Optimizing FANET deployment for mobile sensor tracking in disaster management scenario

Igor Dias da Silva, Christelle Caillouet, David Coudert

Optimizing FANET deployment for mobile sensor tracking in disaster management scenario*

Igor Dias Da Silva[1], Christelle Caillouet[1], and David Coudert[1]

[1]Université Côte d'Azur, Inria, CNRS, I3S, France

Abstract

This paper addresses the data collection problem using the minimum number of unmanned aerial vehicles (UAVs) in a disaster management scenario where mobile sensors are investigating the devastated area. Critical information needs to be quickly gathered for processing by the rescue team, so the use of UAVs in this situation is of great interest. We propose an optimal model for computing the trajectories of the UAVs while guaranteeing the total coverage of the ground mobile sensors and connectivity among the UAVs with a central base station dedicated to data processing. Our model is based on a decomposition model and is solved effectively using column generation. We show that we can provide a plan for deploying the UAVs minimizing the total traveled distance.

Keywords: UAVs; disaster management; linear programming; column generation.

1 Introduction

The use of Unmanned Aerial Vehicles (UAVs or drones) has gained tremendous attention lately [13]. These autonomous flying devices have several advantages in terms of deployment cost and processing capabilities, making them a promising solution for a wide range of military and commercial applications such as aerial monitoring, disaster management, packet delivery, traffic control, smart agriculture, etc. [8]. Flying Ad-hoc Networks (FANET) [1] can fill the lack of networking infrastructure by allowing the aerial devices to cooperate efficiently to collect data from observations or communications with ground sensors, and gather information to a central processing unit [2, 14].

In this paper, we aim at deploying a FANET to collect data from mobiles sensors on the ground. The envisioned scenarios specifically concern the management of disaster scenarios such as oil spills [11] and wildfires [12]. In these situations, we must gather information quickly to avoid as much damage as possible. Drones networks can collect the data and overcome the lack of network infrastructure to provide a connection with the base station responsible for analyzing the data collected. The importance of planning such rescue operations is crucial. The mobile sensors follow a predefined search strategy so their moves are known. We then need to provide a FANET that ensures a communication path between each sensor and the base station at all times. Therefore, whenever a sensor collects critical information, it can immediately send the information to the base station as shown in Fig. 1.

*This work has been supported by the French government, through the UCA$^{\text{JEDI}}$ Investments in the Future project managed by the National Research Agency (ANR) with the reference number ANR-15-IDEX-01.

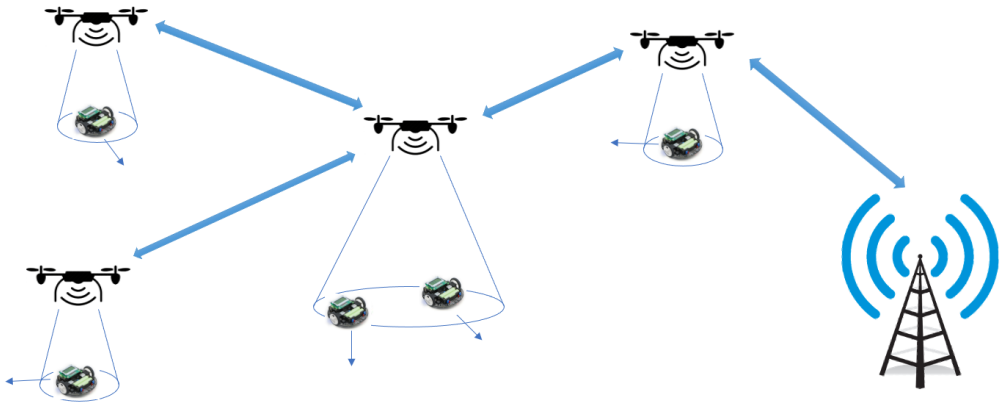

Figure 1: The network architecture.

Drones networks can also be applied to smart farming where mobile sensor robots roam the fields to gather data on soil humidity, temperature, crops quality and perimeter monitoring [10, 16]. In rural areas with poor cellular coverage, establishing a reliable and constant connection between the sensors and a fixed base station for processing can be challenging. When, agricultural applications are not time-critical, data can be stored for some time until the flying network becomes operational. The FANET is an interesting solution thanks to the drone's adaptability provided by their mobility.

Motivated by the need for an efficient computation of the optimal planning of the FANET for gathering mobile sensors data to a central base station, we focus on the optimization of the UAVs trajectories while guaranteeing full coverage of the sensors and connectivity among the FANET at any time. To do so, we present a linear program based on a decomposition model that is solved using column generation. This allows us to generate UAV trajectories to continuously cover the sensors and ensure a connected FANET more effectively than a previous model based on mixed-integer linear programming [7].

Our goal is to help rescue teams in planning their operations. Given the planned movement of ground crews, we determine the expected number of drones needed and their associated trajectories for tracking the ground sensors, to ensure connectivity and coverage for an effective FANET implementation.

This paper is organized as follows: we review the literature on the data collection problem using UAVs in Section 2. In Section 3 we define the problem and then in Section 4 we propose the column generation models. Finally we present our results in Section 5 and conclude our work in Section 6.

2 Related work

To the best of our knowledge, existing work in the literature do not include all the constraints we have here: either there is no connectivity between the UAVs, or the ground nodes are not mobile. In the following, we overview the closest works involving optimization techniques for the data collection problem using drones.

Given a set of sensors, [15] propose a model to optimally place drones such that all sensors are covered by at least one drone. They minimize the energy cost and the number of drones to deploy. The authors also propose efficient heuristics providing good solutions. Later, [6] proposed a new model that considers the drone's battery and replaces the drones when they need to recharge.

However, our goal is not only to cover the sensors but also to connect them to the base station and the models of [6, 15] don't guarantee this connectivity.

To ensure both the sensors' coverage and the connection with the base station, [3] proposed a mixed-integer linear program (MILP) that finds at each time step the optimal set of positions where to deploy the drones. The model minimizes the number of deployed drones and their global distance to the base station, therefore reducing their total consumed energy.

In [9] the authors approach the same problem differently. Instead of using a grid of allowed positions for deploying drones as in [3], they propose possible positions based on the sensors positioning with a clustering algorithm. This results in fewer possible positions and ensures that all the positions considered are around areas of interest, thus avoiding useless positions. Then, they solve a set of covering optimization problems, using linear programming, to decide in which positions the drones should be deployed to cover all targets. They focus on minimizing the number of deployed drones while enforcing coverage to all targets with redundancy. At this point, connectivity between the sensors and the base station is not guaranteed. Also, [9] proposed a final step placing drones between disconnected components until all sensors have a communication path with the base station.

In our scenario the sensors are mobile, hence we need to replace the drones at each time step. The models proposed in [3] and [9] find the positions where the drones should be placed although they don't associate the drones with the positions. This means we know the positions where there should be drones but we don't know which drone should be in which position. In other words, we don't know how to manage the drones from one time step to the other.

Taking the drone's movements into consideration, [7] proposed a MILP to build the network while minimizing the drone's movements and energy consumption. With this model, we can know which drone is in which position and therefore how they move from one time step to the other.

However, associating the drones with the positions turns the problem much more complex. The MILP proposed in [7] has more variables and constraints than the one proposed in [3] and consequently solving it takes more time. In this paper, we extend the work done in [7] to improve its scalability. We use column generation, as in [3], to solve the problem faster.

3 Problem definition

We are working in a discrete observation period with time steps $t \in [0, T]$, which means that we enforce the sensors' coverage and the drones' connectivity to the base station at each time step of the observation period. We consider that the search space is known and the movements of the mobile sensors are predefined depending on the rescuing scenario. Given this strategy, we seek to build the FANET to ensure their coverage. Formally, we define the set S^t for each time step $t \in [0, T]$ that contains the coordinates of each mobile sensor at time t. These sets represent the sensors' search strategy and are known beforehand. Given these sets, we position the drones to build the network. In our experiments, these coordinates are generated as a random walk, where at each time step the sensors move at a fixed speed of 5 m/s in a random direction inside a closed area.

We are given a set P of discrete 3D positions where the drones can be deployed. In Equation (1) we define the coverage radius r_p for each position $p = (x_p, y_p, h_p) \in P$ that depends on the position

altitude and the drones directional antenna half beam-width, or visibility angle θ.

$$r_p \leq h_p \tan \frac{\theta}{2} \tag{1}$$

The coverage radius describes a circle centered at the projection of the position p into the ground. If a mobile sensor is inside this circle it means that the sensor can communicate with a drone deployed at position p. Therefore we say that a drone deployed at position p covers a sensor s if the inequality described in Equation (2) is satisfied.

$$d(p,s) = \sqrt{(x_p - x_s)^2 + (y_p - y_s)^2} \leq r_p \tag{2}$$

Similarly, the communication range R_c is used to determine if a drone at $p \in P$ can communicate with a drone at $q \in P$ by comparing the distance between the two positions with the communication range as shown in Equation (3).

$$\sqrt{(x_p - x_q)^2 + (y_p - y_q)^2 + (h_p - h_q)^2} \leq R_c \tag{3}$$

The communication range of the UAVs can be derived from different channel and propagation models depending on the technology used by the UAVs. Here, we simplify the notations and only derive R_c which can be viewed as the maximum distance such that the packet delivery ratio of each node is guaranteed on average for a given probability. Thus, our model is independent of the communication technology used and can be applied with different settings.

4 Optimization model

In this section, we describe our column generation model to find the drones' optimal positioning to connect the sensors to the base station while minimizing their total traveled distance. The column generation model is divided into two pieces, the master problem and the pricing problem [5].

The master problem is the linear program that select the optimal trajectory for each UAV. In this program, the variables are defined on the possible trajectories, and we seek to select those that minimize the total travelled distance for the drones. Since there is an exponential number of possible trajectories in our search space, it is not efficient to enumerate all of them and solve the program with all the possible variables. We start by considering only a small number of variables allowing a feasible solution for our problem, and we use the pricing problem to add more variables to the master problem that can lead us to better solutions. For several problems, column generation allows us to reach the optimal solution considering only a small subset of all possible variables and consequently much faster [5].

4.1 Master problem (MP)

We define the set C as the set of all possible paths, or trajectories, a drone can follow during the observation period. A path is a sequence of positions $p \in P \cup \{b\}$ (b denotes the base station) through which the drone passes at each time step. We consider a set of predefined positions P where the drones can go and hover to communicate with the ground sensors. The base station is considered as a possible position for the UAVs in case it is not deployed (either it starts flying later than the first time step, or if it goes back earlier than other drones, or if the UAV is useless at some

time because other drones can ensure a valid FANET with less elements). We assume that the drones travel between positions in a straight line and we don't consider collisions. The variables that describe the master problem are:

- y_c is a binary variable for each path $c \in C$ that is 1 if c is selected by a drone and is 0 otherwise.
- f_{pq}^t is a real variable between two positions that are within communication range of each other (i.e. satisfying eq. (3)), and represents the flow at time t from a drone at position p to a drone at position q. The flow here is used for connectivity purpose.

Despite the exponential number of variables we take advantage of column generation and start by considering just a small subset C' of possible paths. Then, we use the pricing problem to add new paths to C'.

Among trajectories registered in C', we need to define the constant δ_{cp}^t that is 1 if trajectory c goes through the position p at time t and $\delta_{cp}^t = 0$ otherwise. Our master problem is described in the standard form by the objective function (4) and the Constraints (5) to (9).

$$\min \quad \sum_{c \in C'} D_c y_c \tag{4}$$

$$-\sum_{c \in C'} \delta_{cp}^t y_c \geq -1 \qquad \forall t \in [0,T], p \in P \tag{5}$$

$$\sum_{l \in N(p)} f_{pl}^t - \sum_{l \in N(p)} f_{lp}^t = |S^t| \qquad \forall t \in [0,T], p = b \tag{6}$$

$$\sum_{l \in N(p)} f_{pl}^t - \sum_{l \in N(p)} f_{lp}^t = 0 \qquad \forall t \in [0,T], p \in P \tag{7}$$

$$\sum_{l \in N(p)} f_{pl}^t - \sum_{l \in N(p)} f_{lp}^t = -1 \qquad \forall t \in [0,T], p \in S^t \tag{8}$$

$$|S^t| \sum_{c \in C} \delta_{cp}^t y_c - \sum_{l \in N(p)} f_{pl}^t \geq 0 \qquad \forall t \in [0,T], p \in P \tag{9}$$

Our goal is to choose a set of trajectories for the UAVs that build a valid FANET during the observation period. This means that if the drones follow the selected paths, the sensors will be covered and connected to the base station at each time step. The objective function (4) is to minimize the total cost of the selected paths. Here we consider the cost D_c to be the total distance of the path c, i.e. the sum of the 3D-distance between two consecutive positions of the trajectory $D_c = \sum_{p,q \in c} D_{p,q}$. Constraints (5) ensure that we cannot choose more than one path going through the same position p at the same time t. The base station is not included in this equation since it can hold as many drones as possible at the same time in case these drones are currently useless. Even though Constraints (5) guarantee that we have no drones at the same position at the same time, it does not include collision avoidance between the time steps.

To guarantee that the selected trajectories build the FANET connecting all the sensors to the base station we verify if it is possible to find a flow that leaves the base station, goes through the positions occupied by drones, and reaches all the sensors at each time step. Finding this flow at

each time step means that there is a communication path between the base station and the mobile sensors. For this, Constraints (6) ensure that the total flow leaving the base station is equal to the number $|S^t|$ of mobile sensors. $N(p)$ denotes the set of positions that are within communication range of p (i.e. satisfying eq. (3)). Constraints (7) ensure that the flow entering a position p occupied by a drone is the same as the flow that leaves p. Hence the positions $p \in P$ are used only to route the flow that leaves the base station. Constraints (8) ensure that the total flow entering each mobile sensor must be 1, and so that each sensor is reached.

Constraints (9) ensure that if there is flow going through a position p at time t, then there must be a path c selected that goes through p at time t. In other words, we must select a path c that places a drone at p at time t otherwise no flow is allowed to go through p. The computed flows ensure that the selected trajectories contain positions fulfilling the connectivity and coverage constraints of our problem.

Solving this path formulation is equivalent to solving the formulation presented in [7]. Satisfying these constraints means finding a set of paths that builds the network connecting the sensors to the base station. Since now we have a variable for each possible trajectory, if we were to consider all variables at once, this problem would be impossible to solve in a reasonable time as we have an exponential number of variables. However the goal of column generation is to find the optimal solution without having to consider all variables, but just a small subset $C' \subseteq C$ of them.

4.2 Pricing problem

The pricing problem is the one used to generate new variables. The variables created by the pricing problem improve our current optimal solution. To find such a variable, we try to break the certificate that ensures the optimality of the current solution. For any feasible solution $y_1, ..., y_n$ of a linear program called primal, we have an equivalent solution $\beta^{(1)}, ..., \beta^{(m)}$ for its dual problem. If this equivalent solution is a feasible solution for the dual problem it means that $y_1, ..., y_n$ is the optimal solution of the primal program. This is called the optimality certificate. And by creating a variable that makes the corresponding solution of the dual problem infeasible, we prove that the new variable allows us to find a better solution because we break the current solution's optimality certificate.

To derive our pricing problem we look at the dual problem. A variable of the dual program corresponds to a set of constraints of the primal program, and the constraints of the dual are associated with variables of the primal. We define the dual variables $\beta_{tp}^{(5)}$ and $\beta_{tp}^{(9)}$ for Constraints (5) and (9) respectively. The dual constraint for variable y_c is as follows.

$$-\sum_{t=0}^{T}\sum_{p\in P}\delta_p^t\beta_{tp}^{(5)} + |S^t|\sum_{t=0}^{T}\sum_{p\in P}\delta_p^t\beta_{tp}^{(9)} \leq D \tag{10}$$

Once we solve the master problem, we obtain the corresponding values for all $\beta_{tp}^{(5)}$ and $\beta_{tp}^{(9)}$ and Constraint (10) is satisfied for all paths in C' because we have the optimal solution of the current master problem.

In the following, we seek to find if it exists new trajectories in the FANET violating Constraints (10). This is the goal of the pricing problem. If such path exists, we add it to the current set C' of variables of the master problem. If there is no new trajectory violating (10), then the separation-optimization theorem ensures that we have reached the optimal solution of the relaxed master program.

We propose two pricing programs: (i) a mixed-integer linear program (CGLP), and (ii) a shortest path algorithm (CGSP).

4.2.1 Linear program

Our goal is to find values for the pricing problem variables that describe a new path that violates Constraint (10). We use the following variables:

- δ_p^t is a binary variable that is 1 if the new path goes through p at time t and is 0 otherwise.
- D is a real variable that represents the new path total distance.
- ω_{pq}^t is a binary variable that is 1 if the new path contains the position p at time $t-1$ and the position q at time t and it is 0 otherwise.

Hence we can describe the pricing problem with objective (11) and Constraints (12)-(16).

$$\max -\sum_{t=0}^{T}\sum_{p\in P}\delta_p^t\beta_{tp}^{(5)} + |S^t|\sum_{t=0}^{T}\sum_{p\in P}\delta_p^t\beta_{tp}^{(9)} - D \tag{11}$$

$$\sum_{p\in P\cup\{b\}}\delta_p^t = 1 \qquad \forall t \tag{12}$$

$$\sum_{t=0}^{T}\sum_{p,q\in P\cup\{b\}} D_{pq}\omega_{pq}^t + \sum_{p\in P\cup\{b\}}(\delta_p^0 D_{pb} + \delta_p^T D_{bp}) = D \tag{13}$$

$$\omega_{pq}^t \le \delta_p^{t-1} \qquad \forall t \text{ and } p,q \in P\cup\{b\} \tag{14}$$

$$\omega_{pq}^t \le \delta_q^t \qquad \forall t \text{ and } p,q \in P\cup\{b\} \tag{15}$$

$$\omega_{pq}^t \ge \delta_p^{t-1} + \delta_q^t - 1 \qquad \forall t \text{ and } p,q \in P\cup\{b\} \tag{16}$$

The objective (11) is to break the optimality of constraint (10). If we can find a solution with positive objective value for our pricing problem it means that we broke the optimality constraint and therefore found a new variable to add to our master problem.

Constraints (12)-(16) ensure that the solution of our pricing problem is a valid new variable. More specifically, we have to ensure that the solution is a valid path. Constraints (12) enforce that at each time step we must select exactly one position that is either in P or is the base station. Hence the solution must be a valid trajectory, i.e. a list of positions for each time step.

Constraint (13) ensures that the variable D corresponds to the new path total distance and Constraints (14) to (16) ensure the definition of the variables ω_{pq}^t.

By satisfying these constraints we have a valid path and if we can find a valid path with a positive value for the objective function (11), it means that this new path will allow us to find a better solution in our master problem and must be added to C'. If we cannot break the optimality constraint then we cannot find a better solution to our master problem.

Column generation can help us reach the solution faster because solving the relaxed master problem can be done in polynomial time. The pricing problem, although an integer linear program that must be solved several times, is smaller than the MILP proposed in [7], and can be solved

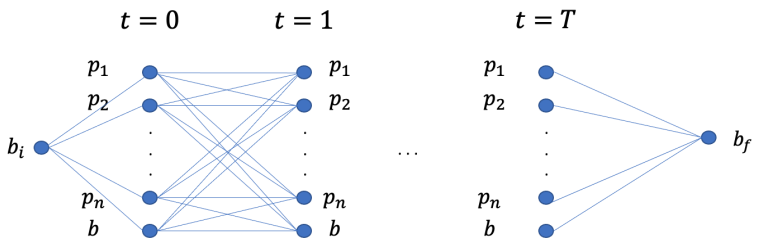

Figure 2: The pricing problem as a shortest path problem.

quickly. And finally, when we need to solve the integer master problem, the subset C' tends to be small and therefore we can reach our final solution fast.

However, as the number of positions and the number of time steps increases, the integer programming pricing problem becomes much bigger and slows down the column generation. We can improve this by substituting the pricing problem as an integer linear program with an algorithm that reaches the same results in polynomial time.

4.2.2 Shortest path algorithm

We can look at our pricing problem as a shortest path problem. Given a graph $G = (V, E)$ as shown in Fig. 2 where the vertices are the positions and the base station at each time step and the edges connect past positions to future positions, we can interpret our pricing problem as finding a path from b_i to b_f with the shortest cost. This means selecting a sequence of positions through time that allows us to break the optimality constraint. For each edge we consider the cost as the negative distance between the positions connected by the edge and for each vertex we consider the cost as $\beta_{tp}^{(5)} - |S^t|\beta_{tp}^{(9)}$. Note that in objective (11), the goal is to maximize the cost so we invert the sign of the cost here to have the equivalent shortest path problem. Finding the path with the shortest total cost in this graph is equivalent as solving the previous pricing problem.

Lemma 1. *The pricing problem can be solved in time* $O(|P|^2|T|)$.

Proof. Let us build the graph $G = (V, E)$ as shown in Fig. 2 with one vertex per position (including the base station) and per time step. So vertex (p, t) represents the position p at time step t. Then, there is an arc from vertex (p, t) to vertex $(p', t + 1)$ representing position p' at time step $t + 1$ if position p' is reachable from position p. Furthermore, there is a vertex b_i in G with arcs to each vertex $(p, t = 0)$ for $p \in P$, and a vertex b_f with arcs from each vertex $(p, t = T)$ for $p \in P$ to d. Vertex b_i (resp. b_f) models the fact that the initial (resp. final) position of a drone is on the base station.

The cost of an arc is the negative distance between the positions it connects and we set the cost of a vertex (p, t) to $\beta_{tp}^{(5)} - |S^t|\beta_{tp}^{(9)}$. The costs of b_i and b_f are null.

Clearly, the graph G is acyclic and so computing a shortest path from b_i to b_f can be done in time $O(|V| + |E|)$ [4]. Since the number of vertices is in $O(|P||T|)$ and the number of arcs is in $O(|P|^2|T|)$, the time complexity follows.
Finally, observe that the shortest path from b_i to b_f in G is a minimum cost trajectory. □

4.3 Solving the column generation

To solve the column generation (CG) we start with a subset C' of possible paths allowing us to obtain a feasible solution of the master program. Then, we solve the relaxed master problem (RMP), that is the master problem in which we relax the binary variables y_c to accept real values in range [0..1]. After, we solve the pricing problem using $\beta_{tp}^{(5)}$ and $\beta_{tp}^{(9)}$, the dual variables of the master. If we obtain a positive value for the objective function of the pricing problem, it means that the solution describes a new path that we must add to C'. After adding this new path, we repeat the process by solving the RMP with the new variable. Otherwise, if we obtain a negative value for the objective function, it means there are no more paths that we need to add to C'. We then solve the integer master problem (MP) with C', where the binary variables y_c can only be 0 or 1, and we get our final solution.

Observe that the solution obtained when solving the MP with C' is not always an optimal solution for the problem. However, this is a valid solution whose cost $\tilde{z}_{\text{MP}}$ is an upper bound of the optimal solution. Furthermore, the objective value z^*_{RMP} obtained when solving the RMP is a lower bound on the value of the optimal solution. Hence, we can evaluate the so-called optimality gap of a solution as $(\tilde{z}_{\text{MP}} - z^*_{\text{RMP}})/z^*_{\text{RMP}}$. We will see in our experiments that this gap is small in practice, meaning that the solutions are of good quality. To always obtain an optimal solution, one must combine the CG with a branch-and-bound process, a much more involved approach.

5 Results

The models presented were implemented using the Java API of IBM Cplex solver 12.10.0 and run on a computer equipped with an Intel(R) Core(TM) i9-10900K CPU 3.70 GHz and 64 GB RAM, running Fedora 33 operating system. The inputs for our models are the mobile ground sensors trajectories inside an area A over a period of time T. In our experiments we consider A to be a 10000 m^2 square with 5 sensors, and the observation period consists of 7 time steps with 2 seconds between them. The sensors coordinates are generated as a random walk, where at each time step the sensors move at a fixed speed of 5 m/s in a random direction inside a closed area. Given the moves of the sensors, we solve our column generation model to find the trajectories of the drones. We analyze the efficiency of the model with both pricing approaches, the linear program and the shortest path algorithm. We make the input parameters vary, and for each considered scenario, we average the results obtained on ten instances.

We distribute the possible positions as a grid dividing the area A into squares of equal length. Each intersection in this grid corresponds to a possible position for the UAVs entered in the set of possible positions P. In our experiments, all possible positions are 45 m above the ground by default, but it is easy to make the altitude of the drones vary by including more positions in the set P.

5.1 Scalability

We first show that our CG model is scalable in comparison to a compact linear model presented in [7]. In Fig. 3 we compare the time required to solve the mixed-integer linear program (MILP) proposed in [7], our column generation model with the standard pricing problem as a linear program (CGLP) and the pricing problem as a shortest path problem (CGSP). Solutions for more than 25 positions was impossible to obtain for the MILP but we can reach 64 positions for CGLP, but

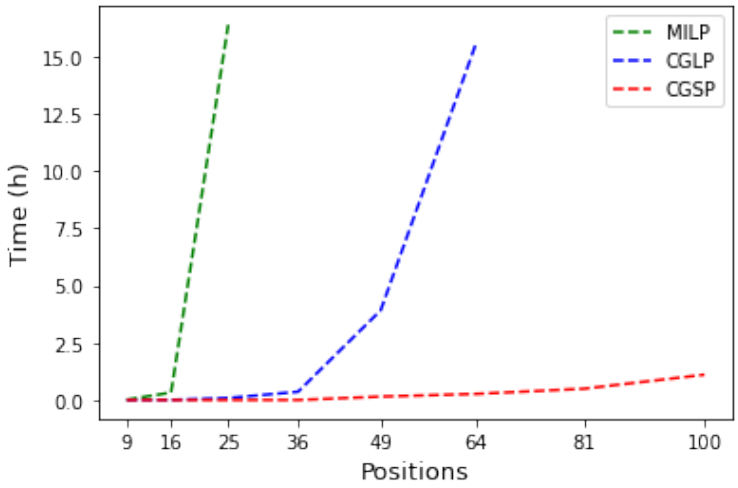

Figure 3: Comparison between the time to solve the problem when considering the compact mixed integer formulation (MILP) and the column generation using either the linear program pricing problem (CGLP) or the shortest path pricing problem (CGSP).

for a long resolution time. Both column generation models are faster than the MILP while the shortest path pricing problem provides a much faster alternative. When considering 64 positions, the shortest path pricing problem reduces the average solution time from 15.54 hours to 13.35 minutes. This validates our scalable approach for the FANET planning in rescue situations. As expected, increasing the number of positions quickly increases the complexity of the problem.

5.2 Optimality

However, as explained in Section 4.3, the column generation doesn't always reach the optimal solution. Nonetheless, the column generation reaches the same solution as the MILP in 54% of our experiments. The average optimality gap of column generation was 5%. When considering 9 positions the average optimality gap was 9% but as we increased the number of positions to 16 and 25, the average optimality gap decreases to 2% and 3.1% respectively.

Although column generation doesn't always reach the optimal solution, the distance to the optimal solution is small and we are able to consider more positions than with the MILP. Moreover, increasing the number of positions improves the solutions thanks to the more precise positioning of the drones. With few positions, whenever a sensor leaves the coverage of a drone, the drone needs to travel a bigger distance than necessary to keep covering the sensor since the positions are far apart from each other. Once we increase the number of positions, the UAVs can be deployed more precisely such that they need to move less, and if they need to move, they can adjust their positioning by moving smaller distances due to the higher density of positions.This is shown in Fig. 4 where the more positions, the less the drones travel on average. Therefore, if we look at the column generation with the shortest path pricing problem (CGSP), when considering 64 positions we find better solutions than the MILP considering 25 positions. Not only the solutions are of better quality but we also find them faster.

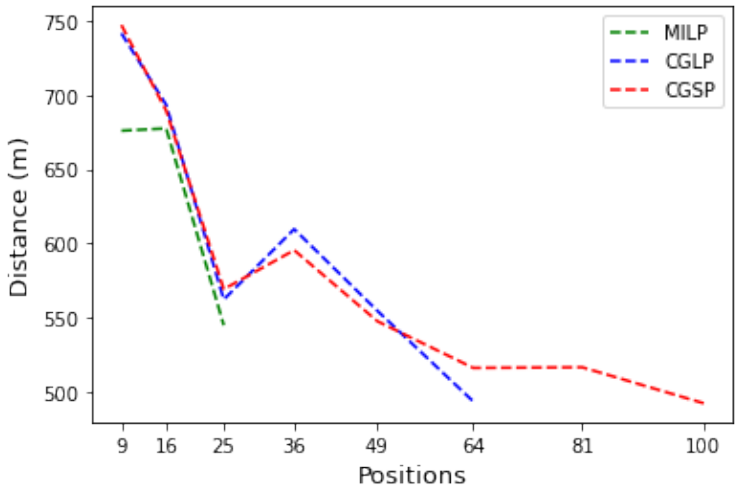

Figure 4: Comparison between the solutions of the problem when considering the compact mixed integer formulation (MILP) and the column generation using either the linear program pricing problem (CGLP) or the shortest path pricing problem (CGSP).

5.3 Pricing policy

In order to improve our CG resolution, we tested several policies when solving the shortest path pricing problem. Given that we may have many paths per iteration with minimal cost that can be added to $C^{'}$ (shortest path is not unique in the graph of Figure 2), we explore adding multiple paths at the same time to $C^{'}$ between two resolutions of the RMP to add several columns in the same iteration and try to accelerate the resolution. Fig. 5 shows that increasing the maximum number of paths that can be added per iteration seems to be interesting when we consider few positions, as the average time required to solve the problem is shorter. However, as the number of positions increases, adding more paths per iteration leads to a longer time to solve the problem. This happens because the more positions we consider, the more solutions can be found for the pricing problem per iteration. Therefore, increasing the maximum number of paths that can be created per iteration leads to many more paths being added to the master problem as shown in Fig. 6. As a result, we need a longer time to solve the master problem per iteration.

5.4 Planning characteristics

In our experiments, we recall that we seek to deploy a connected FANET that covers all the mobile sensor nodes at each time step of the observation period. In such cases, there will be drones dedicated to the coverage and the tracking of the mobile nodes, and other flying devices needed to ensure the connectivity constraints, acting only as relays in the FANET such that data can be collected by the base station at any time.

This second group of UAVs thus has a significantly shorter distance to travel, since they only need to reach their position and then stay hovering. On the opposite, the distance of the trajectory of the covering UAVs is of particular interest, and our model encourages one drone to cover several sensors on the ground to reduce the number of used UAVs.

For this purpose, we have observed that, on a large observation area A of 40000 m^2, covering a

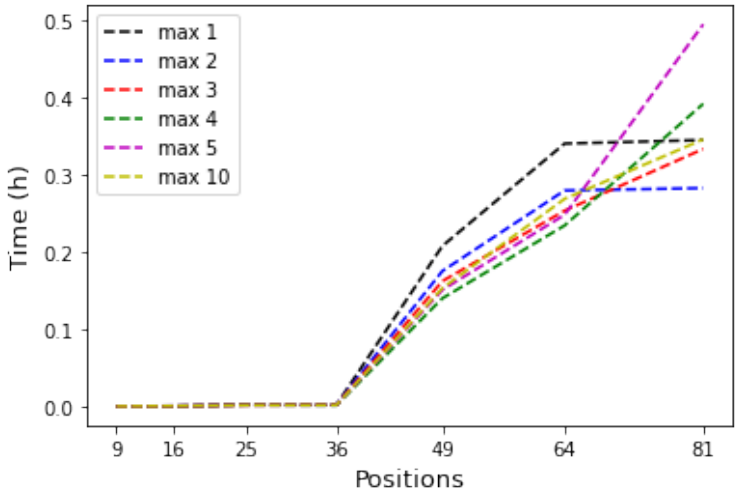

Figure 5: Comparison between the time to solve the pricing problem as a shortest path problem while varying the maximum number of new paths generated per iteration.

single mobile sensor requires on average 5.1 drones, showing that there will be one drone covering the sensor and the other relaying the sensor to the base station. But when we increase the number of sensors, the number of used drones does not increase proportionally. Covering 5 mobile sensors requires on average 8 drones, and for 20 mobile sensors we need on average 14.6 drones to ensure the coverage and keep them connected to the base station at all times. These results show that we can use less drones than the number of mobile sensors on the ground, combining the characteristics of multiple coverage and connectivity envisioned by our model.

6 Conclusion

In this paper, we presented a new model based on column generation with a path formulation extending the performances of the MILP proposed in [7]. The model can build a FANET covering mobile rescue teams deployed in a devastated area while providing a connected path to the base station for quick data collection and processing for efficient disaster management. The objective is to minimize the drones' total traveled distance to maximize the network lifetime.

We proposed two pricing models in the column generation process, and we showed that replacing the linear program pricing problem with a shortest path problem that can be solved in polynomial time improves drastically the resolution time. We showed that both column generation models are scalable for real size scenarios, and although they can't always reach the optimal solution, their average optimality gap is no more than 5%.

With the column generations models, we are able to consider more positions, which increases the complexity of the problem but allows us to place the drones more precisely. Therefore, by considering more positions we can find better solutions with the column generation than we could with the mixed-integer linear program considering fewer positions.

We explored changing the maximum number of paths that can be created per iteration of the column generation. With few positions, more paths per iteration lead us to the solution faster, but with more positions, this leads to many unnecessary paths added resulting in a longer time to find

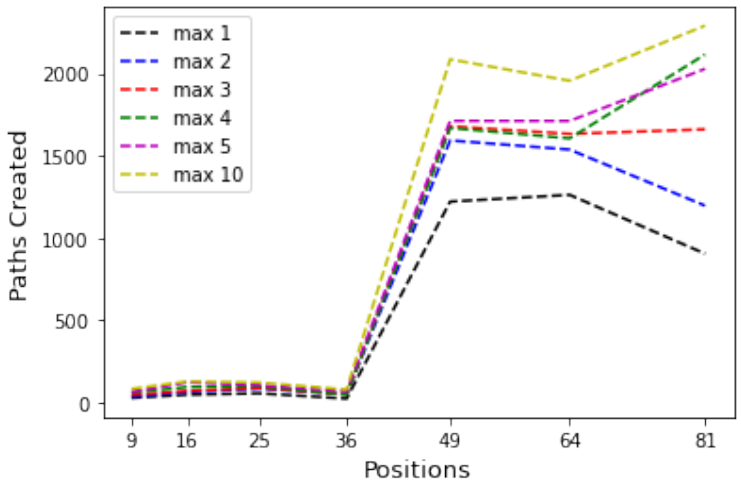

Figure 6: Comparison between the number of paths required to solve the general problem while varying the maximum number of new paths per iteration of the shortest path pricing problem.

the solution. This raises interest in an adaptive maximum number of paths per iteration that could lead us to the solution faster by avoiding unnecessary paths.

This upstream work is dedicated to optimizing the planning of operations the rescue teams when a disaster occurs, because our models can still need a lot of time to find a near-optimal solution. They also require knowing the trajectory of the sensors in advance but we should expect these trajectories to change in these situations. Therefore it is in our interest to investigate further and study methods to update this plan during the disaster scenario to maintain coverage and connectivity as the situation evolves.

References

[1] Ilker Bekmezci, Ozgur Koray Sahingoz, and Şamil Temel. Flying ad-hoc networks: A survey. *Ad Hoc Networks*, 11(3):1254–1270, 2013. doi:10.1016/j.adhoc.2012.12.004.

[2] Yann Busnel, Christelle Caillouet, and David Coudert. Self-organized UAV-based Supervision and Connectivity: Challenges and Opportunities. In *IEEE International Symposium on Network Computing and Applications (NCA)*, pages 1–5, Cambridge, United States, September 2019. IEEE. URL: https://hal.inria.fr/hal-02267396, doi:10.1109/NCA.2019.8935060.

[3] Christelle Caillouet, Frédéric Giroire, and Tahiry Razafindralambo. Efficient Data Collection and Tracking with Flying Drones. *Elsevier Adhoc networks*, 89(C):35–46, 2019. doi:10.1016/j.adhoc.2019.01.011.

[4] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms, Third Edition*. The MIT Press, 3rd edition, 2009.

[5] Guy Desaulniers, Jacques Desrosiers, and Marius M. Solomon. Column generation 1st edn. Springer, Boston, MA, 2005. doi:10.1007/b135457.

[6] Luigi Di Puglia Pugliese, Francesca Guerriero, Dimitrios Zorbas, and Tahiry Razafindralambo. Modelling the mobile target covering problem using flying drones. *Optimization Letters*, 10(5):1021–1052, 2016. doi:10.1007/s11590-015-0932-1.

[7] Igor Dias Da Silva and Christelle Caillouet. Optimizing the trajectory of drones: trade-off between distance and energy. In *2nd International Workshop on Internet of Autonomous Unmanned Vehicles (IAUV) in conjunction with IEEE SECON 2020*, Cuomo, Italy, June 2020. doi:10.1109/SECONWorkshops50264.2020.9149781.

[8] Aicha Idriss Hentati and Lamia Chaari Fourati. Comprehensive survey of uavs communication networks. *Computer Standards & Interfaces*, 72:103451, 2020. doi:10.1016/j.csi.2020.103451.

[9] Donald Mahoro Ntwari, Daniel Gutierrez-Reina, Sergio Luis Toral Marín, and Hissam Tawfik. Time efficient unmanned aircraft systems deployment in disaster scenarios using clustering methods and a set cover approach. *Electronics*, 10(422), 2021. doi:10.3390/electronics10040422.

[10] Prusayon Nintanavongsa and Itarun Pitimon. Impact of sensor mobility on uav-based smart farm communications. In *International Electrical Engineering Congress (iEECON)*, 2017. doi:10.1109/IEECON.2017.8075822.

[11] Philip Odonkor, Zachary Ball, and Souma Chowdhury. Distributed operation of collaborating unmanned aerial vehicles for time-sensitive oil spill mapping. *Swarm and Evolutionary Computation*, 46:52–68, 2019. doi:10.1016/j.swevo.2019.01.005.

[12] Huy Xuan Pham, Hung Manh La, David Feil-Seifer, and Matthew C. Deans. A distributed control framework of multiple unmanned aerial vehicles for dynamic wildfire tracking. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(4):1537–1548, 2020. doi:10.1109/TSMC.2018.2815988.

[13] Kimon P Valavanis and George J Vachtsevanos. *Handbook of unmanned aerial vehicles*, volume 2077. Springer, 2015.

[14] Evşen Yanmaz, Markus Quaritsch, Saeed Yahyanejad, Bernhard Rinner, Hermann Hellwagner, and Christian Bettstetter. Communication and coordination for drone networks. In *Ad Hoc Networks*, pages 79–91. Springer, 2017. doi:10.1007/978-3-319-51204-4_7.

[15] Dimitrios Zorbas, Luigi Di Puglia Pugliese, Tahiry Razafindralambo, and Francesca Guerriero. Optimal drone placement and cost-efficient target coverage. *Journal of Network and Computer Applications*, 75:16–31, 2016. doi:10.1016/j.jnca.2016.08.009.

[16] Dimitrios Zorbas and Brendan O'Flynn. A network architecture for high volume data collection in agricultural applications. In *1st International Workshop on Wireless sensors and Drones in Internet of Things (Wi-DroIT), in conjunction with DCOSS*, May 2019. doi:10.1109/DCOSS.2019.00107.