leq time_M^o \tag{DCW-CO}$$

We can model the time endurance constraint for a particular operation $o \in \mathcal{O}$ and drone $\delta \in \mathcal{D}$ by limiting the time traveled by the drone δ for this operation o:

$$time_\delta^o \leq N_\delta. \qquad \text{(Endurance-CO)}$$

AMMDRPG-Complete Overlapping Formulation (with non-homogeneous fleet of drones)
Putting together all the constraints introduced before, the following formulation minimizes the total time traveled by the mothership (makespan), ensuring the coordination with the fleet of drones while guaranteeing the required coverage of the target graphs.

min $time_M$ (AMMDRPG-Complete Overlapping with a non-homogeneous fleet of drones)

s.t. (??) – (??) or (??),
(??) or (??),
(??) – (??),
(??) – (??),
(??) – (??),
(??),
(??)

The objective function accounts for the time traveled by the mothership (makespan). Constraints (??)-(??) model the route followed by the drone $\delta \in \mathcal{D}$, (??) - (??) or (??) ensure that the displacement of the drone $\delta \in \mathcal{D}$ assigned to the target graph $g \in \mathcal{G}$ is a route, (??) or (??) define what is required in each visit to a target graph. Finally, constraints (??)-(??) set the variables $d_L^{e_g o\delta}$, d^{e_g}, $d^{e_g e'_g}$, $d_R^{e_g o\delta}$, d_{origin}, d_{RL}^o, d_{LR}^o and d_{dest}, defined in Table ??, which represent Euclidean distances needed in the model.

Strengthening the formulations
In this section, we present some results that adjust the bigM constants for each of the models. These constants appear when we linearize the bilinear terms of (??). We use the McCormick's envelopes again by adding variables $p \geq 0$ representing the products. To strengthen the formulations, we provide tight upper and lower bounds for those constants. The reader may note that the same bounds can be used for both models. Therefore, wlog, we focus on the bigM constants that appear in (??).

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 graph $g \in \mathcal{G}$
To linearize the first addend in (??), we define the auxiliary non-negative continuous variables $p_L^{e_g o\delta}$ (resp. $p_R^{e_g o\delta}$) and we model the product by including the following constraints:

$$\begin{aligned}
&p_L^{e_g o\delta} \leq M_L^{e_g o\delta} u^{e_g o\delta},\\
&p_L^{e_g o\delta} \leq d_L^{e_g o\delta},\\
&p_L^{e_g o\delta} \geq m_L^{e_g o\delta} u^{e_g o\delta},\\
&p_L^{e_g o\delta} \geq d_L^{e_g o\delta} - M_L^{e_g o\delta}(1 - u^{e_g o\delta}).
\end{aligned}$$

Note that, among all graph nodes and the origin and destination points, it is possible to identify the pair of points at the maximum distance. From this pair of points, we can build a circle whose diameter is the segment joining them. Hence, because we are minimizing the distance traveled by the mothership, every launching or rendezvous point is inside this circle and the best upper bound $M_L^{e_g o\delta}$ or $M_R^{e_g o\delta}$ can be described as:

$$M_R^{e_g o\delta} = \max_{\{v \in V_g \cup \{\text{origin,dest}\}, v' \in V_{g'} \cup \{\text{origin,dest}\}: g, g' \in \mathcal{G}\}} \|v - v'\| = M_L^{e_g o\delta}.$$

On the other hand, the minimum distance in this case can be zero. This bound is attainable whenever the launching or rendezvous points of the mothership are the same as the rendezvous or launching point on the target graph $g \in \mathcal{G}$.

Bounds on the bigM constants for the distance from the launching to the rendezvous points on the target graph $g \in \mathcal{G}$.
When the drone visits a graph g, it has to go from one edge e_g to another edge e'_g depending on the order given by $z^{e_g e'_g}$. This fact produces a product of variables linearized by the following constraints:

$$\begin{aligned}
&p^{e_g e'_g} \leq M^{e_g e'_g} z^{e_g e'_g},\\
&p^{e_g e'_g} \leq d^{e_g e'_g},\\
&p^{e_g e'_g} \geq m^{e_g e'_g} d^{e_g e'_g},\\
&p^{e_g e'_g} \geq d^{e_g e'_g} - M^{e_g e'_g}(1 - z^{e_g e'_g}).
\end{aligned}$$

Since we are taking into account the distance between two edges $e_g = (B^{e_g}, C^{e_g})$, $e_g{}' = (B^{e'_g}, C^{e'_g}) \in E_g$, the maximum distance between their vertices gives us the upper bound:

$$M^{e_g e'_g} = \max\{\|B^{e_g} - C^{e'_g}\|, \|B^{e_g} - B^{e'_g}\|, \|C^{e_g} - B^{e'_g}\|, \|C^{e_g} - C^{e'_g}\|\}.$$

We observe that the minimum distance between edges $m^{e_g e'_g}$ can easily be obtained computing the minimum distance between two edges, which results in a simple second-order cone program.

Experimental results
In this section we discuss the results obtained testing the formulation of the AMMDRPG with a non-homogeneous fleet of drones, presented in Appendix **??**, on the same set of instances described in Section **??**.
Table **??** reports the results obtained by adopting the commercial solver Gurobi. We consider the exact solution both providing and not providing an initial solution computed by the matheuristic described in Section **??**. More precisely, the first column of Table **??** indicates the number of target graphs to be visited by the fleet of drones, the second column reports the endurance of the drones, the third column distinguishes between the visit of a fraction of each edge (e) and a fraction of each target graph (g). The fourth column reports the size of the fleet of drones. This last column contains three subcolumns reporting, for each cardinality of the set $\mathcal{D}$, respectively, the average gap without initialization (wi), the average gap with initialization of the solution provided by the matheuristic (i) and the solution time, in seconds, of the matheuristic (TimeH) for each combination of the listed parameters. The time limit of these experiments is set equal to 2 hours.
We can observe that the value of the average gap ranges between a minimum of 0.669 and a maximum of 0.974. This shows that the model is hard to solve even with small size instances. Moreover, we can see that in most of the cases, the average gap associated with the variant of the model consisting of visiting a given fraction of each edge, is higher than the one associated with the variant that imposes visiting a given fraction of each target graph. Another thing that we can observe is that the average gap increases with the number of drones and decreases with the drone endurance.
As regards the number of target graphs, we can see that by increasing it from 5 to 10, the exact method without initialization of the solution obtained with the matheuristic, becomes even harder. Indeed, the gray entries of the table mean that some instances could not find a feasible solution within the time limit (note that in the brackets we indicate the number of these instances). The number of not solved instances increases with the number of drones. Moreover, for the minimum level of endurance, the exact solution of the model without initialization provided by the matheuristic, does not provide any solution within the time limit for instances with 10 graphs and 2 or 3 drones.
Considering the comparison with the exact method starting from the solution provided by the matheuristic, we can note that the values of the average gap are very close to those related to the exact solution method without initialization. Thus, the initialization does not speed up the convergence of the solver. However, we can see that the matheuristic is always able to find a feasible solution to the problem, even for the cases in which the solver is not (instances with 10 graphs and 2 or 3 drones and minimum value of endurance). Moreover, the average solution times of the matheuristic range between a minimum of 37 seconds to a maximum of 3 minutes. They increase with the drone endurance for the variant of the model in which a given fraction of each edge must be visited, while they decrease by increasing the number of drones for the variant of the model in which a given fraction of each target graph must be visited. By increasing the number of target graphs from 5 to 10, the average solution times of the matheuristic become more than double for both model variants. Summing up, the results obtained show that the

Table 9: Comparison between exact solution with and without initialization by the matheuristic solution

$\|\mathcal{G}\|$	N_D	v.t.	$\|\mathcal{D}\|$ = 1			$\|\mathcal{D}\|$ = 2			$\|\mathcal{D}\|$ = 3		
			Gap (wi)	Gap (i)	TimeH	Gap (wi)	Gap (i)	TimeH	Gap (wi)	Gap (i)	TimeH
5	20	e	81.70	82.63	61.56	90.61	91.57	63.80	90.93	93.06	60.87
		g	79.63	79.09	44.97	91.85	89.03	37.32	95.80	94.00	39.05
	30	e	80.17	82.70	65.21	82.21	85.14	64.41	90.12	91.90	63.34
		g	71.19	75.80	55.77	88.27	84.36	44.36	91.39	91.02	44.59
	40	e	77.98	80.94	68.81	82.16	83.44	64.80	86.25	91.24	63.19
		g	73.46	74.47	43.92	84.35	81.21	38.27	89.63	85.34	37.51
	50	e	74.41	76.87	66.67	79.57	81.12	63.86	86.16	85.11	63.51
		g	66.90	70.58	43.42	88.84	80.96	43.98	82.81	80.49	44.35
	60	e	71.61	76.39	67.78	79.84	81.63	66.08	82.06	83.82	64.40
		g	72.79	78.17	44.69	86.55	79.35	40.63	84.66	81.74	50.01
10	20	e	84.91	82.56	137.93	-	92.30	128.53	-	94.73	124.44
		g	84.08 (2)	81.00	119.20	96.64 (2)	89.88	83.50	97.43 (3)	96.44	70.00
	30	e	80.93	80.60	159.00	87.58 (3)	87.11	132.15	92.85 (2)	94.56	127.35
		g	82.70 (1)	79.93	132.67	86.13 (3)	86.32	80.29	89.74 (1)	91.12	76.72
	40	e	78.07	79.05	191.37	84.33	85.11	131.26	88.61 (1)	91.88	132.10
		g	79.64	80.23	115.00	84.57 (3)	87.31	68.39	91.86 (1)	96.09	69.40
	50	e	77.81	81.49	188.32	85.51 (1)	87.72	134.01	90.79 (3)	92.68	132.82
		g	80.38	79.92	87.23	84.00 (3)	82.80	66.14	91.96 (2)	92.48	64.94
	60	e	81.57	83.79	155.27	82.96 (2)	85.91	131.94	86.58 (3)	92.24	130.11
		g	78.46	77.57	97.89	88.29 (2)	86.94	76.53	92.23 (3)	94.31	69.53

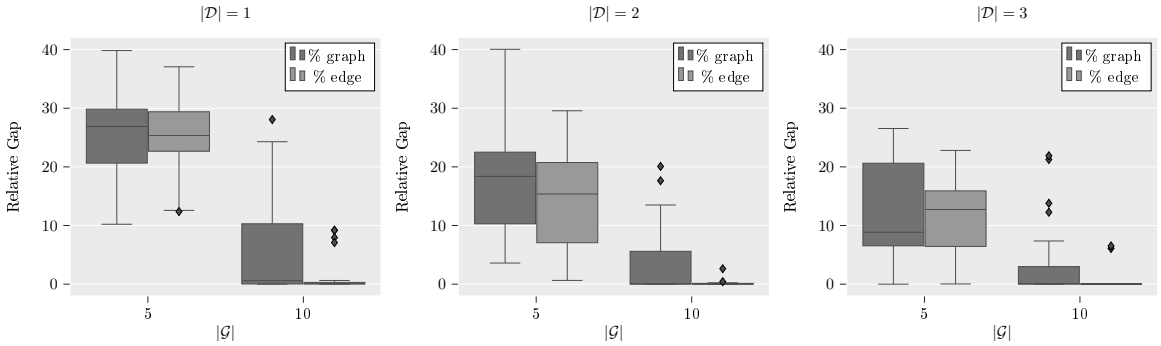


Figure 16: Relative gap boxplots

exact solution method given by solving the formulation is very challenging even for small size instances. However, exploiting it, the matheuristic is able to provide solutions for all instances quite quickly.
The boxplots in Figure ?? represent the relative gap of the solution provided by the matheuristic with respect to that provided by the exact solution of the mathematical programming model within the time limit, with the initialization of the solution found by the matheuristic. We can observe that the maximum value of the relative gap is equal to 40, but it decreases when the number of target graphs increases from 5 to 10 and it tends to be smaller when a given fraction of each edge must be visited with respect to the other case, that is, when a given fraction of each graph to be visited is imposed. In particular, for the instances with 10 target graphs, the relative gap is not greater than 25, with the exception of one outlier in the case with 1 drone and a given fraction of each graph to be visited. When the number of drones is equal to 2, this maximum value further decreases and it becomes lower than 10 when the fleet consists of 3 drones. Additionally, when a given fraction of each edge to be visited is imposed, the relative gap is even smaller and around zero in most of the instances. Thus, we can conclude that the matheuristic, even though it does not speed up the convergence to the optimum, it provides solutions of good quality, especially for instances of large size, both in terms of target graphs and drones, and for the most challenging variant of the problem in which a given fraction of each edge must be visited.