



Deployment Optimization of a Fleet of Drones for Routine Inspection of Networks of Linear Infrastructures

Etienne Petitprez, Frédéric Georges, Nicolas Raballand, Sylvain Bertrand

► To cite this version:

Etienne Petitprez, Frédéric Georges, Nicolas Raballand, Sylvain Bertrand. Deployment Optimization of a Fleet of Drones for Routine Inspection of Networks of Linear Infrastructures. ICUAS 2021, Jun 2021, Athènes, Greece. 10.1109/ICUAS51884.2021.9476674 . hal-03306569

HAL Id: hal-03306569

<https://hal.science/hal-03306569>

Submitted on 29 Jul 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Deployment Optimization of a Fleet of Drones for Routine Inspection of Networks of Linear Infrastructures

E. Petitprez¹, F. Georges¹, N. Raballand¹, S. Bertrand¹

Abstract—In this paper, the problem of optimizing the deployment of a fleet of Unmanned Aerial Vehicles (UAVs) for the inspection of networks of linear infrastructures is addressed. In this optimization problem, two levels need to be considered: the UAVs need to follow a trajectory allowing them to fly over linear infrastructures, with a limited endurance, while ground vehicles need to be deployed to act as platforms that can launch and retrieve the UAVs from fixed parking positions. Both parking positions of ground vehicles and the UAVs fleet routes along the linear infrastructures have to be optimized to improve the inspection efficiency, which is related to multiple contradictory objectives such as minimizing operational cost and maximizing inspection performance. The algorithm proposed in this paper was developed by expanding upon several algorithms from literature. More precisely, we propose to solve a Two-Echelon Vehicle Routing Problem (2E-VRP) with a custom multi-objective Hybrid Genetic Algorithm (HGA) based on a Capacitated Arc Routing Problem (CARP). Simulation results are provided on a benchmark of networks of linear infrastructures along with a sensitivity analysis on the parameters of the algorithm, showing promising results.

I. INTRODUCTION

In recent years, a growing interest has been directed to the use of Unmanned Aerial Vehicles (UAVs) to perform tasks that may be hazardous and costly when performed by manned workforce. One such case, which is the subject of this paper, is the inspection of networks of linear infrastructures spanning several hundreds of kilometers, like rail or road networks [1]–[3] or electric power lines [4], [5], where the use of UAVs could greatly decrease costs and provide complementary data for maintenance.

In order to deploy UAVs over large networks, from an operational perspective, one can consider a multi-level distribution system in which ground vehicles start from a central depot and carry a number of UAVs to be launched and retrieved from "parking spots" along the network. The use of ground vehicles as a first step in the UAVs' deployment is necessary because of their limited flight endurance and their higher cost. A natural issue arising in the operation of such a multi-vehicle system is therefore how to efficiently route vehicles operating at both levels (i.e. select "parking" spots for the ground vehicles and flight routes for the UAVs), while minimizing costs related to the system and maximizing the mission performance. This mission performance can be defined using multiple (and possibly conflicting) criteria which can vary depending on the inspection mission requirements

and on the needs and constraints of the asset manager of the network of infrastructures. This kind of problem is known as a Multi-Objective Two-Echelon Vehicle Routing Problem (MO-2E-VRP).

In the literature, the most common problems, closest to the MO-2E-VRP, are express delivery services in e-commerce and home delivery [6]. Coordination activities arising in city logistics can be performed with a multi-level system with several optimization criteria. Solutions to these problems use specific search methods like Adaptative Large Neighborhood Search (ALNS) [6], Clustering and Nearest Point Search (CNPS) [7] or Genetic Algorithm (GA) [8] which are excellent for routing problems. However, the first two algorithms are efficient only for single objective optimization (i.e. minimizing the traveled distance) while under constraints, and all of them have trouble converging when the number of customers increases. Besides, linear infrastructures inspection cannot be regarded as touring a collection of discrete positions. Instead, they should be considered as arcs or segments.

Consequently, a Capacitated Arc Routing Problem (CARP) [9] would be a better fit for inspection of linear infrastructure networks. The underlying optimization problem is slightly different from those presented previously since the goal in a CARP is to determine the least-cost route of a given subset of required arcs in a capacitated graph while under constraints. Unfortunately, existing algorithms that can be found in the literature are only aimed at single-objective optimization problems as well (i.e. minimizing the total travel cost), which does not reflect properly the various operational requirements that are expressed by companies managing networks of linear infrastructures.

Nonetheless, recent researches on hybridization tend to combine several algorithms to address more complex problems. Hybrid Genetic Algorithms (HGA) have been conceived to solve CARPs [10], [11]. As implied by its name, a HGA is a GA coupled with a local search procedure [11]. It is still based on the theory of evolution like its GA counterpart and works in a very similar way, relying on an evolutionary mechanism to select individuals in a population of possible solutions [12]. Each generation is obtained by breeding two individuals together and introducing mutations in the children that are created. From there, the best individuals according to a fitness function are preserved while the worst ones are rejected, until further improvement of the fitness is no longer possible. The best individual, corresponding to the best fitness, is then considered to be the optimal solution to the problem (see Section III).

¹All authors are with Université Paris-Saclay, ONERA, Traitement de l'Information et Systèmes, 91123, Palaiseau, France. etienne.petitprez@gmail.com, {frederic.georges, nicolas.raballand, sylvain.bertrand}@onera.fr

In [13], a HGA was elaborated to solve a Two-Echelon Vehicle Routing Problem (2E-VRP) for delivery without dissociating the two levels to reduce computing time and the complexity of the algorithm. A suitable interaction between the levels leads to an improved algorithm which fits perfectly for specific problems [4]. Still, these algorithms only consider a single UAV or a launch from a single ground vehicle in a restricted area, whereas multiple UAVs taking-off from the same vehicle might allow for better solutions [13] [14].

In this paper, a hybridization of existing methods with a CARP algorithm is proposed, similarly to the work in [11] [13] but allowing multiple take-offs of UAVs from ground vehicles. Mutation operators from the HGA have also been adapted to tackle the Multi-Objective (MO) optimization issue.

The remainder of this paper is organized as follows. The problem and its modelization is first described in Section II and the optimization method is then presented in detail in Section III. Finally, Section IV highlights the performance of the developed algorithm based on simulation results and sensitivity analysis and Section V summarizes this work contribution.

II. PROBLEM DESCRIPTION AND MODELIZATION

In this work, the routing problem for a cooperation between ground vehicles and UAVs to inspect networks of linear infrastructures is investigated. The network is modelled as a capacitated graph as shown in Fig. 1. The linear sections of the infrastructures that are to be inspected correspond to the arcs of the graph, associated to a weight taking into account their length as well as possible other factors (eg. wind conditions) influencing the energy that UAVs will need to fly over them. The vertices of the graph correspond to segmentation of the infrastructure into arbitrary sections so that the weights of the arcs do not become too large and would not result in non-feasible sections. Furthermore, additional arcs and vertices are added to the graph to allow alternative routes and connections within the initial infrastructure network. These non-required arcs represent flyable air corridors for the UAVs but do not correspond to any infrastructure requiring inspection. The graph is also built so as to reflect real geographical areas above which UAVs are forbidden to fly by the regulations, as shown in Fig. 1: forbidden flyover areas correspond to empty spaces in the graph and constrain its design.

Due to limited endurance, UAVs can only visit arcs of the graph within a maximal range. Therefore, ground carrier vehicles are used to transport, launch and retrieve UAVs on the field in order to extend their operational reach. As a consequence, a subset of the vertices of the graph must correspond to suitable parking positions for ground vehicles, fitting operational requirement. This subset of vertices corresponds to potential starting and stopping points for the UAVs routes. In this work, it is assumed that a ground vehicle can transport several UAVs at once, and that the maximum number of UAVs that can be carried at once is large enough to be of no consequence.

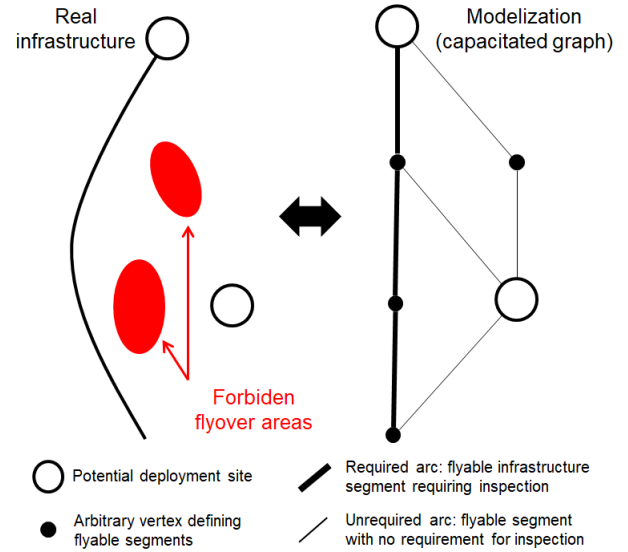


Fig. 1. Modelization of a linear infrastructure as a capacitated graph. The linear infrastructure is represented by the thick arcs of the graph, coupled with a weight reflecting their length. Additional thin arcs allow alternative routes for the UAVs and correspond to flyable corridors. The vertices of the graph are placed arbitrarily to keep arcs reasonably short, and a subset of these vertices correspond to real geographical areas suitable as parking positions for ground vehicles. Finally, forbidden flyover areas constrain the design of the graph.

The developed algorithm will allow to define the number of UAVs, their routes and their take-off and landing sites corresponding to parking spots of ground vehicles.

In this paper, the following notations and assumptions will be used:

- The graph is defined as an undirected graph $G = (V, A)$ where V defines the set of vertices and A the set of arcs. The subset A_r of required arcs is included in A ($A_r \subseteq A$).
- An UAV route is defined by a sequence of arcs, starting and finishing at the same vertex (parking spot for ground vehicle). It is assumed that ground vehicles are not allowed to move to another parking spot while UAVs are deployed, and that UAVs are allocated to their specific ground carrier and cannot be retrieved by another one.
- The possible parking spots $P = \{p_1, p_2, \dots, p_n\}$ are assumed to be known and P is a subset of the graph vertices ($P \subseteq V$).
- The cost c_{ij} of travelling on an arc between V_i and V_j is in adequacy with UAVs speeds and battery consumption during its flight. Arcs in A_r tend to be more "expensive" to travel through since UAVs might be required to fly slowly for inspection tasks and so, spend more time and consume more battery power.
- UAVs are constrained by a known endurance ϵ depending on battery capacities and flight speed.

In this paper, the CARP consists in finding UAV routes by assigning arcs of A_r to an adjustable number of UAVs also to be determined, deployed from parking spots. The solution must be feasible, in the sense that limited endurance of

the vehicles must be accounted for in the route definition. Optimization of a performance index is considered in the search of a solution. The following five criteria will be considered in the definition of a performance index for the inspection missions addressed in this paper:

- The number N_{park} of parking spots (and hence of ground vehicles to be deployed): criterion to be minimized.
- The number N_{UAV} of UAVs launched: criterion to be minimized.
- The ratio of the total distance traveled by UAVs with respect to the cumulative length of all the arcs of the graph, R_{dist} : criterion to be minimized.
- The coverage of infrastructures, defined as the ratio R_{cov}^n of the number of required arcs visited by UAVs with respect to the total number of required arcs: criterion to be maximized.
- The "useful" use rate R_{use}^n of the UAVs, defined for each UAV as the ratio of the number of visited required arcs with respect to the number of visited arcs (required or not): criterion to be maximized. It is assumed that, as a side effect of this last criterion, the algorithm might favor solutions where the UAVs will either use almost all of their endurance, or on the contrary nearly none, in an attempt to minimize the relative number of unrequired arcs being flown over.

These five criteria of interest should all be accounted for in the definition of the CARP and its solution(s). The problem of interest, in its direct and simplest formulation, is therefore a multi-objective optimization problem. A wide range of techniques for adapting genetic algorithms to multi-objective cases can be found in the literature. The difficulty lies in how solutions can be evaluated by a cost (or *fitness*) function and compared to each other to select the better ones, while objectives may compete with one another. In the overview of strategies described in [15], the definition of a scalar single fitness function as a weighted sum of several criteria appears as a highly effective and flexible strategy that is quite easy to enforce. This fitness function can be defined by linearly combining the different objectives:

$$f(s) = \lambda_{N_{park}}(N_{park} - 1) + \lambda_{N_{UAV}}(N_{UAV} - 1) + \lambda_{R_{dist}} R_{dist} + \lambda_{R_{cov}^n} \frac{1}{R_{cov}^n} + \lambda_{R_{use}^n} \sum_{k=1}^{N_{UAV}} \frac{1}{R_{use}^n(k)} \quad (1)$$

where s is the solution for which the cost has to be evaluated (or individual of the population tested by the GA), and where the weights λ_i , $i \in \{N_{park}, N_{UAV}, R_{dist}, R_{cov}^n, R_{use}^n\}$ satisfy

$$\lambda_i \in [0, 1], \text{ and } \sum_i \lambda_i = 1. \quad (2)$$

The weights λ_i of this fitness function can be adjusted according to the requirements of the companies to properly reflect the relative importance of the five criteria corresponding to the various goals they seek to achieve. For instance,

one might want to prioritize infrastructure coverage while another user might focus on operational costs minimization (i.e. minimization of the number of vehicles).

III. OPTIMIZATION ALGORITHM

The HGA proposed in this work requires as inputs:

- a description of the graph G representing the network of linear infrastructures requiring inspection (vertices, arcs, weights, possible parking spots) ;
- an initial value for the number N_{UAV} of available UAVs and their maximum endurance ϵ (considered to be identical for all the UAVs);
- the weights λ_i of the fitness function.

The optimization will then provide as a solution the number of required UAVs and their associated routes (including the deployment vertices) allowing to minimize the fitness function (1). In particular, the initial number of UAVs given as inputs is not binding and can be adjusted by the algorithm.

The algorithm depends on a set of parameters which are listed in Table I. These parameters influence greatly the optimization process and the overall performance of the algorithm. For instance, it is necessary to prevent the population of solutions from becoming too homogeneous (genetic drift) [16] in order to properly explore the field of feasible solutions and ensure that the algorithm does not get locked in a local optimum. If these parameters are not set correctly, the algorithm may suffer from a slow convergence, or stop at a sub-optimal solution like in the case of a genetic drift. In order to set these parameters, a sensitivity analysis has been conducted and its results are presented in Section IV.

$S_{pop_{init}}$	Size of the initial population
S_{pop}	Size of the population kept at each generation
N_{cross}	Number of crossovers performed between two individuals
N_{mut}	Number of mutations performed per crossover
N_{sol}	Number of solutions tested by the stopping criteria
N_{stop}	Number of consecutive population generations complying with the stopping criteria
P_{path}	Probability in the population initialization for a UAV to choose a random path to reach the next required arc

TABLE I

LIST OF TUNING PARAMETERS OF THE PROPOSED ALGORITHM

A. Algorithm overview

As reminded in Section I, a genetic algorithm is based on the evolution of a population of individuals which are possible solutions to the optimization problem. In this work, an individual corresponds to a set of UAV routes and their corresponding parking spots. The developed algorithm can be decomposed in the following steps:

- the *initialization* of a starting population,
- the population *evolution*, performed iteratively through:
 - a *crossover* phase,

- a *mutation* phase, performed on every individuals created by the crossover phase,
- the verification of the *stopping criteria* after each iteration of the population evolution.

These steps are detailed in the next subsections.

Population initialization

First, the Dijkstra algorithm is used to determine the shortest distance (i.e. travel cost) between each possible pair of vertices. Then, a parking spot is associated to each arc from A_r using the *parking_spot_selection* procedure from [13], with an additional reachability constraint: if an arc cannot be reached from any parking within a round trip, it is removed from A_r and will no longer be taken into account in the optimization process.

Then, individuals are created to initialize the population. To generate an UAV route belonging to an individual, first, an arc of A_r is selected, and the UAV takes-off from the associated parking spot. The route followed by the UAV to reach and cross the selected arc has a probability P_{path} of being chosen randomly, otherwise, it is the shortest one, as identified by the Dijkstra algorithm. This procedure allows for alternative routes to be generated, to foster diversity. Then, a new arc is selected from A_r (the same arc cannot be selected twice) and the UAV aims to reach and cross it from its current position (i.e. the extremity of the previous arc), selecting a random or the shortest route according to the probability P_{path} . The UAV route is thus extended. This procedure is repeated as long as the endurance ϵ of the UAV allows for a round trip, otherwise the route is completed in order to allow the UAV to fly back to its associated parking spot, and a new route is generated and added to the individual.

At each step of the route generation procedure, an additional parameter P_{share} , computed as the ratio of the number of elements in A_r with respect to the number of available UAVs given as input, is used as a probability to stop the current UAV's route from being further extended. The goal of this parameter is to foster sharing of elements of A_r between initially available UAVs.

Once every arc from A_r has been allocated (or deleted if not reachable), the total cost of the solution is calculated by evaluating the fitness function (1), and the individual is added to the initial population.

When the initial population reaches its maximum size $S_{pop_{init}}$ (see Table I), the algorithm starts generating new children populations, which are further limited to the size $S_{pop} \leq S_{pop_{initial}}$.

Crossover

Once the population has been initialized, in a similar fashion to usual genetic algorithms, the HGA proceeds with a crossover phase by selecting iteratively two solutions as parents to produce children solutions. The children solutions are created by swapping segments (arcs) of the routes of the UAVs between the two parents. The crossover method proposed in this work is based on [11], but N_{cross} crossovers

are performed between the selected parents (see Table I). If the child's cost is lower than the population median cost, it is added to the population pool. Otherwise, the algorithm selects two other parents for a new generation attempt.

In this paper, two kinds of parents selection have been tested and compared to each other: a random and an elitist one. The random research selects both parents for the crossover method randomly in the population pool whereas the elitist one tends to find all best local solutions before searching for others. The elitist algorithm chooses the parents starting with the best solutions: it crosses the first with itself then the first with the second one and so on, until a solution with a cost smaller than the current one is found. Once that is the case, the algorithm restarts from the new best solution. Results of the comparison of both methods will be discussed in Section IV.

Mutation

For each children created by the crossover process, mutations are performed. Three operators [17] have been implemented: *single_insertion* which adds an arc to a UAV route, *single_deletion* which deletes an arc from a UAV route and *swap* which exchanges two arcs of UAV routes of the child. Each operator is executed N_{mut} times on UAV routes chosen randomly (see Table I). If, as a result of the mutation, the child's cost decreases (improvement of the solution), then the mutated solution is appended to the population pool.

After the crossover and mutation steps, the S_{pop} individuals with the smallest cost are kept in the population, while the other ones are eliminated.

Stopping criteria

Stopping criteria must be used to detect whether or not the search of a better solution needs to continue. As highlighted in [16], this is of a particular importance when:

- the best solution obtained so far already satisfies the expressed requirements;
- a better solution is unlikely to be found;
- the algorithm is unlikely to converge in a computation time deemed reasonable.

The stopping criteria need to detect "success" and "failure" cases thanks to a local progress indicator and population convergence, while population diversity is maintained to ensure global convergence and avoid genetic drift. Thereby, in this paper, three criteria are used to monitor the algorithm convergence at each generation.

The first one is a measurement of the mean value of the spread of the non-dominated individuals, computed as an average across iterations, with a stopping criterion from [18]. As defined by Vilfredo Pareto, non-dominated individuals are feasible balanced solutions such that one cannot improve a criterion without deteriorating another one. This criterion tracks the increase of the number of non-dominated individuals.

The second criterion is an indicator that measures how stable solutions are in an archive that contains the best solutions obtained with the consolidation ratio from [19].

The third and last criterion used in this work is the Offline Convergence Detection from [20], which applies a Kolmogorov-Smirnov statistical test. It is robust but heavy to parameterize and very resource-demanding. However, once the test is validated by the obtained set of solutions, its result does not change subsequently. As a consequence, this criterion is dropped after its first validation to save computation time. On the other hand, the two other criteria still need to be validated over subsequent generations.

Although the occurrence probability is small, it is possible for these three convergence criteria to be simultaneously validated even though the algorithm has not fully converged yet. Thus, the criteria have to be validated N_{stop} consecutive times (see Table I) to ensure global convergence.

In addition, a maximum run time has also been set to stop the algorithm if it is taking too much time to converge. This is a fail safe to prevent the algorithm from running for an unreasonable amount of time. Activating this maximum run time stopping criterion is considered a failure since the algorithm did not converge.

B. Genetic drift

Genetic algorithms are known to be very sensitive to the choice of the initial population as well as the selection of individuals during the crossover step, leading to the genetic drift issue [16]. A solution implemented in this work consists in adding a constraint on the individuals of the population: they must all be different from each others. The population pool cannot be filled by duplicates.

IV. SIMULATION RESULTS

In order to tune the parameters of the algorithm and assess its performance, several simulations were conducted on a graph shown in Figure 4. It is composed of 12 vertices, 4 of which are possible parking spots (vertices 1, 8, 10 and 12), and 22 arcs, 11 of which correspond to sections of infrastructures requiring inspection. The maximum endurance ϵ of the UAVs was set at 110 (arbitrary unit). The values chosen for the weights λ_i used in the fitness function (1) are given in Table II.

$\lambda_{N_{park}}$	$\lambda_{N_{UAV}}$	$\lambda_{R_{dist}}$	$\lambda_{R_{cov}}^n$	$\lambda_{R_{use}}^n$
0.25	0.15	0.3	0.23	0.07

TABLE II

CHOSEN VALUES FOR THE WEIGHTS λ_i OF THE FITNESS FUNCTION (1)

A. Random and elitist selection procedures for crossover

As described in Section III-A two strategies have been considered to select individuals in the genetic population to perform crossovers: a random and an elitist method. Both methods were tested to compare their performance. Figure 2 shows the evolution of the fitness (cost) of the

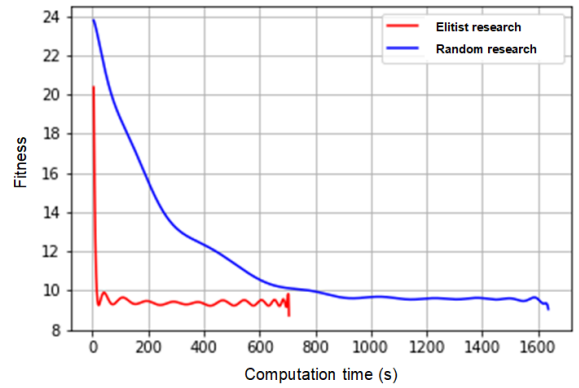


Fig. 2. Comparison of the evolution of the fitness of the best solution against time for both elitist and random research methods.

best solution against computation time. A similar fitness is reached with both methods, but the convergence speed is about 8.1 times faster with the elitist strategy compared to the more traditional random search method.

Moreover, the elitist method is more robust with respect to the population size S_{pop} . When the population kept at each generation becomes too large (over 6000 individuals), the genetic selection method struggles to converge towards an adequate solution, whereas the elitist method still remains effective. The elitist research method was thus favored over the random one.

B. Sensitivity analysis and parameter tuning

A study of the sensitivity of the developed HGA on its tuning parameters (see Table I) was conducted in order to optimize its performance. The elitist research method has been used and simulations were run for different parameter values, using the graph shown in Figure 4. The algorithm maximum run time was set to 3000 seconds. Simulations were performed several times with the same sets of parameters to account for statistical uncertainty.

Results show that an increase of the parameter $S_{pop_{init}}$ results in a decrease of the time required by the algorithm to converge. However, the initialization of this population requires increased computational resources. When $S_{pop_{init}}$ becomes greater than 20000 individuals, the overall computation time actually increases because of the initialization of the population. Thus, the choice of $S_{pop_{init}}$ must result from a trade-off between initialization time and convergence speed.

On the other hand, the parameters N_{UAVs} and P_{path} appear to have no effect on the convergence speed, nor the solution fitness. However, this first approximation needs to be confirmed particularly for larger graphs.

Regarding the number of operations in the algorithm, the crossover needs to be performed at least 3 times per generation to achieve an effective result, and the computation time is minimal for $N_{cross} = 35$ crossovers. A similar result was observed for N_{mut} , with the minimum computation time reached for 70 mutations. However, unlike N_{cross} , an

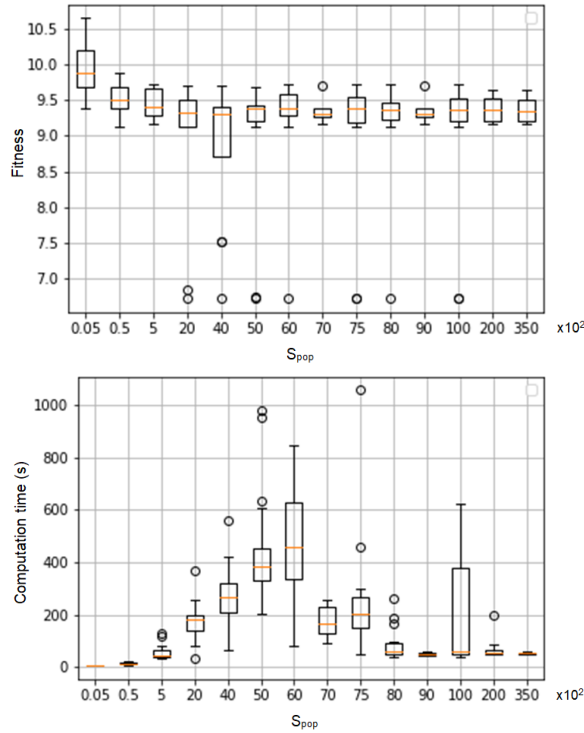


Fig. 3. Influence of the parameter S_{pop} on the optimization process: evolution of the fitness of the best solution (top) and evolution of the computation duration (bottom) with respect to S_{pop} . Values and statistical uncertainties are displayed by box-plots with the median (orange), the box representing the inter-quartile range (IQR), straight lines for extremum values (under 1.5 times the IQR) and circles for outliers.

upper bound was observed for N_{mut} , as the computation time would increase for values of N_{mut} greater than 1000.

Regarding the stopping criteria, as the parameter N_{sol} increases, the fitness of the final solution improves slightly, but at the cost of a dramatic growth of the computation time when N_{sol} becomes greater than 30. This may be explained by the fact that as N_{sol} grows, more individuals need to be stable for the stopping criteria to consider that the algorithm has converged.

The parameter N_{stop} is linearly proportional to the computation time. For values below 10, the obtained solution is quite sub-optimal and the standard deviation increases, which points to a failure to properly detect the convergence of the algorithm.

Finally, as shown in Figure 3 (top), S_{pop} should be at least larger than 1000, otherwise the algorithm converges to a sub-optimal solution. The bottom plot shows that the computation time reaches a maximum for S_{pop} around 6000 individuals. This is due to the fact that N_{sol} is set to be proportional to the population size ($\frac{S_{pop}}{5}$), and as its value increases, so does computation time. One can notice however that for values of S_{pop} greater than 6000 individuals, the computation time decreases. It is hypothesised that the huge increase of the number of individuals N_{sol} tested by the stopping criteria hides small individual variations that would have been picked up otherwise.

These results are preliminary: statistical uncertainties are

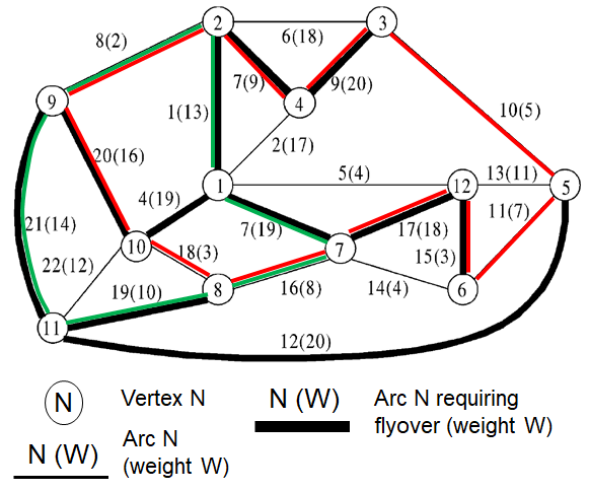


Fig. 4. Capacitated graph used as a test case for the developed algorithm. The red and green outlines correspond to the paths of the two UAVs deployed from vertex 8.

still quite significant, and a bias might have been introduced by the use of a single test-case graph. As such, one should be cautious with the interpretation of these results, and further studies will be necessary. In particular, one can notice the presence of outlier points in Figure 3 (top), which may hint that the optimal solution might actually not be reached.

C. Test cases

Following the previous sensitivity analysis, a test was then conducted with the set of parameters provided in Table III.

The obtained solution uses 2 UAVs, both deployed from vertex 8 (see Fig. 4), thus requiring only one ground vehicle. The UAV assigned to the green route travels a distance of 66, of which 56 are required arcs, and uses 60% of its endurance. The UAV assigned to the red route travels a distance of 91, of which 66 are required arcs, and uses 83% of its endurance. 81% of the required arcs are thus inspected, which corresponds to a 76% coverage of the infrastructure network in terms of distance. The arcs 4 and 12 are not inspected by the UAVs, as the algorithm deemed that the additional cost would outweigh the benefits.

Other tests have also been conducted on even larger graphs, up to 140 vertices and 190 arcs. An illustration of a "middle size" case is shown in Figure 5. It is composed of 77 vertices, 10 of them with possible parking spots (double circles) and 98 arcs, 51 of them corresponding to sections of the infrastructure to be inspected (bold arcs). Weights on the arcs are given in arbitrary unit and do not reflect the visible length of the arcs shown in the figure. The

$S_{pop_{init}}$	S_{pop}	N_{cross}	N_{mut}	N_{sol}	N_{stop}	P_{path}
20000	3000	35	600	600	30	$\frac{1}{6}$

TABLE III
TUNING PARAMETERS OF THE HGA USED FOR THE TEST CASE.

maximum endurance of each UAV has been set as $\epsilon = 500$ (arbitrary unit). A solution with four UAVs has been obtained by running the proposed algorithm. In Figure 5, arcs in magenta correspond to the ones included in the obtained routes for the four UAVs, leading to a coverage of 72% of the infrastructure. The use rates of the four UAVs are 96.6%, 94.6%, 93.8% and 68.6%, which seems to confirm the aimed behavior of the algorithm which is to favor the use of UAVs at the maximum of their capacity, possibly leading to one UAV being used to a much lesser extent. Three UAVs are launched from parking spot 10 and one UAV from parking spot 38, thus requiring only two ground vehicles for their deployment.

As an example, a route obtained for one of the four UAVs is displayed in Figure 6. Take-off and landing are performed from parking spot on vertex 10. Only a few arcs are visited twice in the route. They correspond to arcs to be inspected or used to reach the parking spot.

Even on the larger graphs, the algorithm only required a few (tens of) minutes to run on a usual desktop computer. Thus, it is expected that applications to large scale infrastructure networks can be considered.

V. CONCLUSION

The problem of optimizing the deployment of a fleet of UAVs has been considered in this paper for routine inspection of networks of linear infrastructures. Due to operational concerns related to the UAVs' transportation and deployment from ground vehicles, this problem can be formulated as a Two-Echelon Vehicle Routing Problem (2E-VRPs), involving several criteria to be optimized, such as minimization of the number of vehicles and maximization of the infrastructure coverage.

Within this context, this work has proposed a Multi-Objective Hybrid Genetic Algorithm (MO-HGA) based on a Capacitated Arc Routing Problem (CARP) formulation. The resulting optimization algorithm is able to provide an optimal solution to 2E-VRPs of large sizes. Simulations and sensitivity analysis have been performed to evaluate the proposed algorithm and tune its parameters, so as to reduce its computation time while preserving the quality of the obtained solution. Illustration results have been proposed on a benchmark graph example. Application of the optimization algorithm on capacitated graphs composed of up to 140 vertices and 190 arcs has also been successfully tested, leading to very promising results.

Future work will focus on improvement of the optimization algorithm, including selection process and stopping criteria or reduction of statistical uncertainties in the sensitivity analysis and tuning of its parameters. Besides, the analysis of the stability of these parameters over different graphs will be of great importance. Applications to very large scale networks, such as railways or power line networks, will also be considered. In particular, a scalability analysis would allow to pinpoint the limits of the developed algorithm.

Further study will also explore the possibility for deployed UAV to take-off and land at different vertices (parking spots)

of the graph. This would correspond to a greater cooperation between UAVs and land-based vehicles: ground vehicles could move to different parking spots to retrieve the UAVs at a more convenient location, or UAVs could no longer be bound to a single ground vehicle. This could effectively improve the coverage or reduce the required number of UAVs by eliminating the need for a roundabout trip.

REFERENCES

- [1] S. Bertrand, N. Raballand, F. Viguier, and F. Muller. Ground risk assessment for long-range inspection missions of railways by UAVs. In *International Conference on Unmanned Aircraft Systems*, 2017.
- [2] S. Bertrand, N. Raballand, S. Lala, and F. Viguier. Feasibility analysis of UAV operations for monitoring of infrastructure networks: a risk-based approach. In *International Conference on Unmanned Aircraft Systems*, 2019.
- [3] S. Bertrand, N. Raballand, and F. Viguier. Evaluating ground risks for road networks induced by UAV operations. In *International Conference on Unmanned Aircraft Systems*, 2018.
- [4] Y. Liu, J. Shi, Z. Liu, J. Huang, and T. Zhou. Two-layer routing for high-voltage powerline inspection by cooperated ground vehicle and drone. *Energies*, 12(7), 2019.
- [5] A. la Cour-Harbo. Quantifying risk of ground impact fatalities of power line inspection BVLOS flight with small unmanned aircraft. In *International Conference on Unmanned Aircraft Systems*, 2017.
- [6] V.C. Hemmelmayr, J-F. Cordeau, and T.G. Crainic. An adaptative large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Computers and Operations Research*, 39:3215–3228, 2012.
- [7] Y. Liu, Z. Liu, J. Shi, G. Wu, and C. Chen. Optimization of base location and patrol routes for unmanned aerial vehicles in border intelligence, surveillance, and reconnaissance. *Journal of Advanced Transportation*, 2019.
- [8] R. Gupta, B. Singh, and D. Pandey. Multi-objective fuzzy vehicle routing problem: A case study. *Math. Sciences*, 5(29):1439–1454, 2010.
- [9] Bruce L. Golden and Richard T. Wong. Capacitated arc routing problems. *Networks*, 11:305–315, 1981.
- [10] P. Lacomme, C. Prins, and W. Ramdane-Cherif. Competitive algorithm for arc routing problems. *Annals of Operations Research*, 131:159–185, 2004.
- [11] R.K. Arakaki and F.L. Usberti. Hybrid genetic algorithm for the open capacitated arc routing problem. *Computers and Operations Research*, 90:221–231, 2018.
- [12] J.H. Holland. Adaptation in natural and artificial systems: An introductory analysis with applications to biology. *MIT Press*, 11, 1992.
- [13] K. Peng, W. Liu, Q. Sun, X. Ma, M. Hu, D. Wang, and J. Liu. Wide-area vehicle-drone cooperative sensing: Opportunities and approaches. *IEEE Access*, 7:1818–1828, 2019.
- [14] M. Hu, W. Liu, J. Lu, R. Fu, K. Peng, X. Ma, and J. Liu. On the joint design of routing and scheduling for vehicle-assisted multi-UAV inspection. *Future Generation Computer Systems*, 94:214–223, 2019.
- [15] M-H Mabeed, M. Rahoual, E-G. Talbi, and C. Dhaenens. Algorithmes génétiques multicritères pour les problèmes de flowshop. In *Conception, Analyse et Gestion des systèmes industriels*, 2001.
- [16] N. Sanchez-Pi and L. Marti. Applying stopping criteria in evolutionary multi-objective optimization. In *Group of Research on Intelligence and Optimization*, 2017.
- [17] M. Gen, R. Cheng, and L. Lin. *Network Models and Optimization: Multiobjective Genetic Algorithm Approach*. Springer, 2008.
- [18] O. Rudenko and M. Schoenauer. A steady performance stopping criterion for Pareto-based evolutionary algorithms. In *The 6th International Multi-Objective Programming and Goal Programming Conference*, 1992.
- [19] T. Goel and N. Stander. A non-dominance-based online stopping criterion for multi-objective evolutionary algorithms. *International Journal for Numerical Methods in Engineering*, 84(6):661–684, 2010.
- [20] H. Trautmann, U. Ligges, J. Mehnen, and M. Preuss. A convergence criterion for multiobjective evolutionary algorithms based on systematic statistical testing. *Parallel Problem Solving from Nature*, pages 825–836, 2008.

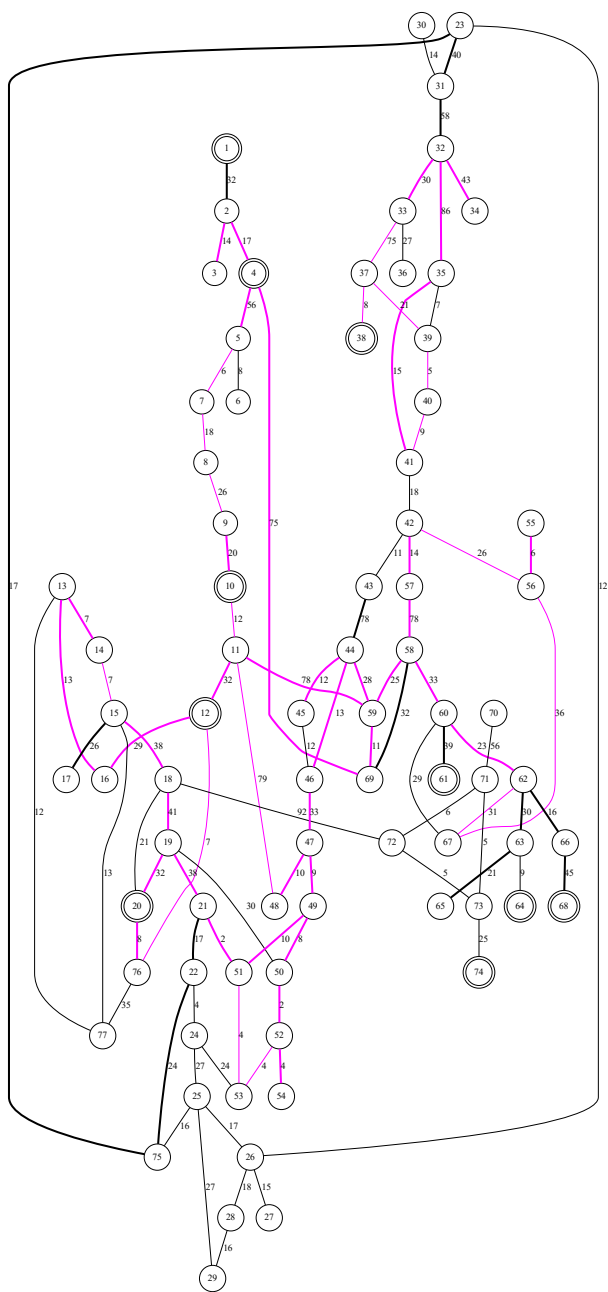


Fig. 5. Infrastructure network benchmark case and coverage with four drones obtained by the algorithm. Required arcs for inspection are in bold, inspected arcs by UAVs are in magenta, vertices with a double circle are possible parking spots, weights on arcs are given in arbitrary unit.

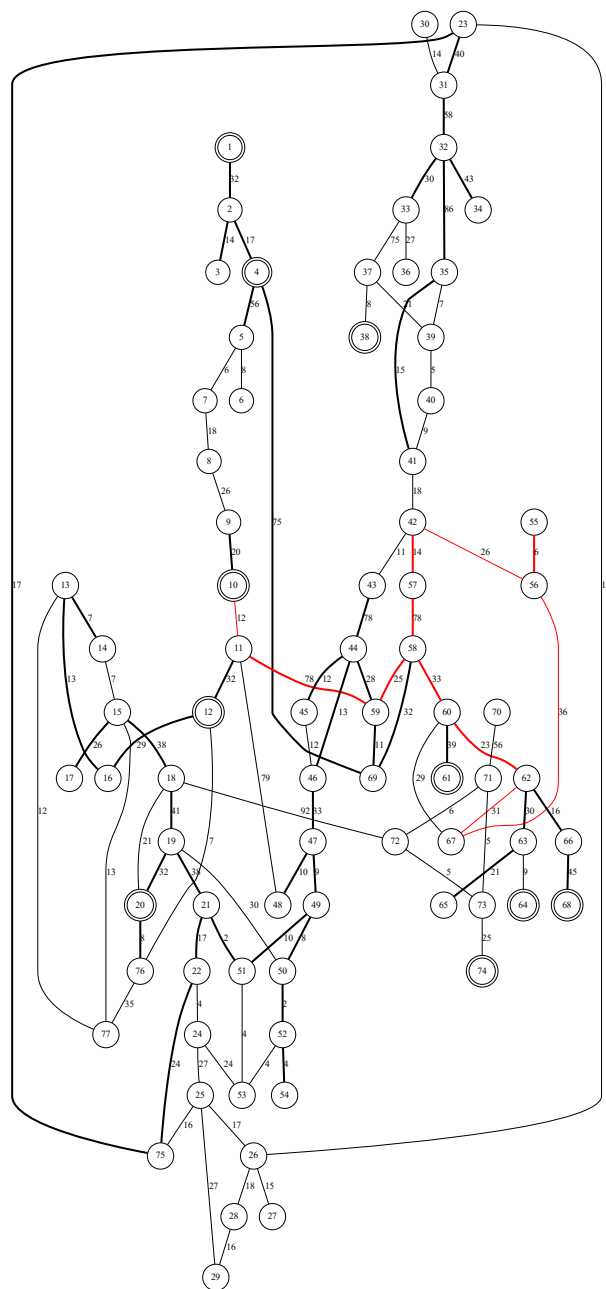


Fig. 6. Route obtained for one of the UAVs (red arcs).