On task assignment for early target inspection in squads of aerial drones

Novella Bartolini, Andrea Coletta, Gaia Maselli
Computer Science Department, Sapienza University of Rome
{bartolini, coletta, maselli}@di.uniroma1.it

***Abstract*—We consider the problem of assigning tasks and related trajectories to a fleet of drones, in critical scenarios requiring early anomaly discovery and intervention. Drones visit target points in consecutive trips, with recharging and data offloading in between. We propose a novel metric, called weighted coverage, which generalizes classic notions of coverage, as well as a new notion of accumulative coverage which prioritizes early inspection of target points. We formulate an ILP problem for weighted coverage maximization and show its NP-hardness. We propose an efficient polynomial algorithm with guaranteed approximation. By means of simulations we show that our algorithm performs close to the optimal solution and outperforms a previous approach in terms of several performance metrics, including coverage, average inspection delay, energy consumption, and computation time, under a wide range of application scenarios.**

***Index Terms*—trajectory planning, uav, drones, task assignment, vehicle routing**

I. Introduction

Unmanned aerial vehicles (UAVs) are a key technology for damage assessment and search and rescue operations in the aftermath of disasters, such as earthquakes, wildfires, or floodings. They are increasingly used in many critical scenarios for many purposes from rapid spot checks, to UAV assisted cellular communications, to delivery of items to be dropped to people in need [1], [2]. Aerial drones are also frequently used for crop health assessment, or mapping glaciers or forest areas [3].

While single remotely controlled drones are commonly used for monitoring regions of interest in multiple applications, there are many situations in which monitoring points of interest in a sequence is not sufficient for the purpose of reconstructing a representative view of ongoing events, and the application requires monitoring from multiple drones working concurrently. For instance, in critical scenarios where drones are looking for survivors, or aim at distributing medicines or water to trapped or disabled humans or animals, it is of uttermost urgency for them to deliver service within strict time constraints. Thus, the benefit of using multiple aerial devices capable of sampling multiple targets in parallel, and ensuring the best tradeoff between monitoring accuracy and time is clear. However, planning drone trajectories along a field of interest is a big challenge under several points of view, including autonomy, cooperation, coordination, and control.

This work was supported by the North Atlantic Treaty Organization (NATO) grant G4936, under the Science for Peace and Security program.

The literature presents several research works in the area of network algorithms and optimization, formulating various algorithmic optimization problems for guiding decision making in controlling mobile vehicles operations. Most of these works study variants of the Traveling Salesman Problem, for multiple vehicles [4], [5], [6] and do not address our problem setting with multiple consecutive trips, under the objective to maximize early target inspection.

Closest to our approach, the work in [7] addresses multi-UAV path planning to minimize mission completion time in search-and-reconnaissance operations. The main contribution of such work is a constant factor approximation algorithm for multi-depot multi-UAV scenarios. Unfortunately this solution assumes unlimited energy for drones, making the approach applicable only in small areas, where the limited energy actually available is sufficient to complete the mission. The problem of path planning is addressed also in the context of intelligence transportation systems. In [8], trajectory optimization aims at finding time-optimal paths for multiple UAVs such that they collectively visit the coverage areas when they fly from a starting point to a final location. However, differently from ours, this work considers only a single-depot and a unique trip for each drone.

In this paper we formulate the problem of assigning monitoring tasks and planning related trajectories for multiple drones forming a cooperative squad, with the purpose of maximizing the network efficiency within device and application related constraints, including battery limitations, and time requirements. Specifically, we want to provide early target inspection, prioritizing early coverage of target points in a round based execution of the monitoring tasks. For this purpose we introduce an original metric of coverage, called *weighted coverage*, which generalizes the classic notion of total coverage, as well as utility based metrics which capture the capability of a trajectory planning algorithm to provide early target inspection. Differently from previous work, our approach considers heterogeneous drones (e.g., with different energy availability), multiple depot stations, and the possibility for each drone to perform multiple consecutive trips.

We contribute an analytic formulation of the weighted coverage maximization problem as an ILP model and show its NP-hardness. Experiments confirm that the optimization problem has prohibitive execution times for problem instances of average to large size. For this reason we propose a polyno-

mial approximation algorithm, with guaranteed performance. We perform a comparative analysis of the proposed algorithm against the optimal solution, and a previous approach proposed by Kim et al. in [7] for a simpler scenario. To make a fair performance comparison, we propose a variant of the above approach, to make it suitable to the wider setting under study in this paper.

A wide range of simulations confirm our theoretical results, and show that our approach outperforms previous algorithms in all the performance aspects considered in this work, including coverage, average inspection delay, as well as energy cost and computation time.

In summary, the main contributions of this paper are:

- We formulate a novel performance metric, called *weighted coverage*, to measure the capability of a trajectory planning algorithm to provide early target inspection, and formulate the related optimization problem. We show that weighted coverage generalizes traditional metrics of coverage, as well as a new notion of accumulative coverage which is directly related to early coverage capabilities.
- We formulate the problem of maximizing weighted coverage as an ILP model and prove its NP-hardness. By means of experiments we show that such optimization has prohibitive computation times even for small problem instances, which preclude its application in safety critical scenarios, motivating the need of polynomial time heuristics.
- We proposed a novel greedy-and-prune polynomial algorithm for approximating the optimal solution of the aforementioned problem. We prove that the above algorithm has a constant factor approximation of $1/2$.
- Through extensive simulations, we show that our algorithms perform close to the optimal and outperform previous solutions in all the performance metrics, including coverage, average inspection delay, energy consumption and computation time.

II. Problem formulation

We consider the scenario in which a squad of aerial drones aims at inspecting a given set of target points Ψ in the region of interest. These points may be the known locations of survivors of a catastrophe, requiring medicines or water to be dropped from aerial drones, or more in general they may be areas of suspected anomalies, requiring immediate surveillance and local inspection.

Let $\mathcal{U}$ be the set of aerial vehicles forming the squad, and let d_u be the home depot of drone $u \in \mathcal{U}$, i.e. the point of the region of interest from which the drone departs, and where it is recollected, its data are offloaded, and batteries are recharged at the end of each flight. The battery life of a drone imposes a limitation on the number of target points that can be inspected in a unique flight, before going back to the depot for recharging.

For example, the DJI Phantom 4 Pro [9] can fly for about 30 minutes with no additional payload, and needs about 1hr to recharge its battery from 15% to 100%.

A. Progressive coverage metrics

We consider a progressive execution of the drone squad monitoring activity in consecutive rounds $n = 1, 2, \ldots, N$, where new target points are inspected at each new round. A round based execution is meant to consider recharging and offloading at the depot between consecutive rounds. We denote with $[N]$ the set $\{1, 2, \ldots, N\}$. The variables $\delta_i(n) \in \{0, 1\}$ represent the *coverage status of target i at round n* for any $i \in \Psi$. If target i is visited exactly at round n, then $\delta_i(n) = 1$, otherwise $\delta_i(n) = 0$. To avoid repeated coverage of a single point i, we consider $\sum_{n=1}^{N} \delta_i(n) \leq 1$.

We denote with $\delta(n) \triangleq \sum_{i \in \Psi} \delta_i(n)$ the *amount of target points covered at round n*.

We define a novel metric of round based coverage called *Weighted Progressive Coverage* $\mathcal{W}(n)$ as follows. We introduce a round based utility function $w(n) : [N] \rightarrow \mathbb{R}_0^+$ defining non increasing utilities for each round, such that $w(n_1) \geq w(n_2),\ \forall n_1, n_2 \in [N]$, with $n_1 < n_2$. The weighted progressive coverage, shortly called weighted coverage, is defined as follows:

$$\mathcal{W}(N) \triangleq \sum_{k=1}^{N} w(k) \cdot \delta(k). \tag{1}$$

The specific setting of the weights $w(k)$ is meant to give different priority to the coverage achieved at each round k.

It is easy to see that, under this generalization, the *total coverage in N rounds*, hereby denoted with $\Delta(N)$ is obtained by setting $w(i) = 1,\ \forall i = 1, \ldots, N$, in Equation 1 as follows:

$$\Delta(N) \triangleq \sum_{k=1}^{N} \delta(k), \tag{2}$$

which represents the number of targets covered either at round N or at any round before N.

We now define a novel coverage metric which prioritizes early coverage of target points, called *accumulative coverage $\mathcal{A}(N)$, at round N*, as follows:

$$\mathcal{A}(N) \triangleq \sum_{k=1}^{N} \Delta(k).$$

We observe that the accumulative coverage metric is also a special case of weighted progressive coverage, by setting the values of the weights $w(k)$, $k = 1, \ldots, N$ as explained in the following observation.

Observation II.1. *The accumulative coverage function $\mathcal{A}(N)$ is a linear combination of the single round coverage variables which can be expressed as:*

$$\mathcal{A}(N) = \sum_{k=1}^{N} (N - k + 1) \cdot \delta(k). \tag{3}$$

Therefore the accumulative coverage function is a special case of weighted coverage obtained by setting the values $w(k) = N - k + 1$, in Equation 1.

Proof. By simple algebraic passages, we see that

$$\mathcal{A}(N) = \sum_{k=1}^{N} \Delta(k) = \sum_{k=1}^{N} \sum_{j=1}^{k} \delta(j) = \sum_{k=1}^{N} (N - k + 1) \cdot \delta(k).$$

□

Equation 3 shows that as a performance objective, $\mathcal{A}(N)$ prioritizes coverage in the early rounds. For example, consider a set of 6 targets $\{a, b, c, d, e, f\}$ and two solutions A and B, where solution A covers $\{a, b\}$ at round 1 and $\{c, d, e, f\}$ at round 2, and solution B covers $\{c, d, e, f\}$ at round 1 and $\{a, b\}$ at round 2. After two rounds, solution A would have an accumulative coverage of $2 + (2 + 4) = 8$, whereas solution B would have an accumulative coverage of $4 + (4 + 2) = 10$, hence the model would choose solution B.

Consider N rounds, and a setting where all the target points of Ψ are covered in a progressive, round based inspection, namely $\sum_{k=1}^{N} \delta(k) = |\Psi|$.

Let $\tau_i(N)$ be the round at which target i is visited, hereby called the *inspection delay* of target $i \in \Psi$ in an N-rounds progressive solution. It holds:

$$\tau_i(N) \triangleq \sum_{k=1}^{N} k \cdot \delta_i(k), \tag{4}$$

We denote with $D(N)$ the *average inspection delay*:

$$D(N) = \sum_{i \in \Psi} \tau_i(N)/|\Psi|. \tag{5}$$

Theorem II.1. *For any progressive coverage solution $\delta_i(k)$, with $i \in \Psi$ and $k \in [N]$, it holds that:*

$$\mathcal{A}(N) = |\Psi| \cdot [N + 1 - D(N)], \tag{6}$$

where $\mathcal{A}(N)$ and $D(N)$ are defined in Equations 3 and 5, respectively. It follows that any progressive coverage solution which guarantees complete coverage of the set Ψ and maximizes the accumulative coverage function $\mathcal{A}(N)$, also minimizes the average inspection delay $D(N)$.

Proof. By applying Observation II.1 we have that

$$\mathcal{A}(N) = (N + 1) \cdot \sum_{k=1}^{N} \delta(k) - \sum_{k=1}^{N} k \cdot \delta(k) =$$

$$= (N+1) \cdot |\Psi| - \sum_{k=1}^{N} \sum_{i \in \Psi} k \cdot \delta_i(k) = (N+1) \cdot |\Psi| - \sum_{i \in \Psi} \tau_i(N) =$$

$$= |\Psi| \cdot [(N + 1) - D(N)].$$

□

Discussion on the setting of the number of rounds N: We clarify that the setting of N — the maximum number of rounds for inspecting the target points in the field of interest — typically responds to application requirements. Considering that maintenance operations (e.g., battery recharging and data offloading), typically require a time that is in the order of 3 to 8 times the maximum flight duration, there can be multiple criteria and motivations to determine this setting, for example if drones are supposed to operate during daylight hours. The setting of N affects the total time of the target inspection mission. Assuming that each drone has enough energy to inspect at least a target of Ψ at each round, a possible setting of N is $|\Psi|/|\mathcal{U}|$, which is the maximum number of rounds required to ensure completion of the target covering mission (i.e., total coverage of the target points).

III. Weighted coverage optimization

We now formalize the problem of maximizing the weighted coverage in terms of ILP. We define the set of paths that each drone $u \in \mathcal{U}$ can traverse as $\mathcal{C}_u$. This set is composed of the only tours that include the depot d_u of drone u and that can be traversed within the constraint b_u on energy consumption imposed by the drone battery. We consider trajectory-based variables as follows: $z_p^u(n) \in \{0, 1\}$, to denote the decision to assign path $p \in \mathcal{C}_u$ to drone $u \in \mathcal{U}$ at round $n \in [N]$. We obtain the following Problem 1.

$$\begin{array}{ll}
\max \mathcal{W}(N) \triangleq \sum_{i=1}^{N} w(i) \cdot \delta(i) & (a) \\
s.t. & \\
\delta_i(n) = \sum_{u \in \mathcal{U}, p \in \mathcal{C}_u : i \in p} z_p^u(n), \forall i \in \Psi,\ \forall n \in [N] & (b) \\
\sum_{p \in \mathcal{C}_u} z_p^u(n) \leq 1,\ \forall u \in \mathcal{U},\ \forall n \in [N] & (c) \\
\sum_{n=1}^{N} \delta_i(n) \leq 1, \forall i \in \Psi & (d) \\
z_p^u(n) \in \{0, 1\},\ \forall u \in \mathcal{U}, p \in \mathcal{C}_u,\ \forall n \in [N] & (e) \\
\delta_i(n) \in \{0, 1\},\ \forall i \in \Psi,\ \forall n \in [N] & (f)
\end{array}$$

Problem 1. Weighted Coverage Optimization.

In Problem 1, constraint (b) defines the variables $\delta_i(n)$ to represent coverage of target i at round n, constraint (c) imposes that each drone traverses at most one tour at any given round n. Constraint (d) precludes redundant coverage of the same target points at different rounds. The remaining constraints (e-f) define the binary domain of the decision variables of the problem.

We observe that the number of variables of Problem 1 grows with the number of cycles of a graph. Furthermore, there is no battery constraint in Problem 1 because energy limitations are incorporated in the definition of the sets $\mathcal{C}_u$, $\forall u \in U$, which only include those cyclic tours that drones can realize within their battery limitation b_u. This formulation has the advantage that we can make use of non linear energy models, such as those proposed by Goss et al. [10], while still keeping the optimization model simple and linear.

IV. Weighted coverage optimization under a restricted sets of paths

As we discussed at the end of section III, the number of variables of Problem 1 grows with the number of cyclic

trajectories that can be defined on Ψ. Computing all these trajectories is time-consuming and can become the bottleneck of the entire trajectory planning problem. Moreover, not all trajectories are equally viable due to energy constraints and limited maneuverability of drones. Trajectories traversing urban areas may be precluded due to the presence of obstacles or to privacy and security related law enforcement, such as no-fly zones.

Provisioning a list of viable trajectories for each drone is in itself an interesting research problem. The work by Milan et al. [11] discusses the inherent non linearity of this problem and addresses the generation of spline trajectories by means of non-linear models which take account of mechanical constraints. Trajectories may be generated by means of clustering algorithms, including density based clustering [12], agglomerative hierarchical clustering [13], and spectral clustering [14], just to mention those that may capture the multivariate nature of the target geographical distribution.

In the following we consider a set of *candidate trajectories* for each drone $\chi_u(b_u, d_u)$, as the set of trajectories that drone $u \in \mathcal{U}$ can use ($\chi_u \subseteq C_u$). The definition of this set depends on the battery limitation b_u of drone u, on the adopted energy consumption model, and on the location of the related depot d_u, as well as on the other limitations discussed above.

In our work we assume that the sets of feasible trajectories χ_u is known in advance, for each drone $u \in \mathcal{U}$, and defines the problem instance. We denote $\chi \triangleq \cup_{u \in \mathcal{U}} \chi_u$.

A. ILP formulation of weighted coverage optimization under restricted sets of paths

Notice that the weighted coverage problem under restricted sets of paths χ_u, for $u \in \mathcal{U}$ requires a more general definition of the coverage variables $\delta_i(n)$ to take account of the fact that the optimization problem could select paths with overlapping coverage of some targets. This is because the use of restricted paths does not ensure the existence of a full coverage solution composed of disjoint paths only, and we cannot exclude the selection of paths traversed by different drones and/or at different rounds, containing the same targets.

In Section IV-D we discuss a pruning technique to be adopted to post-process a solution to the task assignment problem based on restricted paths, for removing redundantly covered target points without affecting weighted coverage.

For the purpose of redefining Problem 1 under a restricted set of paths χ, we generalize the definition of the variables $\delta_i(n)$ given in Section II-A to this new setting. Therefore, we define $\delta_i(n) = 1$ if n is the first round in which i is covered by one or more drones, and $\delta_i(n) = 0$ otherwise. The definition of $\tau_i(N)$ is generalized accordingly, so that $\tau_i(N)$ corresponds to the number of the first round at which target i is traversed for the first time.

Naturally, the generalized variables $\delta_i(n)$ and $\tau_i(N)$ will produce the same values described in section II-A when adopted in a scenario where the set of paths is unrestricted.

In agreement with Equation 4, in which we defined $\tau_i(N)$ for the case of unrestricted path sets, we define

$$\tau_i(N) \triangleq \min\{n \in [N] : \exists (p, u) \text{ with } p \in \chi_u, i \in p, z_p^u(n) = 1\},$$

when i is covered, otherwise we set $\tau_i(N) = N + 1$.

Observation IV.1. *The values of $\delta_i(n),\ i \in \Psi$, are uniquely determined from the values of the variables $z_p^u(n),\ u \in \mathcal{U},\ p \in \chi_u$, defining a round based trajectory assignment. Then $\delta_i(n) = 1$ if $n = \tau_i(N)$ and $\delta_i(n) = 0$ otherwise, for $n \in [N]$.*

To define the variables $\delta_i(n)$ in an ILP formulation, we need to formulate linear constraints. This is done by replacing constraint (b), by means of constraints (b1) and (b2) of the following new formulation of the problem of weighted coverage optimization under restricted sets of paths, i.e. of Problem 1.

$$\begin{array}{ll}
\max \mathcal{W}(N) \triangleq \sum_{i=1}^{N} w(i) \cdot \delta(i) & (a) \\
s.t.,\ \forall n \in [N] & \\
\delta_i(n) \geq \sum_{u \in \mathcal{U}} \sum_{p \in \chi_u : i \in p} \frac{z_p^u(n)}{|\mathcal{U}|} - N \cdot \sum_{k=1}^{n-1} \delta_i(k),\ \forall i \in \Psi & (b1) \\
\delta_i(n) \leq 1 - \sum_{k=1}^{n-1} \frac{\delta_i(k)}{(n-1)},\ \forall i \in \Psi & (b2) \\
\sum_{p \in \chi_u} z_p^u(n) \leq 1,\ \forall u \in \mathcal{U} & (c) \\
z_p^u(n) \in \{0, 1\},\ \forall u \in \mathcal{U}, p \in \chi_u & (d) \\
\delta_i(n) \in \{0, 1\},\ \forall i \in \Psi & (e)
\end{array}$$

Problem 2. Weighted coverage optimization with restricted sets of paths.

The metric $\delta(n) = \sum_{i \in \Psi} \delta_i(n)$ represents the *number of newly covered targets* at round n. It generalizes the definition of $\delta(n)$ given in Section II-A for the case of unrestricted sets of paths.

Theorem IV.1 shows that, despite these simplifications, the problem is still NP-hard, hence we look for polynomial time heuristics with good approximation of the optimal, within the constrained sets of trajectories.

B. Efficient approximation algorithms

While the definition of restricted sets of paths χ_u for each drone $u \in \mathcal{U}$ simplifies the trajectory planning problem significantly, as it reduces the number of cycles to be considered as potential trajectories, the problem of selecting the trajectories to maximize target coverage is still NP-hard. When considering $N = 1$ the weighted coverage optimization problem under restricted sets of paths, reduces to the problem of maximizing the number of targets inspected within a unique round with limited battery availability.

Theorem IV.1. *Problem (2) is NP-hard.*

Sketch. Any instance of the Max-Coverage problem can be reduced to an instance of Problem (2) in polynomial time. Consider a collection $\mathcal{S}$ of sets of elements of $\mathcal{T}$, and an integer value m. The Max-Coverage problem requires finding m elements $S_1, S_2, \ldots, S_m$ of $\mathcal{S}$ such that $\cup_{i=1}^{m} S_i$ is maximized. We can build a graph $G = (\Psi, E)$ where the nodes are the elements of $\mathcal{T}$, therefore $\Psi = \mathcal{T}$, and we have $|\mathcal{U}| = 1$ drone u, with depot d, flying for $N = m$ rounds. The restricted set of

paths χ_u associated to drone u is composed by the following sets of nodes: for any $S \in \mathcal{S}$ we add a set $p = S \cup \{d\}$ to the set χ_u, representing a cycle in G, i.e. a cyclic trajectory on the elements of path p. We then set the weights $w(n) = 1$, for each $n = 1, \ldots, m$. Any solution to problem (2) maximizes the number of elements of $\mathcal{T}$ covered by m sets of $\mathcal{S}$. The hardness of Problem (2) derives from the hardness of Max-Coverage. □

The hardness of Problem (2) motivates us to seek for efficient suboptimal solutions with guaranteed performance. To this end, we identify properties of the set of constraints and of the objective functions that allow for easy approximation.

C. Trajectory Assignment as Matroid Optimization

We introduce the following concepts from combinatorial optimization.

Definition IV.1 (Matroid [15])**.** *A matroid $\mathbb{M}$ is a pair (E, I), where E is a finite ground set and $I \subseteq 2^E$ a non-empty collection of subsets of E, with the following properties:*

1) *$\forall A \subset B \subseteq E$, if $B \in I$, then $A \in I$;*
2) *$\forall A,\ B \in I$ with $|B| > |A|$, $\exists x \in B \setminus A$ such that $A \cup \{x\} \in I$.*

Definition IV.2 (Partition Matroid [16])**.** *A matroid $\mathbb{M} = (E, I)$ is a* partition matroid *if E is partitioned into* disjoint *sets $E_1, E_2, \ldots, E_m$ and, for some given integers $b_1, \ldots, b_m$, $0 \leq b_i \leq |E_i|$,*

$$I = \{X \subseteq E \ :\ |E_i \cap X| \leq b_i,\ \ for\ i = 1, 2, \ldots, m\}.$$

Definition IV.3 (Monotone submodular function [15])**.** *Given a finite ground set E and a function $f : 2^E \to \mathbb{R}$,*

- *f is monotone if $\forall A \subset B \subseteq E$, $f(A) \leq f(B)$;*
- *f is submodular if $\forall A \subset B \subseteq E$ and $e \in E \setminus B$, $f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) - f(B)$.*

The reason why we recall the above definitions is that if we prove that our objective function is monotone submodular and our constraint forms a matroid, then we can apply results from combinatorial optimization to define algorithms with known approximation guarantee. In particular, a greedy algorithm that iteratively selects an element of the matroid that maximizes the objective function achieves a near-optimal approximation ratio of $1/2$ [1]. It holds the following theorem.

Theorem IV.2 ([18])**.** *Consider the maximization of a set function $f : 2^E \to \mathbb{R}$ over a collection $I \subseteq 2^E$ of sets. We denote with f^* the optimal value, and with f^g the value achieved by the greedy algorithm. If $\mathbb{M} = (E, I)$ is a matroid and f is monotone and submodular, then $f^g \geq f^*/2$.*

We show that our trajectory assignment problem can be cast as the problem of maximizing a set function under matroid constraints. In fact, under the assumption that each drone u may be activated a given number of rounds at most equal to k_u, with full battery charge b_u, the constraints of Problem 2 define a set of feasible solutions that can be mapped to a partition matroid.

Each drone selects up to k_u cycles on a round basis, for a total of maximum N rounds, where $N = \max_{u \in \mathcal{U}} k_u$. For simplicity, we hereby consider $k_u = N,\ \forall u \in \mathcal{U}$. Notice that the round-based cycle selection does not imply that the drone activities must be synchronized. Nevertheless as a first study we consider the round based weight decreasing in the number of elapsed rounds (namely, targets visited in the first round have the largest weight, regardless of the time taken by their drone to complete the round).

In the following lemma we show that the feasible region of Problem 2 can be mapped to a matroid.

Lemma IV.1. *The solution space of Problem 2 can be modelled by means of the pair $M = (\mathcal{E}, \mathcal{I})$, defined as follows:*

- *$\mathcal{E}$ is a ground set, $\mathcal{E} = [N] \times \chi$ composed of all the cyclic trajectories for any vehicle $u \in \mathcal{U}$, each cloned N times, one for each round: $\mathcal{E} = \{(i, p) : i \in [N], p \in \chi\}$. Such a ground set is partitioned into $N \cdot |\mathcal{U}|$ disjoint subsets $\mathcal{E}_{i,u} = \{(i, p) :\ p \in \chi_u\},\ \forall i \in [N]$ and $u \in \mathcal{U}$.*
- *$\mathcal{I}$ is nonempty collection of subsets of $\mathcal{E}$ defined as $\mathcal{I} = \{S \subseteq \mathcal{E}\ :\ |S \cap \mathcal{E}_{j,u}| \leq 1, \forall (j, u) \in [N] \times \mathcal{U}\}$.*

The pair $M = (\mathcal{E}, \mathcal{I})$ is a matroid (a partition matroid).

Proof. We first establish a one to one mapping between any feasible solution of Problem 2, identified by the variable vectors $\bar{\delta}$ and $\bar{z}$, and a set $S = S(\bar{\delta}, \bar{z}) \in \mathcal{I}$, then we show that M is a matroid.

(One to one mapping.) Given a solution of Problem 2, we can build the corresponding set $S \in \mathcal{I}$ consisting of all the elements (n, p), with $p \in \chi_u$, for which $z_p^u(n) = 1$. Thanks to constraint (c) of Problem 2, there will be at most one element in $S \cap \mathcal{E}_{n,u}$, for each $n \in [N]$ and $u \in \mathcal{U}$. The opposite is also true. If we take any set $S \in \mathcal{I}$, this will have at most one element in each partition subset $\mathcal{E}_{n,u}$ of the matroid M. More specifically, if $S \cap \mathcal{E}_{n,u} = \emptyset$, no trajectory of χ_u is assigned to drone u at round n and $z_p^u(n) = 0$, for all $p \in \chi_u$ and for the selected drone $u \in U$ at round n. If instead the set $S \cap \mathcal{E}_{n,u}$ is not empty, by the definition of the partition matroid M, S will contain only one element (n, p^*) with $p^* \in \chi_u$. This corresponds to assigning $z_{p^*}^u(n) = 1$, and $z_p^u(n) = 0,\ \forall p \neq p^*$. The variables $\bar{\delta}$ are defined according to the setting of $\bar{z}$ in agreement with constraints (b1-b2) of Problem 2, or equivalently using the definition introduced in Observation IV.1.

(M is a matroid.) In order to prove that the pair $M = (\mathcal{E}, \mathcal{I})$ is actually a matroid, we must verify the two conditions of Definition IV.1 on M. For condition (1), let us consider A and B, with $A \subset B \subseteq \mathcal{E}$. A and B are sets of indexed cyclic trajectories, for selected vehicles and round indexes. As a consequence, if B belongs to the solution space $\mathcal{I}$ then A also belongs to $\mathcal{I}$ (A represents a solution with fewer trajectories than in solution B). For condition (2), given any A and B in $\mathcal{I}$, such that $|B| > |A|$, we can write $|B| =$

[1]The best approximation ratio is $(1 - 1/e)$, achieved by a more complex algorithm based on continuous relaxation and rounding proposed by Calinescu et al. in [17].

$\sum_{(i,u)\in[N]\times U} |B \cap \mathcal{E}_{i,u}|$ and $|A| = \sum_{(i,u)\in[N]\times U} |A \cap \mathcal{E}_{i,u}|$, because the sets $\mathcal{E}_{i,u}$ constitute a partition of $\mathcal{E}$.

We can see that there exists at least an element $x = (i', u') \in [N]\times\mathcal{U}$ such that $|B\cap\mathcal{E}_{i',u'}| = 1$ and $|A\cap\mathcal{E}_{i',u'}| = 0$. Therefore the element x that satisfies the second condition of Definition IV.1 is such that $\{x\} = B\cap\mathcal{E}_{i',u'}$. Indeed, $x \in B\setminus A$ and $A \cup \{x\}$ is still a solution in $\mathcal{I}$, i.e., $A \cup \{x\} \in \mathcal{I}$. It is straightforward to verify that the definition of the matroid M is consistent with Definition IV.2, hence M is a partition matroid. □

In the following, with p we interchangeably denote either the cycle or the set of nodes traversed by the cycle.

Lemma IV.2. *Problem 2 can be cast as a set function optimization on the partition matroid M defined in Lemma IV.1, where the objective function is*

$$\tilde{f}(S) \triangleq \sum_{n=1}^{N} w(n) \cdot |\cup_{(n,p)\in S} p \setminus \cup_{(i,p)\in S:i<n} p|. \quad (7)$$

Proof. We consider the objective function of Problem 2 and express it in terms of the elements of matroid M. We recall that the objective function of Problem 2 is $\mathcal{W}(N) = \sum_{n=1}^{N} w(n)\cdot\delta(n)$. According to Observation IV.1, $\delta_i(n) = 1$ if $n = \tau_i(N)$ and $\delta_i(n) = 0$ otherwise, for $n \in [N]$. Therefore the value of $\delta(n)$ is the cardinality of the set of targets visited at round n for the first time. Consequently, in terms of the matroid M we can write

$$\delta(n) = |\cup_{(n,p)\in S} p \setminus \cup_{(i,p)\in S:i<n} p|. \quad (8)$$

It follows that $\tilde{f}(S) = \sum_{n=1}^{N} w(n)\cdot\delta(n) = \mathcal{W}(N)$. □

Lemmas IV.1 and IV.2 show that the weighted coverage problem can be cast as the optimization of a set function over a matroid constraint. In the following we prove that this set function is monotone submodular.

Theorem IV.3. *Function $\tilde{f} : \mathcal{I} \to \mathbb{R}$ is a monotone submodular function on the matroid $M = (\mathcal{E}, \mathcal{I})$.*

Proof. We discuss monotonicity and submodularity separately.
Monotonicity proof: Let us consider $A, B \in \mathcal{I}$ such that $A \subset B$. We must show that $\tilde{f}(B) \geq \tilde{f}(A)$.

Given a solution S we define with $\tilde{\tau}_S(i)$ the value of

$$\tilde{\tau}_S(i) \triangleq \min\{n \in [N] : \exists (n,p) \in S \text{ with } p \in \chi, i \in p\},$$

if i is covered, and $\tilde{\tau}_S(i) = N + 1$ otherwise. Notice that, given Lemma IV.2, $\tilde{f}(S)$ can be rewritten as

$$\tilde{f}(S) = \sum_{i\in\Psi} w(\tilde{\tau}_S(i)) \quad (9)$$

Since $A \subset B$, it holds $\tilde{\tau}_A(i) \geq \tilde{\tau}_B(i),\ \forall i \in \Psi$. As a consequence $w(\tilde{\tau}_A(i)) \leq w(\tilde{\tau}_B(i)),\ \forall i \in \Psi$, because $w(n)$ is a non-increasing function of n. Therefore, applying Equation 9, we obtain

$$\tilde{f}(B) = \sum_{i\in\Psi} w(\tilde{\tau}_B(i)) \geq \sum_{i\in\Psi} w(\tilde{\tau}_A(i)) = \tilde{f}(A).$$

Submodularity proof:
To prove the submodularity of $\tilde{f}$, according to Definition IV.3 we must prove that $\forall A \subset B \subseteq \mathcal{E}$ and $e \in \mathcal{E}\setminus B$, $\tilde{f}(A\cup\{e\}) - \tilde{f}(A) \geq \tilde{f}(B\cup\{e\}) - \tilde{f}(B)$.

Let $e = (n_e, p_e)$ be the generic element of $\mathcal{E}\setminus B$. For a generic solution S, applying Equation 9, we obtain

$$\tilde{f}(S\cup\{e\}) - \tilde{f}(S) = \sum_{i\in p_e} \max\{w(n_e) - w(\tilde{\tau}_S(i));\ 0\}.$$

As we observed for monotonicity, $A \subset B$ implies $\tilde{\tau}_A(i) \geq \tilde{\tau}_B(i)$, and consequently $w(\tilde{\tau}_A(i)) \leq w(\tilde{\tau}_B(i)),\ \forall i \in \Psi$, as we recall that $w(n)$ is non-increasing in n. Therefore $\tilde{f}(A\cup\{e\}) - \tilde{f}(A) = \sum_{i\in p_e} \max\{w(n_e) - w(\tilde{\tau}_A(i)), 0\} \geq \sum_{i\in p_e} \max\{w(n_e) - w(\tilde{\tau}_B(i)), 0\} = \tilde{f}(B\cup\{e\}) - \tilde{f}(B)$, which proves submodularity.

□

As a consequence of this observation we can apply Theorem IV.2 to define a greedy approach with $1/2$-approximation of the optimal.

D. Algorithm Greedy and Prune (GaP) for weighted coverage

In agreement with the discussion of Section IV-C on the constant factor approximation of the greedy approach, we propose the adoption of an enhanced greedy algorithm, called Greedy and Prune (GaP) described in Algorithm 1. This algorithm provides a preliminary greedy trajectory assignment phase, followed by a pruning step which removes redundant target points from the selected trajectories, without affecting weighted coverage.

Algorithm 1: Greedy and Prune (GaP)

Input: A set of drones $\mathcal{U}$, a family of candidate cyclic trajectories $\{\chi_u : u \in \mathcal{U}\}$, a number of rounds N, and a weight function $w : [N] \to \mathbb{R}$
Output: An assignment of $|\mathcal{U}|$ ordered lists of cycles $\bar{\mathcal{L}}$, where $\mathcal{L}_u = (p_1^u, \ldots, p_N^u) \in \chi_u$ to each drone $u \in \mathcal{U}$

$\mathcal{R}_{\texttt{assigned}} = (r_1, r_2, \ldots, r_{|\mathcal{U}|}) \leftarrow (1, 1, \ldots, 1)$
$\mathcal{V} \leftarrow \emptyset,\ \bar{\mathcal{L}} \leftarrow \emptyset^{|U|}$
while $\exists u \in \mathcal{U}$ *s.t.* $r_u \leq N \wedge \cup_{p\in\chi_u} p \setminus \mathcal{V} \neq \emptyset$ **do**
  $(u^*, p^*) = \arg\max_{u\in\mathcal{U}, p\in\chi_u:\ r_u\leq N} w(r_u)\cdot|p\setminus\mathcal{V}|$
  $\chi_{u^*} = \chi_{u^*} \setminus p^*$
  $r_{u^*} = r_{u^*} + 1$
  $\mathcal{V} = \mathcal{V} \cup \{p^* \cap \Psi\}$
  append p_{u^*} to $\mathcal{L}_{u^*}$
prune of redundantly covered targets
return $\bar{\mathcal{L}}$

Algorithm 1 iteratively builds an assignment of paths for the drones, along the available rounds, by selecting at each step the next tour that maximizes the weighted coverage. When the set of feasible paths χ is restricted, the solution may contain overlapping trajectories, namely trajectories assigned to distinct drones or to the same drone at distinct rounds, containing at least a common target point. In such cases, the pruning algorithm removes this redundancy without affecting weighted coverage. In particular, it removes a target t from a path p if exist an earlier round with a path p' s.t. $t \in p'$. If

overlaps are on the same round the algorithm leaves the target only on a single path while pruning the others.

Clearly, as the algorithm removes only redundant targets and leaves them in earliest rounds, it does not affect the weighted coverage.

Corollary IV.1. *Algorithm 1 for the optimization of the objective function $\tilde{f}(S)$ defined in Equation 7 achieves $1/2$-approximation of the optimal solution to the* weighted coverage maximization problem *under restricted sets of trajectories*[2].

That is, the weighted coverage obtained by the trajectories selected by Algorithm 1 is at least half of the maximum weighted coverage that can be obtained by optimally choosing the drone trajectories within the restricted feasible sets χ_u.

We recall that the formulation of weighted coverage generalizes the other coverage metrics described in Section II-A. The total coverage metric of Equation 2 and the accumulative coverage metric of Equation 3 can be obtained by setting the weight parameters $w(k)$, $k = 1, \ldots, N$, to specific values, such that $w(k)$ is non increasing with k. Therefore Theorem IV.3 is still valid for assessing monotonicity and submodularity of the total and of the accumulative coverage metric, which implies that the Greedy and Prune approach of Algorithm 1 provides a constant factor approximation of $1/2$ for these metrics.

Theorem IV.4. *The complexity of Algorithm 1 is $\mathcal{O}(|\mathcal{U}| \cdot |\mathcal{X}| \cdot |\Psi| \cdot N)$*

Proof. The greedy assignment of tours runs in $\mathcal{O}(|\mathcal{U}| \cdot |\mathcal{X}| \cdot |\Psi| \cdot N)$. In fact, it iterates in the while loop at most $N \cdot |\mathcal{U}|$ times, because at each iteration at least a path is chosen for a drone u, and the related drone index r_u is increased. Its body loop can easily be implemented to run in $\mathcal{O}(|\mathcal{X}| \cdot |\Psi|)$, which is the time of dominant instruction at line 4. In fact, assuming that the weighted function $w(x)$ is constant ($\mathcal{O}(1)$), the $\arg\max$ function runs in $\mathcal{O}(|\mathcal{X}| \cdot |\Psi|)$ while the other instructions in $\mathcal{O}(|\Psi|)$. The pruning step can be implemented in polynomial time in $\mathcal{O}(|\mathcal{U}| \cdot |\Psi| \cdot N)$. In fact, the pruning step iterates over the tours executed by drones in progressive rounds and updates a list of visited targets so that it can remove redundant targets already in the list. ∎

V. Performance evaluation

In the following we give a performance evaluation of the algorithms discussed in this paper.

We recall that both Problem 2 and the GaP algorithm aim at optimizing weighted coverage, in its general formulation given in Equation 1, for any setting of the weight function $w(k)$, $k = 1, \ldots, N$. In this section we consider the two practical coverage metrics introduced in Section II-A, namely *total coverage* $\Delta(N)$ defined by Equation 2, and *accumulative coverage* $\mathcal{A}(N)$ defined by Equation 3. Total and accumulative coverage are instances of weighted coverage obtained by setting $w(k) = 1$ for total coverage in N rounds, and $w(k) = N - k + 1$ for accumulative coverage in N rounds, $\forall k = 1, \ldots, N$.

In this section, we refer to the two variants of the optimal solution with the names TC-OPT and AC-OPT, for total coverage and accumulative coverage optimization, respectively. Similarly, the GaP algorithm variants corresponding to the two objectives are called TC-GaP and AC-GaP.

In the following we compare the aforementioned approaches against a previous proposal, introduced by Kim et al. [7], hereby referred to with the name Tree Cover Trajectory Planning Algorithm (TCTPA) as it is based on the creation of tree covers rooted at the depot locations. In Section V-C we discuss this previous approach, explaining its limitations when applied to the scenario considered in this paper, and showing the way in which they are addressed.

A. Energy model

Although the approaches analyzed in this paper work under any energy consumption model, such as the non linear one proposed by Goss et al. in [10], in the experiments we assume a linear model of energy consumption according to which the energy consumed by a drone is proportional to the traversed distance and length of hovering time.

Given any two points $i, j \in \Psi$ we denote with ℓ_{ij} the distance that a drone needs to traverse to move from point i to point j. With an abuse of notation we say that $i \in p$ if target point i is traversed by p and that $(i, j) \in p$ if the target points i, j are traversed by p in a sequence.

Concerning hovering, if ϕ_i is the necessary time to inspect the target point i, the energy consumption related to target inspection is assumed to be proportional to ϕ_i. We also assume $\phi_{d_u} = 0$ for each depot d_u of any drone $u \in \mathcal{U}$.

We model the energy consumption of a drone along its path as the summation of terms related to both inspected target points and traversed edges. Therefore the energy expenditure for traversing a path p, including hovering on the traversed target points, is $E(p) \triangleq a \cdot \sum_{i \in p} \phi_i + b \cdot \sum_{(i,j) \in p} \ell_{ij}$, where a and b are dimensional coefficients which reflect the energy consumption in energy units (eu) for a second of inspection of a target and a meter of flight, respectively. In all the experiments we consider $a = 1$eu/sec and $b = 1$eu/m.

B. Restricted sets of trajectories

While our algorithms work independently of the specific technique adopted to generate the restricted sets of paths χ_u for each drone, in this experimental section we adopt a candidate tour generation technique based on the Christofides TSP approximation algorithm [19]. In particular for each drone u we calculate a TSP approximated solution over the set of points $\Psi \cup \{d_u\}$ and therefore we determine an ordered sequence of the target points $s =< v_1, v_2, \ldots, v_{|\Psi|} >$. Then we consider all the subtours formed traversing any subsequence $< d_u, v_i, \ldots, v_j, d_u >$ of s for any choice of v_i, v_j, with $i \leq j$,

[2] A relaxation to the continuous and consequent random rounding of the problem 2 conducted with the technique described in [17] guarantees an approximation factor of $1 - 1/e$.

and exclude those that do not meet the energy limitations b_u of the drone battery, according to the energy consumption model provided in Section V-A. By means of this technique, we generate at most $|\Psi| \cdot (|\Psi| - 1)/2$ subtours for each drone.

C. Tree Cover Trajectory Planning Algorithm (TCTPA)

In [7] the authors address the problem of planning multiple drone trajectories to inspect a number of target points positioned in an area of interest. The primary goal is to ensure complete coverage of the target points, while minimizing the mission completion time, i.e. the time at which the last drone returns to its depot.

TCTPA builds a graph whose vertices coincides with the target points and depots. Then, it builds a set of $|\mathcal{U}|$ balanced tree covers rooted at the depot stations, and convert each tree to a drone trajectory by means of a technique inspired to the Christofides algorithm for the Travelling Salesman Problem [19].

The authors consider a multiple depot scenario, but assume each device has unlimited energy, therefore the drone squad is always able to inspect all the targets in a unique round. Unlike this previous approach, our algorithms work in a more general scenario, where the flight autonomy of each vehicle is constrained because of the limited energy availability of each drone. To have fair comparisons, we made TCTPA work under limited energy availability, letting drones fly in multiple rounds N, possibly depleting their batteries before the completion of the target inspection mission. To make TCTPA produce N tours for each drone, we let it work as if the number of available drones were $N \cdot |\mathcal{U}|$, considering N virtual clones for each physical drone, each with its logically replicated depot, and we let each physical drone traverse one of its clones' tours at each round, with recharging and maintenance between consecutive rounds.

It must be underlined that the objective function of TCTPA is related to ours, as it aims at covering *all the targets* in minimum time. Our approach is to privilege solutions in which *most of the targets* are inspected in the early rounds of flight.

Figures 1, 2, and 3, show the execution of AC-OPT, AC-GaP, and TCTPA, respectively, in a field of interest of 600m $\times$600m. Three drones are launched from different depots located on the bottom border of the field, and are required to inspect 50 target points randomly generated in the field area. Each drone speed is 8m/s, the drone battery is set to allow a continuous flight of about 2000m in a single round. The inspection of each target requires 5 sec of hovering time.

The figure shows that AC-OPT and AC-GaP definitely succeed in designing tours which, under the battery limitation, contain more target points in the early rounds. The first round tours produced by AC-OPT and AC-GaP for the three drones are almost the same and contain a number of targets which is much higher than the average of number of targets in the successive rounds. Both these algorithms complete the mission of inspecting the 50 targets in just two rounds.

By contrast, TCTPA produces a more uniform distribution of targets in the exploration time, and ends up requiring one additional round for the two drones on the left of the figure.

This figure shows that, in the addressed scenario, TCTPA performs worse than AC-OPT and AC-GaP in terms of accumulative coverage as it does not give priority to target coverage in the early rounds. It also shows that TCTPA requires a longer completion time (in number of rounds) than the other two algorithms, which is its specific objective. Notice that in emergency critical settings, the time interval between consecutive rounds is devoted to recharging, maintenance and data offloading, which are time consuming activities. Hence a solution which minimizes the number of rounds, together with early coverage, such as AC-GaP, is preferable in these settings.

D. Performance comparisons

In this section we give performance comparisons through simulations, evaluating several key performance metrics under different settings. Where not otherwise stated, the experiments consider a drone fleet of 5 vehicles, monitoring a squared area of interest of 2000m $\times$2000m with randomly located target points requiring 5sec of hovering time, with depots uniformly deployed at 100m of distance, out of the area of interest, to reflect the inaccessibility of the monitored field. Each of the drones executes a maximum number of $N = 20$ rounds of flight, with speed of 8m/sec, under a uniform battery constraint b for each drone, which allows a flight of 7200m at constant speed (for 15min). Each point in the plots is the result of 30 runs, and the error bars denote one standard deviation of uncertainty.

1) Progressive coverage percentage: Progressive total coverage percentage is the percentage of target points covered up to a given round (according to Equation 2), evaluated in progressive rounds. A progressive coverage percentage of $x\%$ at round n reflects a situation in which $x\%$ of the target points have been covered in any round from the first to the n-th.

Fig. 4 compares the two optimal and the two greedy approaches with TCTPA, in a setting with 225 target points, in terms of progressive total coverage achieved round by round. It is worth noting that the area under the plot until round n is equal to the accumulative coverage at round n divided by the total number of target points. The figure shows that, after AC-OPT, the algorithm that performs the best, both in total coverage and in accumulative coverage, is AC-GaP, with an excellent approximation of the optimal.

As a matter of fact, both the algorithms achieve very high coverage in the early rounds of execution because they adopt decreasing weights $w(n)$ as prescribed by Equation 1 to give decreasing priority to the coverage achieved at any round $n = 1, \ldots, N$.

It is interesting to notice that in terms of progressive total coverage, TC-GaP performs better than TC-OPT, until round 20 in which both algorithms achieve 100% of total coverage. This is due to the fact that by setting the weight parameters as $w(n) = 1$, when the algorithms are run for $N = 20$ rounds,

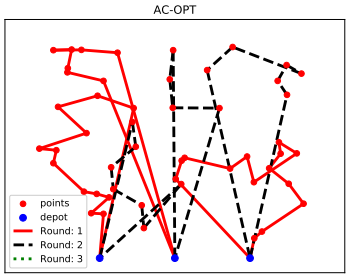

Fig. 1. Consecutive flights under AC-OPT.

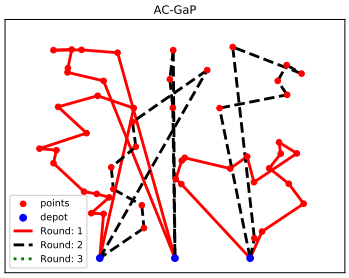

Fig. 2. Consecutive flights under AC-GaP.

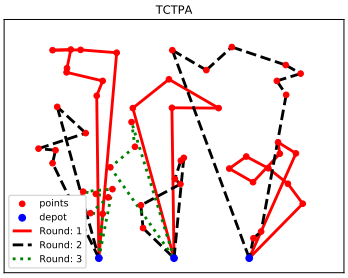

Fig. 3. Consecutive flights under TCTPA.

TC-GaP and TC-OPT make any round as important as the others in terms of coverage. None of these algorithms aim at obtaining high coverage in the early rounds, but only at the 20-th round. An algorithm that obtains zero coverage at the the first 19 rounds and obtains complete coverage at round 20, has the same value of total coverage $\Delta(N = 20)$ as another algorithm that obtains complete coverage since the very first round. Finally, we observe that TCTPA has a very good performance in this setting in terms of total coverage after round 3. Nevertheless in the early rounds TCTPA performs poorly with respect to all the other algorithms, and in particular, in the first round it achieves only less than half the coverage obtained by our AC-GaP.

Figure 5 shows coverage percentage achieved at any round by AC-GaP and TCTPA in settings with a number of target points growing from 100 to 650. Due to the large size of the problem instances, we do not include the optimal policies in this figure, nor do we include the TC-GaP heuristics because its purpose is not to obtain high coverage in the early rounds but only within a given round N. The figure compares the performance of TC-GaP with that of TCTPA and shows that also in this setting, for any round $n = 1, \ldots, N$ the coverage obtained by TC-GaP is alway higher than with TCTPA.

These results match the analysis of Figure 6 which shows, by varying the number of target points, the round at which the algorithms achieve the different coverage quartiles. The box-with-whiskers plot denotes in fact the round at which the algorithms achieve the different quartiles of total coverage. The figure highlights that all the quartiles are met in a lower round for AC-GaP, demonstrating its superiority with respect to TCTPA in total coverage and in accumulative coverage per round.

2) Average inspection delay: The average inspection delay, in number of rounds, has been introduced with Equation 5. With Theorem II.1 we proved that AC-OPT is optimal both for accumulative coverage and for average inspection delay in rounds. Figures 7 compares our algorithms in terms of this metric, for a growing number of target points and confirms the optimality of AC-OPT. Due to the high computational time of the optimal solution (discussed in Section V-D4), we only compare the optimal AC-OPT with the heuristic algorithms AC-GaP and TCTPA for small problem instances in Figure 7, whereas when the size of the problem instance grows, as shown in Figure 8, we compare the two heuristics AC-GaP and TCTPA alone. As explained by Theorem II.1, AC-OPT performs always better than the other algorithms in terms of average inspection delay in rounds. Indeed, Figures 7 and 8 show that the inspection delay in rounds of algorithm TCTPA diverges from the one of AC-OPT and AC-GaP, showing that the latter algorithms excel in achieving early monitoring of target points. Figure 9 shows a different experiment in which we still consider the average inspection delay, but we measure it in seconds, by considering zero waiting time between two consecutive rounds, in order not to penalize TCTPA with this setting. The figure shows that also in these terms AC-GaP performs better than TCTPA, due to the fact that TCTPA prioritizes the completion time of the algorithm, which is the inspection time of the last target point, and does not care about the inspection time of the other targets, possibly increasing their inspection delays. Despite the fact that mission completion time is the specific objective of TCTPA, Figure 10 shows that for the considered scenario, TCTPA always completes the mission with almost twice the number of rounds than AC-GaP.

3) Energy cost: Figure 11 shows the total flight time over all the 20 rounds, considering all 5 drones, for the three heuristics TC-GaP, AC-GaP and TCTPA. Notice that since we use the linear energy model discussed in Section V-A, this measure directly reflects the energy expenditure of drones.

The graph shows that the difference between the algorithms in terms of total energy expenditure is negligible, although in small favor of the greedy approaches.

This explains that the superiority of the greedy approaches resides on the way drone visits are scheduled along the rounds and within single rounds, rather than in the minimization of the traversed paths.

4) Computation time: We conclude the performance evaluation of the discussed algorithms by comparing them in terms of computation time as shown in Figure 12. In order to stress the optimal algorithm while keeping the problem size small enough to be executed in a reasonable time to have enough runs for a reliable experimental evaluation, we designed a new setting with 10 drones, and a growing number of target points, from 50 to 300.

For this and for all the previous experiments we adopted a home made simulator, programmed in Python, and we ran

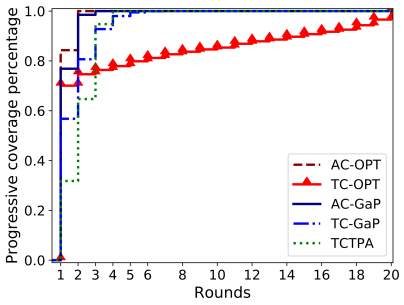

Fig. 4. Progressive coverage (225 target points).

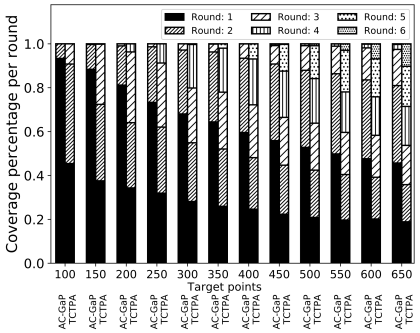

Fig. 5. Coverage percentage (varying nr. of points).

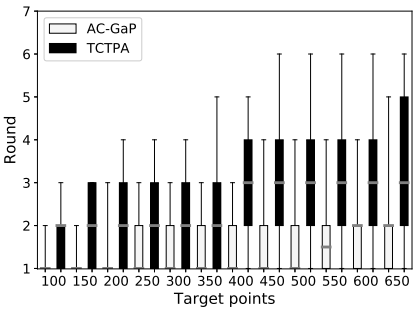

Fig. 6. Quartiles of progressive coverage.

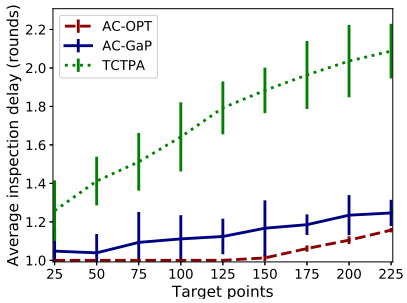

Fig. 7. Average inspection delay in rounds.

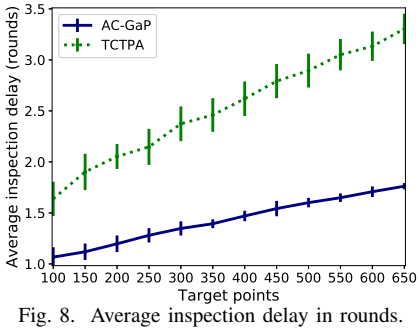

Fig. 8. Average inspection delay in rounds.

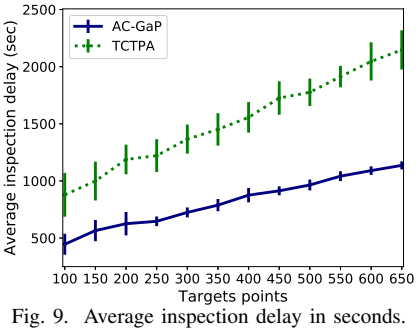

Fig. 9. Average inspection delay in seconds.

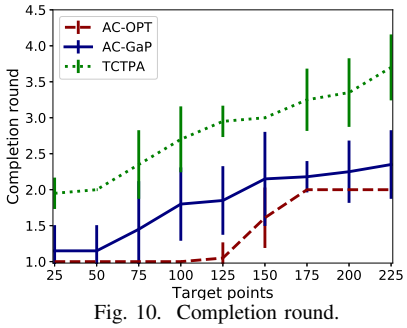

Fig. 10. Completion round.

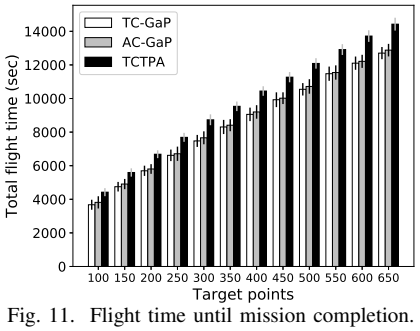

Fig. 11. Flight time until mission completion.

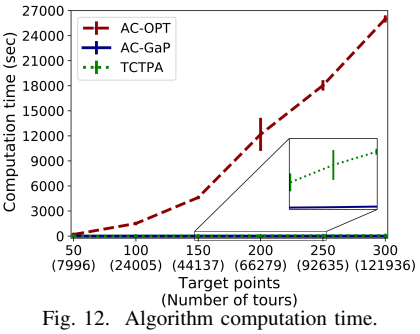

Fig. 12. Algorithm computation time.

the ILP optimization problems using the Gurobi optimizer [20]. We ran all the experiments on a Lenovo X3550 M5, with 2 CPUs Intel(R) XEON(R) E5-2650 @ 2.20GHz with 16 cores and 32 GB RAM [21]. This experiment shows that the computation time of the optimal solution grows prohibitively high also for this small a setting. With 300 target points the computation time is already in the order of 7 hours. Such a high processing time makes the optimal solution inapplicable to emergency critical applications which require prompt intervention of the aerial network and do not tolerate delays. The figure also contains a zoomed plot which shows the difference between TCTPA and AC-GaP in terms of running time. Though negligible, such a performance parameter is also in favor of AC-GaP.

VI. Conclusions

The work addresses the problem of assigning location based tasks to a fleet of drones in an emergency critical scenario, where early visit of target locations is of uttermost importance. Task assignment translates into trajectory planning, where the multiple drones visit targets points in a sequence, within the energy constraints imposed by the batteries. We propose a novel metric, called weighted coverage, to measure the capability of a trajectory planning algorithm to provide early target inspection, and formulate a related optimization problem. We show that weighted coverage generalizes traditional metrics of coverage, as well as a new notion of accumulative coverage which is directly related to early coverage capabilities. Due to the NP-hardness and high computation time of the optimal approach, we propose a new polynomial time algorithm with a constant factor approximation of $1/2$ of the optimal. We study the proposed algorithm by means of extensive simulations, showing that it outperforms previous approaches in terms of all the metrics of coverage, average inspection delay, energy consumption and computation time.

REFERENCES

[1] N. Strochlic, "Surprising ways drones are saving lives, National Geographic," 2017. [Online]. Available: https://www.nationalgeographic.com/magazine/2017/06/explore-drones-for-good/

[2] M. McNabb. (2018) Drones for good: UNICEF is calling on drone operators for vaccine delivery. [Online]. Available: dronelife.com/2018/06/14/drones-for-good-unicef-is-calling-on-drone-operators-for-vaccine-delivery/

[3] FAO, "E-Agriculture in action: drones for agriculture," 2018. [Online]. Available: http://www.fao.org/3/I8494EN/i8494en.pdf

[4] J. Fakcharoenphol, C. Harrelson, and S. Rao, "The k-traveling repairman problem," in *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '03. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2003, pp. 655–664.

[5] C. S. Martin and M. R. Salavatipour, "Minimizing latency of capacitated k-tours," *Algorithmica*, vol. 80, no. 8, pp. 2492–2511, Aug 2018. [Online]. Available: https://doi.org/10.1007/s00453-017-0337-x

[6] P. Vansteenwegen, W. Souffriau, and D. V. Oudheusden, "The orienteering problem: A survey," *European Journal of Operational Research*, vol. 209, pp. 1–10, 2011.

[7] D. Kim, L. Xue, D. Li, Y. Zhu, W. Wang, and A. O. Tokuta, "On theoretical trajectory planning of multiple drones to minimize latency in search-and-reconnaissance operations," *IEEE Transactions on Mobile Computing*, vol. 16, no. 11, pp. 3156–3166, Nov 2017.

[8] H. Binol, E. Bulut, K. Akkaya, and I. Guvenc, "Time optimal multi-uav path planning for gathering its data from roadside units," *IEEE VTC-Fall*, 01 2018.

[9] DJI, "Phantom 4 PRO," 2018. [Online]. Available: https://www.dji.com/phantom-4-pro

[10] K. Goss, R. Musmeci, and S. Silvestri, "Realistic models for characterizing the performance of unmanned aerial vehicles," in *26th International Conference on Computer Communication and Networks, ICCCN*. Vancouver, BC, Canada, July 31 - Aug. 3, 2017, pp. 1–9.

[11] M. B. Milam, R. Franz, and R. Murray, "A new computational approach to real-time trajectory generation for constrained mechanical systems," in *Proceedings of the IEEE Conference on Decision and Control (CDC)*, 2000.

[12] H.-P. Kriegel, P. Krger, J. Sander, and A. Zimek, "Density-based clustering," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 1, no. 3, pp. 231–240, 2011.

[13] I. Davidson and S. S. Ravi, "Agglomerative hierarchical clustering with constraints: Theoretical and empirical results," in *Knowledge Discovery in Databases: PKDD 2005*, A. M. Jorge, L. Torgo, P. Brazdil, R. Camacho, and J. Gama, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 59–70.

[14] F. R. Bach and M. I. Jordan, "Learning spectral clustering," in *Advances in neural information processing systems*, 2004, pp. 305–312.

[15] J. Lee, *A First Course in Combinatorial Optimization*. Cambridge University Press, 2004.

[16] E. Lawler, *Combinatorial optimization - networks and matroids*. New York: Holt, Rinehart and Winston, 1976.

[17] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, "Maximizing a submodular set function subject to a matroid constraint," in *IPCO*, June 2007.

[18] M. Fisher, G. Nemhauser, and L. Wolsey, "An analysis of approximations for maximizing submodular set functions – II," *Math. Prog. Study*, vol. 8, pp. 73–87, 1978.

[19] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem," Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, Tech. Rep., 1976.

[20] "Gurobi," Last accessed on January 2019. [Online]. Available: http://gurobi.com

[21] "Lenovo x3550-m5," Last accessed on January 2019. [Online]. Available: https://www.lenovo.com/it/it/data-center/servers/racks/System-x3550-M5/p/77XS7HV7V38