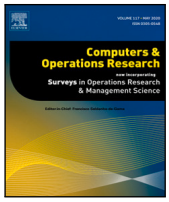
Computers & Operations Research 159 (2023) 106322

Contents lists available at ScienceDirect

Computers and Operations Research

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



A multiple-drone arc routing and mothership coordination problem

Lavinia Amorosi [a,*,1], Justo Puerto [b,1], Carlos Valverde [b,1]

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

ARTICLE INFO

Keywords:
Arc routing problems
Networks
Drones
Conic programming

ABSTRACT

This paper considers the optimisation problems that arise in coordinating 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 be ready for a new task. Tasks consist of (partially) visiting graphs of a given length to provide some services or to carry out a surveillance/inspection activity. The goal is to minimise the overall time of travelling 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 programmes 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.

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 optimising the use of this technology. Examples may be found in many different sectors, such as telecommunications, where drones can be adopted in place of traditional infrastructures to provide connectivity (see, for example, Amorosi et al. (2018), Chiaraviglio et al. (2018), Jiménez et al. (2018), Amorosi et al. (2019), and Chiaraviglio et al. (2019a)), or to temporarily deal with damage caused by a disaster (Chiaraviglio et al., 2019b; Dönmez et al., 2021), deliveries (see, for example, Mathew et al. (2015), Ferrandez et al. (2016), Poikonen and Golden (2020b), Amorosi et al. (2020), and Pei et al. (2021)), also in emergency (Wen et al., 2016), inspection (Trotta et al., 2018) and other contexts. The reader is referred to the recent surveys (Otto et al., 2018; Chung et al., 2020; Dönmez et al., 2021) for further details. In this context, depending on the specific application, we can distinguish three main different systems: one or multiple drones that, starting from a depot, provide a given service, one or multiple drones supported by a traditional vehicle that works only as a mobile depot (and/or recharging station), and one or multiple drones that cooperate with one or more traditional vehicles. In the latter case, the vehicles are also responsible for providing the service.

In the rest of this section, we review the related literature, limiting ourselves to articles that focus on truck-and-drone systems. After the seminal paper (Murray and Chu, 2015) that introduces the *Flying Sidekick Travelling Salesman Problem* (FSTSP), in which a truck and a drone cooperate to make deliveries, Ulmer and Thomas (2018) considers another model in which a fleet of trucks and drones is dispatched when orders are placed and analyses the effect of different policies to decide whether an order must be delivered by a drone or by a vehicle. Other articles, such as Campbell et al. (2017) and Carlsson and Song (2018), also study hybrid truck-and-drone models to mitigate the limited delivery range of drones. Dayarian et al. (2020) focuses on a delivery system in which traditional vehicles are resupplied by drones. Dell'Amico et al. (2021) presents a branch-and-bound algorithm and heuristics based on it to solve large-sized instances of the FSTSP. In Poikonen and Golden (2020b), the authors study the *k-Multi-Visit Drone Routing Problem* (k-MVDRP) in which a truck acts as a mobile depot. The truck is allowed to stop at a predefined set of points and launches drones that can deliver more than one package to their designated destination points (customers). The model also includes a drone energy drain function that takes into account each package weight.

Many of the articles cited assume that the set of allowable locations to launch/retrieve a drone is fixed and known a priori, the operations carried out by the drone consist of delivering to a single point, and coordination is between a truck and a single drone. These assumptions may be appropriate in some contexts, but in other cases it would be better to relax them. In Poikonen and Golden (2020a) the authors progress on the coordination problem by introducing the *Mothership*

* Corresponding author.
E-mail addresses: lavinia.amorosi@uniroma1.it (L. Amorosi), puerto@us.es (J. Puerto), cvalverde@us.es (C. Valverde).
[1] Equally contributing authors.

https://doi.org/10.1016/j.cor.2023.106322
Received 10 August 2022; Received in revised form 6 April 2023; Accepted 18 June 2023
Available online 1 July 2023

and Drone Routing Problem (MDRP) in which a two-vehicle tandem is used to design a route that visits a set of points that allow the mothership (which may be a ship or an aircraft) to launch and recover the drone in a continuous space. However, only a few articles in the literature focus on drone operations that consist of traversing graphs rather than visiting single points. In Campbell et al. (2018) the authors introduce the *Drone Rural Postman Problem* (DRPP). The paper presents 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 UAV can enter or leave. Thus, the problem is solved as a discrete optimisation 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 cannot serve all 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 Campbell et al. (2021) this problem is defined as the *Length Constrained K-Drones Rural Postman Problem* (LC K-DRPP), a continuous optimisation problem in which 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 on arc routing problems involving hybrid systems consisting of one vehicle and one or multiple drones, the number of contributions is rather limited. In Tokekar et al. (2016) the authors study the path planning problem of a system composed of a ground robot and a drone in precision agriculture and solve it by applying orienteering algorithms. Furthermore, the paper Garone et al. (2010) 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 assumes that the carrier vehicle moves in continuous space.

To the best of our knowledge, the papers Amorosi et al. (2021, 2022) are the only that deal with the coordination of a mothership with a drone to visit targets represented by graphs. In particular, in Amorosi et al. (2021) the authors make different assumptions on the route followed by the mothership: (i) it can move on 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 using mixed-integer second-order cone programmes 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 causes damage to the network. In all these cases, drone inspection or monitoring consists of traversing the edges of the network to perform a reconnaissance activity. For this reason, the targets that the drone will visit are modelled as graphs. Note that, in general, the shape of the graph edges may be a straight line or a curve. In this paper, we limit the discussion to the case of straight lines. However, as shown in Campbell et al. (2018), it is possible to deal with curved edges by approximating them through polygonal chains. Thus, this paper represents a first building block in the study of coordination models between a mothership and a fleet of drones that embed drone arc routing problems. The investigation on different edges shape is out of the scope of this paper and it is left for future research. The action of visiting a graph can be of two different types: (i) traversing a given fraction of the length of each one of its edges or (ii) visiting a fraction of the total length of the network. Other types of inspection activities, such as, for example, video surveillance of urban areas in large cities, can also be modelled by adopting the formulations presented in this paper. In this context, the request to visit a certain fraction of the target graphs (e.g., borders of a neighbourhood) may be due to the necessity of "covering" different areas in a limited time interval. Another example that we can mention is traffic flow monitoring. In this case, to verify whether traffic progression is not disrupted, only inspecting a fraction of the edge provides valuable information.

In this article, we deal with an extension of the problem studied in Amorosi et al. (2021), for which we propose a novel mothership-and-multi-drone coordination model. We consider a system in which a base vehicle (mothership) travels in continuous space and must support the launch/retrieval of several drones that must visit graphs. Indeed, depending on the specific application, the tandem system may require the adoption of multiple drones to perform surveillance/monitoring activities, for example, to accelerate data collection in emergency contexts. The presence of several drones opens up different possible working principles of the tandem system that can be more or less appropriate depending on the specific application. In particular, we focus on two different versions.

The contributions of this article to the existing literature can be summarised as follows:

(i) it extends the mothership-drone coordination problem to the more cumbersome case of several drones;
(ii) it focuses on drone arc routing problems in which drone operations consist of traversing graphs rather than visiting single points;
(iii) it studies two versions of the problem that make the presented mathematical programming approach flexible with respect to different working principles of the tandem system;
(iv) it presents matheuristic algorithms able to handle large-sized instances;
(v) it includes in the Appendix the model extension for dealing also with a non-homogeneous fleet of drones.

The rest of the paper is structured as follows. Section 2 provides a detailed description of the problem under consideration. Section 3 develops valid Mixed-Integer Non-Linear Programming (MINLP) 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 4 provides some valid inequalities that strengthen the formulations and derive upper and lower bounds on the bigM constants introduced in the proposed formulations. Section 5 presents details of the matheuristic algorithms designed to handle large-sized instances. In Section 6, we report the results obtained by testing the formulations and the matheuristic algorithm on different classes of planar graphs to assess their effectiveness. In Section 7, we present an illustrative example that applies the coordination models for the Cordoba Courtyard Festival. Section 8 concludes the paper. Finally, for the sake of completeness, we include 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 speeds 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. Furthermore, it is assumed that each drone has a limited endurance N_D, so once 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 freely moves in 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 on its movement. Nowadays, this type of system consisting of a boat and a fleet of drones is used, for example, by coast guards to carry out surveillance activities to identify

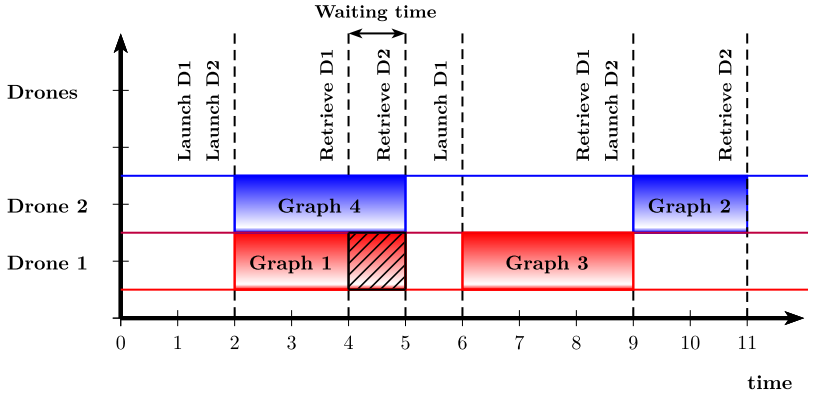


Fig. 1. Model with complete overlapping.

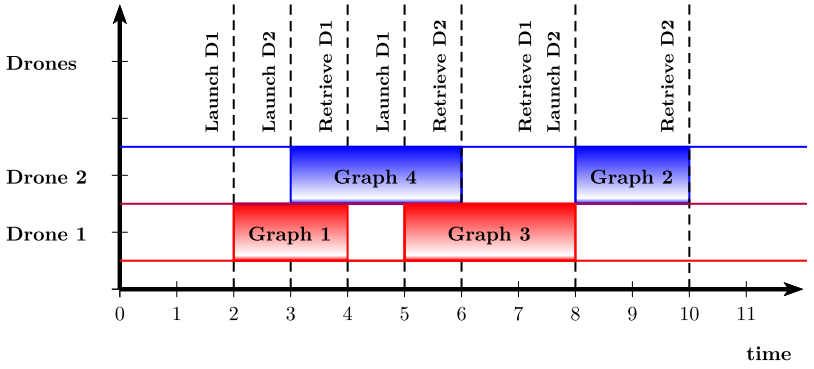


Fig. 2. Model with partial overlapping.

immigrants who need help on the sea (see AltiGator (2015)). The mothership starts at a known location, denoted *orig* where the entire system is ready to depart. Once all operations are completed, the mothership and drones must return together to a final location called *dest*. Moreover, without loss of generality, we assume that the endurance of the drone does not allow it to visit all target graphs in a single trip, starting from the origin and ending at the destination. Otherwise, the problem becomes trivial, and coordination is not 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 visits the graph and has to complete the entire action of traversing this target before returning to the base. We assume that the time each drone spends visiting the graph must be less than or equal to the time the mothership takes to move from the launching point to the retrieval point. Note also that each drone in the fleet cannot be launched from the same base vehicle location to carry out all tasks due to its limited endurance. Additionally, the costs induced by the drone trips are negligible compared to those incurred by the base vehicle. Therefore, the goal is to minimise the makespan, which in this case coincides with the overall time travelled by the mothership. Despite that, the reader may note that from a theoretical point of view, the extension to include in the objective function the times travelled by drones is straightforward and does not increase the complexity of the models or formulations.

The goal of the AMMDRPG is to find the launch and retrieval points of the drone fleet D that meet the visit requirements of the graphs in $\mathcal{G}$ and minimise the makespan (total time travelled by the mothership).

In this paper, we focus on two different versions of AMMDRPG. In the first, called AMMDRPG with *complete overlapping model* (AMMDRPG-CO), operations consisting of the launch and retrieval of a set of drones are carried out sequentially so that no two consecutive launches are possible without the retrieval of previously launched drones (see Fig. 1). The second version, called AMMDRPG with *partial overlapping model* (AMMDRPG-PO), allows consecutive launch or retrieval actions so that the visits of several drones to their target graphs are allowed to partially overlap over time (see Fig. 2). This second version is more difficult to model. Indeed, the possibility for the mothership to retrieve one drone in a different phase from that in which it has been launched implies the introduction of additional concepts and decision variables to properly formulate the problem. Moreover, additional constraints must be included to model the mothership route and its relationship with one of the drones. For both variants of the problem, we present mathematical programming formulations, valid inequalities to strengthen them, and ad hoc matheuristics to deal with medium-sized instances of the problem.

Note that the partial overlapping variant is an extension of the complete overlapping case. In fact, we allow one drone to be launched and retrieved before another different drone is launched to visit another target graph. Therefore, when applicable, the partial overlapping version can provide smaller values of the makespan. This can be observed in Figs. 1 and 2, showing the Gantt diagram of a feasible solution for the two versions of the problem, involving two drones and four target graphs. In particular, in Fig. 1, the visit of Graph 3 starts at 6 and the makespan is equal to 11. This fact is due to the constraint that the mothership must wait for the retrieval of both drones before launching Drone 1 again. In Fig. 2, the possibility to launch both drones in an asynchronous way allows us to move the visit of Graph 3 up, with a makespan equal to 10.

3. Mixed-integer non-linear programming formulations

In this section, we present a MINLP formulation for the AMMDRPG that can be used to solve medium-sized instances of this problem. As mentioned in Section 2, 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, distances are measured by the Euclidean

Table 1
Nomenclature for AMMDRPG.

Problem parameters
$orig$: 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.
$\lvert D\rvert$: number of drones.
v_D: drone speed.
N_D: drone endurance.
$\mathcal{O}$: set of drone operations to perform visits to the target graphs. $\mathcal{O} = \{1, \dots, \lvert\mathcal{O}\rvert\}$.
M: big-M constant.

Table 2
Decision variables for AMMDRPG-CO.

Binary 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 linearising 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.
$v^{e_g}_{\min}$ and $v^{e_g}_{\max} \in [0,1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: auxiliary variables used for linearising expressions.
$x^o_L \in \mathbb{R}^2,\ \forall o \in \mathcal{O}$: coordinates representing the point where the mothership launches the drones at operation o.
$x^o_R \in \mathbb{R}^2,\ \forall o \in \mathcal{O}$: 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.
$d_{orig} \geq 0$: that represents distance from the origin $orig$ to the first launching point x^1_L.
$d^{e_g o}_L \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O}$: representing the distance travelled by one drone from the launching point x^o_L on the mothership at operation o to the first visiting point R^{e_g} on e_g.
$p^{e_g o}_L \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O}$: auxiliary variable used for modelling the product of $d^{e_g o}_L$ and $u^{e_g o}$.
$d^{e_g} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G})$: representing the distance travelled by the drone from the retrieval 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 modelling 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 travelled by the drone from the launching point L^{e_g} on e_g to the retrieval 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 modelling the product of $d^{e_g e'_g}$ and $z^{e_g e'_g}$.
$d^{e_g o}_R \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O}$: representing the distance travelled by one drone from the last visiting point L^{e_g} on e_g to the retrieval point x^o_R on the mothership at operation o.
$p^{e_g o}_R \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall o \in \mathcal{O}$: auxiliary variable used for modelling the product of $d^{e_g o}_R$ and $v^{e_g o}$.
$d^o_{LR} \geq 0,\ \forall o \in \mathcal{O}$: representing the distance travelled by the mothership from the launching point x^o_L to the retrieval point x^o_R at operation o.
$d^o_{RL} \geq 0,\ \forall o \in \mathcal{O} \setminus \lvert\mathcal{O}\rvert$: representing the distance travelled by the mothership from the retrieval point x^o_R at operation o to the launching point $x^{(o+1)}_L$ at operation $o+1$.
$d_{dest} \geq 0$: that represents distance from the last retrieval point $x^{\lvert\mathcal{O}\rvert}_R$ to the destination $dest$.
$time^o_D \geq 0,\ \forall o \in \mathcal{O}$: maximum time spent by a drone during operation o.
$time^o_M \geq 0,\ \forall o \in \mathcal{O}$: time spent by the mothership to go from the launching point x^o_L to the retrieval point x^o_R of operation o.
$time_M \geq 0$: total time spent by the mothership to go from the origin to the destination (makespan).

norm, $\|\cdot\|_2$, although this assumption can be extended to any l_p norm, $1 \leq p \leq \infty$ (see Blanco et al. (2017)).

Table 1 summarises the parameters or input data that formally describe the problem.

In the following, we describe all the constraints required to formulate the AMMDRPG-CO model. Table 2 summarises 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 now 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 graph $g \in G$ and let $\mathcal{L}(e_g)$ be its length. Each edge e_g is parameterised 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, assuming the value one if the drone visits the segment e_g. Furthermore, 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 in the segment.

As discussed in Section 2, we consider two modes of visit to the target graphs $g \in \mathcal{G}$:

(i) Visiting a fraction α^{e_g} of each edge e_g which can be modelled by using the following constraints:

$$|\lambda^{e_g} - \rho^{e_g}| \geq \alpha^{e_g}, \quad \forall e_g \in E_g. \qquad (\alpha\text{-E})$$

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

(ii) Visiting a fraction α^g of the total length of the graph:

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

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

In both cases, the corresponding constraints are non-linear. To linearise them, we need to introduce a binary variable entry^{e_g} that determines the direction of travel on the edge e_g and 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 (α-E) can be represented by:

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

The first four inequalities model the standard trick of linearisation 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, (α-G) can be linearised as follows:

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

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

where p^{e_g} is the auxiliary variable that represents the product of the binary variable μ^{e_g} and the difference in absolute value $|\rho^{e_g} - \lambda^{e_g}|$. The first four inequalities linearise the expression of the absolute value. 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 greater than the fraction of the length of g required to be traversed.

Elimination of subtours

To represent the actual routes of drones on the target graph, subtours are not 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, we can include, among others, the compact formulation that uses Miller-Tucker-Zemlin constraints (MTZ) or subtour elimination constraints (SEC).

For the MTZ formulation, we use the continuous variables s^{e_g}, defined in Table 2, which state the order of visit to 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 $g \in \mathcal{G}$:

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

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 drone fleet have to perform. 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 take-off location, denoted by x^o_L and then takes them back to a retrieval location x^o_R. Here, it is important to realise that both the 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:

$$x^o_L \to R^{e_g} \to L^{e_g} \to \ldots \to R^{e'_g} \to L^{e'_g} \to \ldots \to R^{e''_g} \to L^{e''_g} \to x^o_R.$$

Fig. 3 shows an example of the notation in a configuration with four target graphs that have four nodes and four edges. Here, it is assumed that the number of drones available is equal to two. In particular, Fig. 3(a) represents a feasible solution to the problem for this configuration. The mothership, whose path is represented in black, begins at its starting point *orig*, which coincides with the first launch point x^1_L, where two drones are launched to visit two graphs. There, each drone follows a route (represented by the orange and the green paths) that ensures coverage of one-half of the length of each edge of the graph. The smaller red dots in 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 of the first two graphs, the drones return to the point x^1_R. The mothership moves from this point to the second launch point x^2_L 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 retrieval point x^2_R. Finally, the mothership moves to the point x^3_L from where a drone is launched for the last visit to the fourth graph. The drone is then retrieved by the mothership at the point x^3_R, and then the mothership ends its route at the destination point *dest*.

Fig. 3(b) represents an optimal solution for the same instance of the problem. We can observe that, in this case, from the first launch point x^1_L only one drone is launched, while from the second x^2_L two drones are launched to visit the second and third graphs. The different position in space of this last point, with respect to the feasible solution reported in Fig. 3(a) whose makespan is 158.36, ensures that the makespan of the optimal solution is equal to 152.39, which is shorter.

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

(i) The optimal assignment of drones to visit graphs in a given operation o.

(ii) The order to visit the edges of each graph in its corresponding operation.

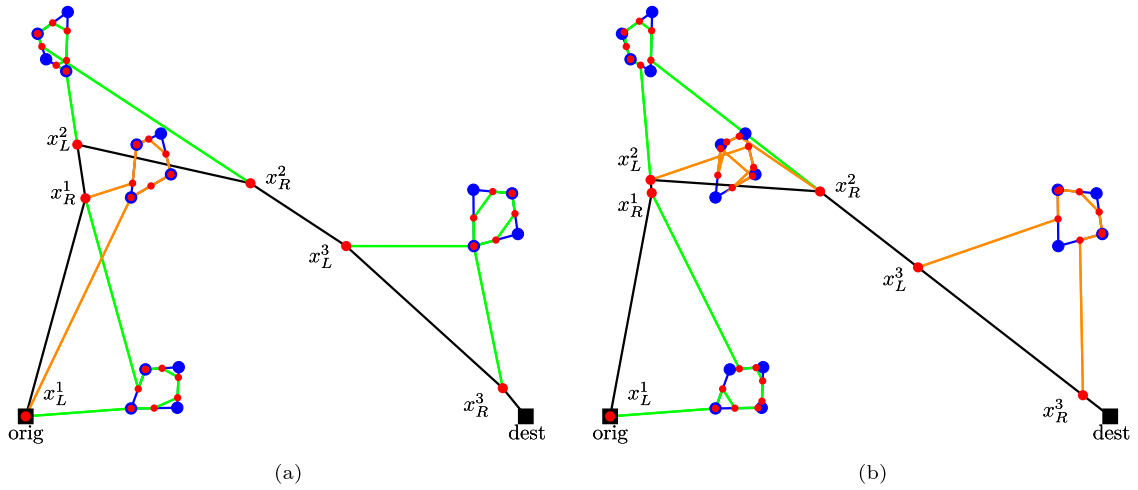


Fig. 3. Problem instance with 4 graphs and 2 drones to visit 50% of each target graph: (a) A feasible solution; and (b) an optimal solution for this case..

Drone constraints

We model the route that the drone follows using the binary variables $u^{e_g o}$, $z^{e_g e'_g}$ and $v^{e_g o}$ defined in Table 2.

$$\sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g o} \le |\mathcal{D}|, \qquad \forall o \in \mathcal{O}, \tag{Drone ROUTE$_1$-CO}$$

$$\sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g o} \le |\mathcal{D}|, \qquad \forall o \in \mathcal{O}, \tag{Drone ROUTE$_2$-CO}$$

$$\sum_{e_g\in E_g}\sum_{o\in\mathcal{O}} u^{e_g o} = 1, \qquad \forall g \in \mathcal{G}, \tag{Drone ROUTE$_3$-CO}$$

$$\sum_{e_g\in E_g}\sum_{o\in\mathcal{O}} v^{e_g o} = 1, \qquad \forall g \in \mathcal{G}, \tag{Drone ROUTE$_4$-CO}$$

$$\sum_{e_g\in E_g} u^{e_g o} = \sum_{e_g\in E_g} v^{e_g o}, \qquad \forall g \in \mathcal{G}, \forall o \in \mathcal{O}, \tag{Drone ROUTE$_5$-CO}$$

$$\sum_{o\in\mathcal{O}} u^{e_g o} + \sum_{e'_g\in E_g} z_g^{e'_g e_g} = \mu^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}, \tag{Drone ROUTE$_6$-CO}$$

$$\sum_{o\in\mathcal{O}} v^{e_g o} + \sum_{e'_g\in E_g} z_g^{e_g e'_g} = \mu^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}. \tag{Drone ROUTE$_7$-CO}$$

The inequalities (Drone ROUTE$_1$-CO) and (Drone ROUTE$_2$-CO) state that it is not possible to use a number of drones larger than the one available in each operation o. Constraints (Drone ROUTE$_3$-CO) and (Drone ROUTE$_4$-CO) ensure that each graph is visited by a drone in an operation o. Eqs. (Drone ROUTE$_5$-CO) ensure that the action of entering and exiting the graph g occurs in the same operation o. Constraints (Drone ROUTE$_6$-CO) state that if a drone visits an edge e of the graph g, one of two alternative situations must occur: e is the first edge of the graph g visited by the drone during operation o, or the edge e is visited by the drone after visiting another edge e' of the graph g. Similarly, the constraints (Drone ROUTE$_7$-CO) state that if a drone visits an edge e of the graph g, e is the last edge of the graph g visited by the drone during operation o, or the drone must move to another edge e' of the graph g after visiting the edge e.

Distance and time constraints

The goal of the AMMDRPG-CO is to find a feasible solution that minimises 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_{orig}, d_{RL}^o, d_{LR}^o and d_{dest} defined in Table 2 (Blanco et al., 2013). This can be done by means of the following constraints:

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

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

$$\|R^{e_g} - L^{e'_g}\| \le d^{e_g e'_g}, \qquad \forall e_g \ne e'_g \in E_g : g \in \mathcal{G}, \tag{Drone DIST$_3$-CO}$$

$$\|L^{e_g} - x_R^o\| \le d_R^{e_g o}, \qquad \forall e_g \in E_g : g \in \mathcal{G},\ \forall o \in \mathcal{O}. \tag{Drone DIST$_4$-CO}$$

$$\|orig - x_L^1\| \le d_{orig}, \tag{Mothership DIST$_1$-CO}$$

$$\|x_L^o - x_R^o\| \le d_{LR}^o, \qquad \forall o \in \mathcal{O}, \tag{Mothership DIST$_2$-CO}$$

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

$$\|x_R^{|\mathcal{O}|} - dest\| \le d_{dest}. \tag{Mothership DIST$_4$-CO}$$

All variables that model 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 modelling the distance travelled by the mothership, namely d_{orig}, d_{LR}^o, d_{RL}^o and d_{dest}, are defined in Table 2.

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

$$time_D^o \ge \frac{1}{v_D}\left(\sum_{e_g\in E_g} u^{e_g o} d_L^{e_g o} + \sum_{e_g, e'_g\in E_g} z^{e_g e'_g} d^{e_g e'_g} + \sum_{e_g\in E_g} \mu^{e_g} d^{e_g} + \sum_{e_g\in E_g} v^{e_g o} d_R^{e_g o}\right) - N_D\left(1 - \sum_{e_g\in E_g} u^{e_g o}\right). \tag{Drone TIME$_o$-CO}$$

The first addend in the brackets represents the time that the drone spends transferring from the launch point x_L^o to the first retrieval point in the graph R^{e_g}. The second addend considers the time consumed by the drone to go from edge e_g to edge e'_g on graph g. The third computes the time required to traverse the required edges in g. The fourth measures the time taken to travel from the last launching point $L^{e''_g}$ to the retrieval point x_R^o. Note that, in the special case where all edges must be visited, the third sum on the right-hand side of the constraint (Drone TIME$_o$-CO) reduces to $\sum_{e_g\in E_g} d^{e_g}$ setting all variables μ^{e_g} equal to one.

The endurance term in the constraint (Drone TIME$_o$-CO) ensures that the constraint becomes active only when a graph g is visited during

the operation o. The reader may observe that the endurance constraint (Endurance-CO) restricts the time the drone spends performing the 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 (Drone TIME$_o$-CO).

Note that, to deal with the bilinear terms of the constraint (Drone TIME$_o$-CO), we use McCormick's envelope to linearise them by adding variables $p \geq 0$ representing the products and introducing the following constraints:

$$p \leq Mz,$$
$$p \leq d,$$
$$p \geq mz,$$
$$p \geq d - M(1-z),$$

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 4.

The constraint (Mothership TIME$_o$-CO) defines the time the mothership must spend to go from the launch point x^o_L to the retrieval point x^o_R associated with the operation o:

$$time^o_M = \frac{d^o_{LR}}{v_M}, \quad \forall o \in \mathcal{O}. \qquad \text{(Mothership TIME}_o\text{-CO)}$$

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

$$time_M = \frac{1}{v_M}(d_{orig} + \sum_{o\in\mathcal{O}} d^o_{LR} + \sum_{o\in\mathcal{O}: o<|\mathcal{O}|} d^o_{RL} + d_{dest}).$$

(Mothership TIME-CO)

Coordination and endurance constraints

The coordination between the drones and the mothership must ensure that the maximum time $time^o_D$ 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 retrieval point during operation o. To this end, we need to define the following coordination constraint for each operation $o \in \mathcal{O}$:

$$time^o_D \leq time^o_M. \qquad \text{(DCW-CO)}$$

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

$$time^o_D \leq N_D. \qquad \text{(Endurance-CO)}$$

AMMDRPG-complete overlapping formulation

Combining all the constraints introduced hitherto, the following formulation minimises the makespan, ensuring coordination with the drone fleet while guaranteeing the required coverage of the target graphs.

$$\min \quad time_M \qquad \text{(AMMDRPG-CO)}$$

s.t. (α-E) or (α-G),
(MTZ$_1$)–(MTZ$_2$) or (SEC),
(Drone ROUTE$_1$-CO)–(Drone ROUTE$_7$-CO),
(Drone DIST$_1$-CO)–(Drone DIST$_4$-CO),
(Mothership DIST$_1$-CO)–(Mothership DIST$_4$-CO),
(Drone TIME$_o$-CO), (Mothership TIME$_o$-CO), (Mothership TIME-CO),
(DCW-CO), (Endurance-CO).

The objective function accounts for the makespan. Constraints (Drone ROUTE$_1$-CO)–(Drone ROUTE$_7$-CO) model the route followed by the drones, (MTZ$_1$)–(MTZ$_2$) or (SEC) ensure that the displacement of a drone assigned to the target graph $g \in \mathcal{G}$ is a route, (α-E) or (α-G) define what is required in each visit to a target graph. Constraints (Drone DIST$_1$-CO)–(Drone DIST$_4$-CO) set the variables $d^{e_g o}_L$, d^{e_g}, $d^{e_g e'_g}$, $d^{e_g o}_R$. The mothership distances d^o_{RL} and d^o_{LR}, are defined by means of constraints (Mothership DIST$_1$-CO)–(Mothership DIST$_4$-CO). Constraints (Drone TIME$_o$-CO), (Mothership TIME$_o$-CO) and (Mothership TIME-CO) define times travelled by the drones and the mothership. Finally, constraints (DCW-CO)–(Endurance-CO) guarantee that coordination and drone endurance are satisfied.

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 3.1, 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 one, 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, different from the complete overlapping version of the problem, the launch of a drone 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 Fig. 5. We notice that when the fleet of drones consists of only one drone, the two versions of the problem coincide. Table 3 summarises 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 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 the stage t and due to the assumptions of the problem, we need to introduce additional binary variables γ^{gt}. Thus, the following constraints model the route followed by the drone while operating on a graph $g \in \mathcal{G}$:

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

(Drone ROUTE$_1$)

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

(Drone ROUTE$_2$)

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

(Drone ROUTE$_3$)

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

(Drone ROUTE$_4$)

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

(Drone ROUTE$_5$)

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

(Drone ROUTE$_6$)

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

(Drone ROUTE$_7$)

Table 3
Decision variables for AMMDRPG-PO.

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 linearising 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 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 retrieval 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 retrieval 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 retrieval point to a retrieval point between stage t and stage $t+1$, 0 otherwise.
$\mathcal{K}(t) \in \{0,1,2,\dots,\lvert\mathcal{D}\rvert\},\ \forall t \in \mathcal{T}$: 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 retrieval 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.
$d^{e_g t}_L \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall t \in \mathcal{T}$: representing the distance travelled 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 travelled by the drone from the retrieval 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 travelled by the drone from the launching point L^{e_g} on e_g to the retrieval point $R^{e'_g}$ on e'_g.
$d^{e_g t}_R \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G}),\ \forall t \in \mathcal{T}$: representing the distance travelled by the drone from the last visiting point L^{e_g} on e_g to the retrieval point x^o_R on the mothership at stage t.
$d_{orig} \geq 0$: distance from the origin $orig$ 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 retrieval point x^{t+1}_R.
$d^t_{RL} \geq 0,\ \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert$: distance from the retrieval 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 retrieval point x^t_R to the retrieval point x^{t+1}_R.
$d_{dest} \geq 0$: distance from the last retrieval point $x^{\lvert\mathcal{T}\rvert}_R$ to the destination $dest$.
$d^g_{LR} \geq 0,\ \forall g \in \mathcal{G}$: representing the distance travelled by the mothership from the launching point x^t_L to the retrieval 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).

$$\gamma^{gt} \geq \sum_{e_g \in E_g} u^{e_g t}, \qquad \forall g \in \mathcal{G},\ \forall t \in \mathcal{T}, \tag{Drone ROUTE$_8$}$$

$$\gamma^{g(t+1)} \geq \gamma^{gt} - \sum_{e_g \in E_g} v^{e_g (t+1)}, \qquad \forall g \in \mathcal{G},\ \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert, \tag{Drone ROUTE$_9$}$$

$$\sum_{t' \in \mathcal{T} : t' < t} \gamma^{gt'} \leq (t-1)\Big(1 - \sum_{e_g \in E_g} u^{e_g t}\Big), \qquad \forall g \in \mathcal{G},\ \forall t \in \mathcal{T}, \tag{Drone ROUTE$_{10}$}$$

$$\sum_{t' \in \mathcal{T} : t' \geq t} \gamma^{gt'} \leq (\lvert\mathcal{T}\rvert - t + 1)\Big(1 - \sum_{e_g \in E_g} v^{e_g t}\Big), \qquad \forall g \in \mathcal{G},\ \forall t \in \mathcal{T}, \tag{Drone ROUTE$_{11}$}$$

$$\mathcal{K}(1) = \lvert\mathcal{D}\rvert, \tag{Drone ROUTE$_{12}$}$$

$$\mathcal{K}(t+1) = \mathcal{K}(t) + \sum_{g \in \mathcal{G}} \sum_{e_g \in E_g} v^{e_g t} - \sum_{g \in \mathcal{G}} \sum_{e_g \in E_g} u^{e_g t}, \qquad \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert. \tag{Drone ROUTE$_{13}$}$$

The constraints (Drone ROUTE$_1$) and (Drone ROUTE$_2$) ensure that a launch point and a retrieval point are associated with each graph g. The restrictions (Drone ROUTE$_3$) allow the mothership to launch a drone in stage t only if a drone is available when stage t occurs. Constraints (Drone ROUTE$_4$) guarantee that a launch or a retrieval occurs at each stage $t \in \mathcal{T}$. Constraints (Drone ROUTE$_5$) indicate that the retrieval stage associated with the graph g occurs after the launch stage associated with the same graph g. Eqs. (Drone ROUTE$_6$) state that if a drone visits an edge e of the graph g, either e is the first edge of the graph g visited by the drone at stage t, or the drone visits the edge e after visiting another edge e' of the graph g. Similarly, constraints (Drone ROUTE$_7$) state that if a drone visits an edge e of the graph g, e is the last edge of the graph g visited by the drone at stage t, or the drone must move to another edge e' of the graph g after visiting the edge e. Constraints (Drone ROUTE$_8$) ensure that the operation associated with graph g starts when the drone is launched during the stage t. Inequalities (Drone ROUTE$_9$) state that the drone is still operating in the graph g for successive stages until it is retrieved at the stage t. Constraints (Drone ROUTE$_{10}$) ensure that drone does not operate in g until launch stage occurs. Constraints (Drone ROUTE$_{11}$) guarantee that the drone finishes operating in graph g when retrieval stage occurs. Finally, the constraints (Drone ROUTE$_{12}$) and (Drone ROUTE$_{13}$) model the number of drones available at the stage t.

Mothership constraints

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

$$y^1_{LL} + y^1_{LR} = 1, \tag{Mothership ROUTE$_1$}$$

$$y^{t+1}_{LL} + y^{t+1}_{LR} \geq y^t_{RL} + y^t_{LL}, \qquad \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert, \tag{Mothership ROUTE$_2$}$$

$$y^{t+1}_{RR} + y^{t+1}_{RL} \geq y^t_{LR} + y^t_{RR}, \qquad \forall t \in \mathcal{T} : t < \lvert\mathcal{T}\rvert, \tag{Mothership ROUTE$_3$}$$

$$y_{LR}^{|\mathcal{T}|-1} + y_{RR}^{|\mathcal{T}|-1} = 1. \qquad \text{(Mothership ROUTE}_4\text{)}$$

The constraints (Mothership ROUTE$_1$) state that at stage 1 the mothership must depart from the launch point x_L^1. Constraints (Mothership ROUTE$_2$) (resp. (Mothership ROUTE$_3$)) ensure that if the mothership goes to the launching point (resp. retrieval) 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}). The constraint (Mothership ROUTE$_4$) guarantees that the path followed by the mothership ends at the retrieval 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:

$$\|x_L^t - R^{e_g}\| \le d_L^{e_g t}, \qquad \forall e_g \in E_g : g \in \mathcal{G},\ \forall t \in \mathcal{T}, \qquad \text{(Drone DIST}_1\text{)}$$

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

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

$$\|L^{e_g} - x_R^t\| \le d_R^{e_g t}, \qquad \forall e_g \in E_g : g \in \mathcal{G},\ \forall t \in \mathcal{T}. \qquad \text{(Drone DIST}_4\text{)}$$

$$\|orig - x_L^1\| \le d_{orig}, \qquad \text{(Mothership DIST}_1\text{)}$$

$$\|x_L^t - x_L^{t+1}\| \le d_{LL}^t, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Mothership DIST}_2\text{)}$$

$$\|x_L^t - x_R^{t+1}\| \le d_{LR}^t, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Mothership DIST}_3\text{)}$$

$$\|x_R^t - x_L^{t+1}\| \le d_{RL}^t, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Mothership DIST}_4\text{)}$$

$$\|x_R^t - x_R^{t+1}\| \le d_{RR}^t, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Mothership DIST}_5\text{)}$$

$$\|x_R^{|\mathcal{T}|} - dest\| \le d_{dest}. \qquad \text{(Mothership DIST}_6\text{)}$$

All variables that model 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 that model the distances travelled by the mothership, namely d_{orig}, d_{LL}^t, d_{LR}^t, d_{RL}^t, d_{RR}^t and d_{dest}, are defined in Table 3.

The time that the drone spends performing the operation of visiting the graph g is given by:

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

(Drone TIME$_g$)

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

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

(Mothership TIME$_g$)

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

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

(Mothership TIME)

Coordination and endurance constraints

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

$$time_D^g \le time_M^g, \quad \forall g \in \mathcal{G}. \qquad \text{(DCW)}$$

In addition, the time the drone spends to operate on the graph g must not exceed its endurance:

$$time_D^g \le N_D \qquad \text{(Endurance)}$$

Linearisation constraints

This subsection is devoted to linearising 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 following non-linear expressions:

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

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

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

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

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

$$y_{LL}^t \le \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Linearization}_1\text{)}$$

$$y_{LL}^t \le \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g (t+1)}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Linearization}_2\text{)}$$

$$y_{LL}^t \ge \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t} + \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g (t+1)} - 1, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Linearization}_3\text{)}$$

$$y_{LR}^t \le \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Linearization}_4\text{)}$$

$$y_{LR}^t \le \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g (t+1)}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Linearization}_5\text{)}$$

$$y_{LR}^t \ge \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g t} + \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g (t+1)} - 1, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Linearization}_6\text{)}$$

$$y_{RL}^t \le \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g t}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Linearization}_7\text{)}$$

$$y_{RL}^t \le \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g (t+1)}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Linearization}_8\text{)}$$

$$y_{RL}^t \ge \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g t} + \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g (t+1)} - 1, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Linearization}_9\text{)}$$

$$y_{RR}^t \le \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g t}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Linearization}_{10}\text{)}$$

$$y_{RR}^t \le \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g (t+1)}, \qquad \forall t \in \mathcal{T} : t < |\mathcal{T}|, \qquad \text{(Linearization}_{11}\text{)}$$

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

(Linearization$_{12}$)

AMMDRPG-Partial overlapping formulation

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

min $time_M$ (AMMDRPG-PO)

s.t. (α-E) or (α-G),
(MTZ_1)–(MTZ_2) or (SEC),
(Drone ROUTE$_1$)–(Drone ROUTE$_{13}$),
(Mothership ROUTE$_1$)–(Mothership ROUTE$_4$),
(Drone DIST$_1$)–(Drone DIST$_1$),
(Mothership DIST$_1$)–(Mothership DIST$_6$),
(Drone TIME$_g$), (Mothership TIME$_g$), (Mothership TIME),
(DCW), (Endurance),
(Linearization$_1$)–(Linearization$_{12}$)

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 those that were launched previously. Fig. 5 shows a solution that is not possible for the model with complete overlapping. In fact, we can see that the first drone is launched at x^1_L to visit P_1 and retrieved at x^1_R. However, the mothership has launched another drone at x^2_L that goes to visit P_2 before retrieved the first drone. Clearly, this solution does not satisfy the assumptions of 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, we first 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 obtained directly once the variables $\hat{u}^{e_g t}$ and $\hat{v}^{e_g t}$ are set through the constraints (Drone ROUTE$_8$)–(Drone ROUTE$_{13}$). Thus, we can limit ourselves to showing 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 that define the launch and retrieval actions that occur in operation o. More in detail, we can identify the first element $t(o)$ of this set as follows:

$$t(0) = 1;$$

$$t(o+1) = t(o) + \sum_{g\in\bar{\mathcal{G}}(o)}\sum_{e_g\in E_g} (\bar{u}^{e_g o} + \bar{v}^{e_g o}).$$

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

$$\mathcal{T}_u(o) = \{t \in \mathcal{T} : t(o) \leq t \leq t(o) + |\bar{\mathcal{G}}(o)| - 1\},$$

$$\mathcal{T}_v(o) = \{t \in \mathcal{T} : t(o) \leq t - |\bar{\mathcal{G}}(o)| \leq t(o) + |\bar{\mathcal{G}}(o)| - 1\}.$$

Since the cardinality of the set $\mathcal{T}_u(o)$ is equal to the cardinality of the 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 the assignment between the graphs and stages. Note that any assignment defined by the functions $\bar{\varphi}_{u(o)}$ and $\bar{\varphi}_{v(o)}$ is feasible.

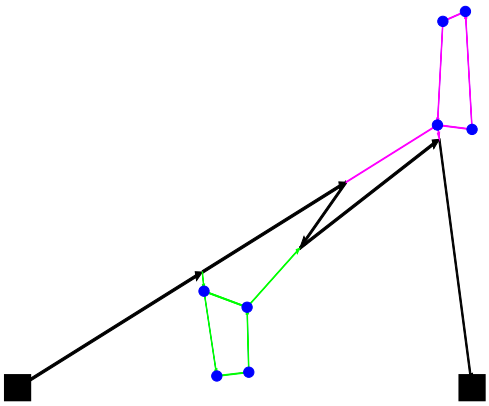


Fig. 4. Feasible solution with partial overlapping that is not feasible for the complete overlapping model.

Using 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 variables $\hat{u}^{e_g t}$ and $\hat{v}^{e_g t}$ that must be equal to 1:

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

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

The remaining $\hat{u}^{e_g t}$ and $\hat{v}^{e_g t}$ variables are set equal to 0. To show that binary variables $\hat{u}^{e_g t}$ and $\hat{v}^{e_g t}$ are feasible for the AMMDRPG-PO model, it can be easily checked they satisfy the constraints (Drone ROUTE$_1$)–(Drone ROUTE$_7$).

Moreover, it is easy to show that the mothership constraints are also satisfied by the variables $\hat{y}^t = (\hat{y}^t_{LL}, \hat{y}^t_{LR}, \hat{y}^t_{RL}, \hat{y}^t_{RR})$, induced by the variables $\hat{u}^{e_g t}$ and $\hat{v}^{e_g t}$.

With regard to continuous variables, they can be directly derived from the setting of variables $\hat{x}^t_L$ and $\hat{x}^t_R$ which 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 see that the distances between two consecutive launch points or two consecutive retrieval points are equal to 0 by the definition of the variables $\hat{x}^t_L$ and $\hat{x}^t_R$. Consequently, the time $\widehat{time}^g_M$ spent by the mothership while the drone visits the graph $g \in \bar{\mathcal{G}}(o) : o \in \bar{\mathcal{O}}$ is equal to $\overline{time}^o_M$.

To complete the proof, it suffices to notice that, in contrast, there exist feasible solutions of the AMMDRPG-PO characterised by partial overlaps between operations, as shown, for example, in Fig. 4, which are not feasible for the AMMDRPG-CO. □

To present our next result, wlog, we restrict ourselves to the degenerate case where the graphs are reduced 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 at the point which is at the same time as the one required to traverse the edges of the graph. We simplify the proof by 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 launch (resp. retrieval) points associated with the visit to the target points P_1 and P_2. If there exist*

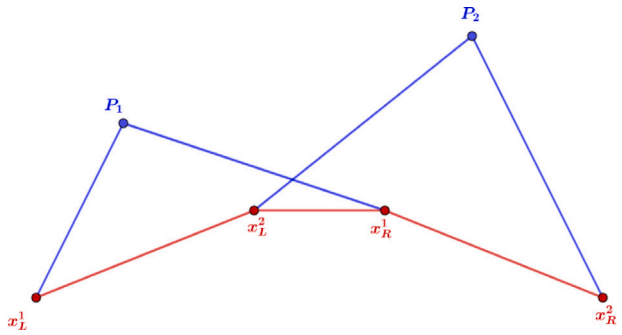


Fig. 5. The mothership launches two drones sequentially.

two points x_L and x_R verifying

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

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 launch and retrieval points. Therefore, the only constraints involved are those related to these points. These are the conditions in the following statement: The first two are the (DCW-CO) inequalities. The third is the constraint (Endurance-CO) and the last ensures that the distance travelled 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.

4. Strengthening the formulations

In this section, we present some valid inequalities for (AMMDRPG-CO) that reinforce the formulation given in Section 3.1. Moreover, constraints (DCW-CO) and (DCW) have products of binary and continuous variables that, when linearised, produce bigM constants that must be tightened. This section also provides some bounds for these constants when it is possible.

4.1. Valid inequalities for the (AMMDRPG-CO)

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 Amorosi et al. (2021). 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. Therefore, 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 target graphs are visited when the operation o begins, and zero otherwise. Note that if all graphs have already been visited before operation o then they were also completed before operation $o+1$. Therefore, the variables β must satisfy the following constraints:

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

Let k^o denote the number of graphs visited in operation o. This number can be computed using variables u since $u^{e_g o}$ takes the value of 1 if the 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 is equal to 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 allowed to have an operation o without 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 we propose includes bigM constants. In this work, we define different bigM constants. To strengthen the formulations, we provide tight upper and lower bounds for these 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 (AMMDRPG-CO).

Big M constants bounding the distance from the launch/retrieval point on the path followed by the mothership to the retrieval/launch point on the target graph $g \in \mathcal{G}$

To linearise the first addend in (DCW-CO), we define the non-negative continuous auxiliary 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 that joins them. Hence, because we minimise the distance travelled by the mothership, every launch or retrieval 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:

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

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

Bounds on the bigM constants for the distance from the launch to the retrieval 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 linearised by the following constraints:

$$\begin{aligned} & p^{e_g e'_g} \leq M^{e_g e'_g} z^{e_g e'_g}, \\ & p^{e_g e'_g} \leq d^{e_g e'_g}, \end{aligned}$$

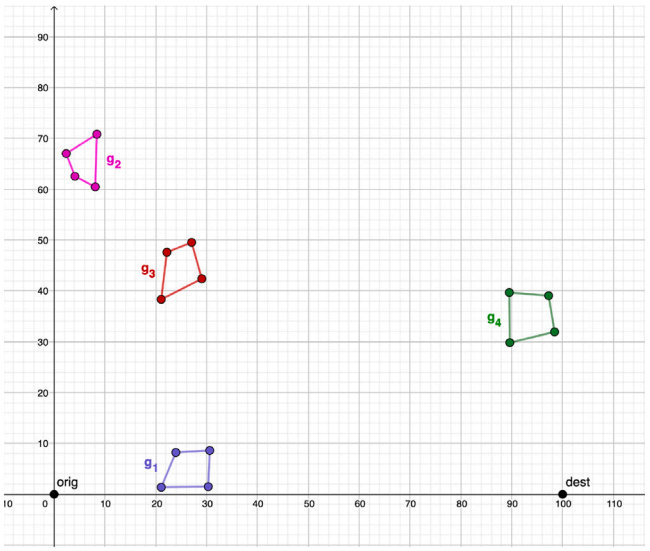


Fig. 6. Illustrative example.

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

Since we take 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 by computing the minimum distance between two edges, which results in a simple second-order cone program.

5. A matheuristic for the All Terrain Mothership and Multiple-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 3 is time-consuming. 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 take 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 several times to consider different cluster structures. At each iteration, the new clusters are evaluated by computing the cost of the route that visits 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 $v^{e_g o}$ which determine 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 solution.

Algorithm 1 reports the pseudocode of this algorithm.

Fig. 6 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 carried out by a fleet of two drones supported by a mothership whose path starts from the origin $(0, 0)$ and ends at the destination point $(100, 0)$.

Algorithm 1 Matheuristic algorithm for AMMDRPG-CO

Data: $\mathcal{G}$, $|D|$, 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 $|K_i \cup K_j| < |D|$ **then**
    Search for point P satisfying the following endurance constraint:

$$\frac{d(P, R^{e_g}) + \mathcal{L}(e_g, e'_g) + d(L^{e'_g}, P)}{v_D} \leq N_D, \quad \forall R^{e_g}, L^{e'_g} \in K_i, K_j. \quad (1)$$

    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 minimisation problem:

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

subject to (1).

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 AMMDRPG-CO

Fig. 7 illustrates a zoom in on each target graph, showing the tours generated by STEP 1 of the matheuristic procedure. A pair of points representing the launch and retrieval points, together with an arrow

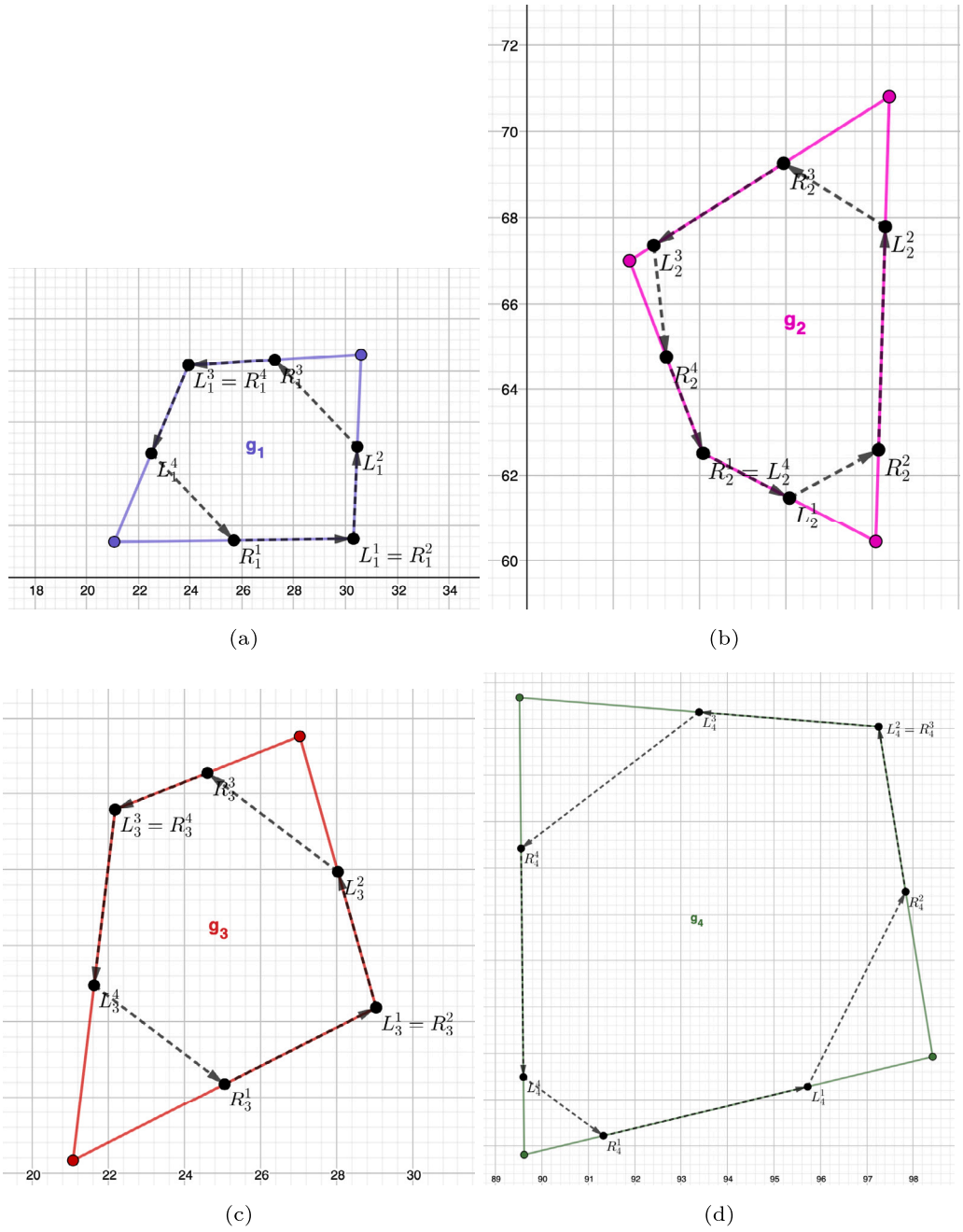


Fig. 7. STEP 1 for the illustrative example.

that points the direction followed by the drone according to the order in which the edges are visited, is depicted on each edge.

Applying STEP 2 to this illustrative example, we obtain three clusters, as shown in Fig. 8(a). One cluster contains graphs g_1 and g_3 (in lilac), while graphs 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 Fig. 8(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 Fig. 9(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 allows us to set the values of the variables $u^{e_{g^o}}$ and $v^{e_{g^o}}$ of the AMMDRPG-CO 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-CO model and returns the final feasible solution shown in Fig. 9(b). From it, we can observe that the sequence of visits of the target graphs does not change with respect to that provided by STEP 4. The two-drone fleet first visits the graphs g_1 and g_3 starting from the launch point x^1_L. Then, both drones are retrieved by the mothership at point x^1_R. The mothership moves to the point x^2_L where a drone is launched to visit graph g_2. The mothership then reaches the point x^2_R to retrieve the drone, and from the same point it launches the other drone to visit graph g_4. Then, the mothership retrieves this drone at the point x^3_R before moving to the final destination point.

Focussing on each target graph, Fig. 10 shows the zoom in on the tours followed by the drones. For example, Fig. 10[a] reports the

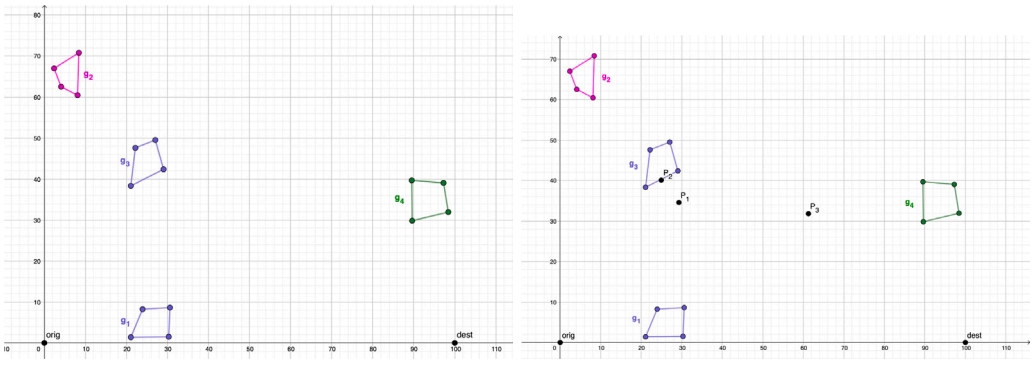


Fig. 8. (a) STEP 2, (b) STEP 3 for the illustrative example.

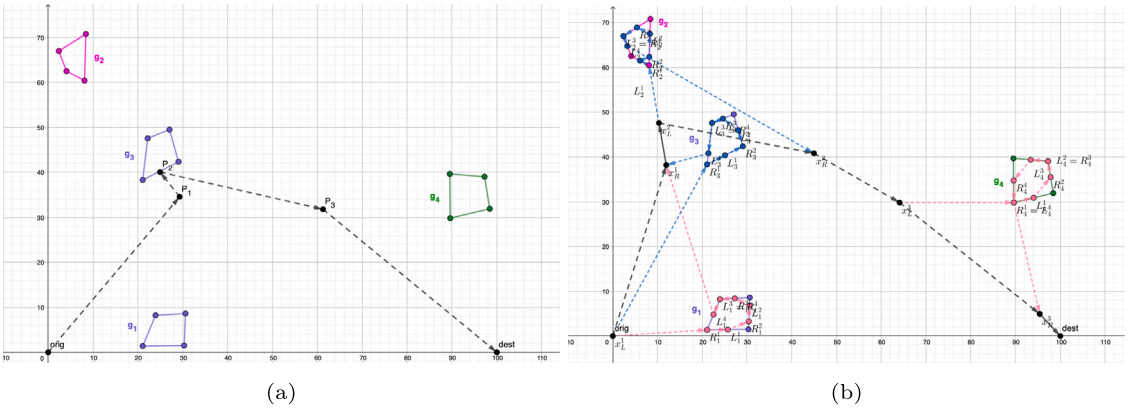


Fig. 9. (a) STEP 4, (b) STEP 5 for the illustrative example.

tour performed by the drone that visits the graph g_1 (the dashed pink segments) and the drone that visits the graph g_3 (the dashed light blue segments). Both drones start from the mothership at the point x_L^1, that is, the *orig*. 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 traversing the segment $\overline{R_1^2 L_1^2}$. Then, it moves to the third visited edge of the graph g_1, 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 the graph g_1 up to point L_1^4. Finally, the drone leaves the graph g_1 at this last point and is retrieved by the mothership at the point x_R^1. Similarly, the second drone, which visits the graph g_3, after traversing the segment $\overline{R_3^1 L_3^1}$, moves to the second visited edge of the same graph and traverses the segment $\overline{R_3^2 L_3^2}$. Then, it flies to the third visited edge, traversing the segment $\overline{R_3^3 L_3^3}$. Finally, it moves to the last visited edge of the graph g_3, flying over the segment $\overline{R_3^4 L_3^4}$. The drone leaves the graph g_3 at the point L_3^4 and reaches the mothership at the 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.

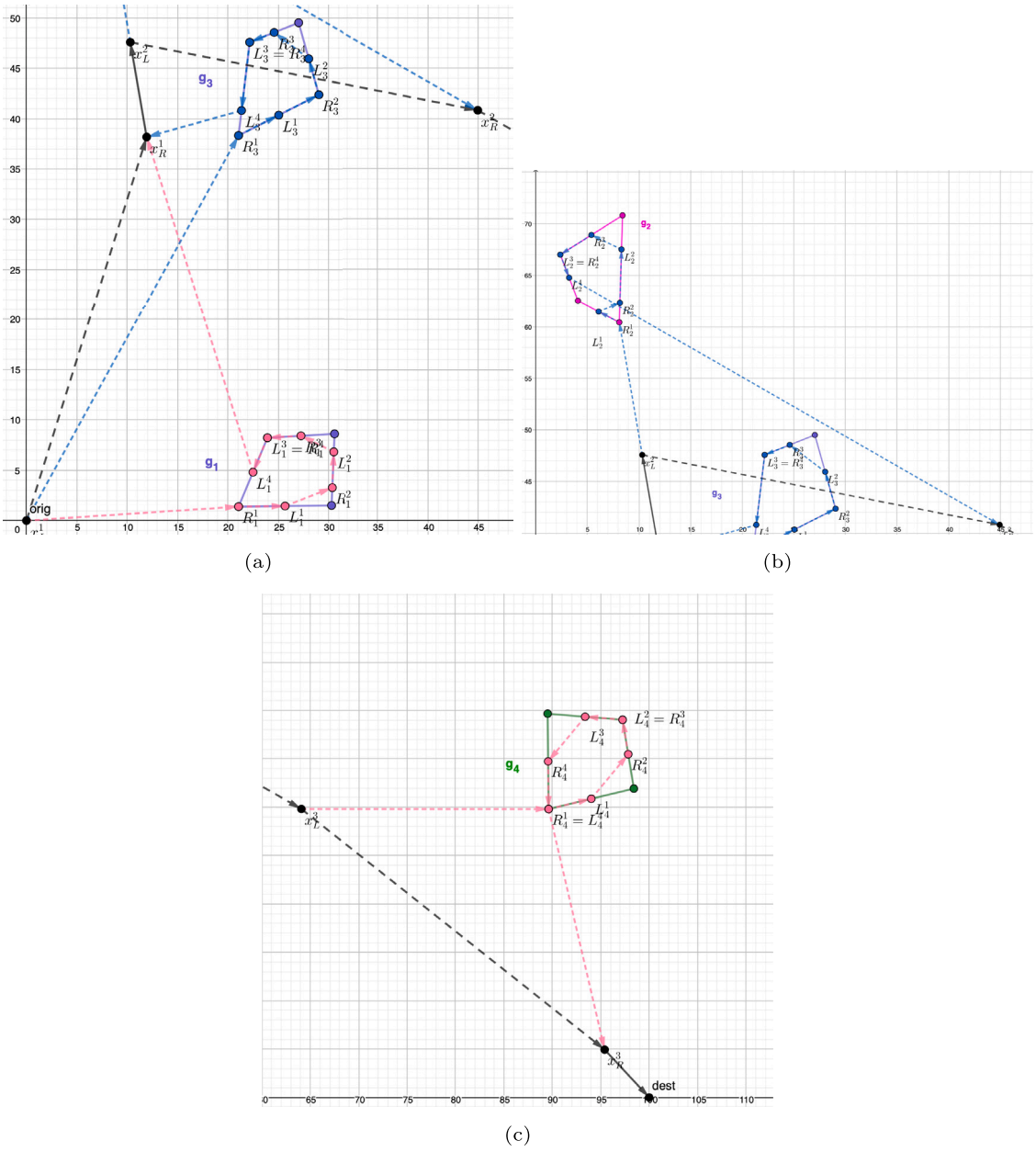
The reader may notice that the algorithm above can also be used to generate solutions for the partial overlapping model presented in Section 3.2 as any solution of the model AMMDRPG-CO is also feasible for the model AMMDRPG-PO one, as shown in Theorem 3.1.

6. Experimental results

In this section, we discuss the experimental results obtained testing the formulations presented in Section 3 and the matheuristic procedure proposed in Section 5 on a testbed of instances. In particular, we consider instances such as those used in Amorosi et al. (2021), where the targets to be visited are represented by grid graphs. More precisely, we generate a set of five instances each with five graphs having 4, 6, 8, 10, and 12 nodes, respectively. Similarly, we generate a set of five instances each with ten graphs, equally distributed with respect to the number of nodes (which always ranges between 4 and 12).

In order to place the graphs of a single instance, we follow the same procedure described in Amorosi et al. (2021). This is based on the division of an initial square of side 100 units in sub-squares and on the random selection of the graph locations among them. Then, in order to build the single graph, each of the randomly selected sub-squares is further partitioned in a number of sub-squares equal to the cardinality n of the set of nodes of the graph to build. Successively, two opposite corner sub-squares are considered to randomly select two points and build a rectangle whose diagonal joins these two points. Finally, a grid of n points is identified by locating $\frac{n}{m}$ equally spaced points on the two sides square, where m is randomly selected in the set of divisors of n. A perturbation on the coordinates of the points so obtained is applied, imposing that the perturbed points still belong to the original square. The resulting grid graph is obtained connecting each point to its adjacent ones lying on the same side and with the one located on the opposite side of the square. The reader can find and download all the instances used in this paper from Puerto and Valverde (2021). Moreover, we assume that the speed of the drones is twice that of the mothership and that the fleet of drones must visit a random fraction of each target graph or of each of its edges. These fractions are uniformly randomly sampled in the interval $(0, 1)$.

In our experiments, we consider that the number of drones varies between 1 and 3 and that the endurance of the drone (expressed as the maximum time the drone can operate when fully recharged) ranges between 20 and 60. Note that the case of a single drone is also included



(a) (b) (c)

Fig. 10. Zoom on the tour on each target graph provided by STEP 5.

Table 4
Instance parameter values.

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

in our experiments to compare the results and complexity of using one or more than one drone. The interested reader is referred to Amorosi et al. (2021) to analyse the complexity in terms of the gap of the model with a single drone. Table 4 reports a summary of the characteristics of our instances.

We code the matheuristic and the exact resolution of the model in Python 3.8.10. The mathematical programming formulation is implemented in Gurobi 9.1.2. All tests are run on an AMD® Epyc 7402p with 24-core processor × 8. Table 5 reports the results obtained by solving both variants of the AMMDRPG model on instances described above, by adopting Gurobi commercial solver. We consider the exact solution, setting a time limit of one hour, providing and not providing to the solver an initial solution computed by the matheuristic described in Section 5. More precisely, the first row of Table 5 indicates the variant of the model, and the second row reports the number of target graphs to be visited by the drone fleet (5 or 10). From the third row, we split each column into three sub-columns. The first three subcolumns 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 subcolumn 4 to subcolumn 15, we report, for each combination of the listed parameters characterising the instances, respectively, the average gap of the best solution found by Gurobi in one hour without initialisation by the solution provided by the matheuristic (wi), the average gap of the best solution found by Gurobi in one hour with initialisation by the solution obtained by the matheuristic (i), and the solution time, in seconds, of the matheuristic (TimeH).

Table 5
Comparison between partial and complete overlapping models.

Model			Complete overlapping						Partial overlapping					
$\|\mathcal{G}\|$			5			10			5			10		
N_D	$\|D\|$	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

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 difficult to solve even with small-sized 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 greater than that associated with the variant obligating visiting a given fraction of each target graph. Another thing we can observe is that the average gap increases with the number of target graphs for both variants of the problem.

Furthermore, 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 behaviour due to the fact that the feasible region of the partial overlapping variant contains that associated with the complete overlapping variant, as proven in Theorem 3.1. 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 initialisation of the solution obtained with the matheuristic becomes even harder. Indeed, the red 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 initialisation 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 be observed also for the exact solution of the complete overlapping model without initialisation 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. Taking into account 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 initialisation. Thus, initialisation 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 where the solver is not.

Furthermore, the average solution times of the matheuristic range between a minimum of 4 s and a maximum of 9.5 min. 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 behaviour can be observed for the partial overlapping version of the problem, for which the difference in terms of running time is even greater. 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 increase 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-sized instances. However, by exploiting this, the matheuristic is able to provide solutions for all instances quite quickly. Moreover, the quality of the solutions found by the matheuristic algorithm in a few seconds (or minutes) is comparable with the one of the solutions provided by Gurobi solver in one hour. Indeed, there are no significant differences between the average gaps associated with the solutions obtained with and without initialisation.

The boxplots in Fig. 11 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

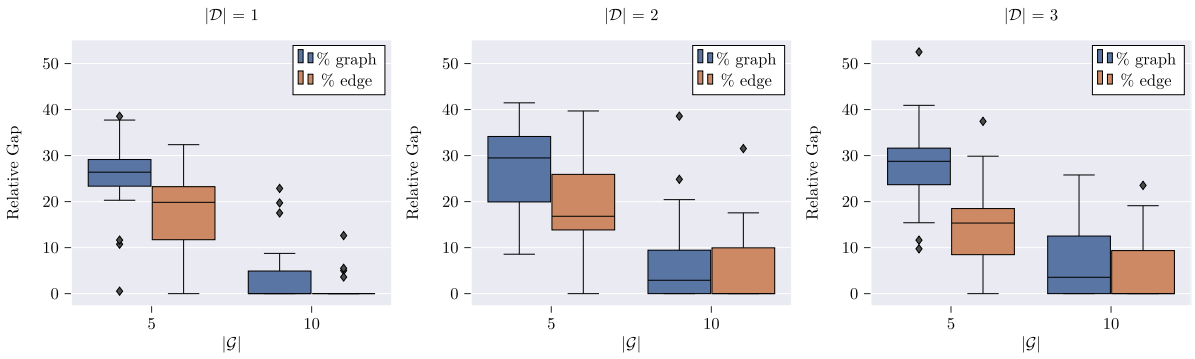


Fig. 11. Relative gap boxplots for AMMDRPG-CO.

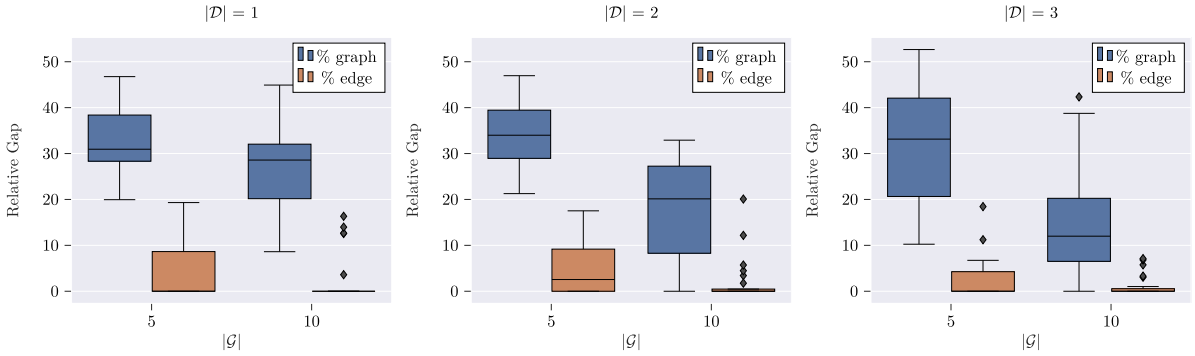


Fig. 12. Relative gap boxplots for AMMDRPG-PO.

of the mathematical programming model within the time limit, with the initialisation of the solution found by the matheuristic. Similarly, Fig. 12 reports the same information for the partial overlapping version of the problem.

From Fig. 11, 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 greater when a given fraction of each graph must be visited, independently of the size of the drone fleet. Additionally, its values decrease with the number of target graphs. A similar behaviour can also be observed for the partial overlapping variant of the problem, from Fig. 12. In the latter case, we can notice a larger difference between the relative gap values 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 a short computing times, especially for the version of the problem in which a given fraction of each edge must be visited.

For illustrative purposes, Fig. 13 shows one of the instances with five graphs and two drones adopted for the experimental results. Fig. 14 reports the solution of the partial overlapping version of the problem for this instance and a zoom on the mothership tour. We can observe that the origin (the black square) is the first launching point for the first drone (whose path is represented with green dotted lines) to visit the graph g_3. Then, the mothership moves to point x_L^2 for launching the second drone (whose path is represented with red dotted lines) that visits the graph g_5. Successively, the mothership moves to point x_R^3 for retrieving the same drone. From point x_L^4 the mothership launches again the second drone to visit the graph g_2. We can notice that this latter launching point coincides also with the entering point in the first visited edge of the graph g_2. From point x_R^5 the mothership retrieves the first drone that is launched again from point x_L^6 to visit the graph g_4. From the same point the mothership also retrieves the second drone ($x_L^6 = x_R^7$). Then, it moves to point x_L^8 for launching again the second drone for visiting the graph g_1. From point x_R^9 the mothership retrieves the first drone and, eventually, from point x_R^{10} it retrieves also the second drone. We can notice that the retrieving point x_R^{10} coincides with the exiting point for the drone at the end of its visit of the graph g_3. The mothership tours ends at the destination, that for this example coincides with the origin.

In Fig. 15 we show also the zoom on the tour on each target graph. The blue point represents the entering point in the graph and the yellow one represents the exiting point from the graph. The dotted arrows shows the path followed by the drones to visit each target graph.

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 $(0, 1)$.

6.1. Comparing the solutions for different configurations of the problem

In this subsection, we compare the relationship between the number of drones available and their endurance and the value of the objective function 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 6.

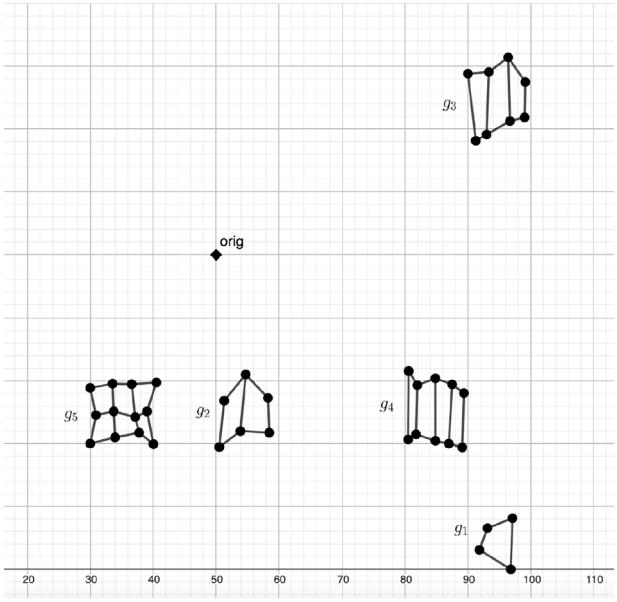


Fig. 13. Example of instance with 5 graphs.

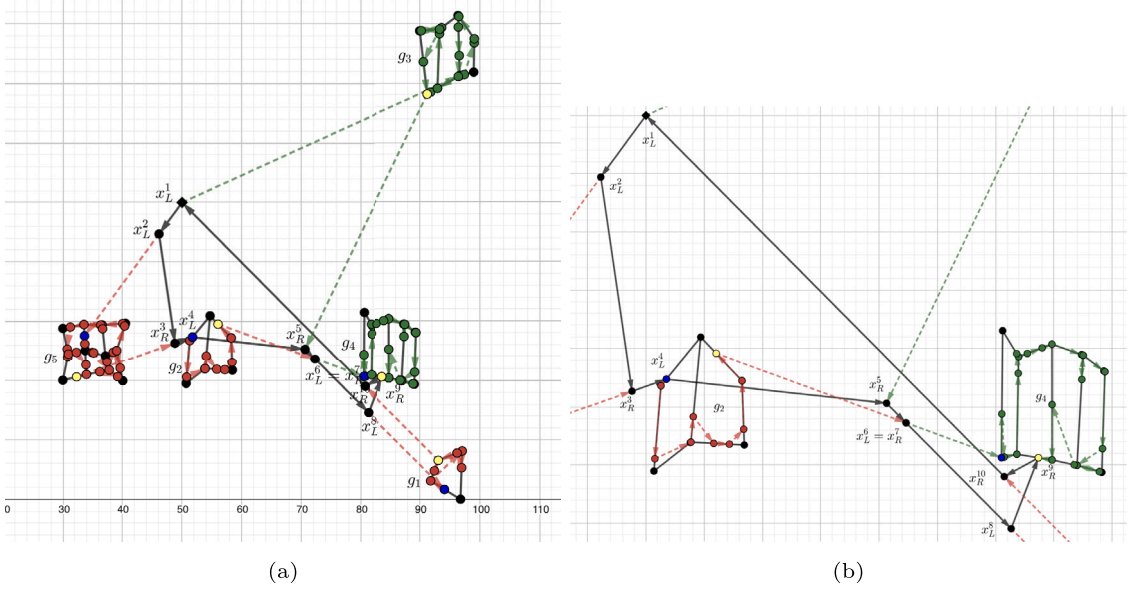


Fig. 14. Solution for the instance of Fig. 13 and zoom on the mothership tour.

Fig. 16 reports the objective value that depends on these parameters. The darker the intensity of the colour, the lower the objective value. As expected, our experiment confirms that both the larger number of drones and the greater 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 experienced 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 Cordoba Courtyards Festival (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. The owners of the courts decorate their houses with many flowers trying to win the award offered by the Municipality. During this competition, a festival is run in parallel with several artistic performances along six different paths located in different areas of the city, as shown in Fig. 17. In context of the pandemic, to monitor the situation to avoid concentration of people, we propose applying a system consisting of a 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 Reim (2020)). In our application, the reason for adopting a similar system is the possibility of simultaneously inspecting different paths in real-time, also reducing the risk of flying the helicopter over populated areas and the cost of moving the helicopter by minimising the total length of its tour.

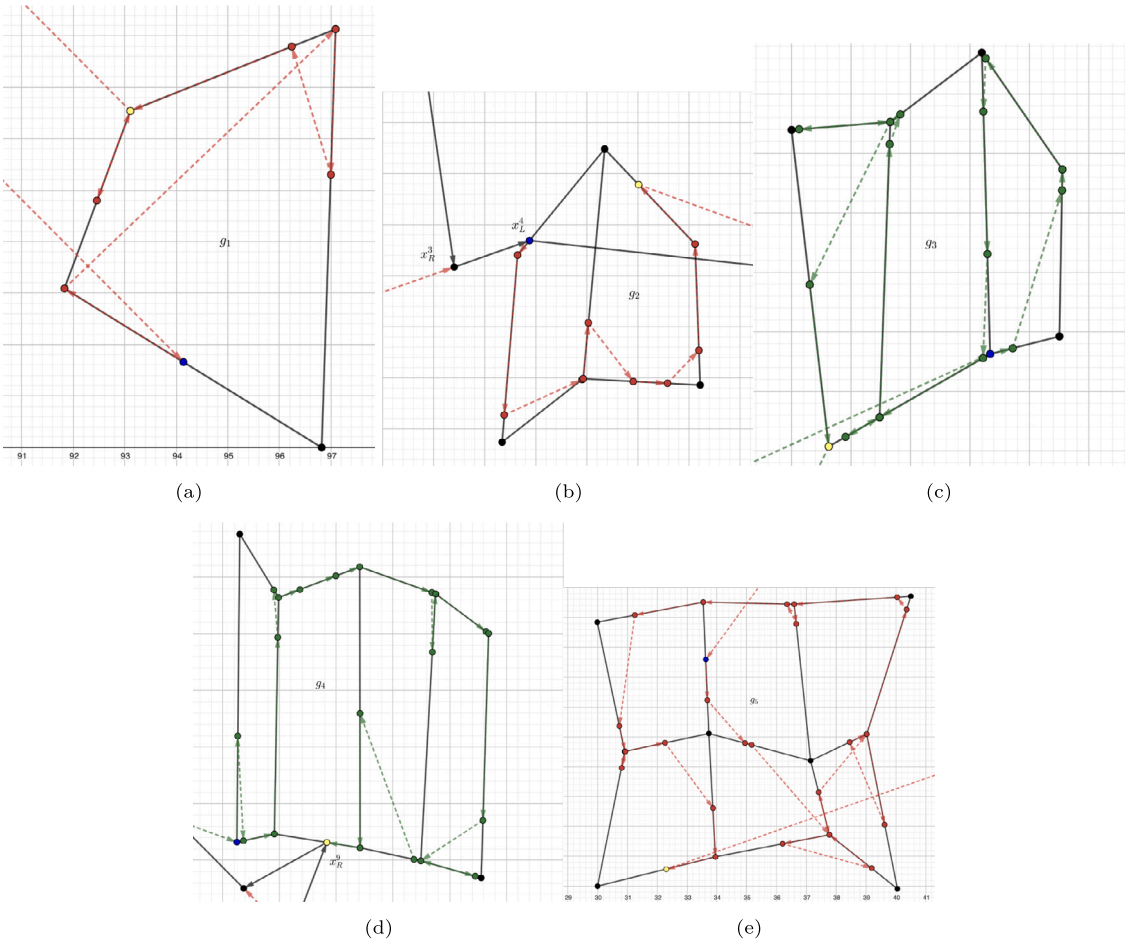


Fig. 15. Zoom on the tour on each target graphs.

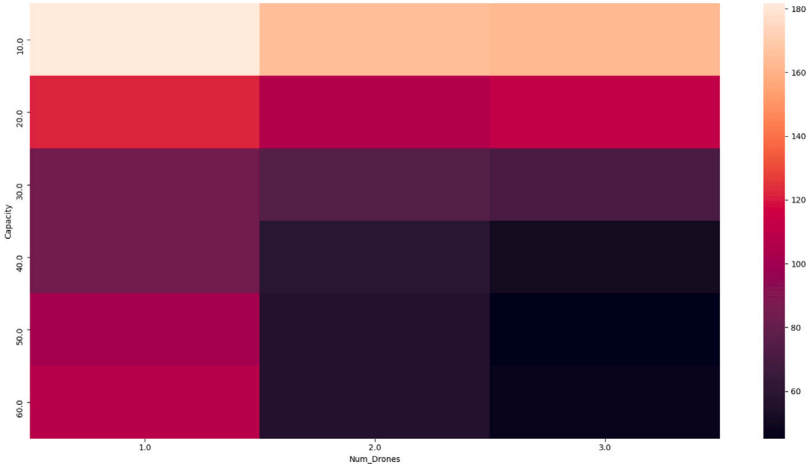


Fig. 16. Heatmap of the values of the objective function depending on the the number of drones and drone endurances. The darker the colour intensity, the lower the objective value.

We run the models presented in Section 3 in this scenario starting from the initial solution provided by the matheuristic, where the 6 coloured paths reported on the map of Fig. 17 represent the 6 target graphs to be visited, in this case, inspected, by the drone fleet. In addition, we assume that the drone speed is 43 km/h, while that of the helicopter is 30 km/h with the aim to minimise costs. Furthermore, we assume that the fleet consists of two drones with endurance equal to 7.5 min, and we impose that each target graph must be fully visited (inspected). As we can see in Figs. 18 and 19, the origin of the mothership tour coincides with the destination and is located in an area of the city where it is possible to take off and land the helicopter. Fig. 18 reports the tour followed by the helicopter in solving the complete overlapping version of the problem, after 4 h of running time. We can observe that the helicopter, starting from the origin, flies to the point x_R^1 which is the first retrieval 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

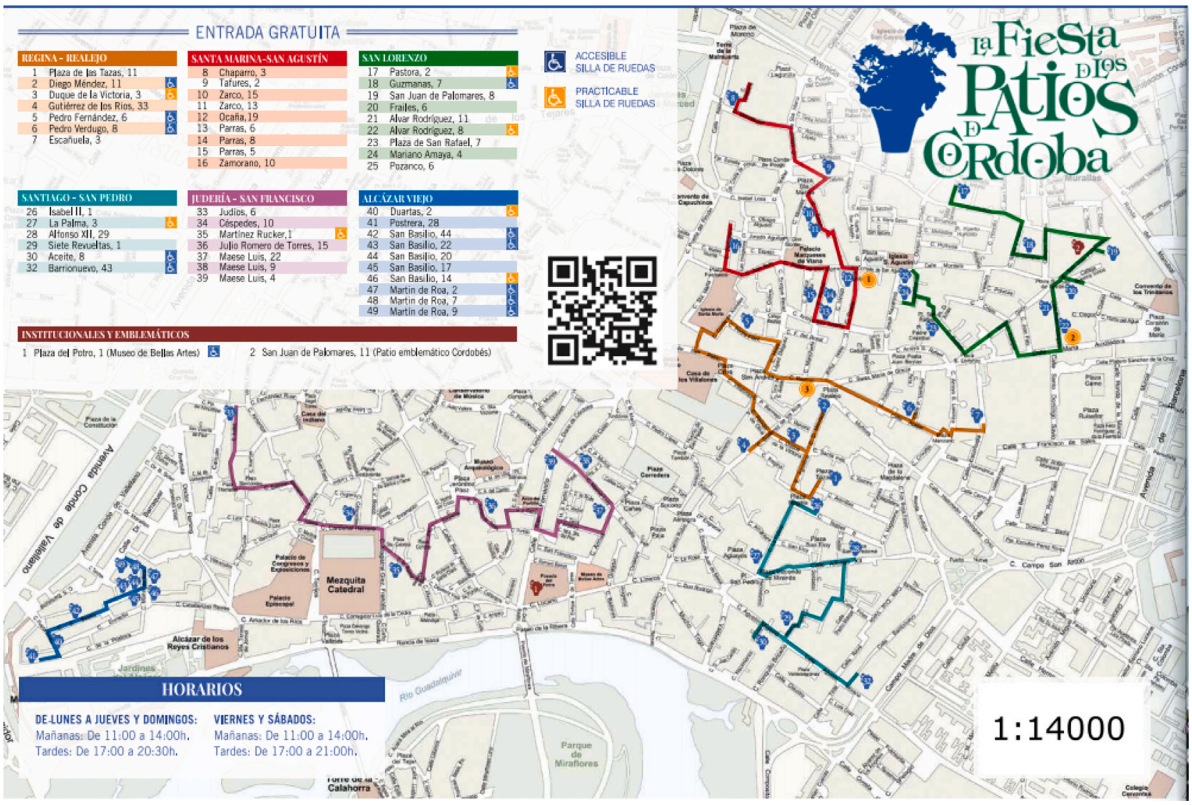


Fig. 17. Map of the Courtyards Festival in Cordoba.

second retrieval point, which coincides with the third launching point x_L^3. The helicopter then flies to x_R^3 to retrieve the drone completing the third mission. From the same point, the fourth and last mission starts and ends at the point x_R^4, which is also the final destination of the helicopter tour.

Fig. 18 also shows the tour (the violet and the red dotted paths) followed by the two drones to inspect the six paths. In particular, one drone starts from the origin ($orig = x_L^1$) to visit the path of "Alcazar Viejo". It is retrieved by the helicopter at the 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 the point x_R^2. From this last point, they are launched to perform the visits to the paths of "San Lorenzo" and "Regina-Realejo". Then, both drones are retrieved by the helicopter at the point x_R^3 where only one drone starts its last mission to visit the path of "Santiago-San Pedro". Meanwhile, the helicopter, which contains the other drone, flies to the point $x_R^4 = dest$ where it retrieves the other and ends its tour. Total time taken by helicopter is approximately 21 min. We can observe that in the drone tour on the graphs "San Lorenzo" and "Regina-Realejo", there are two edges whose duplicate is represented by a dotted segment in Fig. 18. They are associated with edges of the graph that are visited once, but travelled twice by the drone, in order to perform the inspection of the entire graph. Fig. 19 shows the solution of the partial overlapping version of the problem, always obtained by setting a time limit of 4 h. In this case, we can observe that the helicopter follows a different tour and that there are more launch and retrieval points due to the possibility of launching the drone before retrieving the other. From Fig. 19 we can also see that, unlike the complete overlapping version, both drones start their first mission from the origin $orig = x_L^1 = x_L^2$. 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 the 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". Meanwhile, 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 where 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}. 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", which are traversed twice represented with dotted segments in Fig. 19. The total travel time of the helicopter is 19 min. It is slightly lower than that associated with the solution of the complete overlapping version of the problem. Therefore, even if in this scenario we cannot observe significant 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 drone missions and a different location of the launch and retrieval points.

All details of this case study, including map coordinates, .lp models and solutions can be found in Puerto and Valverde (2021).

8. Concluding remarks

This paper has analysed the coordination problem that arises between a mothership vehicle and a fleet of drones that must coordinate their routes to minimise the makespan while visiting a set of targets modelled by graphs. We have presented exact mixed-integer non-linear programming formulations of the problem for its complete and partial overlapping versions. Furthermore, we strengthen the models with some valid inequalities for them.

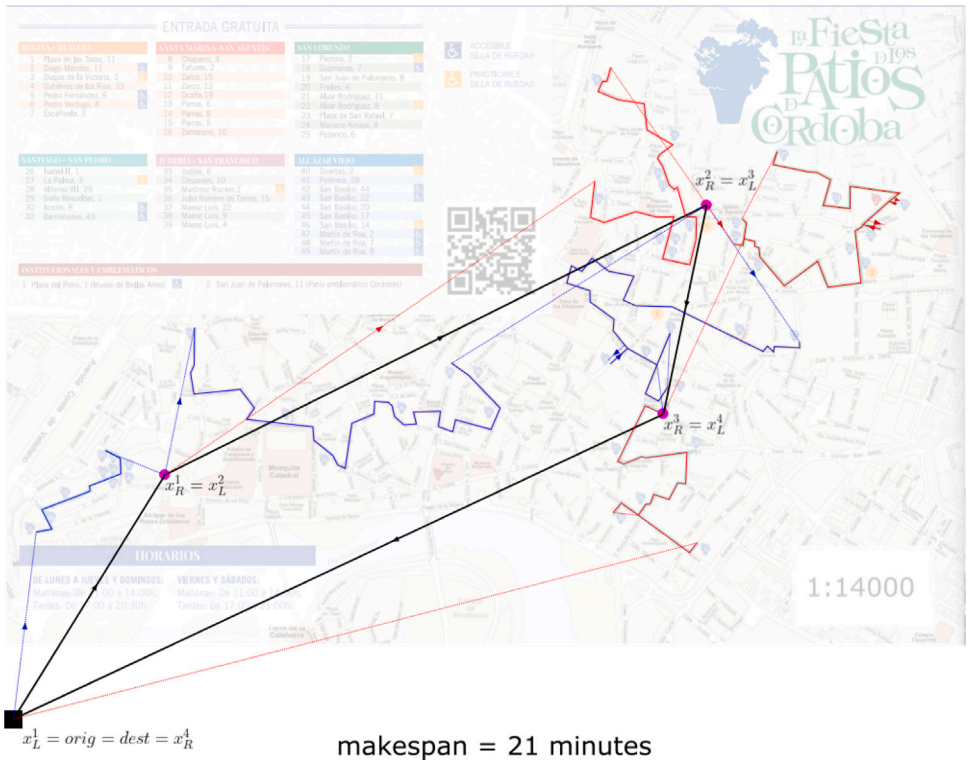


Fig. 18. The complete overlapping solution (CO).

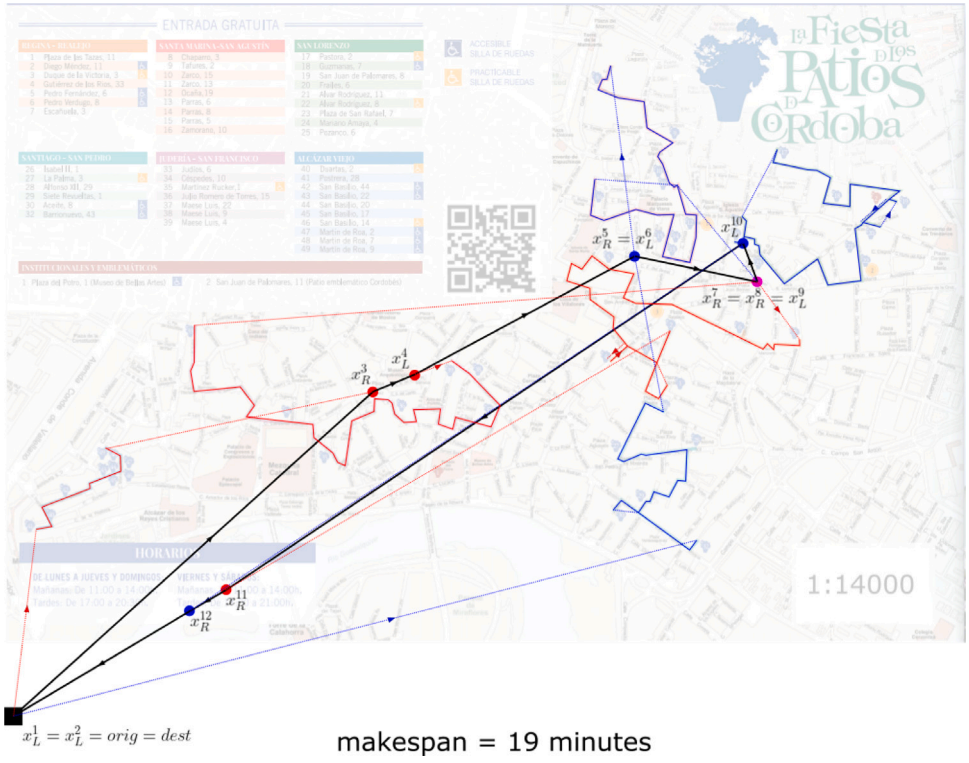


Fig. 19. The partial overlapping solution (PO).

Our computational results show that problem considered is very challenging to solve even in small and medium-sized 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. Furthermore, 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 article in the framework of the Courtyards Festival in the city of Cordoba, illustrating the solution obtained by adopting the formulation of the problem, in both versions of the model, and its solution by means of the initialisation provided by the proposed matheuristic.

The formulation and algorithms proposed in this paper can be seen as the first building block for coordination systems composed of a base vehicle and several drones. Further research on this topic must focus on finding faster and more accurate algorithms capable of solving larger instances. Other extensions that may be considered can take into account that the time the mothership takes to launch and retrieve the drones is not negligible, as well as handle the speed of the mothership and the drones as decision variables. Eventually, it is also possible to consider different shape of the graphs edges, like curve lines instead of straight lines. These problems are very interesting and beyond the scope of this paper and will be the focus of a follow-up research line.

CRediT authorship contribution statement

Lavinia Amorosi: Conceptualization, Methodology, Formal analysis, Writing – original draft, Writing – review & editing, Visualization. **Justo Puerto:** Supervision, Conceptualization, Methodology, Formal analysis, Writing – original draft, Writing – review & editing. **Carlos Valverde:** Conceptualization, Methodology, Formal analysis, Software, Writing – original draft, Writing – review & editing, Visualization.

Data availability

Data will be made available on request.

Acknowledgements

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, University of Rome, Sapienza grant number RM11916B7F962975 and Ministerio de Universidades, grant number FPU2018-04591.

Appendix

In this section, we report an extension of the MINLP formulation presented in Section 3 to deal with the case of non-homogeneous fleets of drones. We introduce the parameters or input data that formally describe the problem and are summarised in Table 7.

The formulation in this appendix is quite similar to the one in Section 3.1 but, due to 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 variables. For the sake of completeness, we include the complete set of constraints of these formulations, although some of them are similar to those in Section 3.1. Table 8 summarises all the decision variables used in this formulation.

Visits to graphs

Regarding the case of a homogeneous fleet of drones presented in Section 3, 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.

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

Problem parameters
$orig$: 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.
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.

Each edge e_g is parameterised 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. Furthermore, 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 in the segment.

As discussed in Section 2, we consider two modes of visit to the target graphs $g \in \mathcal{G}$:

(i) Visiting a fraction α^{e_g} of each edge e_g which can be modelled 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 parameterisations of the entry and exit points associated with with each edge e_g must be greater than the fraction of the length of e_g required for traverse.

(ii) Visit 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 the edges chosen to be crossed must be greater than the fraction of the length of g that must be traversed.

In both cases, the corresponding constraints are non-linear. To linearise them, we need to introduce a binary variable entry^{e_g} that determines the direction of travel at the edge e_g, as well as the definition of the auxiliary variables $v_{\min}^{e_g}$ and $v_{\max}^{e_g}$ of the access and exit points in that segment. Then, for each edge e_g, the absolute value constraint (α-E) can be represented by:

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

The first four inequalities model the standard trick of linearisation of the absolute value. The last constraint ensures that the value of

Table 8
Decision variables for AMMDRPG with a 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 linearising 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.
$\nu^{e_g}_{\min}$ and $\nu^{e_g}_{\max} \in [0,1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: auxiliary variables used for linearising expressions.
$p^{e_g} \in [0,1],\ \forall e_g \in E_g\ (g \in \mathcal{G})$: auxiliary variable used for modelling the product of μ^{e_g} and $\lvert\lambda^{e_g} - \rho^{e_g}\rvert$.
$x^o_L \in \mathbb{R}^2,\ \forall o \in \mathcal{O}$: coordinates representing the point where the mothership launches the drones at operation o.
$x^o_R \in \mathbb{R}^2,\ \forall o \in \mathcal{O}$: 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 travelled by the drone δ from the launching point x^o_L 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 modelling 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 travelled by the drone from the retrieval 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 travelled by the drone from the launching point L^{e_g} on e_g to the retrieval 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 modelling 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 travelled by the drone δ from the last visiting point L^{e_g} on e_g to the retrieval point x^o_R 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 modelling the product of $d_R^{e_g o\delta}$ and $v^{e_g o\delta}$.
$d_{orig} \geq 0$: distance from the origin $orig$ to the first launching point x^1_L.
$d^o_{LR} \geq 0,\ \forall o \in \mathcal{O}$: representing the distance travelled by the mothership from the launching point x^o_L to the retrieval point x^o_R at operation o.
$d^o_{RL} \geq 0,\ \forall o \in \mathcal{O} \setminus \lvert\mathcal{O}\rvert$: representing the distance travelled by the mothership from the retrieval point x^o_R at operation o to the launching point $x^{(o+1)}_L$ at operation $o+1$.
$d_{dest} \geq 0$: distance from the last retrieval point $x^{\lvert\mathcal{O}\rvert}_R$ to the destination $dest$.
$time^o_D \geq 0,\ \forall o \in \mathcal{O}$: maximum time spent by a drone during operation o.
$time^o_M \geq 0,\ \forall o \in \mathcal{O}$: time spent by the mothership to go from the launching point x^o_L to the retrieval point x^o_R of operation o.
$time_M \geq 0$: total time spent by the mothership to go from the origin to the destination (makespan).

the linear expression of the absolute value is higher than the required fraction α^{e_g}.

Similarly, (α-G) can be linearised as follows:

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

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

where p^{e_g} is the auxiliary variable that represents the product of the binary variable μ^{e_g} and the difference in absolute values $|\rho^{e_g} - \lambda^{e_g}|$. The first four inequalities again linearise the absolute value expression. 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 greater than the fraction of the length of g required to be traversed.

Elimination of subtours

As already presented in Section 3, to prevent the existence of subtours within each graph $g \in \mathcal{G}$ that the drone must visit, one can include, among others, the compact formulation that uses Miller-Tucker-Zemlin constraints (MTZ) or subtour elimination constraints (SEC).

For the MTZ formulation, we use continuous variables s^{e_g}, defined in Table 8, which 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 $g \in \mathcal{G}$:

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

To find the 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 described in Section 3.1, we adopt the concept of operation. Let us denote by $\mathcal{O}$ the set of operations that the

mothership and drone fleet have to perform. These operations are visits to the different graphs in $\mathcal{G}$ with the required constraints. An operation $o \in \mathcal{O}$ refers to the event in which the mothership launches some drones from a take-off location, denoted by x_L^o and then takes them back to a retrieval location x_R^o.

For each operation $o \in \mathcal{O}$, each of the drones launched from the mothership must follow a path that starts from and returns 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:

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

We model the route that the drone follows using the binary variables $u^{e_g o\delta}$, $z^{e_g e'_g}$ and $v^{e_g o\delta}$ defined in Table 8.

$$\sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} u^{e_g o\delta} \le 1, \qquad \forall o\in\mathcal{O}, \forall \delta\in\mathcal{D}. \tag{Drone ROUTE$^{\Delta}_1$-CO}$$

$$\sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g o\delta} \le 1, \qquad \forall o\in\mathcal{O}, \forall \delta\in\mathcal{D}. \tag{Drone ROUTE$^{\Delta}_2$-CO}$$

$$\sum_{e_g\in E_g}\sum_{o\in\mathcal{O}}\sum_{\delta\in\mathcal{D}} u^{e_g o\delta} = 1, \qquad \forall g\in\mathcal{G}, \tag{Drone ROUTE$^{\Delta}_3$-CO}$$

$$\sum_{e_g\in E_g}\sum_{o\in\mathcal{O}}\sum_{\delta\in\mathcal{D}} v^{e_g o\delta} = 1, \qquad \forall g\in\mathcal{G}, \tag{Drone ROUTE$^{\Delta}_4$-CO}$$

$$\sum_{e_g\in E_g} u^{e_g o\delta} = \sum_{e_g\in E_g} v^{e_g o\delta}, \qquad \forall g\in\mathcal{G}, \forall o\in\mathcal{O}, \forall \delta\in\mathcal{D}, \tag{Drone ROUTE$^{\Delta}_5$-CO}$$

$$\sum_{o\in\mathcal{O}}\sum_{\delta\in\mathcal{D}} u^{e_g o\delta} + \sum_{e'_g\in E_g} z_g^{e'_g e_g} = \mu^{e_g}, \qquad \forall e_g\in E_g : g\in\mathcal{G}, \tag{Drone ROUTE$^{\Delta}_6$-CO}$$

$$\sum_{o\in\mathcal{O}}\sum_{\delta\in\mathcal{D}} v^{e_g o\delta} + \sum_{e'_g\in E_g} z_g^{e_g e'_g} = \mu^{e_g}, \qquad \forall e_g\in E_g : g\in\mathcal{G}. \tag{Drone ROUTE$^{\Delta}_7$-CO}$$

The inequalities (Drone ROUTE$^{\Delta}_1$-CO) and (Drone ROUTE$^{\Delta}_2$-CO) state that a drone δ visits at most one graph g at operation o. Constraints (Drone ROUTE$^{\Delta}_3$-CO) and (Drone ROUTE$^{\Delta}_4$-CO) ensure that each graph is visited at some operation o by some drone δ. Eqs. (Drone ROUTE$^{\Delta}_5$-CO) ensure that the operation of entering and exiting the graph g occurs in the same operation o and is performed by the same drone δ. Constraints (Drone ROUTE$^{\Delta}_6$-CO) 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 (Drone ROUTE$^{\Delta}_7$-CO) 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 is to find a feasible solution that minimises the total time taken 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_{orig}, d^o_{RL}, d^o_{LR} and d_{dest} defined in Table 8. This can be done by means of the following constraints:

$$\|x_L^o - R^{e_g}\| \le d_L^{e_g o\delta}, \qquad \forall e_g\in E_g : g\in\mathcal{G}, \forall o\in\mathcal{O}, \forall\delta\in\mathcal{D}, \tag{Drone DIST$^{\Delta}_1$-CO}$$

$$\|R^{e_g} - L^{e_g}\| \le d^{e_g}, \qquad \forall e_g\in E_g : g\in\mathcal{G}, \tag{Drone DIST$^{\Delta}_2$-CO}$$

$$\|R^{e_g} - L^{e'_g}\| \le d^{e_g e'_g}, \qquad \forall e_g\ne e'_g\in E_g : g\in\mathcal{G}, \tag{Drone DIST$^{\Delta}_3$-CO}$$

$$\|L^{e_g} - x_R^o\| \le d_R^{e_g o\delta}, \qquad \forall e_g : g\in\mathcal{G}, \forall o\in\mathcal{O}, \forall\delta\in\mathcal{D}, \tag{Drone DIST$^{\Delta}_4$-CO}$$

$$\|orig - x_L^1\| \le d_{orig}, \tag{Mothership DIST$^{\Delta}_1$-CO}$$

$$\|x_L^o - x_R^o\| \le d^o_{LR}, \qquad \forall o\in\mathcal{O}. \tag{Mothership DIST$^{\Delta}_2$-CO}$$

$$\|x_R^o - x_L^{o+1}\| \le d^o_{RL}, \qquad \forall o\in\mathcal{O} : o < |\mathcal{O}|, \tag{Mothership DIST$^{\Delta}_3$-CO}$$

$$\|x_R^{|\mathcal{O}|} - dest\| \le d_{dest}, \tag{Mothership DIST$^{\Delta}_4$-CO}$$

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

$$time^o_\delta \ge \frac{1}{v_\delta}\left(\sum_{e_g\in E_g} u^{e_g o\delta} d_L^{e_g o\delta} + \sum_{e_g, e'_g\in E_g} z^{e_g e'_g} d^{e_g e'_g} + \sum_{e_g\in E_g}\mu^{e_g} d^{e_g} + \sum_{e_g\in E_g} v^{e_g o\delta} d_R^{e_g o\delta}\right) - N_\delta\left(1 - \sum_{e_g\in E_g} u^{e_g o\delta}\right) \tag{Drone TIME$^{\Delta}_o$-CO}$$

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

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

$$time^o_D \ge time^o_\delta \ \ \forall\delta\in\mathcal{D} \tag{Drone MAX TIME$^{\Delta}_o$-CO}$$

Constraints (Mothership TIME$^{\Delta}_o$-CO) define the time that the mothership takes to go from the launch point x_L^o to the retrieval point x_R^o associated with the operation o.

$$time^o_M = \frac{d^o_{LR}}{v_M} \ \ \forall o\in\mathcal{O} \tag{Mothership TIME$^{\Delta}_o$-CO}$$

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

$$time_M = \frac{1}{v_M}(d_{orig} + \sum_{o\in\mathcal{O}}(d^o_{LR} + d^o_{RL}) + d_{dest}) \tag{Mothership TIME$^{\Delta}$-CO}$$

Coordination and endurance constraints

The coordination between the drones and the mothership must ensure that the maximum time $time^o_D$ 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 retrieval point during operation o. To this end, we need to define the following coordination constraint for each operation $o \in \mathcal{O}$:

$$time^o_D \le time^o_M. \tag{DCW$^{\Delta}$-CO}$$

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

$$time^o_\delta \leq N_\delta. \qquad \text{(Endurance}^\Delta\text{-CO)}$$

AMMDRPG-complete overlapping formulation (with non-homogeneous fleet of drones)

Putting together all the constraints introduced before, the following formulation minimises the total time travelled by the mothership (makespan), ensuring coordination with the drone fleet while guaranteeing the required coverage of the target graphs.

$$\min \quad time_M$$

(AMMDRPG-CO with a non-homogeneous fleet of drones)

s.t. (α-E) or (α-G),
(MTZ_1)–(MTZ_2) or (SEC),
(Drone ROUTE$^\Delta_1$-CO)–(Drone ROUTE$^\Delta_7$-CO),
(Drone DIST$^\Delta_1$-CO)–(Drone DIST$^\Delta_4$-CO),
(Mothership DIST$^\Delta_1$-CO)–(Mothership DIST$^\Delta_4$-CO),
(Drone TIME$^\Delta_o$-CO), (Drone MAX TIME$^\Delta_o$-CO),
(Mothership TIME$^\Delta_o$-CO), (Mothership TIME$^\Delta$-CO),
(DCW$^\Delta$-CO), (Endurance$^\Delta$-CO)

The objective function accounts for the time travelled by the mothership (makespan). Constraints (Drone ROUTE$^\Delta_1$-CO)–(Drone ROUTE$^\Delta_7$-CO) model the route followed by the drone $\delta \in \mathcal{D}$, (MTZ_1)–(MTZ_2) or (SEC) ensure that the displacement of the drone $\delta \in \mathcal{D}$ assigned to the target graph $g \in \mathcal{G}$ is a route, (α-E) or (α-G) define what is required in each visit to a target graph. Finally, constraints (Drone DIST$^\Delta_1$-CO)–(Mothership DIST$^\Delta_4$-CO) set the variables $d_L^{e_g o\delta}$, d^{e_g}, $d^{e_g e'_g}$, $d_R^{e_g o\delta}$, d_{orig}, d^o_{RL}, d^o_{LR} and d_{dest}, defined in Table 8, which represent the 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 linearise the bilinear terms of (Drone TIME$^\Delta_o$-CO). We use the McCormick's envelopes by adding variables $p \geq 0$ representing the products. To strengthen the formulations, we provide tight upper and lower bounds for these 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 (AMMDRPG-CO with a non-homogeneous fleet of drones).

Big M constants bounding the distance from the launch/retrieval point on the path followed by the mothership to the retrieval/launch point on the target graph $g \in \mathcal{G}$

To linearise the first addend in (DCW-CO), 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:

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

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 that joins them. Hence, because we are minimising the distance travelled by the mothership, every launch or retrieval 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:

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

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

Bounds on the bigM constants for the distance from the launch to the retrieval 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 linearised by the following constraints:

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

Since we take 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 by 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 by testing the formulation of AMMDRPG with a non-homogeneous fleet of drones, presented in Appendix, on the same set of instances described in Section 6.

Table 5 reports the results obtained by adopting the Gurobi commercial solver. We consider the exact solution that provides and does not provide an initial solution computed by the matheuristic described in Section 5. More precisely, the first column of Table 9 indicates the number of target graphs to be visited by the drone fleet, the second column reports the endurance of the drones, and 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 drone fleet. This last column contains three subcolumns reporting, for each cardinality of the set $\mathcal{D}$, respectively, the average gap without initialisation (wi), the average gap with initialisation 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 for these experiments is set equal to 2 h.

We can observe that the value of the average gap ranges between a minimum of 0.67 and a maximum of 0.97. This shows that the model is hard to solve even with small-sized instances. Furthermore, we can see that, in most cases, the average gap associated with the variant of the model consisting of visiting a given fraction of each edge is greater than the one associated with the variant that imposes visiting a given fraction of each target graph. Another thing we can observe is that the average gap increases with the number of drones and decreases with the drone endurance.

Regarding the number of target graphs, we can see that, increasing it from 5 to 10, the exact method without initialisation of the solution obtained with the matheuristic is even harder. Indeed, the red 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

Table 9
Comparison between an exact solution with and without initialisation by the matheuristic solution.

$\|\mathcal{G}\|$	N_D	v.t.	$\|\mathcal{D}\|$								
			1			2			3		
			Gap (wi)	Gap (i)	TimeH	Gap (wi)	Gap (i)	TimeH	Gap (wi)	Gap (i)	TimeH
5	20	e	0.82	0.83	61.56	0.9	0.92	63.80	0.91	0.93	60.87
		g	0.80	0.79	44.97	0.92	0.89	37.32	0.96	0.94	39.05
	30	e	0.80	0.83	65.21	0.82	0.85	64.41	0.90	0.92	63.34
		g	0.71	0.76	55.77	0.88	0.84	44.36	0.91	0.91	44.59
	40	e	0.78	0.81	68.81	0.82	0.83	64.80	0.86	0.91	63.19
		g	0.73	0.74	43.92	0.84	0.81	38.27	0.90	0.85	37.51
	50	e	0.74	0.77	66.67	0.80	0.81	63.86	0.86	0.85	63.51
		g	0.67	0.71	43.42	0.89	0.81	43.98	0.83	0.80	44.35
	60	e	0.72	0.76	67.68	0.80	0.82	66.08	0.82	0.84	64.40
		g	0.73	0.78	44.69	0.86	0.79	40.63	0.85	0.82	50.01
10	20	e	0.85	0.83	137.93	–	0.92	128.53	–	0.95	124.44
		g	0.85 (2)	0.81	119.20	0.97 (2)	0.90	83.50	0.97 (3)	0.97	70.00
	30	e	0.81	0.81	159.00	0.88 (3)	0.87	132.15	0.93 (2)	0.95	127.35
		g	0.83 (1)	0.80	132.67	0.86 (3)	0.86	80.29	0.9 (1)	0.91	76.72
	40	e	0.78	0.79	191.37	0.84	0.85	131.26	0.89 (1)	0.92	132.10
		g	0.80	0.80	115.00	0.85 (3)	0.87	68.39	0.92 (1)	0.96	69.40
	50	e	0.78	0.81	188.32	0.85 (1)	0.88	134.01	0.91 (3)	0.93	132.82
		g	0.80	0.80	87.23	0.84 (3)	0.83	66.14	0.92 (2)	0.92	64.94
	60	e	0.82	0.84	155.27	0.83 (2)	0.86	131.94	0.87 (3)	0.92	130.11
		g	0.78	0.77	97.89	0.88 (2)	0.87	76.53	0.92 (3)	0.94	69.53

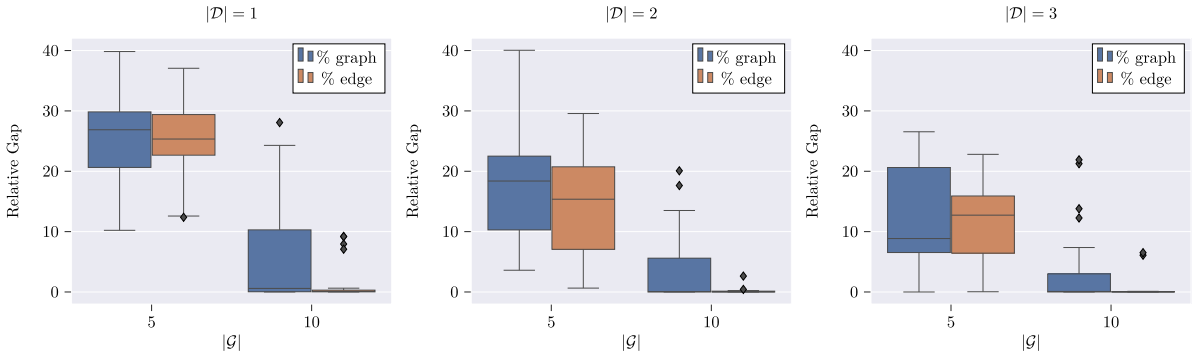


Fig. 20. Relative gap boxplots.

of these instances). The number of instances not solved increases with the number of drones. Furthermore, for the minimum level of endurance, the exact solution of the model without initialisation 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 initialisation. Thus, initialisation 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 s to a maximum of 3 min. 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 exact solution method given by solving the formulation is very challenging even for small-sized instances. However, by exploiting this, the matheuristic is able to provide solutions for all instances quite quickly.

The boxplots in Fig. 20 represent the relative gap of the solution provided by the matheuristic concerning that provided by the exact solution of the mathematical programming model within the time limit, with the initialisation 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 concerning the other case, that is, when a given fraction of each graph to be visited is imposed. In particular, for instances with 10 target graphs, the relative gap is not greater than 25, except an 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 decreases further, and is reduced to 10 when the fleet consists of 3 drones. Furthermore, when a given fraction of each edge to be visited is imposed, the relative gap is even smaller, around zero, in most of cases. 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.

References

AltiGator, 2015. A drone to rescue immigrants. URL: https://altigator.com/en/rescue-immigrants-with-a-drone/.

Amorosi, L., Caprari, R., Crainic, T., Dell'Olmo, P., Ricciardi, N., 2020. CIRRELT-2020-17 An Integrated Routing-Scheduling Model for a Hybrid UAV-Based Delivery System. CIRRELT.

Amorosi, L., Chiaraviglio, L., D'Andreagiovanni, F., Blefari-Melazzi, N., 2018. Energy-efficient mission planning of UAVs for 5G coverage in rural zones. In: 2018 IEEE International Conference on Environmental Engineering. EE, pp. 1–9. http://dx.doi.org/10.1109/EE1.2018.8385250.

Amorosi, L., Chiaraviglio, L., Galan-Jimenez, J., 2019. Optimal energy management of uav-based cellular networks powered by solar panels and batteries: Formulation and solutions. IEEE Access 7, 53698–53717. http://dx.doi.org/10.1109/ACCESS.2019.2913448.

Amorosi, L., Puerto, J., Valverde, C., 2021. Coordinating drones with mothership vehicles: The mothership and drone routing problem with graphs. Comput. Oper. Res. 136, 105445. http://dx.doi.org/10.1016/j.cor.2021.105445.

Amorosi, L., Puerto, J., Valverde, C., 2022. An extended model of coordination of an all-terrain vehicle and a multivisit drone. Int. Trans. Oper. Res. http://dx.doi.org/10.1111/itor.13179.

Blanco, V., Fernández, E., Puerto, J., 2017. Minimum spanning trees with neighborhoods: Mathematical programming formulations and solution methods. European J. Oper. Res. 262 (3), 863–878. http://dx.doi.org/10.1016/j.ejor.2017.04.023.

Blanco, V., Puerto, J., Ben-Ali, S.E.-H., 2013. Revisiting several problems and algorithms in continuous location with ℓ_τ norms. Comput. Optim. Appl. 58, 563–595.

Campbell, J.F., Corberán, Á., Plana, I., Sanchis, J.M., 2018. Drone arc routing problems. Networks 72 (4), 543–559. http://dx.doi.org/10.1002/net.21858.

Campbell, J.F., Corberán, Á., Plana, I., Sanchis, J.M., Segura, P., 2021. Solving the length constrained K-drones rural postman problem. European J. Oper. Res. 292 (1), 60–72. http://dx.doi.org/10.1016/j.ejor.2020.10.035.

Campbell, J.F., Sweeney, D., Zhang, J., 2017. Strategic design for delivery with trucks and drones. Comput. Sci..

Carlsson, J.G., Song, S., 2018. Coordinated logistics with a truck and a drone. Manage. Sci. 64 (9), 4052–4069. http://dx.doi.org/10.1287/mnsc.2017.2824.

Chiaraviglio, L., Amorosi, L., Blefari-Melazzi, N., Dell'olmo, P., Lo Mastro, A., Natalino, C., Monti, P., 2019a. Minimum cost design of cellular networks in rural areas with UAVs, optical rings, solar panels, and batteries. IEEE Trans. Green Commun. Netw. 3 (4), 901–918. http://dx.doi.org/10.1109/TGCN.2019.2936012.

Chiaraviglio, L., Amorosi, L., Blefari-Melazzi, N., Dell'Olmo, P., Natalino, C., Monti, P., 2018. Optimal design of 5G networks in rural zones with UAVs, optical rings, solar panels and batteries. In: 2018 20th International Conference on Transparent Optical Networks. ICTON, pp. 1–4. http://dx.doi.org/10.1109/ICTON.2018.8473712.

Chiaraviglio, L., Amorosi, L., Malandrino, F., Chiasserini, C.F., Dell'Olmo, P., Casetti, C., 2019b. Optimal throughput management in UAV-based networks during disasters. In: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). pp. 307–312. http://dx.doi.org/10.1109/INFCOMW.2019.8845190.

Chung, S.H., Sah, B., Lee, J., 2020. Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions. Comput. Oper. Res. 123, 105004. http://dx.doi.org/10.1016/j.cor.2020.105004.

Dayarian, I., Savelsbergh, M., Clarke, J.-P., 2020. Same-day delivery with drone resupply. Transp. Sci. 54 (1), 229–249. http://dx.doi.org/10.1287/trsc.2019.0944.

Dell'Amico, M., Montemanni, R., Novellani, S., 2021. Algorithms based on branch and bound for the flying sidekick traveling salesman problem. Omega 104, 102493. http://dx.doi.org/10.1016/j.omega.2021.102493.

Dönmez, Z., Kara, B.Y., Karsu, Ö., Saldanha-da Gama, F., 2021. Humanitarian facility location under uncertainty: Critical review and future prospects. Omega 102, 102393. http://dx.doi.org/10.1016/j.omega.2021.102393.

Ferrandez, S.M., Harbison, T., Weber, T., Sturges, R., Rich, R., 2016. Optimization of a truck-drone in tandem delivery network using k-means and genetic algorithm. J. Ind. Eng. Manage. 9 (2), 374–388. http://dx.doi.org/10.3926/jiem.1929.

Garone, E., Naldi, R., Casavola, A., Frazzoli, E., 2010. Cooperative mission planning for a class of carrier-vehicle systems. In: 49th IEEE Conference on Decision and Control. CDC, pp. 1354–1359. http://dx.doi.org/10.1109/CDC.2010.5717171.

Jiménez, J.G., Chiaraviglio, L., Amorosi, L., Blefari-Melazzi, N., 2018. Multi-period mission planning of UAVs for 5G coverage in rural areas: a heuristic approach. In: 2018 9th International Conference on the Network of the Future. NOF, pp. 52–59. http://dx.doi.org/10.1109/NOF.2018.8598123.

Mathew, N., Smith, S.L., Waslander, S.L., 2015. Planning paths for package delivery in heterogeneous multirobot teams. IEEE Trans. Autom. Sci. Eng. 12 (4), 1298–1308. http://dx.doi.org/10.1109/TASE.2015.2461213.

Murray, C.C., Chu, A.G., 2015. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. Transp. Res. C 54, 86–109. http://dx.doi.org/10.1016/j.trc.2015.03.005.

Otto, A., Agatz, N., Campbell, J., Golden, B., Pesch, E., 2018. Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey. Networks 72, 1–48. http://dx.doi.org/10.1002/net.21818.

Pei, Z., Dai, X., Yuan, Y., Du, R., Liu, C., 2021. Managing price and fleet size for courier service with shared drones. Omega 104, 102482. http://dx.doi.org/10.1016/j.omega.2021.102482.

Poikonen, S., Golden, B., 2020a. The mothership and drone routing problem. INFORMS J. Comput. 32 (2), 249–262. http://dx.doi.org/10.1287/ijoc.2018.0879.

Poikonen, S., Golden, B., 2020b. Multi-visit drone routing problem. Comput. Oper. Res. 113, 104802. http://dx.doi.org/10.1016/j.cor.2019.104802.

Puerto, J., Valverde, C., 2021. Project: Instances for the case study of the all terrain mothership multiple drone routing problem with graphs (AMMDRPG). URL: https://github.com/z72vamac/case_study_AMMDRPGST.

Reim, G., 2020. US army catches 'air-launched effect' drones in mid-air using another UAV. URL: https://www.flightglobal.com/military-uavs/us-army-catches-air-launched-effect-drones-in-mid-air-using-another-uav/140498.article.

Tokekar, P., Hook, J.V., Mulla, D., Isler, V., 2016. Sensor planning for a symbiotic UAV and UGV system for precision agriculture. IEEE Trans. Robot. 32 (6), 1498–1511. http://dx.doi.org/10.1109/TRO.2016.2603528.

Trotta, A., Andreagiovanni, F.D., Di Felice, M., Natalizio, E., Chowdhury, K.R., 2018. When UAVs ride a bus: Towards energy-efficient city-scale video surveillance. In: IEEE INFOCOM 2018 - IEEE Conference on Computer Communications. pp. 1043–1051. http://dx.doi.org/10.1109/INFOCOM.2018.8485863.

Ulmer, M.W., Thomas, B.W., 2018. Same-day delivery with heterogeneous fleets of drones and vehicles. Networks 72 (4), 475–505. http://dx.doi.org/10.1002/net.21855.

Wen, T., Zhang, Z., Wong, K.K.L., 2016. Multi-objective algorithm for blood supply via unmanned aerial vehicles to the wounded in an emergency situation. PLoS One 11 (5), http://dx.doi.org/10.1371/journal.pone.0155176.