Coordinating drones with mothership vehicles: The mothership and drone routing problem with Graphs

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

[a] *Department of Statistics, Sapienza, University of Rome, Rome, 00185, Italy*
[b] *Department of Statistics and Operations Research, University of Seville, Seville, 41012, Spain*
[c] *Department of Statistics and Operations Research, University of Seville, Seville, 41012, Spain*

Abstract

This paper addresses the optimization of routing problems with drones. It analyzes the coordination of one mothership with one drone to obtain optimal routes that have to visit some target objects modeled as general graphs. The goal is to minimize the overall weighted distance traveled by both vehicles while satisfying the requirements in terms of percentages of visits to targets. We discuss different approaches depending on the assumption made on the route followed by the mothership: i) the mothership can move on a continuous framework (the Euclidean plane), ii) on a connected piecewise linear polygonal chain or iii) on a general graph. In all cases, we develop exact formulations resorting to mixed integer second order cone programs that are compared on a testbed of instances to assess their performance. The high complexity of the exact methods makes it difficult to find optimal solutions in short computing time. For that reason, besides the exact formulations we also provide a tailored matheuristic algorithm that allows one to obtain high quality solutions in reasonable time. Computational experiments show the usefulness of our methods in different scenarios.

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

1. Introduction

In recent years the progress in the field of automation has led to the increasingly widespread use of drone technology in many sectors (see [23] and [12] for a survey in civil applications). Depending on the application, these devices are used to support or replace humans in carrying out operations, and also in the cases of lack of infrastructures (see [24] also for future applications and research directions). We can find several examples of this trend in the telecommunication field, where drones can be used to provide connectivity in rural areas without antennas (see for example [3], [10], [9], [16] and [2]) or in areas affected by natural disasters which have compromised the existing infrastructures (see for example [8]). In goods delivery activities, especially in the last mile, drones represent a valid tool to support or replace the tasks of drivers by speeding up the service and relieving traffic from big cities or cities with particular configurations where standard vehicles cannot proceed (see for example [13] and [1]). This technology allows to provide a faster and safer response even in emergency contexts, for example for the delivery of medicines or blood bags, [30]. Other uses are also for achieving safer and faster activities of inspection and monitoring, both of networks (such as electricity, gas, telecommunications, railways, roads, etc.) and areas or their portions, depending on the application context. Indeed drones can reach quickly sections of a network that have suffered damage to verify the actual conditions (for example road networks after a storm, electrical or telecommunication networks that have suffered a breakdown, etc.), or allow, for example, to check the state and progress of a fire or an oil spill at sea. The use of this technology in all these different contexts is made advantageous by the fact that, compared to traditional means of transportation (trucks, ships, helicopters), Unmanned Aerial Vehicles (UAVs) have a lower cost per mile, produce less CO2 emissions and can arrive in places that cannot be reached with traditional means. On the other hand, their main limitation is the limited flight time which does not make them usable in full endurance in a number of contexts. For this reason, for some applications, hybrid systems that involve the combined use of drones with other means of transport may represent a more efficient alternative.

*Equally contributing authors
Email addresses: lavinia.amorosi@uniroma1.it (Lavinia Amorosi), puerto@us.es (Justo Puerto), cvalverde@us.es (Carlos Valverde)

In this system configurations it is necessary to coordinate and synchronize the operations of drones and other means of transport, taking into account the constraints of limited endurance of the drones and the movement of the other involved vehicles. The different configurations from which these hybrid systems can be characterized have given rise in the literature to different combinatorial optimization problems. Most of the works in the literature have focused on approaches that involve a discretization of the movement space of all the vehicles involved in the system. This provides the advantage of being able to mathematically model the problem more easily and obtain linear or linearizable formulations. However, on the other hand, it does not allow to fully exploit the freedom of movement of the drone which, unlike other means of transport which are constrained to road networks, can move from a point to any other in the continuous space. This paper studies the problem of coordinating a system composed of a mothership (the base vehicle) which supports the operations of one drone which have to visit a set of targets represented by graphs, with the goal of minimizing the total distance travelled by both vehicles. This system configuration can model, for example, monitoring and inspection activities on portions of networks where traditional vehicles cannot arrive, due, for example, to the presence of narrow streets, or because of a natural disaster or a terrorist attack that caused damages on the network. In all these cases, the inspection or monitoring of the drone consists in traversing edges of the network to perform a reconnaissance activity. For this reason we model the targets, to be visited by the drone, as graphs. Differently from previous works in the literature, we assume that the base vehicle and the drone can move freely on the continuous space and we present new Mixed Integer Non-Linear Programming (MINLP) formulations for this problem and a heuristic algorithm derived from the formulation to deal with larger instances. In particular, we consider three variants of the problem, depending on the assumptions made on the route followed by the mothership: i) the case in which it can move on a continuous framework (the Euclidean plane), ii) the case in which it is constrained to move on a polygonal, and ii) the case of a general undirected network. Also for these cases we propose alternative MINLP formulations that exploit the specific characteristics of these models.
The rest of the paper is structured as follows: Section 2 provides a detailed description of the problem under consideration. Section 3 reports the state of the art on routing problems with drones, mainly focusing on hybrid systems involving different transportation means. Section 4 presents alternative MINLP formulations proposed to model the different variants of the problem. At the end of this section, subsection 4.3 provides upper and lower bounds on the big-M constants introduced in the proposed formulations to tighten them. Section 5 presents the details of the matheuristic algorithm designed to handle large instances. In Section 6 we report the results obtained testing the formulations presented in Section 4 on different classes of planar graphs and the comparison with the ones provided by the matheuristic procedure in order to evaluate its effectiveness. Finally, Section 7 concludes the paper. For the sake of presentation, we have included an Appendix at the end of the paper including some details on some of the formulations and also on how to strengthen them.

2. Description of the Problem

In the Mothership and Drone Routing Problem with Graphs (MDRPG), there is one mothership (the base vehicle) and one drone that must operate in coordination to visit a set of targets, represented by graphs, located in a continuous framework that can be modeled as the Euclidean 2-or-3 dimension space. The goal is the minimization of the total distance travelled by both vehicles. The mothership and the drone begin at a common starting location denoted *orig* and they have to coordinate their movements so that the drone visits the set $\mathcal{G}$ of graphs (target objects) whose locations are given. These assumptions allow to model several real situations like inspections and monitoring of roads or wired networks, that can be performed by the drone more easily and safely than standard vehicles.
For each graph $g \in \{1, \ldots, |\mathcal{G}|\}$, we require that the drone performs a task consisting in the following four operations: (i) it is launched from the current mothership location (to be determined), (ii) it flies to the graph g that has to be visited, (iii) it traverses the required edges of graph g and then (iv) it returns to the current position of the mothership (to be determined), that most likely is different from the launching point. Indeed, what makes the problem more challenging is that we assume that the mothership can move while the drone visits the targets and thus coordination is a very important issue. Once all target graphs have been visited, the mothership and the drone return to a final location (depot), denoted by *dest*. In all its movements, the drone flies following straight lines.
Figure 1 shows an example of the problem framework, where the black square represents the origin and destination of the mothership tour. The three blue graphs represent the targets to be visited by the drone, located in the plane.

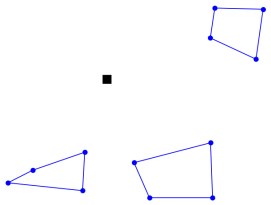


Figure 1: Example of problem framework

We assume wlog that the mothership and the drone do not need to arrive at each rendezvous location at the same time: the fastest arriving vehicle may wait for the other at the rendezvous point. In addition, we also assume that vehicles move at constant speeds, although this hypothesis could be relaxed. The mothership and the drone travel at speeds v_M and v_D, respectively. The mothership and the drone must travel together from *orig* to the first launching point. Similarly, after the drone visits the last target graph, the mothership and the drone must meet at the final rendezvous point before traveling together back to *dest*. The first launching and final rendezvous points are allowed to be *orig* and *dest*, respectively, but it is not mandatory. For the ease of presentation, in this paper we will assume that *orig* and *dest* are at the same location. However, all results extend easily to the case that *orig* and *dest* are at different locations.

Summarizing, the MDRPG consists in coordinating the drone with the mothership to minimize the overall distance travelled imposing that the drone has to visit a set of target graphs. In order to do that, it is required to determine: (i) the tour of the mothership starting at *orig*, deciding the different launching and rendezvous points, and returning to *dest*; (ii) the order of visits of the target graphs followed by the drone, determining the corresponding launching and rendezvous points of the drone on each visited graph; and (iii) the tour followed by the drone on each target graph $g \in \mathcal{G}$. The reader should observe that, since we assume constant velocities, the minimization of the travel distances is a natural proxy for the minimization of the overall time, that is the sum of traveling time of the mothership and traveling time of the drone, needed to complete the visits to all target graphs. Indeed, differently from a scheduling problem, we are not minimizing the makespan of the system. In fact, in this context, the length travelled by the drone, and implicitly the travelled time of the drone, has a cost that must be considered in addition to the one associated with the mothership.

Depending on the assumptions made on the movements of the mothership vehicle, this problem gives rise to three different versions: a) the mothership vehicle can move freely on the continuous space (all terrain ground vehicle, boat on the water or aircraft vehicle); b) the mothership can move on a connected piecewise linear polygonal chain; and c) the mothership can move on a road network (that is, it is a normal truck or van). In the former case, that we will call All terrain Mothership-Drone Routing Problem with Graphs (AMDRPG), each launching and rendezvous location may be chosen from a continuous space (the Euclidean 2-or-3 dimension space). In the latter case, that we will call Network Mothership-Drone Routing Problem with Graphs (NMDRPG), each launching and rendezvous locations must be chosen on a given graph embedded in the considered space. For the sake of presentation and due to the length of the paper, we will mainly focus, on the first model (AMDRPG) and second model (PMDRPG). The third model, namely NMDRPG, is addressed using similar techniques but providing slightly less details reported in the Appendix. Moreover, we consider two different ways for visiting the target graphs: (i) visiting a percentage of each edge of a graph, (ii) visiting a percentage of each graph. These variants derive, for example, from monitoring activities in which it is sufficient a partial visit of the targets. This is the case, for example, of traffic flows monitoring, where, in order to verify if the traffic progression is not disrupted, only inspecting a portion of edge provides a valuable information.

In addition, for the sake of simplicity, we restrict ourselves to the case where there are no obstacles to prevent the drone travelling in straight line. Extensions of this problem with obstacles are very interesting to be further considered although they require different techniques which are beyond the scope of this paper.

3. State of the art

In this section we focus mainly on the previous works in literature related to systems where UAVs are assisted by a vehicle in order to serve a set of targets. In these configurations the vehicle represents a launching and a recharging station for the UAVs and the main problem consists in coordinating the operations performed by one or multiple drones and the mothership. Most of the previous works in literature on this subject focus on node routing problems (NRPs), where the vehicle moves on a road network and the drone is used to visit target points outside the road network. Many of them are related to applications in the delivery sector where the set of targets to be visited is represented by a set of customers. For example, [18] studies a delivery system consisting in one drone and one truck. The UAV visits one customer for each trip and the truck can wait at the launching node for the drone to come back or move to a different rendezvous node. In [7] the authors study a continuous approximation on the Horse Fly Problem, where the truck is used as a mobile depot for the drone. In [6] the authors evaluate the economic impact of truck-and-drone hybrid models for deliveries by means of a continuous approximation model, considering different model parameters and customer densities. The paper [20] focuses on a delivery system where one truck supports the operations of multiple drones. The authors first clusterize the customers demand by adopting a K-means algorithm to find truck stops that represent hubs for drone deliveries. Then, they determine a TSP of the truck among centroids of these clusters, by means of a genetic algorithm, assuming that drones are not constrained by flight range. In [19] the authors consider a similar delivery system where at each stop site the truck waits until all drones come back before moving to the next site. The goal is the minimization of the latency in a customer-oriented distribution system. In [25] the authors formalize the k-Multi-visit Drone Routing Problem (k-MVDRP) considering a tandem between a truck and a fleet of k drones. The authors assumed that each drone can deliver one or more packages to customers in a single mission. Each drone may return to the truck to swap/recharge batteries, pick up a new set of packages, and depart again to customer locations. The article presents a mathematical formulation including a drone energy drain function that takes into account each package weight, but the problem is then solved by means of a heuristic algorithm. The paper [1] presents a multi-objective mixed integer linear programming model for the management of a hybrid delivery system consisting in one base vehicle and a fleet of drones. The problem consists in determining the tour of the base vehicle and the assignments of the customers to the UAVs simultaneously optimizing the distance travelled by the vehicle, the one travelled by the drones and the maximum completion time. The model is solved on two realistic urban scenarios providing a partial exploration of the Pareto frontier of the problem by means of the weighted sum method.
Other examples of similar configurations in which a base vehicle supports the operations of one or multiple drones can be found also in other sectors. [28] studies, for example, the city-scale video monitoring of a set of points of interest performed by a fleet of UAVs whose operations are supported by buses. However, in all the works mentioned so far, the combined operations of vehicles and drones examine routing for a set of locations and these configurations exploit only part of the advantages of the use of drones. Indeed, UAVs can move between any two points in the space not following the road network. Thus, they can be used also for other kind of services in which the targets are represented by edges or part of them. This leads to another class of models, that is arc routing problems (ARPs).
Differently from NRPs, there are relatively few papers on ARP with drones. In [22] and [21] the authors study a coordinated road network search problem with multiple UAVs and they formulate it as a Multi-choice Multi-dimensional Knapsack Problem minimizing the flight time. This is a modified Chinese Postman Problem taking into account the UAVs energy capacity constraints. The authors solve the problem by means of a greedy insertion heuristic that models drone travel distance between the road components as a Dubins path. [14] faces the area coverage problem in sparse environments with multiple UAVs. Also in this case the UAVs motion is modeled using Dubins paths and it is assumed that they are equipped with a coverage sensor of a given radius. The edge covering problem is solved by discretizing the network in orbits and then solving a TSP among this set of orbits.

In [11] the authors deal with the problem of dynamically allocate a finite set of UAVs to links in a network that need monitoring, over multiple time periods. The need of drone monitoring is based on data related to traffic conditions. Another work related to multiple-period real-time monitoring of road traffic, adopting UAVs, is [17]. The authors propose a mixed integer programming model combining a capacitated arc routing with an inventory routing problem and design a local branching based method to deal with large instances.
The ARPs with one UAV, have common characteristics with problems arising in path generation for laser cutting machines or drawing plotters. In particular, the authors of [5] study the Drone Rural Postman Problem (DRPP) showing the relation with the Intermittent Cutting Problem (ICP). They

present a solution algorithm based on the approximation of curves in the plane by polygonal chains that sequentially increases the number of points in the polygonal where the UAV can enter or leave. Thus, they solve the problem as a discrete optimization problem trying to better define the line by increasing the number of points.
As mentioned above, the number of references about ARPs with drones is limited as compared with the one on Drone NRPs. Moreover, as far as we know, the number of contributions in literature on arc routing problems involving a hybrid system consisting in one vehicle and one or multiple drones is simply reduced to [27] and [15].
In [27] the authors study the path planning problem of a system composed by a ground robot and one drone in precision agriculture and solved it by applying orienteering algorithms. On the other hand, [15] studies the paths planning problem for systems consisting in a carrier vehicle and a carried one to visit a set of target points and assuming that the carrier vehicle moves in the continuous space. Heuristic approaches have been proposed to deal with these problems.
To the best of our knowledge none of the previous papers deals with a drone arc routing problem combined with a node routing problem for a hybrid system consisting of one mothership vehicle and one drone.
In this paper we present new Mixed Integer Non-Linear Programming (MINLP) formulations for the problem of coordinating a system composed by one mothership (the base vehicle) which supports the operations of one drone which has to visit a set of targets represented by graphs, minimizing the total distance travelled by both vehicles. Indeed, differently from the previous works in literature, we assume that the drone can move freely on the continuous space, whereas for the mothership, we study two cases: 1) the base vehicle can move freely on the continuous space; and 2) it is constrained to move on a road network where launching and rendezvous points must be chosen. The main contributions of this article can be summarized as follows:

- A formal analysis of the coordination problem of one base vehicle and one drone in the continuous space that combine several characteristics that have not been previously analyzed simultaneously;
- First mathematical formalization of the problem by means of new second order cone MINLP formulations;
- Development of a new matheuristic algorithm to deal with large instances able to provide high quality solutions in limited computing time;
- Extensive experimental analysis comparing the exact solution of the formulations and the matheuristic on a set of generated instances involving different typologies of planar graphs.

4. Mixed Integer Non Linear Programming Formulations

In this section we present alternative MINLP formulations for the MDRPG depending on the nature of the mothership that will be compared computationally in later sections. We start analyzing first the situation where the mothership moves in a continuous space, namely the AMDRPG.

4.1. All terrain Mothership-Drone Routing Problem with Graphs

In this problem, we assume that the mothership is allowed to move freely in a continuous space: $\mathbb{R}^2$ or $\mathbb{R}^3$. In addition, distances are measured by the Euclidean norm, $\|\cdot\|_2$, although this assumption can be extended to any l_p norm, $1 \leq p \leq \infty$ (see [4]). Our goal is to develop an exact mathematical programming formulation that can be used to solve instances of this problem. In the following, we introduce the elements that formally describe the problem and that are summarized in Table 1.
Let $g = (V_g, E_g)$ be a graph in $\mathcal{G}$, where V_g denotes the set of nodes and E_g denotes the set of edges connecting pairs of nodes. Let e_g be edge e of the graph $g \in G$ and let $\mathcal{L}(e_g)$ be its length. Each edge e_g is parametrized 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 indicate its length $\mathcal{L}(e_g) = \|C^{e_g} - B^{e_g}\|$. Moreover, we denote with $\mathcal{L}(g) = \sum_{e_g \in E_g} \mathcal{L}(e_g)$ the total length of graph g.
For each edge e_g, with length $\mathcal{L}(e_g)$, we assign a binary variable μ^{e_g} that indicates whether or not the drone visits the segment e_g and define entry and exit points $R^{e_g} = (B^{e_g}, C^{e_g}, \rho^{e_g})$ and $L^{e_g} = (B^{e_g}, C^{e_g}, \lambda^{e_g})$, respectively, that determine the portion of the edge visited by the drone. Indeed, the coordinates of the points R^{e_g} and L^{e_g} are given, respectively by $R^{e_g} = (\rho^{e_g} B^{e_g}(x_1) + (1-\rho^{e_g}) C^{e_g}(x_1), \rho^{e_g} B^{e_g}(x_2) + (1-\rho^{e_g}) C^{e_g}(x_2))$ and $L^{e_g} = (\lambda^{e_g} B^{e_g}(x_1) + (1-\lambda^{e_g}) C^{e_g}(x_1), \lambda^{e_g} B^{e_g}(x_2) + (1-\lambda^{e_g}) C^{e_g}(x_2))$ where $\rho^{e_g} \in [0,1]$ and $\lambda^{e_g} \in [0,1]$ are variables to determine the position of the points on the segment.

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 $g \in \mathcal{G}$.
B^{e_g}, C^{e_g}: coordinates of the endpoints of edge e of graph $g \in \mathcal{G}$.
α^{e_g}: percentage of edge e of graph $g \in \mathcal{G}$ that must be visited.
α^{g}: percentage of graph $g \in \mathcal{G}$ that must be visited.
v_D: drone speed.
v_M: mothership speed.
M: big-M constant.

Table 1: Nomenclature for AMDRPG

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, and 0 otherwise.
$entry^{e_g} \in \{0,1\}\ \forall e_g \in E_g\ (g \in \mathcal{G})$: auxiliary binary variable used for linearizing expressions.
$u^{e_g t} \in \{0,1\}\ \forall e_g \in E_g\ (g \in \mathcal{G})\ \forall t \in T$: equal to 1 if the drone enters in graph g by e_g at stage t, 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.
$v^{e_g t} \in \{0,1\}\ \forall e_g \in E_g\ (g \in \mathcal{G})\ \forall t \in T$: equal to 1 if the drone exits from graph g by e_g at stage t, 0 otherwise.
s^{e_g}, $\forall e_g \in E_g\ (g \in \mathcal{G})$: integer non negative variable representing the order of visit of edge e of graph g.
Continuous Decision Variables
$\rho^{e_g} \in [0,1]$ 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 linearizing expressions.
$x^t_L\ \forall t \in T$: coordinates representing the point where the mothership launches the drone at stage t.
$x^t_R\ \forall t \in T$: coordinates representing the point where the mothership retrieves the drone at stage t.
$R^{e_g}\ \forall e_g \in E_g\ (g \in \mathcal{G})$: coordinates representing the entry point on edge e of graph g.
$L^{e_g}\ \forall e_g \in E_g\ (g \in \mathcal{G})$: coordinates representing the exit point on edge e of graph g.
$d_L^{e_g t} \geq 0,\ \forall e_g \in E_g\ (g \in \mathcal{G})\ \forall t \in T$: 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.
$d^{e_g e_g'} \geq 0,\ \forall e_g, e_g' \in E_g\ (g \in \mathcal{G})$: representing the distance travelled by the drone from the launching point L^{e_g} on e_g to the rendezvous 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 rendezvous point R^{e_g} to the launching point L^{e_g} on e_g.
$d_R^{e_g t} \geq 0\ \forall e_g \in E_g\ (g \in \mathcal{G})\ \forall t \in T$: representing the distance travelled by the drone from the last visiting point L^{e_g} on e_g to the rendezvous point x^t_R on the mothership at stage t.
$d^t_{LR} \geq 0\ \forall t \in T$: representing the distance travelled by the mothership from the launching point x^t_L to the rendezvous point x^t_R at stage t.
$d^t_{RL} \geq 0\ \forall t \in T$: representing the distance travelled by the mothership from the rendezvous point x^t_R at stage t to the launching point $x^{(t+1)}_L$ at the stage $t+1$.

Table 2: Decision Variables for AMDRPG-ST

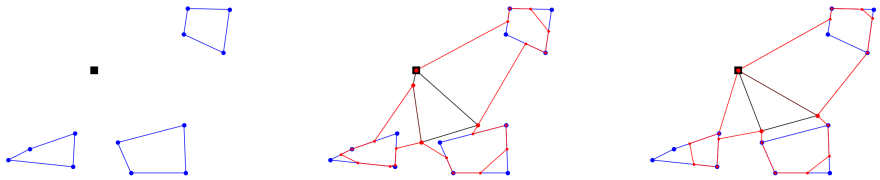
As mentioned in Section 2, we have considered two modes of visit to the target graphs $g \in \mathcal{G}$ that must be represented by their corresponding constraints:

- Visiting a percentage α^{e_g} of each edge e_g which can be modeled by:

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

- Visiting a percentage α^{g} of the total length $\mathcal{L}(g)$ of the graph g modeled by:

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



(a) (b) (c)

Figure 2: (a) Origin and target graphs, (b) Visit of $\alpha^{e_g}\%$ of each edge, (c) Visit of $\alpha_g\%$ of each graph.

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

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

The linearization of (α-G) is similar to (α-E) and only requires changing the last inequality in (α-E) for

$$\sum_{e_g \in E_g} \mu^{e_g}(\nu^{e_g}_{\max} + \nu^{e_g}_{\min})\mathcal{L}(e_g) \geq \alpha_g \mathcal{L}(g). \qquad (\alpha\text{-G})$$

Figure 2(a) shows an example of a configuration with three target graphs, each one with four nodes and four edges. The mothership begins at its starting point, denoted by the black square and it follows the black tour, like the ones in figure 2(b) and 2(c), where the red points represent launching and rendezvous points for the drone. Figure 2(b) and 2(c) show, through red segments, respectively the visit of a percentage α^{e_g} of each edge of the target graphs and the visit of a percentage α_g of each target graph. In this latter case we can observe that for each target graph one edge is not visited by the drone.

4.1.1. A first formulation for AMDRPG *based on stages*

Our first proposal to model this problem uses stages identified with the order in which the different elements in the problem are visited. We identify each visit to one of the target graphs with a stage of the process. Then, by using the notation below, we define the stages associated with graphs $T := \{1, \ldots, |\mathcal{G}|\}$ and those associated with the launching and rendezvous points including *orig* and *dest* $T' = \{0, \ldots, |\mathcal{G}| + 1\}$. For each stage $t \in T$, the drone follows a path starting from the mothership, visiting the required edges of g and returning to the mothership. Using the notation in Table 2 the sequence of points in the path are the following:

$$x^t_L \to R^{e_g} \to L^{e_g} \to \ldots \to R^{e'_g} \to L^{e'_g} \to \ldots \to R^{e''_g} \to x^t_R \to x^{t+1}_L.$$

This path calls in a natural way for the definition of binary variables that choose:

- The optimal order to visit each graph $g \in \mathcal{G}$. In other words, defining the sequence of the stages.
- The optimal order to visit the edges of each graph in its corresponding stage.

Thus, we can model the route that the drone follows by using the binary variables $u^{e_g t}$, $z^{e_g e'_g}$ and $v^{e_g t}$ defined in Table 2.

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

$$\sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} v^{e_g t} = 1, \qquad \forall t \in T, \tag{2}$$

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

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

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

$$\sum_{e'_g\in E_g} z_g^{e'_g e_g} + \sum_{t\in T} u^{e_g t} = \mu^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}, \tag{6}$$

$$\sum_{e'_g\in E_g} z_g^{e_g e'_g} + \sum_{t\in T} v^{e_g t} = \mu^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}. \tag{7}$$

Equations (1) and (2) state that in each stage the drone visits (enter and exit, respectively) only one graph. Constraints (3) and (4) assure that each graph is visited at some stage. Constraints (5) enforce that at stage t the drone enters and exits of exactly the same graph. Constraints (6) state that if edge e of graph g is visited by the drone, one of two alternative situations must occur: either e is the first edge of graph g visited by the drone at stage t, or edge e is visited by the drone after visiting another edge e' of graph g. Similarly, constraints (7) state that if edge e of graph g is visited by the drone, either e is the last edge of graph g visited by the drone at stage t, or the drone must move to another edge e' of graph g after visiting edge e.

Elimination of subtours

To prevent the existence of subtours within each graph $g \in \mathcal{G}$ that the drone must visit, one can include, among others, either a formulation that uses the Miller-Tucker-Zemlin constraints (MTZ) or another one that applies the subtour elimination constraints (SEC).

For the MTZ formulation, we use the continuous variables s^{e_g}, defined in Table 2, and establish the following constraints for each $g \in \mathcal{G}$:

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

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

Alternatively, we can also use the family of subtour elimination constraints:

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

Since there is an exponential number of SEC constraints, when we implement this formulation we need to perform a row generation procedure including constraints whenever they are required by a separation oracle. To find SEC inequalities, as usual, we search for disconnected components in the current solution. Among them, we choose the shortest subtour found in the solution to be added as a lazy constraint to the model.

The goal of the AMDRPG is to find a feasible solution that minimizes the total distance traveled by the drone and the mothership. To account for the different distances among the decision variables of the model we need to set the continuous variables $d_L^{e_g t}$, d^{e_g}, $d^{e_g e'_g}$, $d_R^{e_g t}$, d_{RL}^t and d_{LR}^t, defined in Table 2. This can be done by means of the following constraints:

$$\|x_L^t - R^{e_g}\| \le d_L^{e_g t}, \qquad \forall e_g : g \in \mathcal{G}, \forall t \in T, \qquad (\text{DIST}_1\text{-t})$$
$$\|R^{e_g} - L^{e_g}\| \le d^{e_g}, \qquad \forall e_g : g \in \mathcal{G}, \forall t \in T, \qquad (\text{DIST}_2\text{-t})$$
$$\|R^{e_g} - L^{e'_g}\| \le d^{e_g e'_g}, \qquad \forall e_g \ne e'_g \in E_g : g \in \mathcal{G}, \qquad (\text{DIST}_3\text{-t})$$
$$\|L^{e_g} - x_R^t\| \le d_R^{e_g t}, \qquad \forall e_g : g \in \mathcal{G}, \forall t \in T, \qquad (\text{DIST}_4\text{-t})$$
$$\|x_R^t - x_L^{t+1}\| \le d_{RL}^t, \qquad \forall t \in T, \qquad (\text{DIST}_5\text{-t})$$
$$\|x_L^t - x_R^t\| \le d_{LR}^t, \qquad \forall t \in T. \qquad (\text{DIST}_6\text{-t})$$

To ensure that the time spent by the drone to visit graph g at stage t is less than or equal to the time that the mothership needs to move from the launching point to the rendezvous point at stage t, we need to define the following coordination constraint for each $g \in \mathcal{G}$ and $t \in T$:

$$\frac{1}{v_D}\left(\sum_{e_g \in E_g} u^{e_g t} d_L^{e_g t} + \sum_{e_g, e'_g \in E_g} z^{e_g e'_g} d^{e_g e'_g} + \sum_{e_g \in E_g} \mu^{e_g} d^{e_g} + \sum_{e_g \in E_g} v^{e_g t} d_R^{e_g t}\right) \le \frac{d_{RL}^t}{v_M} + M(1 - \sum_{e_g \in E_g} u^{e_g t}). \qquad (\text{DCW-t})$$

Eventually, we have to impose that the tour of the mothership, together with the drone, starts from the origin *orig* and ends at the destination *dest*. To this end, we define the following constraints:

$$x_L^0 = orig, \qquad (\text{ORIG}_1)$$
$$x_R^0 = orig, \qquad (\text{ORIG}_2)$$
$$x_L^{|\mathcal{G}|+1} = dest, \qquad (\text{DEST}_1)$$
$$x_R^{|\mathcal{G}|+1} = dest. \qquad (\text{DEST}_2)$$

Therefore, putting together all the constraints introduced before, the following formulation minimizes the overall distance traveled by the mothership and drone, coordinating their movements and ensuring the required coverage of the targets.

$$\min \quad \sum_{g \in \mathcal{G}} \sum_{e_g \in E_g} \sum_{t \in T} (u^{e_g t} d_L^{e_g t} + v^{e_g t} d_R^{e_g t}) + \sum_{g \in \mathcal{G}} \sum_{e_g \in E_g} \mu^{e_g} d^{e_g} + \sum_{g \in \mathcal{G}} \sum_{e_g, e'_g \in E_g} z^{e_g e'_g} d^{e_g e'_g} + \sum_{t \in T} (d_{RL}^t + d_{LR}^t) \qquad (\text{AMDRPG-ST})$$

s.t. (1) − (7),
(MTZ$_1$) − (MTZ$_2$) or (SEC),
(α-E) or (α-G),
(DCW-t),
(DIST$_1$-t) − (DIST$_6$-t),
(ORIG$_1$) − (DEST$_2$).

The objective function has four terms: the first one computes the distances to go to and to return from the mothership to the target graph g visited at stage t, the second one accounts for the distance traveled by the drone on the edges of the target graph g between two consecutive *jumps* between edges, the third one computes the distances traveled by the drone while it *jumps* between two consecutive edges on the target graph g and the fourth term expresses the distances made by the mothership. Constraints (1)-(7) model the route followed by the drone, (MTZ$_1$) - (MTZ$_2$) or (SEC) ensure that the displacement of the drone in each visit to a target graph is a route, (α-E) or (α-G) define what is required in each visit to a target graph. Constraints (DIST$_1$-t)-(DIST$_6$-t) set the variables $d_L^{e_g t}$, d^{e_g}, $d^{e_g e'_g}$, $d_R^{e_g t}$, d_{RL}^t and d_{LR}^t, defined in Table 2, which represent Euclidean distances needed in the model.
Note that, to deal with the bilinear terms of the objective function, we use McCormick's envelopes to linearize these terms by adding variables $p \ge 0$ that represent these products, and by introducing the following constraints:

$$p \ge mz, \qquad (8)$$
$$p \le d - M(1 - z), \qquad (9)$$

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 4.3.

4.1.2. Alternative formulations for AMDRPG *based on enforcing connectivity: MTZ-constraints and sub-tour elimination-constraints (SEC)*

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, and 0 otherwise.
$entry^{e_g} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: auxiliary binary variables for linearization.
$u^{e_g} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: equal to 1 if the drone enters in graph g by 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.
$v^{e_g} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: equal to 1 if the drone exits from graph g by e_g, 0 otherwise.
$w^{gg'} \in \{0,1\}$ $\forall g, g' \in \mathcal{G}$: equal to 1 if the mothership moves from x_R^g to $x_L^{g'}$, 0 otherwise.
s^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: integer non negative variables representing the order of visit of edge e of graph g.
Continuous Decision Variables
$\rho^{e_g} \in [0,1]$ and $\lambda^{e_g} \in [0,1]$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: defining the entry and exit points on e_g.
$\nu_{\min}^{e_g}$ and $\nu_{\max}^{e_g} \in [0,1]$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: auxiliary variables for linearization.
x_L^g $\forall g \in \mathcal{G}$: pairs of coordinates representing the point where the mothership launches the drone to visit graph g.
x_R^g $\forall g \in \mathcal{G}$: pairs of coordinates representing the point where the mothership retrieves the drone after visit graph g.
R^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: coordinates representing the entry point on edge e of graph g.
L^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: coordinates representing the exit point on edge e of graph g.
$d_L^{e_g} \geq 0$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: representing the distance travelled by the drone from the launching point on the mothership x_L^g to the first visiting point R^{e_g} on edge 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 launching point L^{e_g} on e_g to the rendezvous 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 rendezvous point R^{e_g} to the launching point L^{e_g} on e_g.
$d_R^{e_g} \geq 0$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: representing the distance travelled by the drone from the last visiting point L^{e_g} on e_g to the rendezvous point x_R^g on the mothership.
$d_{LR}^g \geq 0$ $\forall g \in \mathcal{G}$: representing the distance travelled by the mothership from the launching point x_L^g to the rendezvous point x_R^g while the drone is visiting g.
$d_{RL}^{gg\prime} \geq 0$ $\forall g, g' \in \mathcal{G}$: representing the distance travelled by the mothership from the rendezvous point x_R^g for graph g to the launching point $x_L^{g'}$ for graph $g\prime$.

Table 3: Decision Variables for AMDRPG-MTZ

In this family of formulations we replace the variables $u^{\cdot t}$, $v^{\cdot t}$ and constraints that model the tour using stages, namely (1)-(7), by constraints that ensure connectivity. We will distinguish two different approaches. The first one is based on the rationale of Miller-Tucker-Zemlin (MTZ) whereas the second one uses an extended formulation based on SEC.
We start presenting the formulation based on MTZ. It requires the definition of two families of binary variables that choose:

- The optimal order to visit the graphs $g \in \mathcal{G}$ in the tour of the mothership.
- The optimal order to visit the edges of each graph.

The above concepts can be modeled with the binary variables u^{e_g}, $z^{e_g e_g'}$, v^{e_g} and $w^{gg'}$, summarized in Table 3.
By using these binary variables, we can model the route that the drone follows in each particular graph

$g \in \mathcal{G}$:

$$\sum_{e_g \in E_g} u^{e_g} = 1, \qquad \forall g \in \mathcal{G}, \tag{10}$$

$$\sum_{e_g \in E_g} v^{e_g} = 1, \qquad \forall g \in \mathcal{G}, \tag{11}$$

$$\sum_{e'_g \in E_g} z^{e'_g e_g} + u^{e_g} = \mu^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}, \tag{12}$$

$$\sum_{e'_g \in E_g} z^{e_g e'_g} + v^{e_g} = \mu^{e_g}, \qquad \forall e_g \in E_g : g \in \mathcal{G}. \tag{13}$$

On the other hand, to model the tour followed by the mothership, we have to include the following new constraints:

$$\sum_{g \in \mathcal{G}} w^{g0} = 0, \tag{14}$$

$$\sum_{g' \in \mathcal{G}} w^{(n_G+1)g'} = 0, \tag{15}$$

$$\sum_{g' \in \mathcal{G} \setminus \{g\}} w^{gg'} = 1, \qquad \forall g \in \mathcal{G}, \tag{16}$$

$$\sum_{g \in \mathcal{G} \setminus \{g'\}} w^{gg'} = 1, \qquad \forall g' \in \mathcal{G}, \tag{17}$$

$$s_g - s_{g'} + |\mathcal{G}| w^{gg'} \leq |\mathcal{G}| - 1, \qquad \forall g \neq g', \tag{MTZ$_3$}$$

$$0 \leq s_g \leq |\mathcal{G}| - 1, \qquad \forall g \in \mathcal{G}, \tag{MTZ$_4$}$$

$$s_0 = 0, \tag{MTZ$_5$}$$

$$s_{n_G+1} = n_G + 1. \tag{MTZ$_6$}$$

The reader may note that constraints (16), (17) and (MTZ$_3$)-(MTZ$_6$) are of the type MTZ for the complete graph $\hat{G} = (\hat{N}, \hat{E})$ induced by the graphs $g \in \mathcal{G}$ to be visited, where a node $\hat{v}_g \in \hat{N}$ if $g \in \mathcal{G}$. Thus, we enforce that the mothership route follows a tour on the graphs to be visited by the drone. Next, we have to connect the visits of the drone to the graphs with the tour followed by the mothership. To this end, we require to introduce variables defining the launching and rendezvous points to account for the distances traveled and to define the inner tour of the mothership.

In a similar way, we need to set the continuous variables $d_L^{e_g}$, $d^{e_g e'_g}$, d^{e_g}, $d_R^{e_g}$, d_{LR}^{g} and $d_{RL}^{gg'}$, defined in Table 3, that account for the distances among distinguished points. Thus, we introduce the following constraints:

$$\|x_L^g - R^{e_g}\| \leq d_L^{e_g}, \qquad \forall e_g : g \in \mathcal{G}, \tag{DIST$_1$-g}$$

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

$$\|R^{e_g} - L^{e'_g}\| \leq d^{e_g e'_g}, \qquad \forall e_g \neq e'_g : g \in \mathcal{G}, \tag{DIST$_3$-g}$$

$$\|L^{e_g} - x_R^g\| \leq d_R^{e_g}, \qquad \forall e_g : g \in \mathcal{G}, \tag{DIST$_4$-g}$$

$$\|x_R^g - x_L^{g'}\| \leq d_{RL}^{gg'}, \qquad \forall g, g' \in \mathcal{G}, \tag{DIST$_5$-g}$$

$$\|x_L^g - x_R^g\| \leq d_{LR}^{g}, \qquad \forall g \in \mathcal{G}. \tag{DIST$_6$-g}$$

In this formulation of AMDRPG the goal is again to find a feasible solution that minimizes the total distance traveled by the drone and the mothership but without using stages. Instead the model ensures the Hamiltonian nature of the routes using the rationale of connectivity by the MTZ or SEC family of constraints.

Again, we need to be sure that the time spent by the drone to visit the graph g is less than or equal to the time that the mothership needs to move from the launching point to the rendezvous point associated to this graph g. Hence, by using the same argument, as the one used in (DCW-t), we define for each

$g \in \mathcal{G}$:

$$\frac{1}{v_D}\left(\sum_{e_g \in E_g} u^{e_g} d_L^{e_g} + \sum_{e_g, e'_g \in E_g} z^{e_g e'_g} d^{e_g e'_g} + \sum_{e_g \in E_g} \mu^{e_g} d^{e_g} + \sum_{e_g \in E_g} v^{e_g} d_R^{e_g}\right) \leq \frac{d_{RL}^g}{v_M}, \quad \forall g \in \mathcal{G}.. \quad \text{(DCW-g)}$$

Therefore, modifying the previous formulation (AMDRPG-ST), replacing constraints based on stages, one obtains the following alternative valid formulation:

$$\min \quad \sum_{g \in \mathcal{G}} \sum_{e_g \in E_g} (u^{e_g} d_L^{e_g} + v^{e_g} d_R^{e_g}) + \sum_{g \in \mathcal{G}} \sum_{e_g \in E_g} \mu^{e_g} d^{e_g} + \sum_{g \in \mathcal{G}} \sum_{e_g, e'_g \in E_g} z^{e_g e'_g} d^{e_g e'_g} + \sum_{g \in \mathcal{G}} d_{LR}^g + \sum_{g, g' \in \mathcal{G}} d_{RL}^{gg'} w^{gg'} \quad \text{(AMDRPG-MTZ)}$$

s.t. (10) – (17),
(MTZ_1) – (MTZ_2) or (SEC),
(MTZ_3) – (MTZ_6),
(α-E) or (α-G),
(DCW-g),
(DIST_1-g) – (DIST_6-g),
(ORIG_1) – (DEST_2).

The formulation above (AMDRPG-MTZ) can be slightly modified replacing constraints (MTZ_3) – (MTZ_6) by

$$\sum_{g, g' \in \mathcal{G}} w^{gg'} \leq |S| - 1, \quad \forall S \subseteq \{1, \ldots, |\mathcal{G}|\}. \tag{18}$$

In doing so we obtain another formulation for AMDRPG using SEC rather than MTZ constraints to represent the mothership tour. We will call this formulation (AMDRPG-SEC). Later, we will compare their performance in Section 6.

4.1.3. Comparing the formulations

A straightforward analysis of the two types of formulations, namely by stages and based on connectivity, shows that (AMDRPG-MTZ) has much less variables and constraints than (AMDRPG-ST). Indeed, the latter requires definition of distance variables for each e_g and t which is a quadratic function rather than a linear one as used by the former. In addition, to enforce coordination (AMDRPG-ST) needs constraints (DCW) for each $g \in \mathcal{G},\ t \in T$ whereas (AMDRPG-MTZ) only uses (DCW) constraints for $g \in \mathcal{G}$. This is an important reduction in the dimension of the problems to be solved. This observation is confirmed by its better performance as discussed in Section 6.

4.2. Network Mothership-Drone Routing Problem on a Polygonal with Graphs (PMDRPG)

The most significant difference between AMDRPG and NMDRPG is that, in the latter, the mothership cannot move freely in the framework space and instead it has to move on a network that models a road system where the base vehicle (for example a truck) is constrained to move. An intermediate situation between moving freely and moving on a general road network is to assume that the mothership moves on a closed polygonal, namely a piecewise linear, chain. Modeling this situation is a transition step to achieve the more general case of the mothership moving on general networks. We will present the formulations for the PMDRPG following a scheme similar to that of AMDRPG. First, we present a model by stages and then another one based on connectivity constraints.

Let $\mathcal{P}$ denote a polygonal chain parametrized by its breakpoints $V^1, \ldots, V^p$ ordered in the direction of travel. Observe that we are assuming that there exists a pre-specified orientation that determines the direction of the mothership's displacement.

As in the previous section, we present a summary of the notation, parameters and variables of this model in Tables 4 and 5.

We begin giving a formal description of the distance traveled by the mothership on the polygonal $\mathcal{P}$, between two consecutive events induced by drone operations, either launching-retrieving or retrieving-launching. To this end, we use the variables γ_L^{et} and γ_R^{et}, defined in Table 5. Recall that γ_L^{et} and γ_R^{et} are the values of the parametrizations that define the points x_L^t and x_R^t on the edge e of $\mathcal{P}$, respectively. Figure 3 illustrates the meaning of the parameters γ_L^{et} and γ_R^{et} on a polygonal chain with 4 pieces.

Problem Parameters
$V^1, ..., V^p$: pairs of coordinates defining the breakpoints of the polygonal chain $\mathcal{P}$ representing the road system where the mothership can move.
$orig$: pair of coordinates defining the origin of the mothership path (or tour).
$dest$: pair of coordinates 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 $g \in \mathcal{G}$.
B^{e_g}, C^{e_g}: endpoints of edge e of graph $g \in \mathcal{G}$.
α^{e_g}: percentage of edge e of graph $g \in \mathcal{G}$ that must be visited.
α^g: percentage of graph $g \in \mathcal{G}$ that must be visited.
v_D: drone speed.
v_M: mothership speed.
M: big M.

Table 4: Nomenclature for PMDRPG

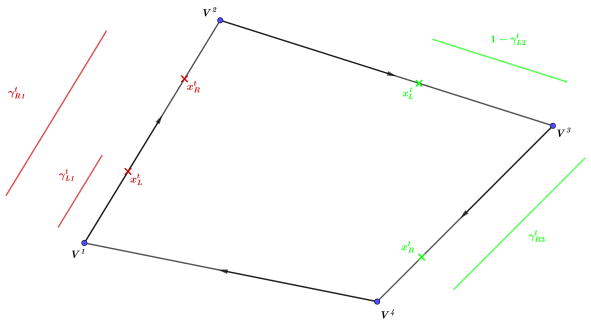


Figure 3: An example of a parametrization of a mothership route that goes from V^1 to x_L^t to launching the drone, then moves to x_R^t on the edge $\overline{V^1V^2}$ to retrieve it, then it goes to x_L^t on $\overline{V^2V^3}$ to launching it again, and travels until x_R^t on $\overline{V^3V^4}$ to retrieve the drone and returns to the starting point at V^1.

Then, the distance measured on the polygonal can be obtained by the following expressions for each $t \in T$ and for each $e, e' \in \mathcal{P}$:

$$d_{LR}^{ee't} = \begin{cases} |\gamma_R^{et} - \gamma_L^{et}|\mathcal{L}(e), & \text{if } e = e', \\ (1-\gamma_L^{et})\mathcal{L}(e) + \sum_{e''=e+1}^{e'-1}\mathcal{L}(e'') + \gamma_R^{e't}\mathcal{L}(e'), & \text{if } e < e', \\ \gamma_L^{et}\mathcal{L}(e) + \sum_{e''=e'+1}^{e-1}\mathcal{L}(e'') + (1-\gamma_R^{e't})\mathcal{L}(e'), & \text{if } e > e'. \end{cases} \tag{$d_{LR}^{t\mathcal{P}}$}$$

$$d_{RL}^{ee't} = \begin{cases} |\gamma_L^{et+1} - \gamma_R^{et}|\mathcal{L}(e), & \text{if } e = e', \\ (1-\gamma_R^{et})\mathcal{L}(e) + \sum_{e''=e+1}^{e'-1}\mathcal{L}(e'') + \gamma_L^{e't+1}\mathcal{L}(e'), & \text{if } e < e', \\ \gamma_R^{et}\mathcal{L}(e) + \sum_{e''=e'+1}^{e-1}\mathcal{L}(e'') + (1-\gamma_L^{e't+1})\mathcal{L}(e'), & \text{if } e > e'. \end{cases} \tag{$d_{RL}^{t\mathcal{P}}$}$$

The reader should observe that, as described in Table 5, $d_{LR}^{ee't}$ accounts for the distance traveled by the mothership on $\mathcal{P}$ between consecutive launching and rendezvous points, whereas $d_{RL}^{ee't}$ does it for consecutive retrieving and next launching points.

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.
$entry^{e_g} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: auxiliary binary variables for linearization.
$u^{e_g t} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$, $\forall t \in T$: equal to 1 if the drone enters in graph g by e_g at stage t, 0 otherwise.
$z^{e_g e_g'} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: equal to 1 if the drone goes from e_g to e_g' of graph g, 0 otherwise.
$v^{e_g t} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$, $\forall t \in T$: equal to 1 if the drone exits from graph g by e_g at stage t, 0 otherwise.

s^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: integer non negative variables representing the order of visit of edge e of graph g.
$z_{LR}^{ee't} \in \{0,1\}$ $\forall e, e' \in \mathcal{P}$ $\forall t \in T$: equal to 1 if the launching point x_L^t is located on e and the rendezvous point x_R^t is located on e' at stage t.
$z_{RL}^{ee't} \in \{0,1\}$ $\forall e, e' \in \mathcal{P}$ $\forall t \in T$: equal to 1 if the rendezvous point x_R^t is located on e at stage t and the launching point x_L^{t+1} is located on e' at stage $t+1$.
μ_L^{et} $\forall e \in \mathcal{P}$ $\forall t \in T$: equal to 1 if the launching point x_L^t is located on e at stage t.
μ_R^{et} $\forall e \in \mathcal{P}$ $\forall t \in T$: equal to 1 if the rendezvous point x_R^t is located on e at stage t.
Continuous Decision Variables
$\rho^{e_g} \in [0,1]$ 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.
$\gamma_R^{et} \in [0,1]$ and $\gamma_L^{et} \in [0,1]$ $\forall e \in \mathcal{P}$ $\forall t \in T$: defining the launching and rendezvous points on edge e of the polygonal $\mathcal{P}$ at stage t.
$\nu_{\min}^{e_g}$ and $\nu_{\max}^{e_g} \in [0,1]$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: auxiliary variables for linearization.
x_L^t $\forall t \in T$: coordinates representing the point where the mothership launches the drone at stage t.
x_R^t $\forall t \in T$: coordinates representing the point where the mothership retrieves the drone at stage t.
R^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: coordinates representing the entry point on edge e of graph g.
L^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: coordinates representing the exit point on edge e of graph g.
$d_L^{e_g t} \geq 0$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$, $\forall t \in \mathcal{T}$: representing the distance travelled by the drone from the launching point x_L^t on the mothership at stage t to the first visiting point R^{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 launching point L^{e_g} on e_g to the rendezvous 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 rendezvous point R^{e_g} to the launching point L^{e_g} on e_g.
$d_R^{e_g t} \geq 0$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$ $\forall t \in T$: representing the distance travelled by the drone from the last visiting point L^{e_g} on e_g to the rendezvous point x_R^t on the mothership at stage t.
$d_{LR}^{ee't} \geq 0$ $\forall e \in \mathcal{P}$ $\forall t \in T$: representing the distance travelled by the mothership from the launching point x_L^t on e to the rendezvous point x_R^t on e' at stage t.
$d_{RL}^{ee't} \geq 0$ $\forall e \in \mathcal{P}$ $\forall t \in T$: representing the distance travelled by the mothership from the rendezvous point x_R^t on e at stage t to the launching point $x_L^{(t+1)}$ on e' at the stage $t+1$.
$d_{LR}^t \geq 0$ $\forall t \in T$: representing the total distance travelled by the mothership between the launching point x_L^t and the rendezvous point x_R^t at stage t.
$d_{RL}^t \geq 0$ $\forall t \in T$: representing the total distance travelled by the mothership between the rendezvous point x_R^t at stage t and the launching point x_L^{t+1} at stage $t+1$.
$\lambda_L^t \geq 0$ $\forall t \in T$: representing the absolute position of point x_L^t in the polygonal $\mathcal{P}$.
$\lambda_R^t \geq 0$ $\forall t \in T$: representing the absolute position of point x_R^t in the polygonal $\mathcal{P}$.

Table 5: Decision Variables for PMDRPG-ST

In order to actually include these expressions in a formulation one has to decide which one of the two definitions ($e < e'$ or $e > e'$) is to be used in the objective function. This is the goal of the binary variables $z_{LR}^{ee't}$ that determines whether $d_{LR}^{ee'}$ is defined by the first or the second expression in the formula:

$$z_{LR}^{ee't} = 1 \text{ if } \mu_L^{et} = 1 \text{ and } \mu_R^{e't} = 1,$$

where μ_L^{et} and μ_R^{et} are indicator variables that attain the value one if the launching point (resp. rendezvous point) is placed in the segment e of $\mathcal{P}$ at the stage t.
Similarly, we define $z_{RL}^{ee't}$:

$$z_{RL}^{ee't} = 1 \text{ if } \mu_R^{et} = 1 \text{ and } \mu_L^{e't+1} = 1.$$

With all this, we can model the movement of the mothership on the polygonal. For each stage $t \in T$, we have

$$x_L^t = \sum_{e=(i,i+1)\in\mathcal{P}} \mu_L^{et}\left[V^i + \gamma_L^{et}(V^{i+1} - V^i)\right], \tag{19}$$

$$x_R^t = \sum_{e=(i,i+1)\in\mathcal{P}} \mu_R^{et}\left[V^i + \gamma_R^{et}(V^{i+1} - V^i)\right], \tag{20}$$

$$z_{LR}^{ee't} = \mu_L^{et}\mu_R^{e't}, \tag{21}$$

$$z_{RL}^{ee't} = \mu_R^{et}\mu_L^{e't+1}, \tag{22}$$

$$\sum_e \mu_L^{et} = 1, \tag{23}$$

$$\sum_e \mu_R^{et} = 1, \tag{24}$$

$$d_{LR}^t = \sum_{e,e'} z_{LR}^{ee't} d_{LR}^{ee'}, \tag{25}$$

$$d_{RL}^t = \sum_{e,e'} z_{RL}^{ee't} d_{RL}^{ee't}, \tag{26}$$

where $d_{LR}^{ee'}$ and $d_{RL}^{ee't}$ are defined in $(d_{LR}^{t\mathcal{P}})$ and $(d_{RL}^{t\mathcal{P}})$. Constraints (19) and (20) parameterize the launching and rendezvous points on the polygonal chain. Constraints (21) and (22) set the variable z by means of the binary variables μ_L^{et} and μ_R^{et}. Constraints (23) and (24) state that, in each stage, the launching and rendezvous points can be only in one segment, not necessarily the same. Constraints (25) and (26) compute the total distance between the launching and the rendezvous points in each stage.
In addition, for each stage $t \in T$ we can define continuous variables λ_R^t and λ_L^t that model the absolute position of the points x_L^t and x_R^t in the polygonal to ensure that the movement of the mothership goes always in the allowed direction of travel and never comes back when it traverses the polygonal chain.

$$\lambda_L^t - i \geq \mu_L^{et} - (p-1)(1-\mu_L^{et}), \qquad \forall e = (i, i+1) \in \mathcal{P}, \tag{27}$$

$$\lambda_L^t - i \leq \mu_L^{et} - (p-1)(1-\mu_L^{et}), \qquad \forall e = (i, i+1) \in \mathcal{P}, \tag{28}$$

$$\lambda_R^t - i \geq \mu_R^{et} - (p-1)(1-\mu_R^{et}), \qquad \forall e = (i, i+1) \in \mathcal{P}, \tag{29}$$

$$\lambda_R^t - i \leq \mu_R^{et} + (p-1)(1-\mu_R^{et}), \qquad \forall e = (i, i+1) \in \mathcal{P}, \tag{30}$$

$$\lambda_R^t \geq \lambda_L^t, \qquad \forall t \in T, \tag{31}$$

$$\lambda_L^{t+1} \geq \lambda_R^t, \qquad \forall t \in T. \tag{32}$$

Inequalities (27)-(30) link μ_L^{et} and λ_L^t (resp. μ_R^{et} and λ_R^t). Finally, constraints (31) and (32) ensure that the mothership moves in the right direction on the polygonal.
The goal of the (PMDRPG) is, once again, to find a feasible solution that minimizes the total distance covered by the mothership and the drone. Thus, gathering all the above constraints one gets the following valid formulation:

$$\min \quad \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g}\sum_{t\in T}(u^{e_g t} d_L^{e_g t} + v^{e_g t} d_R^{e_g t}) + \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g} \mu^{e_g} d^{e_g} + \sum_{g\in\mathcal{G}}\sum_{e_g, e_g'\in E_g} z^{e_g e_g'} d^{e_g e_g'} + \sum_{t\in T}(d_{RL}^t + d_{LR}^t) \tag{PMDRPG}$$

s.t. (1) – (7),
(19) – (32),
(MTZ_1) – (MTZ_2) or (SEC),
(α-E) or (α-G),
(DCW-t),
$(d_{LR}^{t\mathcal{P}}), (d_{RL}^{t\mathcal{P}})$,
(DIST_1-t) – (DIST_6-t),
(ORIG_1) – (DEST_2).

4.2.1. Alternative formulation for PMDRPG *based on MTZ-constraints*

Analogously to the case of AMDRPG one can also model PMDRPG without any explicit reference to stages. Here, we will follow a very similar approach to that in Section 4.1.2 to describe the formulation for PMDRPG based on MTZ-constraints.

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, and 0 otherwise.
$entry^{e_g} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: auxiliary binary variables for linearization.
$u^{e_g} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: equal to 1 if the drone enters in graph g by 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 $e_g^{'}$ of graph g, 0 otherwise.
$v^{e_g} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: equal to 1 if the drone exits from graph g by e_g, 0 otherwise.
$w^{gg'} \in \{0,1\}$: equal to 1 if the mothership moves from x_R^g to $x_L^{g'}$, 0 otherwise.
s^{e_g} $\forall e_g \in g$ $(g \in \mathcal{G})$: integer non negative variables representing the order of visit of edge e of graph g.
$z_{LR}^{ee'g} \in \{0,1\}$ $\forall e, e' \in \mathcal{P}$, $\forall g \in \mathcal{G}$: equal to 1 if the launching point x_L^g, associated with graph g, is located on e and the rendezvous point x_R^g is located on e'.
$z_{RL}^{ee'gg'} \in \{0,1\}$ $e, e' \in \mathcal{P}$, $g, g' \in \mathcal{G}$: equal to 1 if the rendezvous point x_R^g, associated with graph g, is located on e and the launching point $x_L^{g'}$, associated with graph g', is located on e'.
μ_L^{eg} $\forall e \in \mathcal{P}$, $\forall g \in \mathcal{G}$: equal to 1 if the launching point x_L^g, associated with graph g, is located on e.
μ_R^{eg} $\forall e \in \mathcal{P}$, $\forall g \in \mathcal{G}$: equal to 1 if the rendezvous point x_R^g, associated with graph g, is located on e.
Continuous Decision Variables
$\rho^{e_g} \in [0,1]$ and $\lambda^{e_g} \in [0,1]$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: defining the entry and exit points on edge e_g.
$\gamma_R^{eg} \in [0,1]$ and $\gamma_L^{eg} \in [0,1]$ $\forall e \in \mathcal{P}$, $\forall g \in \mathcal{G}$: defining the launching and rendezvous points, associated with graph g, on edge e.
$\nu_{\min}^{eg}$ and $\nu_{\max}^{eg} \in [0,1]$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: auxiliary variables for linearization.
x_L^g $\forall g \in \mathcal{G}$: coordinates representing the point where the mothership launches the drone to visit graph g.
x_R^g $\forall g \in \mathcal{G}$: coordinates representing the point where the mothership retrieves the drone to visit graph g.
R^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: coordinates representing the entry point on edge e of graph g.
L^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: coordinates representing the exit point on edge e of graph g.
$d_L^{e_g} \geq 0$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: distance travelled by drone from launching point x_L^g to the first visiting point R^{e_g} on e_g.
$d^{e_g e_g^{'}} \geq 0$ $\forall e_g, e_g' \in E_g$ $(g \in \mathcal{G})$: distance travelled by drone from launching point L^{e_g} on e_g to the rendezvous 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 rendezvous point R^{e_g} to the launching point L^{e_g} on e_g.
$d_R^{e_g} \geq 0$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: representing the distance travelled by the drone from the last visiting point L^{e_g} on graph g to the rendezvous point x_R^g on the mothership.
$d_{LR}^{ee'g} \geq 0$ $\forall e, e' \in \mathcal{P}$, $\forall g \in \mathcal{G}$: representing the distance travelled by the mothership from the launching point x_L^g on e to the rendezvous point x_R^g on e' to visit graph g.
$d_{RL}^{ee'gg'} \geq 0$ $\forall e, e' \in \mathcal{P}$, $\forall g, g' \in \mathcal{G}$: representing the distance travelled by the mothership from the rendezvous point x_R^g on e, for graph g, to the launching point $x_L^{g'}$ on e', for graph g'.
$d_{LR}^g \geq 0$ $\forall g \in \mathcal{G}$: representing the total distance travelled by the mothership between the launching point x_L^g and the rendezvous point x_R^g to visit graph g.
$d_{RL}^{gg'} \geq 0$ $\forall g, g' \in \mathcal{G}$: representing the total distance travelled by the mothership between the rendezvous point x_R^g, for graph g, and the launching point $x_L^{g'}$, for graph g'.

Table 6: Decision Variables for PMDRPG-MTZ

Let $\mathcal{P}$ denote, as before, the polygonal chain parameterized by its endpoints $V^1, \ldots, V^p$ ordered in the direction of travel. In addition, let also γ_L^{eg} and γ_R^{eg} be the parameter values that define the points x_L^g and x_R^g on the line segment $\overline{V^i V^{i+1}}$ in $\mathcal{P}$, respectively. Then, the distance traveled on the polygonal, by the mothership, can be obtained by one of the following expressions. The first one refers to distances covered by the mothership between the launching and rendezvous points of a drone operation, whereas the second one models the distance traveled by the mothership between the rendezvous and launching points of two consecutive target graphs in its route.

$$d_{LR}^{ee'g} = \begin{cases} |\gamma_L^{eg} - \gamma_R^{eg}|\mathcal{L}(e), & \text{if } e = e', \\ (1-\gamma_L^{eg})\mathcal{L}(e) + \sum_{e''=e+1}^{e'-1} \mathcal{L}(e'') + \gamma_R^{e'g}\mathcal{L}(e'), & \text{if } e < e', \\ \gamma_L^{eg}\mathcal{L}(e) + \sum_{e''=e+1}^{e'-1} \mathcal{L}(e'') + (1-\gamma_R^{e'g})\mathcal{L}(e'), & \text{if } e > e'. \end{cases} \quad (d_{LR}^{g\mathcal{P}})$$

$$d_{RL}^{ee'gg'} = \begin{cases} |\gamma_L^{eg} - \gamma_R^{eg'}|\mathcal{L}(e), & \text{if } e = e', \\ (1-\gamma_R^{eg})\mathcal{L}(e) + \sum_{e''=e+1}^{e'-1}\mathcal{L}(e'') + \gamma_L^{e'g'}\mathcal{L}(e'), & \text{if } e < e', \\ \gamma_R^{eg}\mathcal{L}(e) + \sum_{e''=e+1}^{e'-1}\mathcal{L}(e'') + (1-\gamma_L^{e'g'})\mathcal{L}(e'), & \text{if } e > e'. \end{cases} \quad (d_{RL}^{g\mathcal{P}})$$

This distance requires binary variables $z_{LR}^{ee'g}$ and $z_{RL}^{ee'gg'}$ that determines which of the expressions in $(d_{LR}^{g\mathcal{P}})$ and $(d_{RL}^{g\mathcal{P}})$ define $d_{LR}^{ee'g}$ and $d_{RL}^{ee'gg'}$, respectively.

$$z_{LR}^{ee'g} = 1 \text{ iff } \mu_L^{eg} = 1 \text{ and } \mu_R^{e'g} = 1, \qquad z_{RL}^{ee'gg'} = 1 \text{ iff } \mu_L^{eg} = 1 \text{ and } \mu_R^{e'g'} = 1,$$

where μ_L^{eg} and μ_R^{eg} are indicator variables that attain the value one if the launching point (resp. rendezvous point) associated with the graph g is placed in the segment $\overline{V^iV^{i+1}}$ of the edge $e = (i, i+1) \in \mathcal{P}$. With all this, we can model the movement of the mothership on the polygonal as follows:

$$x_L^g = \sum_{e=(i,i+1)\in\mathcal{P}} \mu_L^{eg}\left[V^i + \gamma_L^{eg}(V^{i+1} - V^i)\right] \qquad \forall g \in \mathcal{G}, \quad (33)$$

$$x_R^g = \sum_{e=(i,i+1)\in\mathcal{P}} \mu_R^{eg}\left[V^i + \gamma_R^{eg}(V^{i+1} - V^i)\right] \qquad \forall g \in \mathcal{G}, \quad (34)$$

$$z_{LR}^{ee'g} = \mu_L^{eg}\mu_R^{e'g} \qquad \forall e, e' \in \mathcal{P}\ \forall g \in \mathcal{G}, \quad (35)$$

$$z_{RL}^{ee'gg'} = \mu_R^{eg}\mu_L^{e'g'} \qquad \forall e, e' \in \mathcal{P}\ \forall g, g' \in \mathcal{G}, \quad (36)$$

$$\sum_e \mu_L^{eg} = 1 \qquad \forall g \in \mathcal{G}, \quad (37)$$

$$\sum_e \mu_R^{eg} = 1 \qquad \forall g \in \mathcal{G}, \quad (38)$$

$$d_{LR}^g = \sum_{e,e'} z_{LR}^{ee'g} d_{LR}^{ee'g} \qquad \forall g \in \mathcal{G}, \quad (39)$$

$$d_{RL}^{gg'} = \sum_{e,e'} z_{RL}^{ee'gg'} d_{RL}^{ee'gg'} \qquad \forall g, g' \in \mathcal{G}, \quad (40)$$

where $d_{LR}^{ee'g}$ and $d_{RL}^{ee'gg'}$ are defined in $(d_{LR}^{g\mathcal{P}})$ and $(d_{RL}^{g\mathcal{P}})$, respectively. Constraints (33) and (34) parameterize the launching and rendezvous points on the polygonal chain. Constraints (35) and (36) set the binary variables $z_{LR}^{ee'g}$ and $z_{RL}^{ee'gg'}$ by means of the binary variables μ_L^{eg} and μ_R^{eg}. Constraints (37) and (38) state that, for each target graph, the launching and rendezvous points can be only in one segment, not necessarily the same. Constraints (39) and (40) compute the total distance between the launching and the rendezvous points for each pair of points. The reader may note that the above representation is similar to that given by (19)-(26) with the exception that constraints (33)-(40) are indexed in the set of graphs $g \in \mathcal{G}$ instead that on the stages. This allows a remarkable reduction in the number of distance variables and constraints as already discussed in Section 4.1.3.

The goal of the PMDRPG is to find a feasible solution that minimizes the total distance covered by the

drone and the mothership, i.e.:

$$\min \quad \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g}(u^{e_g}d_L^{e_g}+v^{e_g}d_R^{e_g})+\sum_{g\in\mathcal{G}}\sum_{e_g\in E_g}\mu^{e_g}d^{e_g}+\sum_{g\in\mathcal{G}}\sum_{e_g,e'_g\in E_g}z^{e_ge'_g}d^{e_ge'_g}+\sum_{g\in\mathcal{G}}d_{LR}^{g}+\sum_{g,g'\in\mathcal{G}}d_{RL}^{gg'}w^{gg'} \tag{PMDRPG-MTZ}$$

s.t. (10) – (17),
(MTZ_1) – (MTZ_2) or (SEC),
(MTZ_3) – (MTZ_6) or (SEC),
(33) – (40),
(α-E) or (α-G),
(DCW-g),
($d_{LR}^{g\mathcal{P}}$), ($d_{RL}^{g\mathcal{P}}$),
(DIST_1-g) – (DIST_6-g),
(ORIG_1) – (DEST_2).

The reader may note that the above formulation actually encloses two different ones depending whether it includes constraints (MTZ_1) - (MTZ_2) or (SEC). However, for the sake of presentation we have preferred to present them in this compact form to reduce the length of the paper.
As mentioned at the beginning of this section, the PMDRPG is an intermediate variant of the problem, between the case in which the mothership moves freely in the continuous space and the more general situation in which the mothership moves on a general network. For modelling details of this latter case we refer the reader to the Appendix.

4.3. Strengthening the formulations of MDRPG

The different models that we have proposed include in one way or another big-M constants. We have defined different big-M constants along this work. In order to strengthen the formulations we provide tight upper and lower bounds for those constants. The computation of these bounds is rather technical and may distract the reader from the main focus of the paper. Therefore, we have preferred to present in the Appendix, at the end of the paper, all the details and results that adjust the different bounds for each one of the the models studied in this paper.

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

This section is devoted to present our matheuristic approach to address the solution of the MDRPG. Our motivation comes from the fact that the exact methods presented in the previous section are highly time demanding. Alternatively, the matheuristic provides good quality solution in limited computing times.
First, we focus on the case in which the mothership can move freely on the continuous space, namely AMDRPG. Since dealing with the exact model is a hard task for medium size instances, our strategy is to split the problem. The rationale of this algorithm rests on decomposing the problem in simpler subproblems decoupling the decisions made on the route followed by the mothership and the ones made on the drone. To do that, first we solve a TSP for the mothership over the neighborhoods of centroids of target graphs, that is an extension of the crossing postman problem where the Hamiltonian routes have to visit neighborhoods or polygonal chains rather than edges (XPPN, see [26]) and then we solve drone operations one at a time based on the order previously defined for the mothership. In this way each subproblem is much simpler and once their solutions are found one can integrate them into a feasible solution of the whole problem. Specifically, the basic idea of the algorithm is to determine first the sequence of visits to the target graphs in $\mathcal{G}$. This route is obtained replacing each graph in $\mathcal{G}$ by its centroid and then solving a TSP over the neighborhoods of those centroids, namely a XPPN. Then, given such an order of visits, the launching/rendezvous points where the mothership must stop, are determined by solving an AMDRPG problem, but limited to one single graph each time, following the sequence previously computed. Successively, the solution obtained is improved by providing it to the solver as initial partial solution and by solving, once more, another AMDRPG problem, where now we fix the launching/rendezvous points but leave free the variables w that give the sequence of visits to the targets graphs, i.e., without fixing the order of visits. If this solution induces a different order of visits to the graphs with respect to the previous one, the procedure is repeated to generate a new feasible solution. The stopping criterion is to find a solution that does not change with respect to the one obtained in the previous cycle of the algorithm. In the following, we present the pseudo-code of this algorithm:

STEP 1 (Centroids of the graphs)
- Let *orig* be the origin/destination of the mothership tour (orange point with label 0 in Figure 4 [a]).
For each graph $g \in \mathcal{G}$:
- identify its centroid c_g and consider its neighborhood defined as the circle $\rho(c_g, 2)$ centered at c_g and with radius 2 (red points in Figure 4 [a]).

STEP 2 (Order of visit of the graphs) Determine an order of visit for the graphs in $\mathcal{G}$ by solving the XPPN of the mothership over the set of the neighborhoods associated with the centroids of those graphs (blue tour 0, 1, 2, 3, 4, 0 in Figure 4 [a]).

STEP 3 (Determining the location of launching/rendezvous points) Let $\bar{w}_{gg'}$, $\forall g, g^{'} \in \mathcal{G}$ be the optimal values of the variables $w_{gg'}$ generated by STEP 2.
Following this order of visit, set the launching point for the first graph as the depot, then solve the resulting AMDRPG limited to the first graph.
Repeat the same procedure for the remaining graphs to be visited, by solving AMDRPG on one single graph each time, by fixing as launching point of the current graph the rendezvous point of the previous graph.

STEP 4 (Solution update) Let $\bar{z}$ be the solution obtained by STEP 3, consisting of the tour of the drone on each target (Figure 4 [b]), and let $\bar{x}^g_L$, $\bar{x}^g_R$ $\forall g \in \mathcal{G}$ be the associated launching/rendezvous points (green and blue points in Figure 4 [b]).
Solve the model AMDRPG with these launching/rendezvous points but leaving free the $w_{gg'}$ $\forall g, g^{'} \in \mathcal{G}$ variables and providing to the solver $\bar{z}$ as initial partial solution.

STEP 5 Let $\hat{z}$ be the updated solution obtained by STEP 4 (see Figure 4 [b] for this illustrative example) and $\hat{w}_{gg'}$ the associated order of visit of the graphs.
If the $\hat{w}_{gg'} \neq \bar{w}_{gg'}$ repeat from STEP 3, otherwise stop.

Note that in the example shown in Figure 4 we solved the AMDRPG model by imposing that at least a given percentage of each target graph must be visited. For this reason the solution consists in drone's missions which visit only part of each graph, as represented by the red segments in Figure 4 [b]. Moreover, for this example, most of the launching/rendezvous points coincide. More specifically, considering Figure 4 [b], the drone starts from the origin and visits part of the first target. The mothership retrieves it at the point labelled 1. From the same point the drone starts the second mission to visit part of the second target and then reaches the mothership at the point labelled 2. From that point the drone starts for visiting the third target meeting the mothership at the point labelled 3. From this latter point the last mission of the drone starts and ends on the mothership at the point labelled 4. Then the mothership with the drone go back to the origin. In this example the solution obtained at STEP 4, by fixing the launching/rendezvous points but not the order of visit, does not change with respect to the one obtained at STEP 3. Thus the procedure stops after the first iteration and provides the solution shown in Figure 4 [b]. We note in passing that the exact solution of AMDRPG on STEP 4 can still be hard, specially on large instances. For this reason the stopping criterion of each call of the solver within the heuristic procedure is set to a maximum number of feasible solutions found.
The procedure followed for the NMDRPG problem consists in the same steps of the matheuristic presented above for the AMDRPG problem but, at STEP 4, the partial solution, given as input to the solver, includes also the values of the variables modelling the routing of the mothership on the graph.

6. Experimental results

In this section we discuss the experimental results obtained testing the formulations presented in Section 4 and the matheuristic procedure proposed in Section 5 on different sets of instances. In particular, we generate two sets of instances of the AMDRPG problem. The first one consists of targets, to be visited by the drone, that are represented by grid graphs, while the second one involves a different typology of targets, namely, Delaunay graphs. For both typologies we generate 5 instances of 10 graphs each, with different cardinality of the set of nodes. More precisely, each instance is composed of 3 graphs with 4 nodes, 3 graphs with 6 nodes, 3 graphs with 8 nodes and 1 graph with 10 nodes. Moreover, we assume that the drone's speed is twice that of the mothership and that a percentage equal to 80% of each target must be visited by the drone.

[a] [b]

Figure 4: Illustrative example

In order to place the 10 graphs of a single instance, we consider a square of side 100 units. Then we divide the original square in subsquares of side 5 and we randomly select among them the locations for the ten target graphs of the instance. The generation procedure of the single graph in a selected square, depends on the graph typology. As regards grid graphs, the single subsquare, like the one represented in Figure 5, is further partitioned in subsquares of side $\frac{5}{n}$ where n is the cardinality of the set of nodes of the graph to build. Two opposite corner subsquares are considered (like V_1 and V_2 in Figure 5) and one point inside each of them is randomly selected (black points in the subsquares V_1 and V_2 in Figure 5). Then, a rectangle whose diagonal joins these two points is built (the black dotted one in Figure 5). A grid of n points is identified by locating $\frac{n}{m}$ equally spaced points on the two sides square (the red ones and the two original points in black in Figure 5), where m is randomly selected in the set of divisors of the number of points of the graph. The links of the graphs connect each point to its adjacent ones lying on the same side and with the one located on the opposite side of the square. Let $width_x$ and $width_y$ be the lengths of these edges as show in Figure 5. In order to perturb the coordinates of these points, we randomly add a value, ranging between $-\frac{width_x}{3}$ and $\frac{width_x}{3}$ to the x coordinate and between $-\frac{width_y}{3}$ and $\frac{width_y}{3}$ to the y coordinate, always imposing that the perturbed point still belongs to the square. The resulting grid graph is obtained connecting the same pairs of points but with perturbed coordinates (blue graph in Figure 5).
As regards the instances related to Delaunay graphs, we select their locations following the same procedure as for the grid ones. Then, for each randomly chosen subsquare, given the set of n nodes of the graph to be generated, a Delaunay triangulation of them is computed by using the Python class scipy.spatial.Delaunay (see [29] for further details).

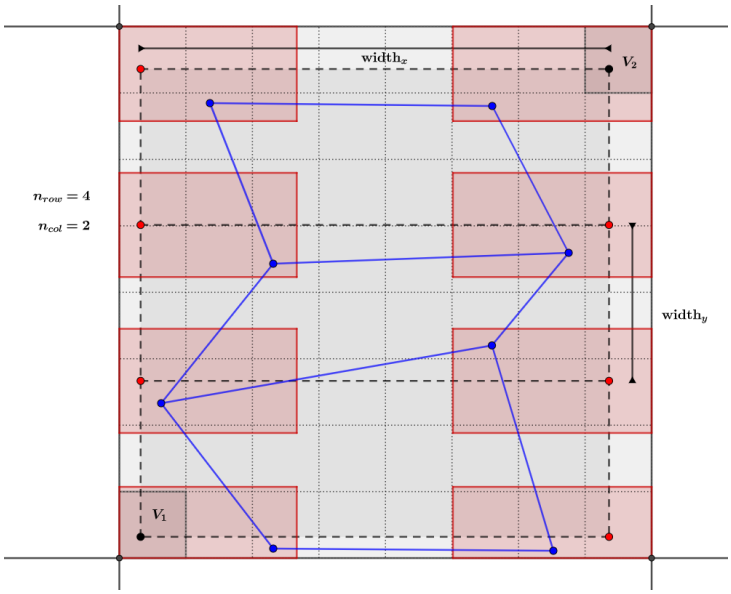


Figure 5: Example of generation of a grid graph

We run the three formulations proposed in Section 4 for the AMDRPG problem (Stages, MTZ and SEC) with two different commercial solvers, Cplex 12.8 and Gurobi 9.03, called by means of Python, by setting

a time limit for each run equal to 3600 sec. In Table 7 we report the results obtained, in terms of average, minimum and maximum percentage gap with both solvers, on the instances consisting of grid graphs. First, we can observe that Gurobi has better performances with respect to Cplex for all the instances. Moreover, for this set of instances, the SEC formulation is the best one among the three proposed, as the associated average gap is equal to 0.61% and also the minimum and the maximum percentage gap are smaller than the ones associated with formulations Stages and MTZ.

Table 7: Comparison between formulations for grid instances

Gap %	Average		Min		Max	
Solver	Cplex	Gurobi	Cplex	Gurobi	Cplex	Gurobi
Formulation						
Stages	0,87	0,87	0,85	0,84	0,88	0,88
MTZ	0,66	0,62	0,59	0,58	0,72	0,65
SEC	0,65	0,61	0,59	0,57	0,70	0,64

Similarly, Table 8 summarizes the results obtained on the instances with Delaunay graphs. Also in this case the Gurobi performance is better than Cplex. However, among the three formulations, the MTZ one provides the best results in terms of average and maximum percentage gap.

Table 8: Comparison between formulations for Delauney instances

Gap %	Average		Min		Max	
Solver	Cplex	Gurobi	Cplex	Gurobi	Cplex	Gurobi
Formulation						
Stages	0,91	0,91	0,90	0,89	0,93	0,93
MTZ	0,78	0,74	0,74	0,70	0,82	0,79
SEC	0,77	0,75	0,73	0,69	0,82	0,81

In order to test the performance of the matheuristic proposed in Section 5, we code it in Python and we run it on the same sets of instances (Grid and Delaunay) on which the three exact formulations were solved. Table 9 reports for each of the five instances, numbered from 0 to 4, distinguishing between Grid and Delauney, respectively, the best objective function provided by the best formulation, the objective function provided by the matheuristic and the associated CPU time. As already noticed from Table 8, for grid graph instances, SEC formulation has the best behaviour, with the exception of the instance number 3 for which the MTZ provides a smaller value of the objective function. As for the Delaunay graph instances, MTZ is the best formulation, but also in this case there is an exception on the instance number 2 for which SEC formulation returns a smaller value of the objective function.
The results show that the matheuristic returns a solution with value of the objective function that is higher than the one provided by the SEC formulation on grid instances. However, these values are smaller than the ones provided by the Stages and the MTZ formulations. Moreover, the saving in terms of solution time is very significant as the maximum CPU time is less than 1 minute. As regards the Delaunay instances, the matheuristic performance is even better, as it finds a solution that is better than the best one provided by the MTZ formulation and in a solution time that is at most 28 minutes.

Table 9: Heuristic performances

#	**Grid**			**Delauney**		
	Best Obj	Obj Heuristic	CPU Time	Best Obj	Obj Heuristic	CPU Time
0	1087,87	1117,83	50,99	947,01	934,46	52,49
1	1100,38	1319,64	24,64	986,22	938,68	72,73
2	1350,67	1126,35	46,06	888,48	865,66	1073,80
3	1218,66	1476,36	27,18	1249,69	1154,62	1703,33
4	1297,77	1424,37	40,91	1239,93	1184,67	81,15

We perform a second set of experiments by observing that, even if there are small differences between the SEC and the MTZ formulations depending on the type of instances, their performances are comparable. Thus, in the rest of the tests we focus on the MTZ formulation. We compare its performance, with or without providing the initial solution found by the matheuristic, on a set of larger instances. More precisely, we generate 20 instances with targets represented by grid graphs and 20 instances with targets represented by Delauney graphs. The instances of each typology are split in 4 groups of 5 instances each, consisting respectively of 5, 10, 15 and 20 targets to be visited. In each instance the same percentage of graphs (20%) has respectively 4, 6, 8, 10 and 12 nodes. Moreover, we assume that the origin coincides with the destination in all instances and we randomly generate with uniform distribution between 0 and 1, two values representing the percentage of each edge and of each graph to be visited. As regards the speeds, we set the speed of the drone three times the one of the mothership. We run the MTZ

formulation using Gurobi and setting a time limit of 7200 sec. for each instance. On the same instances also the matheuristic is applied. Note that, in order to define a stopping rule for the exact solution of the AMDRPG model within the matheuristic procedure (STEP 3 and STEP 4), we set a maximum number of solutions generated by the solver equal to five. For each instance, the solution provided by the matheuristic is then used to initialize the exact application of the MTZ formulation in order to try to speed up the solution process. Table 10 shows the results of the comparison between the exact formulation with and without initialization. In the first column, named List, we report the size of the instances in terms of number of targets to be visited (0, 1, 2 and 3 identifies instances respectively with 5, 10, 15 and 20 graphs). The second column refers to the two variants of the model, that is, a given percentage of each edge of the targets (e) or a given percentage of each target (g) must be visited by the drone. The other columns report respectively the average percentage gap of the solutions found within the time limit starting from the initial solution provided by the matheuristic (% Gap (i)), the average running time of the matheuristic (Time h) and the average percentage gap of the solutions found within the time limit without initialization (% Gap (wi)). This information is reported for both Grid and Delauney instances.

Table 10: Comparison between exact resolution with and without initialization

		Grid			**Delauney**		
List	%	% Gap (i)	Time h	% Gap (wi)	% Gap (i)	Time h	% Gap (wi)
0	e	0.72	105.12	0.73	0.78	154.92	0.74
	g	0.55	58.92	0.54	0.62	92.64	0.67
1	e	0.76	241.99	0.76	0.80	314.69	0.79
	g	0.71	182.61	0.70	0.74	353.04	0.75
2	e	0.76	367.69	0.76	0.80	447.61	0.80
	g	0.71	326.49	0.72	0.76	429.16	0.76
3	e	0.75	481.68	0.74	0.80	514.98	0.76*
	g	0.71	492.27	0.70	0.77	582.90	0.77

From Table 10 we can notice that in most of the cases the average gaps associated with the solution found within the time limit, with and without initialization by the solution found by the matheuristic, are the same or very close (note that in the last column the $*$ indicates that only one instance has been solved within the time limit). As regards the running time of the matheuristic, we can see also from the boxplots in Figure 6, that it increases with the number of targets to be visited both for Grid and Delaunay instances. Considering the model variants based on the minimum percentage of each edge or each graph to visit, we can observe that for Grid instances the average running time of the model imposing a minimum percentage of each edge to be visited, is greater than the one associated with the other variant, with the exception of the instances of biggest size (List=3).
The boxplots in Figure 7 represent the percentage gap of the solution provided by the matheuristic with respect to the one provided by the exact resolution of the MTZ model within the time limit, with initialization by the solution found by the matheuristic. From them we can notice that the gap increases with the size both for Grid and Delaunay instances but it is always less than 0.5%. Figure 8 shows the percentage gap of the solution provided by the exact solution of the MTZ formulation within the time limit without the initialization, with respect to the one found with the initialization. This gap is very close to 0 both for Grid and Delaunay instances. Only for the biggest size we observe values ranging between 0.1% and 0.6%. These observations suggest that, even if the initialization of the model by the solution provided by the matheuristic does not speed up the convergence to the optimal solution, the matheuristic provides solutions of very good quality. Indeed, it generates in less than 10 minutes solutions that are very close to the ones provided by the model within 2 hours.

As regards the NMDRPG problem, we generate three sets of instances with targets represented by grid graphs considering different structures of the network where the mothership can move. In particular, we define a first set of instances where the mothership network is represented by a graph of 6 nodes with a tree structure with origin of the path of the base vehicle different from the destination. A second set of instances involving a mothership network consisting of a complete graph of 4 vertices with origin of the path of the base vehicle different from the destination. A third set of instances characterized by star graphs of 7 nodes representing the mothership network, where the origin coincides with the destination and it is located at the centre of the star. We generate 10 instances for each of these three classes, 5 of them with 5 targets and 5 with 10 targets to be visited. Moreover, as for the AMDRPG, for each of these 10 instances we randomly generate two values representing the percentage of each edge and of each graph that must be visited by the drone. We run on these sets of instances both Stages and MTZ formulations. Table 11 summarizes the results obtained comparing them. The first column identifies the size of the instances, similarly to Table 10, (0 for instances with 5 targets and 1 for instances with 10 targets). The

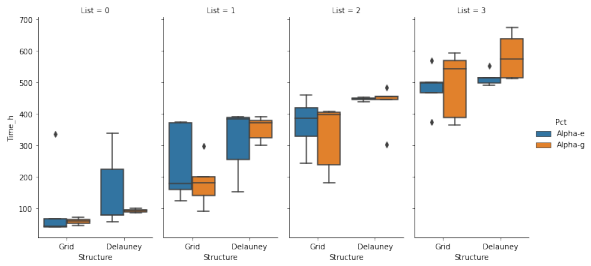


Figure 6: Matheuristic running time

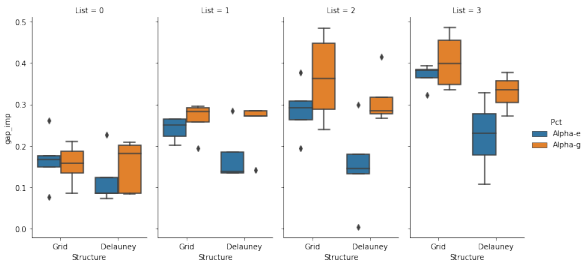


Figure 7: Matheuristic improved gap

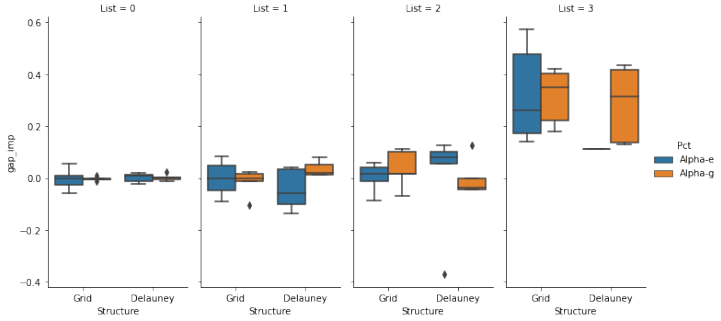


Figure 8: Improved gap of MTZ formulation with and without initialization

second column distinguishes between minimum percentage of each edge (e) or of each graph (g) to be visited by the drone. The remaining columns refer to the three different classes of instances described above (1 for the networks with a tree structure, 2 for complete networks and 3 for start networks). For each of these sets of instances the average percentage gap of the solutions found within the time limit of 7200 sec. by the two formulations (Stages and MTZ) is reported.

Table 11: Comparison between formulations of NMDRPG

	Net Struct	1		2		3	
List	%	Stages	MTZ	Stages	MTZ	Stages	MTZ
0	e	0.89	0.33	0.88	0.24	0.87	0.39
	g	0.86	0.29	0.89	0.18	0.90	0.42
1	e	0.92	0.43	0.92	0.33	0.92	0.46
	g	0.91	0.36	0.92	0.23	0.92	0.39

We can observe that for each class of instances and model variants, based on the percentage of each edge or each graph to be visited, the MTZ formulation performs better than the Stages one. In all the cases the percentage average gap associated with the MTZ formulation is one third or half of that associated with the Stages formulation. For this reason, in the following tests, related to the comparison between the exact solution of the NMDRPG model with and without the initialization by the solution found by the matheuristic, we focus only on the MTZ formulation.
Table 12 summarizes the results of this comparison distinguishing again between the different network structures (columns labelled 1, 2 and 3), the different size (rows labelled 0 and 1) characterizing the instances and model variants (minimum percentage of each edge (e) or each graph (g) to be visited). For each combination of network structure, size and model variant we report the average percentage gap with initialization (% Gap (i)), the solution time of the matheuristic (T_h) and the average percentage gap without initialization by the solution found by the matheuristic (% Gap (wi))

Table 12: Comparison between exact resolution with and without initialization of NMDRPG

Net Struct		1			2			3	
List	%	% Gap (i)	T_h	% Gap (wi)	% Gap (i)	T_h	% Gap (wi)	% Gap (i)	T_h	% Gap (wi)
0	e	0.32	109.96	0.33	0.24	207	0.24	0.39	177.57	0.39
	g	0.30	110.92	0.29	0.18	163.36	0.18	0.45	149.68	0.42
1	e	0.48	1030.64	0.43	0.39	802.3	0.33	0.53	770.05	0.46
	g	0.33	479.36	0.36	0.35	639.09	0.23	0.42	689.51	0.39

We can observe that, similarly to the AMDRPG problem, the average gaps associated with the solution found within the time limit, with and without initialization by the solution found by the matheuristic, are very close. Considering the average running time we can notice that the NMDRPG problem is more

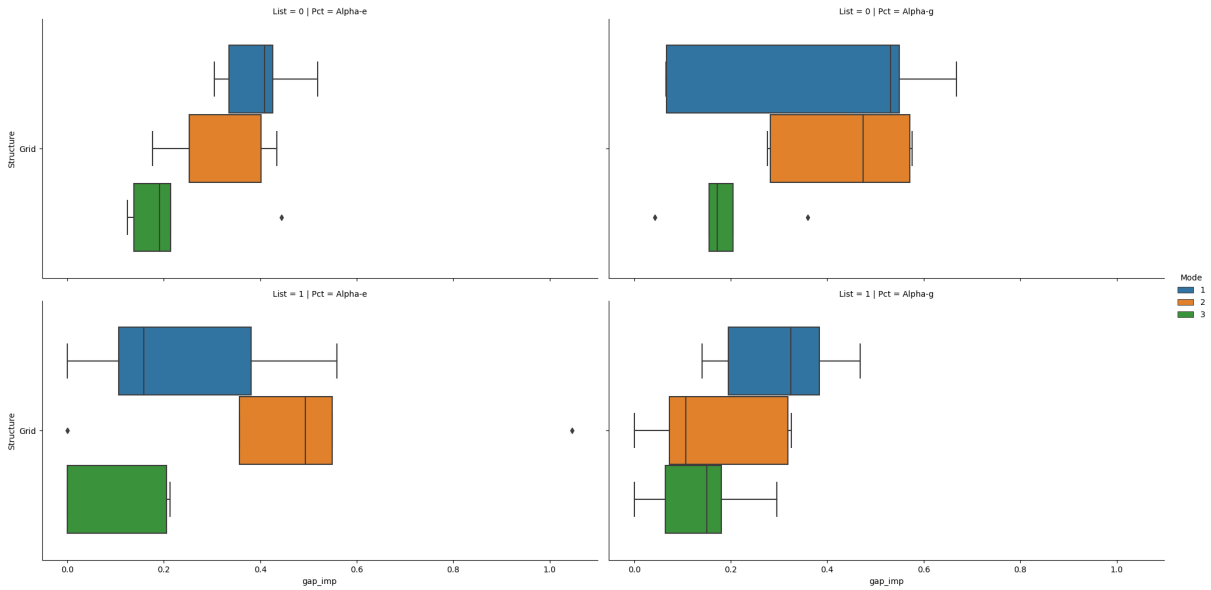


Figure 9: Matheuristic improved gap for NMDRPG

challenging to be solved than the AMDRPG. The solution time increases very fast with the size of the instances especially for the case in which the network where the mothership moves has a tree structure. Moreover, as for the Grid instances in the continuous case, the model variant imposing a minimum percentage of each edge to be visited takes more time to be solved.
The boxplots showed in Figure 9 report the percentage gap of the solution provided by the matheuristic with respect to the one provided by the exact solution of the MTZ model within the time limit, with initialization by the solution found by the matheuristic. We can notice that, excluding the outliers, this gap ranges between 0% and 0.7% and its lowest values are observed for the instances in which the network where the mothership moves has a star structure (green boxplots). From the previous observations, similarly to the AMDRPG, we can conclude that the behaviour of the matheuristic is very good in terms of quality of the solutions provided, even if used as initialization for the MTZ model does not help in speeding up the convergence to the optimal solution.

7. Concluding remarks

This papers has analyzed the coordination problem that arises between a mothership vehicle and a drone that must adjust their routes to minimize travel distances while visiting a set of targets modeled by graphs. We have presented exact formulations for different versions of the problem depending on the constraints imposed to the mothership movement (free on a continuous space or on a given network). Our computational results show that the considered problem is rather hard and only small to medium size problems can be solved to optimality. Additionally, we have proposed a matheuristic algorithm, applicable to all the versions of the problem with minimum changes, that provides acceptable feasible solutions in short computing time; so that it is a good alternative to the exact methods.
Further research in this topic includes the coordination of the operations of several drones with a mothership, the possibility of visiting more than one target per operation and combinations of both cases. These problems being very interesting are beyond the scope of this paper and will be the focus of a follow up paper.

Acknowledgements

This research has been partially supported by Spanish Ministry of Education and Science/FEDER grant number MTM2016-74983-C02-(01-02), Junta de Andalucia project P18-FR-1422 and projects FEDER-US-1256951, CEI-3-FQM331 and *NetmeetData*: Ayudas Fundación BBVA a equipos de investigación científica 2019, and by University of Rome, Sapienza grant number: RM11916B7F962975.

References

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

[2] Amorosi, L., Chiaraviglio, L., D'Andreagiovanni, F., and Blefari-Melazzi, N. (2018). Energy-efficient mission planning of UAVs for 5G coverage in rural zones. Proceedings of the IEEE International Conference on Environmental Engineering, EE 2018.

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

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

[5] Campbell, J. F., Corberán, Á., Plana, I., and Sanchis, J. M. (2018). Drone arc routing problems. Networks, 72(4):543–559.

[6] Campbell, J. F., Sweeney, D., and Zhang, J. (2017). Strategic design for delivery with trucks and drones. Computer Science.

[7] Carlsson, J. G. and Song, S. (2018). Coordinated logistics with a truck and a drone. Management Science, 64(9):4052–4069.

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

[9] Chiaraviglio, L., Amorosi, L., Blefari-Melazzi, N., Dell'Olmo, P., Natalino, C., and Monti, P. (2018). Optimal Design of 5G Networks in Rural Zones with UAVs, Optical Rings, Solar Panels and Batteries. Proceedings of the 20th International Conference on Transparent Optical Networks, ICTON 2018.

[10] Chiaraviglio, L., Amorosi, L., Malandrino, F., Chiasserini, C. F., Dell'Olmo, P., and Casetti, C. (2019b). Optimal Throughput Management in UAV-based Networks during Disasters. INFOCOM 2019 - IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS 2019, pages 307–312.

[11] Chow, J. Y. (2016). Dynamic UAV-based traffic monitoring under uncertainty as a stochastic arc-inventory routing policy. International Journal of Transportation Science and Technology, 5(3):167–185.

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

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

[14] Dille, M. and Singh, S. (2013). Efficient aerial coverage search in road networks. AIAA Guidance, Navigation, and Control (GNC) Conference.

[15] Garone, E., Naldi, R., Casavola, A., and Frazzoli, E. (2010). Cooperative mission planning for a class of carrier-vehicle systems. Proceedings of the IEEE Conference on Decision and Control, pages 1354–1359.

[16] Jimenez, J. G., Chiaraviglio, L., Amorosi, L., and Blefari-Melazzi, N. (2018). Multi-Period Mission Planning of UAVs for 5G Coverage in Rural Areas: A Heuristic Approach. Proceedings of the 2018 9th International Conference on the Network of the Future, NOF 2018, pages 52–59.

[17] Li, M., Zhen, L., Wang, S., Lv, W., and Qu, X. (2018). Unmanned aerial vehicle scheduling problem for traffic monitoring. Computers and Industrial Engineering, 122:15–23.

[18] Mathew, N., Smith, S. L., and Waslander, S. L. (2015). Planning Paths for Package Delivery in Heterogeneous Multirobot Teams. IEEE Transactions on Automation Science and Engineering, 12(4):1298–1308.

[19] Moshref-Javadi, M. and Lee, S. (2017). Using drones to minimize latency in distribution systems. In IE Annual Conference, Proceedings, pages 235–240.

[20] Mourelo Ferrandez, S., Harbison, T., Weber, T., Sturges, R., and Rich, R. (2016). Optimization of a truck-drone in tandem delivery network using k-means and genetic algorithm. Journal of Industrial Engineering and Management; Vol 9, No 2 (2016)DO - 10.3926/jiem.1929.

[21] Oh, H., Kim, S., Tsourdos, A., and White, B. A. (2014). Coordinated road-network search route planning by a team of UAVs. International Journal of Systems Science, 45(5):825–840.

[22] Oh, H., Shin, H. S., Tsourdos, A., White, B. A., and Silson, P. (2011). Coordinated Road Network Search for Multiple UAVs Using Dubins Path BT - Advances in Aerospace Guidance, Navigation and Control. pages 55–65, Berlin, Heidelberg. Springer Berlin Heidelberg.

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

[24] Poikonen, S. and Campbell, J. (2020). Future directions in drone routing research. Networks, pages 1–11.

[25] Poikonen, S. and Golden, B. (2020). Multi-visit drone routing problem. Computers and Operations Research, 113:104802.

[26] Puerto, J. and Valverde, C. (2020). Routing for unmanned aerial vehicles: Touring dimensional sets. arXiv.

[27] Tokekar, P., Hook, J. V., Mulla, D., and Isler, V. (2016). Sensor Planning for a Symbiotic UAV and UGV System for Precision Agriculture. IEEE Transactions on Robotics, 32(6):1498–1511.

[28] Trotta, A., Andreagiovanni, F. D., Di Felice, M., Natalizio, E., and Chowdhury, K. R. (2018). When UAVs Ride A Bus: Towards Energy-efficient City-scale Video Surveillance. Proceedings - IEEE INFOCOM, 2018-April:1043–1051.

[29] Virtanen, P., Gommers, R., Oliphant, T. E., and et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. Nature Methods, 17(3):261–272.

[30] Wen, T., Zhang, Z., and 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).

Appendix

7.1. Strengthening the formulations of MDRPG

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

- AMDRPG. To linearize the first term of the objective function in AMDRPG, we define the auxiliar non-negative continuous variables $p_L^{e_g t}$ (resp. $p_R^{e_g t}$) to model the product $d_L^{eg} u^{e_g t}$ and include the constraints (8) and (9), namely: $p_L^{e_g t} \geq m_L^{e_g} u^{e_g t}$, $p_L^{e_g t} \leq d_L^{e_g} - M_L^{e_g t}(1 - u^{e_g t})$.

  The best upper bound $M_R^{e_g t}$ or $M_L^{e_g t}$ that we can consider is the full diameter of the data, that is the maximum distance between every pair of vertices of the graphs $g \in \mathcal{G}$. every launching or rendezvous point is inside the circle whose diametrically opposite points are determined by the following expression.

$$M_R^{e_g t} = \max_{\{v \in V_g, v' \in V_{g'} : g, g' \in \mathcal{G}\}} \|v - v'\| = M_L^{e_g t}.$$

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

- NMDRPG. In this case, the best upper bounds for $M_R^{e_g t}$ or $M_L^{e_g t}$ is the maximum distance between the polygonal chain $\mathcal{P}$ or the graph $\mathcal{N}$ and any of the target graphs $g \in \mathcal{G}$:

$$M_R^{e_g t} = \max_{\{v \in V_g, w \in \mathcal{N}\}} \|v - w\| = M_L^{i_g t}.$$

  On the other hand, the minimum distance can be computed by taking the closest points between the graph g and the network $\mathcal{N}$:

$$m_R^{e_g t} = \min_{\{v \in V_g, w \in \mathcal{N}\}} \|v - w\| = m_L^{e_g t}.$$

Bounds on big-M constants for distances from the launching to the rendezvous points for the MTZ/SEC formulations in AMDRPG

To linearize the product of $d_{RL}^{gg'} w^{gg'}$ we use the constraints:

$$\begin{aligned} p^{gg'} &\geq m_{RL}^{gg'} d_{RL}^{gg'}, \\ p^{gg'} &\leq d_{RL}^{gg'} - M_{RL}^{gg'}(1 - w^{gg'}). \end{aligned}$$

The upper bound on $M_{RL}^{gg'}$ is given by the diameter of $g \cup g'$, namely $M_{RL}^{gg'} = \max_{\{v \in V_g, v' \in V_{g'}\}} \|v - v'\|$.

Bounds on big-M constants for distances from launching to rendezvous points on target graph $e \in \mathcal{G}$.

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 another product of variables linearized by the following constraints:

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

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

$$\begin{aligned} M^{e_g e'_g} &= \max\{\|B^{e_g} - C^{e'_g}\|, \|B^{e_g} - B^{e'_g}\|, \|C^{e_g} - B^{e'_g}\|, \|C^{e_g} - C^{j_g}\|\}, \\ m^{e_g e'_g} &= \min\{\|B^{e_g} - C^{e'_g}\|, \|B^{e_g} - B^{e'_g}\|, \|C^{e_g} - B^{e'_g}\|, \|C^{e_g} - C^{e'_g}\|\}. \end{aligned}$$

Bounds on big-M constants for distances covered by the mothership on the polygonal for the PMDRPG *model during one drone operation.*

In the case of PMDRPG, we can also set tighter upper bounds for the distance covered by the drone inside the polygonal during an operation that starts in e and finishes at e' (or vice versa) (see (25) and (26)). This is clearly bounded above by the total length of the line segments where the mothership is located.

$$M_{LR}^{ee't} = M_{RL}^{ee't} = \begin{cases} \mathcal{L}(e), & \text{if } e = e', \\ \sum_{e < e'' < e'} \mathcal{L}(e'') & \text{if } e < e', \\ \sum_{e' < e'' < e} \mathcal{L}(e'') & \text{if } e > e'. \end{cases}$$

Bounds on big-M constants for distances covered by the drone during an operation in models by stages.

To link the drone operation with the mothership travel in the models by stages, we have defined the constraint (DCW-t) that includes the M:

$$\left(\sum_{e_g \in E_g} u^{e_g t} d_L^{e_g t} + \sum_{e_g, e'_g \in E_g} z^{e_g e'_g} d^{e_g e'_g} + \sum_{e_g \in E_g} \mu^{e_g} d^{e_g} + \sum_{e_g \in E_g} v^{e_g t} d_R^{e_g t} \right) / v_D \leq d_{RL}^t / v_M + M(1 - \sum_{e_g \in E_g} u^{e_g t})$$

To obtain this upper bound M we add to the length of the graph $\mathcal{L}(g)$ the big-Ms computed for $u^{e_g t}$ and $v^{e_g t}$, that is, $M_L^{e_g t}$ and $M_R^{e_g t}$, and the upper bounds on the distances traveled by the drone to move between edges on g. Thus,:

$$M = \mathcal{L}(g) + M_L^{e_g t} + M_R^{e_g t} + \sum_{e_g, e'_g \in E_g} M^{e_g e'_g}.$$

7.2. Network Mothership-Drone Routing Problem with Graphs (NMDRPG)

This model is a refinement of the one presented in Section 4.2. Indeed, in this case the mothership has to move on a general undirected network rather than on a polygonal chain. Even though the model in Section 4.2, namely PMDRPG, can be seen as a particular case of NMDRPG, for the sake of presentation, we have preferred to include it first to ease the understanding of the more advanced NMDRPG model. As before, we start with an approach that models the problem by stages. Later, we shall extend the analysis using the rationale of connectivity, as already done in the the previous models AMDRPG and PMDRPG.

Problem Parameters
$\mathcal{N} = (V, E)$: set of nodes and edges of the network representing the road system where the mothership can move.
$e = (i, j)$,$e' = (i', j')$: starting edge and ending edge of the mothership 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 $g \in \mathcal{G}$.
B^{e_g}, C^{e_g}: endpoints of edge e of graph $g \in \mathcal{G}$.
α^{e_g}: percentage of edge e of graph $g \in \mathcal{G}$ that must be visited.
α^{g}: percentage of graph $g \in \mathcal{G}$ that must be visited.
v_D: drone speed.
v_M: mothership speed.
M: big M.

Table 13: Nomenclature for NMDRPG

Let $\mathcal{N} = (V, E)$ be the network that models the space of movement for the mothership. For each stage $t \in T := \{1, \ldots, |\mathcal{G}|\}$, the mothership can start from an edge $e = (i, j) \in E$ and end in another one $e' = (i', j') \in E$, moving between each other following a route on $\mathcal{N}$. Thus, it must follow a route from the launching to the rendezvous point and vice versa, determined by a sequence of intermediate points as shown in Figure 10:

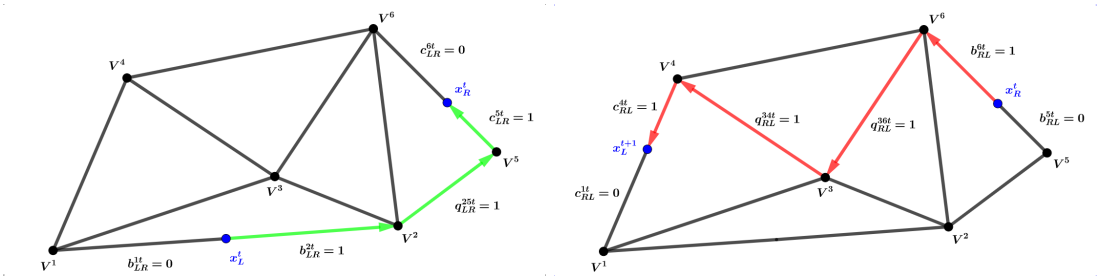


Figure 10: An example of a parameterization of a mothership route for a stage t. In the first picture, the mothership launches the drone at the point x^t_L on the edge $\overline{V^1V^2}$, then traverses the edge $\overline{V^2V^3}$ and finally moves to x^t_R on the edge $\overline{V^4V^5}$ to retrieve it. In the second one, the mothership retrieves the drone at the point x^t_R on the edge $\overline{V^4V^5}$, then traverses the edge $\overline{V^3V^6}$, the edge $\overline{V^3V^4}$ and finally moves to x^{t+1}_L on the edge $\overline{V^1V^4}$ to launch the drone for the next stage.

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, and 0 otherwise.
$entry^{e_g} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: auxiliary binary variables for linearization.
$u^{e_g t} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$, $\forall t \in T$: equal to 1 if the drone enters in graph g by e_g at stage t, 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.
$v^{e_g t} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$, $\forall t \in T$: equal to 1 if the drone exits from graph g by e_g at stage t, 0 otherwise.
s^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: integer non negative variable representing the order of visit of edge e of graph g.
$\mu^{et}_L \in \{0,1\}$, $\forall e \in E$, $\forall t \in T$: equal to 1 if the launching point x^t_L is located on e at stage t.
$\mu^{et}_R \in \{0,1\}$, $\forall e \in E$, $\forall t \in T$: equal to 1 if the rendezvous point x^t_R is located on e at stage t.
$z^{ee't}_{LR} \in \{0,1\}$ $\forall e, e' \in E$, $\forall t \in T$: equal to 1 if the launching point x^t_L is located on e and the rendezvous point x^t_R is located on e' at stage t.
$b^{it}_{LR} \in \{0,1\}$, $\forall i : e = (i,j) \in E$, $\forall t \in T$, equal to 1 if the mothership exits from x^t_L by the vertex V^i of the edge e.
$c^{it}_{LR} \in \{0,1\}$, $\forall i : e = (i,j) \in E$, $\forall t \in T$, equal to 1 if the mothership enters in x^t_R by the vertex V^i of the edge e.
$q^{et}_{LR} \geq 0$ $\forall e \in E$, $\forall t \in T$, integer variable counting the number of times the mothership fully traverses edge e to move between the launching point x^t_L on e to the rendezvous point x^t_R on e' at stage t.
$z^{ee't}_{RL} \in \{0,1\}$ $\forall e, e' \in E$, $\forall t \in T$: equal to 1 if the rendezvous point x^t_R is located on e at stage t and the launching point x^{t+1}_L is located on e' at stage $t+1$.
$b^{it}_{RL} \in \{0,1\}$, $\forall i : e = (i,j) \in E$, $\forall t \in T$, equal to 1 if the mothership exits from x^t_R by the vertex V^i of the edge e.
$c^{it}_{RL} \in \{0,1\}$, $\forall i : e = (i,j) \in E$, $\forall t \in T$, equal to 1 if the mothership enters in x^{t+1}_L by the vertex V^i of the edge e.
$q^{et}_{RL} \geq 0$ $\forall e \in E$, $\forall t \in T$, integer variable counting the number of times the mothership fully traverses edge e to move between the rendezvous point x^t_R on e to the launch point for the next stage x^{t+1}_L.

Continuous Decision Variables
$\rho^{e_g} \in [0,1]$ and $\lambda^{e_g} \in [0,1]$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: defining the entry and exit points on edge e_g.
$\gamma_R^{et} \in [0,1]$ and $\gamma_L^{et} \in [0,1]$ $\forall e \in E$, $\forall t \in T$: defining the launching and rendezvous points on edge e of the network $\mathcal{N}$at stage t.
$\nu_{\min}^{e_g}$ and $\nu_{\max}^{e_g} \in [0,1]$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: auxiliary variables for linearization.
x_L^t $\forall t \in T$: coordinates representing the point where the mothership launches the drone at stage t.
x_R^t $\forall t \in T$: coordinates representing the point where the mothership retrieves the drone at stage t.
R^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: coordinates representing the entry point in edge e of graph g.
L^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: coordinates representing the exit point from edge e of graph g.
$d_L^{e_g t} \geq 0$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$, $\forall t \in T$: representing the distance travelled by the drone from the launching point x_L^t on the mothership at stage t to the first visiting point R^{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 launching point L^{e_g} on e_g to the rendezvous 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 rendezvous point R^{e_g} to the launching point L^{e_g} on e_g.
$d_R^{e_g t} \geq 0$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$, $\forall t \in T$: representing the distance travelled by the drone from the last visiting point L^{e_g} to the rendezvous point x_R^t on the mothership at stage t.
$d_{LR}^{ee't} \geq 0$ $\forall e, e' \in E$ $\forall t \in T$: representing the distance travelled by the mothership from the launching point x_L^t on e to the rendezvous point x_R^t on e' at stage t.
$d_{RL}^{ee't} \geq 0$ $\forall e, e' \in E$ $\forall t \in T$: representing the distance travelled by the mothership from the rendezvous point x_R^t on e at stage t to the launching point $x_L^{(t+1)}$ on e' at stage $t+1$.
$d_{LR}^t \geq 0$ $\forall t \in T$: representing the total distance travelled by the mothership between the launching point x_L^t and the rendezvous point x_R^t at stage t.
$d_{RL}^t \geq 0$ $\forall t \in T$, representing the total distance travelled by the mothership between the rendezvous point x_R^t at stage t and the launching point x_L^{t+1} at stage $t+1$.

Table 14: Decision Variables for NMDRPG-ST

As already explained before in previous sections, the distance traveled by the mothership, between two consecutive launching and rendezvous points in two edges, not necessarily distinct, of the graph can be represented as:

$$d_{LR}^{ee't} = \begin{cases} |\gamma_L^{et} - \gamma_R^{et}|\mathcal{L}(e), & \text{if } e = e', \\ \left[b_{LR}^{it}\gamma_L^{et} + b_{LR}^{jt}(1-\gamma_L^{et})\right]\mathcal{L}(e) + \sum\limits_{e'' \in \mathcal{N}} q_{LR}^{e''t}\mathcal{L}(e'') + \left[c_{LR}^{i't}\gamma_R^{e't} + c_{LR}^{j't}(1-\gamma_R^{e't})\right]\mathcal{L}(e'), & \text{if } e \neq e', \end{cases} \tag{$d_{LR}^{t\mathcal{N}}$}$$

where b_{LR}^{it} (resp. c_{LR}^{it}) are binary variables that determine from which of the end-points of e (respectively e') one has to account for the distance and q_{LR}^{et} is an integer variable that counts how many times the mothership fully traverses the edge e. Furthermore, we need to define another binary variable $z_{LR}^{ee't}$ that models the correct definition of the distance in ($d_{LR}^{t\mathcal{N}}$). With the above definition one can account for the movement of the mothership at each stage $t \in T$ from a launching to a rendezvous point:

$$x_L^t = \sum_{e=(i,j)\in E} \mu_L^{et}\left[V^i + \gamma_L^{et}(V^j - V^i)\right], \qquad \forall t \in T \tag{41}$$

$$x_R^t = \sum_{e=(i,j)\in E} \mu_R^{et}\left[V^i + \gamma_R^{et}(V^j - V^i)\right], \qquad \forall t \in T \tag{42}$$

$$z_{LR}^{ee't} = \mu_L^{et}\mu_R^{e't}, \qquad \forall e, e' \in E \ \forall t \in T \tag{43}$$

$$b_{LR}^{it} \leq \sum_{e\in\delta(i)} \mu_L^{et}, \qquad \forall i \in V \ \forall t \in T \tag{44}$$

$$c_{LR}^{it} \leq \sum_{e\in\delta(i)} \mu_R^{et}, \qquad \forall i \in V \ \forall t \in T \tag{45}$$

$$b_{LR}^{it} + b_{LR}^{jt} \geq \mu_L^{et}, \qquad \forall e = (i,j) \in E \ \forall t \in T \tag{46}$$

$$c_{LR}^{it} + c_{LR}^{jt} \geq \mu_R^{et}, \qquad \forall e = (i,j) \in E \ \forall t \in T \tag{47}$$

$$b_{LR}^{it} + \sum_{\{j:(i,j)\in E\}} q_{LR}^{jit} = \sum_{\{j:(i,j)\in E\}} q_{LR}^{ijt} + c_{LR}^{it}, \qquad \forall i \in V \ \forall t \in T \tag{48}$$

$$\sum_{e\in E} \mu_L^{et} = 1, \qquad \forall t \in T \tag{49}$$

$$\sum_{e\in E} \mu_R^{et} = 1, \qquad \forall t \in T \tag{50}$$

$$d_{LR}^t = \sum_{e,e'\in E} z_{LR}^{ee't} d_{LR}^{ee't}, \qquad \forall t \in T \tag{51}$$

Constraints (41) and (42) parameterize the launching and rendezvous points in the network $\mathcal{N}$ at stage t. Constraints (43) set the binary variables $z_{LR}^{ee't}$ by means of the binary variables μ_L^{et} and $\mu_R^{e't}$. Constraints (44) and (45) state that if the launching point (resp. rendezvous point) is not on the edge e, the mothership cannot go (resp. exit) to the vertex that is incident to e. Constraints (46) state that if the mothership leaves the edge e to go to e', it must exit from one of the incident vertices to e. In the same way, constraints (47) express that if the mothership leaves the edge e' to go to e, it must enter to the edge e from one of its incident vertices. Flow conservation constraints (48) ensure that the number of incoming edges to each vertex i is equal to the number of outgoing edges in the route followed by the mothership. Constraints (49) and (50) state that, in each stage, the selected points can be located only on one edge. Finally, constraints (51) define the total distance between the launching and the rendezvous points at stage t.
Similarly, the distance covered by the mothership along the path on the network from the rendezvous point x_R^t to the next launching point x_L^{t+1} can be modeled using the following definition of distance:

$$d_{RL}^{ee't} = \begin{cases} |\gamma_R^{et} - \gamma_L^{et+1}|\mathcal{L}(e), & \text{if } e = e', \\ \left[b_{RL}^{it}\gamma_R^{et} + b_{RL}^{jt}(1-\gamma_R^{et})\right]\mathcal{L}(e) + \sum\limits_{e''\in\mathcal{N}} q_{RL}^{e''t}\mathcal{L}(e'') + \left[c_{RL}^{i't}\gamma_L^{e't+1} + c_{RL}^{j't}(1-\gamma_L^{e't+1})\right]\mathcal{L}(e'), & \text{if } e \neq e'. \end{cases} \qquad (d_{RL}^{t\mathcal{N}})$$

In this case, the binary variable $z_{RL}^{ee't}$ links the rendezvous point at stage t with the launching point at stage $t+1$. Then, we can use a set of constraints similar to those used above and the distance from x_R^t to x_L^{t+1} can be computed by means of the following additional constraints:

$$z_{RL}^{ee't} = \mu_R^{et}\mu_L^{e't+1}, \qquad (52)$$

$$b_{RL}^{it} \leq \sum_{e\in\delta(i)} \mu_R^{et}, \qquad \forall i \in V \qquad (53)$$

$$c_{RL}^{it} \leq \sum_{e\in\delta(i)} \mu_L^{et}, \qquad \forall e \in \delta(i), \forall i \in V \qquad (54)$$

$$b_{RL}^{it} + b_{RL}^{jt} \geq \mu_R^{et}, \qquad \forall e = (i,j) \in E \qquad (55)$$

$$c_{RL}^{it} + c_{RL}^{jt} \geq \mu_L^{et+1}, \qquad \forall e = (i,j) \in E \qquad (56)$$

$$b_{RL}^{it} + \sum_{\{j:(i,j)\in E\}} q_{RL}^{jit} = \sum_{\{j:(i,j)\in E\}} q_{RL}^{ijt} + c_{RL}^{it}, \qquad \forall i \in V \qquad (57)$$

$$\sum_{e\in E} \mu_L^{et} = 1, \qquad (58)$$

$$\sum_{e\in E} \mu_R^{et} = 1, \qquad (59)$$

$$d_{RL}^{t} = \sum_{e,e'\in E} z_{RL}^{ee't} d_{RL}^{ee't}. \qquad (60)$$

Constraints (52) set the binary variables $z_{RL}^{ee't}$ by means of the binary variables μ_R^{et} and $\mu_L^{e't+1}$. Constraints (53) and (54) state that if the rendezvous point (resp. launching point) is not on the edge e, the mothership cannot go (resp. exit) to the end vertices of the edge e. Constraints (55) state that if the mothership leaves the edge e, it must exit via one of the end vertices of e. In the same way, constraints (56) state that if the mothership goes to the edge e, it necessarily must enter to e from one incident vertex of e. Flow conservation constraints (57) ensure that in the route followed by the mothership the number of used incoming edges to each vertex i is equal to the number of used outgoing edges. Constraints (58) and (59) state that, in each stage, the selected points can be only on one edge. Constraints (60) express the total distance between the rendezvous and the launching points at stage t.
Hence, once the distances inside the graph are set with the above two families of constraints, we can state the mathematical programming formulation of the problem as:

$$\min \quad \sum_{g\in\mathcal{G}}\sum_{e_g\in E_g}\sum_{t\in T}(u^{e_g t}d_L^{e_g t}+v^{e_g t}d_R^{e_g t})+\sum_{g\in\mathcal{G}}\sum_{e_g\in E_g}\mu^{e_g}d^{e_g}+\sum_{g\in\mathcal{G}}\sum_{e_g,e'_g\in E_g}z^{e_g e'_g}d^{e_g e'_g}+\sum_{t\in T}(d^t_{RL}+d^t_{LR}) \tag{NDRPG-ST}$$

s.t. (1) – (7),
(41) – (60),
(MTZ_1) – (MTZ_2) or (SEC),
(α-E) or (α-G),
(DCW-t),
$(d^{t\mathcal{N}}_{LR}), (d^{t\mathcal{N}}_{RL})$,
(DIST_1-t) – (DIST_6-t),
(ORIG_1) – (DEST_2)

Again, the objective function has four terms: the first three compute the distances traveled by the drone, while the last one computes the distances traveled by the mothership. Constraints (1)-(7) model the tour made by the drone. Constraints (41)-(60) model the path followed by the mothership in the graph. The rest of the constraints are similar to those explained in the formulation (AMDRPG-ST).

7.2.1. MTZ formulation for the Network Mothership-Drone Routing Problem with Graphs

In order to apply this type of constraints to model connectivity of the routes we have to reformulate the expressions for the distances so that they are not related to stages. Table 15 summarizes the variables required to formulate this model.

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, and 0 otherwise.
$entry^{e_g} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: auxiliary binary variables for linearization.
$u^{e_g} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: equal to 1 if the drone enter in graph g by 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.
$v^{e_g} \in \{0,1\}$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: equal to 1 if the drone exits from graph g by e_g, 0 otherwise.
s^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: integer non negative variables representing the order of visit of edge e of graph g.
μ_L^{eg} $\forall e \in E,\ \forall g \in \mathcal{G}$: equal to 1 if the launching point x_L^g to visit graph g is located on e.
μ_R^{eg} $\forall e \in E,\ \forall g \in \mathcal{G}$: equal to 1 if the rendezvous point to visit graph g is located on e.
$z_{LR}^{ee'g} \in \{0,1\}$ $\forall e, e' \in E,\ \forall g \in \mathcal{G}$: equal to 1 if the launching point x_L^g to visit graph g is located on e and the rendezvous point x_R^g is located on e'.
$b_{LR}^{ig} \in \{0,1\},\ \forall i : e = (i,j) \in E,\ \forall g \in \mathcal{G}$, equal to 1 if the mothership exits from x_L^g by the vertex V^i of the edge e.
$c_{LR}^{ig} \in \{0,1\},\ \forall i : e = (i,j) \in E,\ \forall g \in \mathcal{G}$, equal to 1 if the mothership enters in x_R^g by the vertex V^i of the edge e.
$q_{LR}^{eg} \geq 0,\ \forall e \in E,\ \forall g \in \mathcal{G}$, integer variable counting the number of times the mothership fully traverses edge e to move between the launching point x_L^g on e to the rendezvous point x_R^g on e' for graph g.
$z_{RL}^{ee'gg'} \in \{0,1\}$ $\forall e, e' \in E,\ \forall g, g' \in G$: equal to 1 if the rendezvous point x_R^g, for graph g, is located on e and the launching point $x_L^{g'}$, for graph g', is located on e'.
$b_{RL}^{ig} \in \{0,1\},\ \forall i : e = (i,j) \in E,\ \forall g \in \mathcal{G}$, equal to 1 if the mothership exits from x_R^g by the vertex V^i of the edge e.
$c_{RL}^{ig} \in \{0,1\},\ \forall i : e = (i,j) \in E,\ \forall g \in \mathcal{G}$, equal to 1 if the mothership enters in x_L^g by the vertex V^i of the edge e.
$q_{RL}^{egg'} \geq 0,\ \forall e \in E,\ \forall g \in \mathcal{G}$, integer variable counting the number of times the mothership fully traverses edge e to move between the rendezvous point x_R^g to the launching point $x_L^{g'}$ for graph g.
Continuous Decision Variables
$\rho^{e_g} \in [0,1]$ and $\lambda^{e_g} \in [0,1]$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: defining the entry and exit points on edge e_g.
$\gamma_R^{eg} \in [0,1]$ and $\gamma_L^{eg} \in [0,1]$ $\forall e \in E,\ \forall g \in \mathcal{G}$: defining the launching and rendezvous points, associated with graph g, located on edge e of the network $\mathcal{N}$.
$\nu_{\min}^{e_g}$ and $\nu_{\max}^{e_g} \in [0,1]$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: auxiliary variables for linearization.
x_L^g $g \in \mathcal{G}$: coordinates representing the point where the mothership launches the drone to visit graph g.
x_R^g $g \in \mathcal{G}$: coordinates representing the point where the mothership retrieves the drone to visit graph g.
R^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: coordinates representing the entry point on edge e of graph g.
L^{e_g} $\forall e_g \in E_g$ $(g \in \mathcal{G})$: coordinates representing the exit point on edge e of graph g.
$d_L^{e_g} \geq 0$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: representing the distance travelled by the drone from the launching point x_L^g on the mothership to the first visiting point R^{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 launching point L^{e_g} on e_g to the rendezvous 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 rendezvous point R^{e_g} to the launching point L^{e_g} on e_g.
$d_R^{e_g} \geq 0$ $\forall e_g \in E_g$ $(g \in \mathcal{G})$: representing the distance travelled by the drone from the last visiting point for graph g L^{e_g} to the rendezvous point x_R^g on the mothership.
$d_{LR}^{ee'g} \geq 0$ $\forall e,\ e' \in E,\ \forall g \in \mathcal{G}$: representing the distance travelled by the mothership from the launching point x_L^g on e and the rendezvous point x_R^g on e' to visit graph g.
$d_{RL}^{ee'gg'} \geq 0$ $\forall e,\ e' \in E$ $\forall g,\ g' \in \mathcal{G}$: representing the distance travelled by the mothership from the rendezvous point x_R^g, for graph g, on e to the launching point $x_L^{g'}$, for graph g', on e'.
$d_{LR}^{g} \geq 0$ $\forall g \in \mathcal{G}$: representing the total distance travelled by the mothership between the launching point

	x_L^g and the rendezvous point x_R^g to visit graph g.
$d_{RL}^{gg'} \geq 0$	$\forall g,\ g' \in \mathcal{G}$, representing the total distance travelled by the mothership between the rendezvous point x_R^g, for graph g, and the launching point $x_L^{g'}$, for graph g'.

Table 15: Decision Variables for NMDRPG-MTZ

In this case, we observe that the distance between launching and rendezvous points in two edges $e = (i,j)$, $e' = (i',j') \in E$, not necessarily distinct, of the graph can be represented as:

$$d_{LR}^{ee'g} = \begin{cases} |\gamma_L^{eg} - \gamma_R^{eg}|\mathcal{L}(e), & \text{if } e = e', \\ \left[b_{LR}^{ig}\gamma_L^{eg} + b_{LR}^{jg}(1-\gamma_L^{eg})\right]\mathcal{L}(e) + \sum\limits_{e'' \in \mathcal{N}} q_{LR}^{e''g}\mathcal{L}(e'') + \left[c_{LR}^{i'g}\gamma_R^{e'g} + c_{LR}^{j'g}(1-\gamma_R^{e'g})\right]\mathcal{L}(e'), & \text{if } e \neq e', \end{cases} \qquad (d_{LR}^{g\mathcal{N}})$$

where b_{LR}^{ig} (resp. c_{LR}^{ig}) are binary variables that determine from which of the end-points of e (respectively e') one has to account for the distance and q_{LR}^{eg} is a binary variable that is equal to one when the mothership fully traverses the edge e. Furthermore, we need to define binary variables $z_{LR}^{ee'g}$ that model the choice of the model for right definition (first or second formulas of the distance in $(d_{LR}^{g\mathcal{N}})$). All the above arguments give the necessary elements to account for the movement of the mothership on the network $\mathcal{N} = (V, E)$:

$$x_L^g = \sum_{e=(i,j)\in E} \mu_L^{eg}\left[V^i + \gamma_L^{eg}(V^j - V^i)\right], \qquad \forall g \in \mathcal{G} \qquad (61)$$

$$x_R^g = \sum_{e=(i,j)\in E} \mu_R^{eg}\left[V^i + \gamma_R^{eg}(V^j - V^i)\right], \qquad \forall g \in \mathcal{G} \qquad (62)$$

$$z_{LR}^{ee'g} = \mu_L^{eg}\mu_R^{e'g}, \qquad \forall e, e' \in E,\ \forall g \in \mathcal{G} \qquad (63)$$

$$b_{LR}^{ig} \leq \sum_{e\in\delta(i)} \mu_L^{eg}, \qquad \forall i \in V \quad \forall g \in \mathcal{G} \qquad (64)$$

$$c_{LR}^{ig} \leq \sum_{e\in\delta(i)} \mu_R^{eg}, \qquad \forall i \in V \quad \forall g \in \mathcal{G} \qquad (65)$$

$$b_{LR}^{ig} + b_{LR}^{jg} \geq \mu_L^{eg}, \qquad \forall e = (i,j) \in E \quad \forall g \in \mathcal{G} \qquad (66)$$

$$c_{LR}^{ig} + c_{LR}^{jg} \geq \mu_R^{eg}, \qquad \forall e = (i,j) \in E \quad \forall g \in \mathcal{G} \qquad (67)$$

$$b_{LR}^{ig} + \sum_{\{j:(i,j)\in E\}} q_{LR}^{jig} = \sum_{\{j:(i,j)\in E\}} q_{LR}^{ijg} + c_{LR}^{ig}, \qquad \forall i \in V \quad \forall g \in \mathcal{G} \qquad (68)$$

$$\sum_{e\in E} \mu_L^{eg} = 1, \qquad \forall g \in \mathcal{G} \qquad (69)$$

$$\sum_{e\in E} \mu_R^{eg} = 1, \qquad \forall g \in \mathcal{G} \qquad (70)$$

$$d_{LR}^{g} = \sum_{e,e'\in E} z_{LR}^{ee'g} d_{LR}^{ee'g}. \qquad \forall g \in \mathcal{G} \qquad (71)$$

Constraints (61) and (62) parameterize the launching and rendezvous points in the network $\mathcal{N}$ induced by the visit to the graph $g \in \mathcal{G}$. Constraints (63) set the binary variables $z_{LR}^{ee'g}$ by means of the binary variables μ_L^{eg} and $\mu_R^{e'g}$. Constraints (64) and (65) state that if the launching point (resp. rendezvous point) is not on the edge e, the mothership cannot go (resp. exit) to the vertices of the edge e. Constraints (66) state that if the mothership leaves the edge e, it must exit from one of the incident vertices to e. In the same way, constraints (67) state that if the mothership leaves the edge e', it necessarily must enter to e from one incident vertex of e. Flow conservation constraints (68) ensure that, in the route to be defined, the number of incoming edges to each vertex i is equal to the number of outgoing edges. Constraints (69) and (70) state that, for the visit to the graph $g \in \mathcal{G}$, launching and rendezvous points can be only on one edge. Constraints (71) returns the total distance traveled by the drone on the graph $g \in \mathcal{G}$.

Similarly, the distance covered by the mothership along the path on the network $\mathcal{N}$, from the rendezvous point $x_R^g \in e$, after the visit to $g \in \mathcal{G}$ to the next launching point $x_L^{g'} \in e'$ (to go to the graph $g' \in \mathcal{G}$), can be modeled using the following definition of distance:

$$d_{RL}^{ee'gg'} = \begin{cases} |\gamma_R^{eg} - \gamma_L^{eg'}|\mathcal{L}(e), & \text{if } e = e', \\ \left[b_{RL}^{ig}\gamma_R^{eg} + b_{RL}^{jg}(1-\gamma_R^{eg})\right]\mathcal{L}(e) + \sum_{e''\in\mathcal{N}} q_{RL}^{e''gg'}\mathcal{L}(e'') + \left[c_{RL}^{i'g'}\gamma_L^{e'g'} + c_{RL}^{j'g'}(1-\gamma_L^{e'g'})\right]\mathcal{L}(e'), & \text{if } e \neq e'. \end{cases} \quad (d_{RL}^{g\mathcal{N}})$$

In this case, the binary variable $z_{RL}^{ee'gg'}$ links the rendezvous point at g with the launching point at g'. Then, we can use a set of constraints similar to those used above and the distance from x_R^g to $x_L^{g'}$ can be computed by means of the following additional constraints:

$$z_{RL}^{ee'gg'} = \mu_R^{eg}\mu_L^{e'g'}, \qquad \forall e, e' \in E, \quad \forall g, g' \in \mathcal{G} \qquad (72)$$

$$b_{RL}^{ig} \leq \sum_{e\in\delta(i)} \mu_R^{eg}, \qquad \forall i \in V \ \forall g \in \mathcal{G} \qquad (73)$$

$$c_{RL}^{ig} \leq \sum_{e\in\delta(i)} \mu_L^{eg}, \qquad \forall i \in V \ \forall g \in \mathcal{G} \qquad (74)$$

$$b_{RL}^{ig} + b_{RL}^{jg} \geq \mu_R^{eg}, \qquad \forall e = (i,j) \in E \ \forall g \in \mathcal{G} \qquad (75)$$

$$c_{RL}^{ig} + c_{RL}^{jg} \geq \mu_L^{eg}, \qquad \forall e = (i,j) \in E \ \forall g \in \mathcal{G} \qquad (76)$$

$$b_{RL}^{ig} + \sum_{\{j:(i,j)\in E} q_{RL}^{jigg'} = \sum_{\{j:(i,j)\in E\}} q_{RL}^{ijgg'} + c_{RL}^{ig'}, \qquad \forall i \in V \ \forall g \in \mathcal{G} \qquad (77)$$

$$d_{RL}^{gg'} = \sum_{e,e'} z_{RL}^{ee'gg'} d_{RL}^{ee'gg'}. \qquad \forall g, g' \in \mathcal{G} \qquad (78)$$

Hence, once the distances inside the graph are set with the above two families of constraints, we can state the mathematical programming formulation of the problem as:

$$\min \quad \sum_{g\in E_g} (u^{e_g} d_L^{e_g} + v^{e_g} d_R^{e_g}) + \sum_{e_g\in E_g} \mu^{e_g} d^{e_g} + \sum_{e_g, e'_g \in E_g} z^{e_g e'_g} d^{e_g e'_g} + \sum_{g\in E_g} d_{LR}^{g} + \sum_{g,g'\in E_g} d_{RL}^{gg'} w^{gg'} \quad \text{(NDRPG-MTZ)}$$

s.t. (1) – (7),
(61) – (78),
(MTZ_1) – (MTZ_2) or (SEC),
(α-E) or (α-G),
(DCW-g),
($d_{LR}^{g\mathcal{N}}$), ($d_{RL}^{g\mathcal{N}}$),
(DIST_1-g) – (DIST_6-g),
(ORIG_1) – (DEST_2)