Les Cahiers du GERAD ISSN: 0711–2440

The Covering-Assignment Problem for swarm-powered ad-hoc clouds: A distributed 3D mapping use-case

L. R. Costa, D. Aloise,
L. G. Gianoli, A. Lodi

G–2020–28

April 2020

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2020
– Bibliothèque et Archives Canada, 2020

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2020
– Library and Archives Canada, 2020

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

The Covering-Assignment Problem for swarm-powered ad-hoc clouds: A distributed 3D mapping use-case

Leandro R. Costa [a,b]

Daniel Aloise [a,c]

Luca G. Gianoli [d]

Andrea Lodi [a,b]

[a] *GERAD, Montréal (Québec), Canada, H3T 2A7*

[b] *Department of Mathematics and Industrial Engineering, Polytechnique Montréal (Québec) Canada, H3C 3A7*

[c] *Department of Computer Engineering, Polytechnique Montréal, Polytechnique Montréal (Québec) Canada, H3C 3A7*

[d] *Humanitas Solutions, Montréal (Québec) Canada, H2Y 1N3*

leandro.costa@polymtl.ca
daniel.aloise@gerad.ca
andrea.lodi@polymtl.ca

April 2020
Les Cahiers du GERAD
G–2020–28

Abstract: The popularity of drones is rapidly increasing across the different sectors of the economy. Aerial capabilities and relatively low costs make drones the perfect solution to improve the efficiency of those operations that are typically carried out by humans (e.g., building inspection, photo collection). The potential of drone applications can be pushed even further when they are operated in fleets and in a fully autonomous manner, acting de facto as a drone swarm. Besides automating field operations, a drone swarm can serve as an ad-hoc cloud infrastructure built on top of computing and storage resources available across the swarm members and other connected elements. Even in the absence of Internet connectivity, this cloud can serve the workloads generated by the swarm members themselves, as well as by the field agents operating within the area of interest. By considering the practical example of a swarm-powered 3D reconstruction application, we present a new optimization problem for the efficient generation and execution, on top of swarm-powered ad-hoc cloud infrastructure, of multi-node computing workloads subject to data geolocation and clustering constraints. The objective is the minimization of the overall computing times, including both networking delays caused by the inter-drone data transmission and computation delays. We prove that the problem is NP-hard and present two combinatorial formulations to model it. Computational results on the solution of the formulations show that one of them can be used to solve, within the configured time-limit, more than 50% of the considered real-world instances involving up to two hundred images and six drones.

Keywords: Cloud Computing, swarm, 3D reconstruction, workload optimization

Acknowledgments: The authors would like to thank all support from Humanitas Solutions and the Canada Excellence Research Chair in Data Science for Real-Time Decision-Making. This research was enabled in part by the support provided by SHARCNET (www.sharcnet.ca) and Compute Canada (www.computecanada.ca).

1 Introduction

An Unmanned Aerial Vehicle (UAV) — otherwise commonly known as drone — is a flying vehicle whose weight can vary, according to the targeted applications, from a few hundreds grams to hundreds of kilos. The popularity of drones is rapidly increasing across the different sectors of the economy. Aerial capabilities and relatively low CAPEX/OPEX costs make UAVs the perfect solution to improve the efficiency of those operations that are typically carried out by humans, e.g., building inspection, photo collection, area surveillance, etc.

In normal operations, drones are remotely controlled by human pilots through wireless remote controls. However, by setting the UAV autopilot in *auto off-board* mode, a drone can operate in a fully autonomous manner by following the inputs generated by an on-board flight computer directly connected to the autopilot. This capability can be leveraged to create fleets of autonomous UAVs collaborating to fulfill the desired missions. This is achieved by installing a collaborative drone application on each on-board flight computer of the fleet and by connecting these latter on the same wireless network.

The possibility of organizing drones in fleets of autonomous and collaborating entities naturally attracted the attention of swarm robotics scientists [1]. Swarm robotics studies how to reproduce, with the help of artificial agents, those swarming behaviors typically observed in nature — in ant colonies, bee swarms, bird flocks, etc. Swarm behaviors have the potential to revolutionize the world of robotized applications — including UAV applications — because of their promise of jointly achieving maximum performance and maximum resilience through the power of distributed interactions that do not require any form of centralized supervision.

Swarming UAVs can be deployed to support operations in a long list of domains [2], including forestry [3, 4, 5], archaeology and architecture [6, 7, 8, 9, 10], environment monitoring [11, 12, 13, 14, 15], emergency management [16, 17, 18] and precision agriculture [19, 20, 21].

Accordingly, the operations research community has been investigating approaches to improve the efficiency of UAV-powered applications [22, 23]. In particular, decentralized optimization methods have fostered search problems [24, 25, 26, 27, 28, 29], target assignment problems [30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40], node covering problems [41, 42], scheduling problems [43], and other cases [44, 45, 46].

In a complex mission, the UAV swarming component is typically dedicated to data collection duties, e.g., taking pictures, producing video feeds, sniffing wireless signals. Other technologies are then involved in processing the collected data and producing the desired output. For instance, digital photogrammetry algorithms [47, 48], can be leveraged to elaborate the images collected by the swarming UAVs, by extracting and displaying the relevant 2D/3D geometric information.

The data-processing phase — 3D processing phase in the photogrammetry use-case [49, 50, 51, 52, 53, 54] — is typically executed in the cloud and in a centralized fashion [55]. However, to mitigate Internet connectivity and network latency issues, the distributed power of the UAV swarm can be leveraged to establish an ad-hoc cloud infrastructure on top of which executing the data processing phases of the considered mission [55, 56, 57, 58]. This approach aims to exploit the *power of the many* — many embedded microcomputers of limited power are installed on the swarming UAVs — to replace the computer power typically available in a powerful work station reserved in the cloud.

This paper addresses the problem of optimizing the exploitation of a swarm-powered ad-hoc cloud, by jointly dealing with two interrelated aspects of the data-processing stage:

- the workload generation, i.e., definition of the computing application elements and of the corresponding set of data input.
- the workload scheduling/assignment, i.e., mapping of computing application elements and physical swarm members.

In particular, we consider the generation and placement of workloads whose input data are subject to geolocation/clustering constraints. Practically speaking, the collected data are bound to a location and must be processed in batch of neighboring samples: in the 3D reconstruction use-case, this means that groups of neighboring pictures have to be processed by the same computing element. This additional constraint has a non-negligible impact when dealing with a swarm powered ad-hoc cloud: due to the distributed nature of the swarm-powered data collection stage, the whole input data-set may end up being completely scattered across the UAVs of the swarm. If not properly dealt with during the workload generation/scheduling processes, this aspect may severely deteriorate the whole process performance due to unnecessary data transmission delays (input data must be received by the corresponding computing application element).

For the purpose of illustrating the applicability of our approach to a real-life application, we adjust the proposed solution to a relevant use-case scenario from the emergency response field. Our use-case is a perfect example of a real-life application subject to geolocation constraints that highly benefits from swarm-powered ad-hoc cloud infrastructure [59]. In fact, the drone swarm is able to perform 3D reconstruction of a region of interest — comprising five hundred photos — on top of twenty Raspberry Pis [60] microcomputers [61].

In such context, the 3D map of a given region of interest is used to improve the decision making process and the operator situational awareness through the availability of a 3D digital twin of the operation area, where the elements of interest can be even selectively highlighted, e.g., building or road damages, risky areas, etc. Producing the relevant 3D maps in a timely manner (near real-time), even when the cloud connectivity is not available, is crucial to increase the chances of success of an operation. To this purpose, we introduce a new optimization problem, namely the Covering-Assignment Problem for swarm-powered ad-hoc clouds (CAPsac). Given a set of geo-positioned aerial pictures (data) that are physically distributed across a set of UAVs (stored on the embedded microcomputers on the drones), CAPsac minimizes the 3D mapping (data-processing phase) completion time by jointly computing:

- the optimal workload configuration/the optimal covering of photos, i.e., splitting the overall photographed region across multiple convex sub-regions, and
- the optimal workload scheduling/the optimal assignment of photographed sub-regions to UAVs, i.e., deciding which drone (its embedded microcomputer) is responsible for the 3D reconstruction of a photographed sub-region.

It is worth pointing out that, differently from the *decompose-then-allocate* and the *allocate-then-decompose* paradigms [62] broadly adopted in (both the cloud computing optimization and) the multi-robot task allocation literature, CAPsac is an integrated decision model that handles workload generation (photo covering or sub-region splitting) and workload assignment (sub-region to UAV assignment) at the same time.

The remainder of the paper is organized as follows. The next section formally introduces the CAPsac problem, by clearly highlighting how the general problem designed for swarm-powered ad-hoc clouds is naturally applied to optimize the execution of a distributed 3D mapping application. Once the relationship between the general problem and the specific 3D mapping use-case will have been clearly proved through Section 2, it will be possible to present the remainder of the paper by directly referring to the latter. We believe that this editing approach will allow the reader to better grasp the details and the added value of the proposed solution. Two mathematical programming formulations to solve the CAPsac problem are described in Section 3 while Section 4 presents the NP-hardness proof for the problem. Finally, Section 5 presents and discusses the computational results obtained by experimenting with realistic 3D reconstruction instances, while Section 6 summarizes our concluding remarks.

2 Covering-Assignment Problem for Swarm-powered Ad-hoc Clouds - CAPsac

A swarm-powered mission can be typically decomposed in two phases:

i **Data collection**: the UAVs of the swarm dynamically collaborate to collect all the necessary information within the area of interest. In a swarm-powered 3D mapping mission, this phase corresponds to the photo-collection process meant to shoot the required aerial photos of the selected area. Note that the set of required pictures is typically computed by a dedicated mapping software and is merely an input of the mapping mission.

ii **Data processing**: the collected data are collaboratively processed by the ad-hoc cloud built on top of the microcomputers installed on the UAVs to produce the desired output. During this process, thanks to the swarm-powered ad-hoc cloud, the computing workload can be parallelized over the available computing units. Furthermore, the collected data can be transferred over the inter-drone wireless network to satisfy the input requirements of the distributed processing tasks. In a swarm-powered 3D mapping mission, this phase corresponds to the 3D-processing process meant to compute a 3D point cloud and/or a 3D mesh of the selected area.

The proposed CAPsac problem deals with the optimization of the data (3D) processing phase and has no direct control on the data (photo) collection strategy. Given the set of data (aerial pictures) just collected by the UAVs, CAPsac aims at minimizing the overall processing time required to compute the desired output (3D map).

An explanatory CAPsac problem instance involving a swarm of four drones performing a 3D mapping mission is represented in Figure 1. The full lines delimit the area of interest represented by the set P of aerial pictures just captured by the four drones during the photo collection phase. Each photo $p \in P$ was taken by a specific drone (which also stores it in memory). Furthermore, each picture must be processed during the 3D processing phase to guarantee a proper reconstruction.

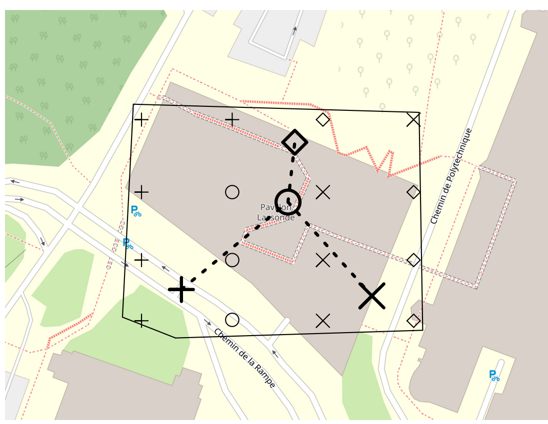

Figure 1: Explanatory instance of CAPsac problem accordingly to the 3D mapping use-case.

In Figure 1, the positions of the four drones are represented by the large symbols "$\times$", "$+$", "$\diamond$", and "$\circ$". Not all drones may be equipped with microcomputers powerful enough to support the 3D processing workload. In the example of Figure 1, only two drones are considered *3D-capable*, those represented by the $+$ and the $\times$ symbols.

Each photo is characterized by its shooting location, which is denoted by the small versions of the symbols previously used to represent the UAVs: the pictures represented by a small $+$ were shooted by the drone represented by the large $+$, and so on. Note that, given the dynamic nature of the decentralized decision-making process employed by the swarm of drones [62], it is impossible to know a-priori which UAV will shoot which picture.

A solution of the CAPsac problem describes how to:

- Split the processing workload into multiple processing (application) components, each responsible for dealing with a specific subset of the collected data, e.g., of the aerial pictures. Note that in the 3D reconstruction use-case each 3D reconstruction sub-task corresponds to a specific sub-region and requires as input all the aerial pictures that belong to that sub-region.

- Assign each processing component and all its corresponding input data to at least one of the computing elements available within the swarm-powered ad-hoc cloud, i.e., the microcomputers installed on the swarming-UAVs or on any other ground element connected to the swarm itself.

The optimal solution of CAPsac minimizes the latest processing time among all the involved computing elements, which corresponds to minimizing the makespan of the whole 3D reconstruction process. Three main issues cannot be ignored when assigning the photos (and thus the sub-regions) to the optimal 3D processing drones. Note that for sake of simplicity, we consider the case of not more than one computing device available on each drone.

i A feasible region (workload) subdivision is characterized by the creation of a *spatial-convex covering*: the union of the sub-regions corresponds to the whole region and the photos associated to each sub-region must be a *spatial-convex set*. Accordingly, a photo can be assigned to a drone if and only if it lies inside the convex hull of all the photos assigned to that drone. Figure 2 illustrates a set of photos which is **not** spatial-convex. The assigned photos are represented by colored "•" symbols. The "◦" symbols represent photos that do not belong to the set, which are hence assigned to other drones. Spatial-convex sets are crucial to perform the 3D mapping procedure since the presence of non-overlapping photo footprints (represented by the dashed colored rectangles in Figure 2) makes the 3D reconstruction of the associated region impossible. Figure 3 shows an example of a photo spatial-convex set assigned to one 3D-capable drone. It is worth pointing out that the creation of a spatial-convex covering is required by any workload operating over geo-located/clustered data to be processed in neighboring batches.

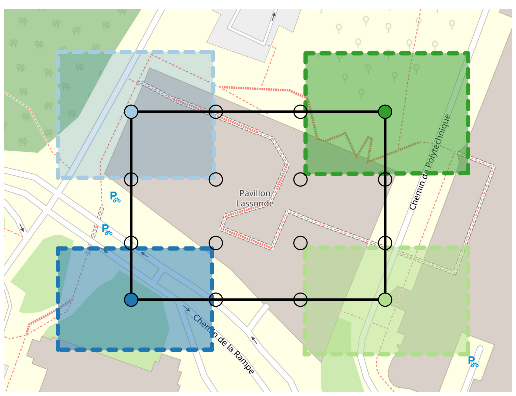

Figure 2: Ordinary set and the respective convex hull.

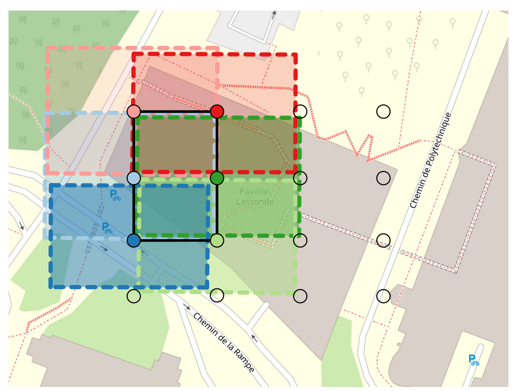

Figure 3: Spatial-convex set and the respective convex hull.

ii As the sub-regions (and their corresponding pictures) are assigned to the 3D-capable drones, a drone may need to require some input pictures (data) from the other swarm members. The CAPsac considers a pre-defined single-tree network topology built by a networking middleware running on the swarming drones [63]. Note that in Figure 4, the network links are represented by the pointed lines. Besides, in a single-tree network topology, only one routing path exists to connect each pair of UAVs. A drone cannot start the 3D reconstruction of the assigned sub-region until all the required photos are received. The TCP communication protocol is widely applied in engineering to achieve reliable transmissions and flow control, and it is adopted to model the swarm communications in the CAPsac problem. According to [64], a good way to approximate the transmission behavior concerning the TCP protocol is to assume that the transmission rate allocation follows the Max-Min Fairness - MMF paradigm [65]. Thus, minimizing the makespan of the 3D reconstruction requires that all the transmission rates of the network follow the MMF paradigm.

iii Finally, a *reliability factor* should be considered to immunize the CAPsac assignment with respect to drone malfunctions. The reliability factor defines the minimum number of drones (computing elements) that should process each sub-region.

Figure 4 shows a feasible solution to the CAPsac problem optimizing the makespan of a 3D mapping mission and considering a reliability factor equal to one. The dashed and the dashed-and-pointed lines define a feasible spatial-convex covering. The number of sets comprising the covering is equal to the number of 3D-capable drones. For instance, the covering in Figure 4 has only two sets. The photos lying within the left sub-region are processed by drone $+$, whereas those in the right sub-region are elaborated by drone $\times$.

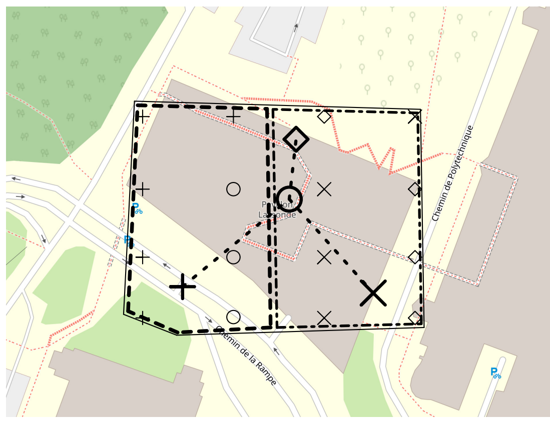

Figure 4: Spatial-convex covering and its assignment optimizing the makespan of a 3D mapping mission.

Given that the relationship between the general CAPsac and the swarm-powered 3D mission use-case has been clearly established, we will present the remainder of the paper referring directly to the specific use-case. Thus, it will be possible to the reader to better grasp the details and the added value of our proposed optimization problem.

3 Mathematical programming formulations

3.1 Common definitions

Let us consider a swarm of drones D of different types which is responsible for the 3D reconstruction of a region described by the set P of photos. Let $\bar{D} \subseteq D$, with $|\bar{D}| = m$, be the set of 3D-capable drones that have enough computing power to support the 3D reconstruction workloads. The location of all photos $p \in P$ are also known.

Given a photo $p \in P$, let λ_p and μ_p be the non-negative real parameters representing, respectively, the estimated processing time of p and the storage space occupied by p (expressed in megabytes). Also, for each drone $d \in D$, let θ_{dp} be the binary parameter equal to 1 if photo $p \in P$ is stored on drone d. The subset of photos processed by a drone directly corresponds to a specific sub-region. Therefore, note that the number of sub-regions are hence equal to the number of 3D-capable drones available in the swarm.

Some pictures may have to be transferred among different drones to respect the computed sub-region assignment configuration. The picture transmission is supported by an undirected transmission tree $T{=}(N, A)$, where the nodes of set N correspond to the swarming drones and the arcs of set A represent the device-to-device communication links (e.g., Wi-Fi links) between the drone themselves. Furthermore, let F be the set of traffic demands defined for each pair of drones $(h, d) \in D \times \bar{D}$, where f^{hd} denotes the demand (possibly null) between drones h and d. For all demands in F, let V^{hd} be the set of links $(i, j) \in A$ in the sole routing path between h and d in T. Also for each $(i, j) \in A$, denote c_{ij}

the transmission capacity of the link (i, j), and $\overline{F}_{ij} = \{F^{hd} \in F | (i, j) \in V^{hd}\}$, i.e., the set of demands that use the link (i, j) for transmission purposes. Finally, the maximum allowed time for transmitting the traffic demands through the network is denoted by $\hat{T}$.

With the support of the notation just introduced — grouped in Table 1, we present two different Mixed-Integer Linear Programming (MILP) formulations to optimize the 3D-processing phase of 3D mapping missions with UAV swarms:

- The Photo-based CAPsac (pCAPsac), where sub-regions are defined by explicitly assigning each picture $p \in P$ to the corresponding sub-regions.
- The Region-based CAPsac (rCAPsac), where all the feasible rectangular sub-regions are given; the formulation is responsible for selecting the optimal set of sub-regions among those available.

Table 1: CAPsac parameters in the context of the 3D mapping use-case.

Parameters	Description
λ_p	estimated processing time of a photo p
μ_p	amount of data of a photo p in Mb
θ_{dp}	equal to 1 if drone d has the photo p stored in its memory
F	set of traffic demands between each pair of drones
V^{hd}	routing path of a demand $f^{hd} \in F$ from the drone h to the drone d
c_{ij}	transmission capacity of the link $(i, j) \in A$
$\bar{c}^{hd}$	minimum c_{ij} for $(i, j) \in V^{hd}$
$\overline{F}_{ij}$	set of demands that use link $(i, j) \in A$
σ	reliability factor
m	number of drones (equiv. number of sub-regions) which can perform 3D reconstruction
$\hat{T}$	maximum allowed time for exchanging photos between drones

3.2 Photo-based CAPsac

In the pCAPsac formulation, the decision variables will be optimized to compose $R = \{1, \ldots, m\}$ sub-regions (equiv. subsets of photos) to be reconstructed by the set of drones. The formulation aims to jointly perform two assignment operations:

- each picture $p \in P$ is assigned to one sub-region $r \in R$,
- each non-empty sub-region $r \in R$ is assigned to one 3D-capable drone $d \in \bar{D}$.

To this purpose, let y_p^r and x_d^r be the binary variables equal to 1 when, respectively, photo $p \in P$ is assigned to sub-region $r \in R$, and sub-region $r \in R$ is assigned to drone $d \in \bar{D}$. Furthermore, let g_{dp}^r be the binary variables equal to 1 if drone $d \in \bar{D}$ is assigned to a sub-region $r \in R$ that contains picture $p \in P$.

3.2.1 Assignment constraints

To obtain a proper 3D-reconstruction, each photo must be processed at least one time, i.e., it must belong to at least one sub-region:

$$\sum_{r \in R} y_p^r \geq 1 \qquad \forall p \in P. \tag{1}$$

Similarly, each sub-region must be assigned to at least σ 3D-capable drones, with σ representing the previously introduced reliability factor meant to immunize the system toward possible drone failures:

$$\sum_{d \in \bar{D}} x_d^r \geq \sigma \qquad \forall r \in R. \tag{2}$$

Finally, let us introduce the group of constraints necessary to correctly compute the g variables without introducing any non-linearity:

$$g_{dp}^r \leq x_d^r \qquad \forall p \in P, \forall r \in R, \forall d \in \bar{D} \tag{3}$$

$$g_{dp}^r \leq y_p^r \qquad \forall p \in P, \forall r \in R, \forall d \in \bar{D} \tag{4}$$

$$g_{dp}^r \geq y_p^r + x_d^r - 1 \qquad \forall p \in P, \forall r \in R, \forall d \in \bar{D}. \tag{5}$$

That is, constraints (3)–(5) are the classical McCormick inequalities [66].

3.2.2 Spatial-convexity constraints

To properly work, state-of-the-art 3D reconstruction algorithms have to deal with convex regions and/or sub-regions, which is also equivalent to work with spatial-convex sets of photos. To this purpose, we approximate the convex hull of the set of photos assigned to a drone by its smallest enclosing hyperrectangle. This is not a bad approximation since 3D mapping missions often considers 50% to 80% of photo overlapping [67].

Since the GPS position of each photo shooting point is known, let C be the set of distinct picture longitudes and let L be the set of distinct photo latitudes. Note that the following relations are always respected: $1 \leq |C| \leq |P|$ and $1 \leq |L| \leq |P|$. For each sub-region, there exists a finite set of photo latitudes and longitudes that represents the bounding rectangular box, i.e. the approximated boundaries of the sub-region.

The boundary of a sub-region r is defined by its left (α^r), right (β^r), bottom (γ^r), and top (ω^r) borders. Binary variables α_c^r, β_c^r, γ_ℓ^r, and ω_ℓ^r are used to designate the latitudes and the longitudes defining these borders:

- Binary variable α_c^r is equal to one if longitude $c \in C$ delimits the left border of sub-region $r \in R$,
- Binary variable β_c^r is equal to one if longitude $c \in C$ delimits the right border of sub-region $r \in R$,
- Binary variable γ_ℓ^r is equal to one if latitude $\ell \in L$ delimits the bottom (inferior) border of sub-region $r \in R$,
- Binary variable ω_ℓ^r is equal to one if latitude $\ell \in L$ delimits the top (superior) border of sub-region $r \in R$.

Each sub-region $r \in R$ must be associated to a unique tuple of borders:

$$\sum_{c \in C} \alpha_c^r = 1 \qquad \forall r \in R \tag{6}$$

$$\sum_{c \in C} \beta_c^r = 1 \qquad \forall r \in R \tag{7}$$

$$\sum_{\ell \in L} \gamma_l^r = 1 \qquad \forall r \in R \tag{8}$$

$$\sum_{\ell \in L} \omega_l^r = 1 \qquad \forall r \in R. \tag{9}$$

To respect the sub-region convexity, a photo $p \in P$ can be assigned to sub-region $r \in R$ if and only if p is contained within the boundary defined for r. Geometrically, such constraint is fulfilled when (i) $lng_{\alpha^r} \leq lng_p \leq lng_{\beta^r}$ and (ii) $lat_{\gamma^r} \leq lat_p \leq lat_{\omega^r}$, where the lat stands for the latitude and lng for longitude. To capture this geometrical pattern, for each photo p, the sets $\mathcal{L}_\alpha^p$, $\mathcal{L}_\beta^p$, $\mathcal{L}_\gamma^p$, $\mathcal{L}_\omega^p$ are defined as:

- $\mathcal{L}_\alpha^p = \{c \in C | lng_c \leq lng_p\}$, i.e., $\mathcal{L}_\alpha^p$ contains the longitudes on the left of lng_p,
- $\mathcal{L}_\beta^p = \{c \in C | lng_c \geq lng_p\}$, i.e., $\mathcal{L}_\beta^p$ contains the longitudes on the right of lng_p,
- $\mathcal{L}_\gamma^p = \{\ell \in L | lat_\ell \leq lat_p\}$, i.e., $\mathcal{L}_\gamma^p$ contains the latitudes below lat_p,
- $\mathcal{L}_\omega^p = \{\ell \in L | lat_\ell \geq lat_p\}$, i.e., $\mathcal{L}_\omega^p$ contains the latitudes above lat_p.

Finally, the sub-region convexity is modeled by the *Boundary Constraints* - BC_1, expressed as:

$$(BC_1^\alpha) \qquad y_p^r \leq \sum_{c \in \mathcal{L}_\alpha^p} \alpha_c^r \qquad \forall p \in P, \forall r \in R \tag{10}$$

$$(BC_1^{\beta}) \qquad y_p^r \leq \sum_{c \in \mathcal{L}_{\beta}^p} \beta_c^r \qquad \forall p \in P, \forall r \in R \tag{11}$$

$$(BC_1^{\gamma}) \qquad y_p^r \leq \sum_{\ell \in \mathcal{L}_{\gamma}^p} \gamma_\ell^r \qquad \forall p \in P, \forall r \in R \tag{12}$$

$$(BC_1^{\omega}) \qquad y_p^r \leq \sum_{\ell \in \mathcal{L}_{\omega}^p} \omega_\ell^r \qquad \forall p \in P, \forall r \in R. \tag{13}$$

Constraints (10) restrict the longitudes which can compose the left border α^r to the left of the photo p's longitude. Similarly, Constraints (11)–(13) impose restrictions on the right (longitude), the bottom (latitude) and the top (latitude) borders, respectively.

However, a photo p is not assigned to sub-region r if and only if it lies outside the boundary of r, i.e., if $lng_p < lng_{\alpha^r}$, or $lng_p > lng_{\beta^r}$, or $lat_p < lat_{\gamma^r}$, or $lat_p > lat_{\omega^r}$, which may be expressed by either

$$(\overline{BC}_0) \qquad \sum_{c \in C - \mathcal{L}_{\alpha}^p} \alpha_c^r + \sum_{c \in C - \mathcal{L}_{\beta}^p} \beta_c^r + \sum_{\ell \in L - \mathcal{L}_{\gamma}^p} \gamma_\ell^r + \sum_{\ell \in L - \mathcal{L}_{\omega}^p} \omega_\ell^r \geq 1 - y_p^r \qquad \forall p \in P, \forall r \in R \tag{14}$$

or

$$(BC_0) \qquad \sum_{c \in \mathcal{L}_{\alpha}^p} \alpha_c^r + \sum_{c \in \mathcal{L}_{\beta}^p} \beta_c^r + \sum_{\ell \in \mathcal{L}_{\gamma}^p} \gamma_\ell^r + \sum_{\ell \in \mathcal{L}_{\omega}^p} \omega_\ell^r \leq 3 + y_p^r \qquad \forall p \in P, \forall r \in R. \tag{15}$$

Given constraints (6)–(9), constraints (14) guarantee that at least one of left-side summations is equal to one when p is not assigned to the sub-region r (i.e., $y_p^r = 0$). Consequently, at least one boundary of r makes the photo p to lie outside r. In a complimentary way, constraints (15) force that at most three boundaries are satisfied when the photo p is not assigned to the sub-region r.

Moreover, let us define valid inequalities (namely Ordering inequalities) to preemptively remove the infeasible boundaries in the continuous space for any possible sub-region. For instance, a boundary is infeasible if the the right border is placed on the left side of the left border. Such boundaries are removed through the following set of ordering inequalities:

$$\alpha_c^r \leq \sum_{j \in C: lng_j > lng_c} \beta_j^r \qquad \forall c \in C, \forall r \in R \tag{16}$$

$$\beta_c^r \leq \sum_{j \in C: lng_j < lng_c} \alpha_j^r \qquad \forall c \in C, \forall r \in R \tag{17}$$

$$\gamma_\ell^r \leq \sum_{j \in L: lat_j > lat_\ell} \omega_j^r \qquad \forall \ell \in L, \forall r \in R \tag{18}$$

$$\omega_\ell^r \leq \sum_{j \in L: lat_j < lat_\ell} \gamma_j^r \qquad \forall \ell \in L, \forall r \in R \tag{19}$$

3.2.3 Photo transmission constraints

For the purpose of minimizing the 3D processing computation time, it cannot be ignored that an additional delay is introduced any time a picture is transmitted by the drone where it is currently stored, to the drone that is responsible for reconstructing the corresponding sub-region. Given a demand $f^{hd} \in F$, let $\bar{c}^{hd}$ represent the minimum link capacity on routing path V^{hd} and let z^{hd} be the binary variable equal to 1 if traffic demand $f^{hd} > 0$ is active, i.e., if at least one picture has to be transferred from drone $h \in D$ to drone $d \in \bar{D}$. In this case, non-negative real variables ϕ^{hd} are used to represent the transmission rate achieved by traffic demand f^{hd} on its routing path.

The following two groups of constraints are introduced to, respectively, correctly activate binary variables z, and force flow variables ϕ to 0 when the corresponding traffic demands are idle and force the upper bound on the transmission rate, otherwise:

$$z^{hd} \leq \sum_{r\in R,p\in P} g^r_{dp}\theta_{hp} \qquad \forall(h,d)\in D\times\bar{D}, \tag{20}$$

$$\phi^{hd} \leq \bar{c}^{hd} z^{hd} \qquad \forall(h,d)\in D\times\bar{D}. \tag{21}$$

As mentioned in Section 2, the transmission times of the traffic demands are computed by considering the MMF paradigm for computing the traffic demand transmission rates. A flow (transmission rate) allocation vector is MMF if and only if there is at least one bottleneck link $(i,j)\in A$ on the routing path V^{hd} of each active traffic demand $f^{hd}\in F$ [68]. A link (i,j) is considered as a bottleneck of traffic demand f^{hd} if and only if [68]:

(i) its capacity is saturated, i.e., $\sum_{f^{ab}\in\overline{F}_{ij}}\phi^{ab}=c_{ij}$ and

(ii) the transmission rate ϕ^{hd} of traffic demand f^{hd} is the highest among those of the other traffic demands routed over link (i,j), i.e., $\phi^{hd}\geq\phi^{ab} \quad \forall f^{ab}\in\overline{F}_{ij}$.

Given a demand f^{hd}, let w^{hd}_{ij} be the binary variable equal to one if link (i,j) is a bottleneck of f^{hd}, as well as let u_{ij} be the highest transmission rate among all the traffic demands carried by a link $(i,j)\in A$, i.e., $u_{ij}=\max\limits_{f^{ab}\in\overline{F}_{ij}}\{\phi^{ab}\}$.

Thus, the following groups of constraints are used to impose the MMF paradigm for all the swarm communications (the photo transmissions) in pCAPsac [69]:

$$\sum_{(i,j)\in V^{hd}} w^{hd}_{ij} \geq z^{hd} \qquad \forall(h,d)\in D\times\bar{D} \tag{22}$$

$$\sum_{f^{ab}\in\overline{F}_{ij}} \phi^{ab} \leq c_{ij} \qquad \forall(i,j)\in A \tag{23}$$

$$\sum_{f^{ab}\in\overline{F}_{ij}} \phi^{ab} \geq c_{ij}w^{hd}_{ij} \qquad \forall(i,j)\in A,\forall f^{hd}\in\overline{F}_{ij} \tag{24}$$

$$u_{ij} \geq \phi^{hd} \qquad \forall(i,j)\in A,\forall f^{hd}\in\overline{F}_{ij} \tag{25}$$

$$\phi^{hd} \geq u_{ij}-c_{ij}(1-w^{hd}_{ij}) \qquad \forall(i,j)\in A,\forall f^{hd}\in\overline{F}_{ij}. \tag{26}$$

Constraints (22) ensure that all the active demands have at least one bottleneck link on their routing path. The capacity of each link $(i,j)\in A$ is respected through inequalities (23). The first condition required to consider a link (i,j) to be a bottleneck is jointly handled by constraints (23) and (24), which ensure that any bottleneck link (for at least one traffic demand) is saturated. The second bottleneck condition is instead fulfilled via constraints (25) and (26). Constraints (25) force u_{ij} to to be greater or equal to the highest transmission rate among the demands that are flowing through link (i,j). Finally, constraints (26) guarantee that ϕ^{hd} will not be exceeded by any other transmission rate of traffic demands routed over link $(i,j)\in A$ when (i,j) is a bottleneck link of traffic demand f^{hd}.

In pCAPsac formulation, a maximum networking/transmission latency of $\hat{T}$ seconds is imposed for each activated traffic demand:

$$\hat{T}\cdot\phi^{hd} \geq \sum_{r\in R,p\in P} g^r_{dp}\theta_{hp}\mu_p \qquad \forall(h,d)\in D\times\bar{D}. \tag{27}$$

The summation term $\sum_{r\in R,p\in P} g^r_{dp}\theta_{hp}\mu_p$ computes the overall amount of data to be transferred from drone $h\in D$ to the drone $d\in\bar{D}$. Limiting the transmission times means to ensure every drone receives all the photos belonging to the assigned sub-region within a maximum prefixed time set by the domain expert. This constraint can be relaxed by setting $\hat{T}$ to a suitable very high value.

3.2.4 Symmetry breaking constraints

Formulation pCAPsac suffers from symmetry in both photo-to-sub-region (i.e., y_p^r) and sub-region-to-drone (i.e., x_d^r) assignments. It is possible to partially break the symmetry of the sub-region-to-drone assignments. Each sub-region can be assigned to one distinct drone in advance. That is, m variables x_d^r are fixed where the fixed pairs $\{(r_1, d_1), \ldots, (r_m, d_m)\} \in \{R \times \bar{D}\}$ have distinct indexes. Considering Figure 1, the sub-region 1 could be assigned to the drone "+" and the sub-region 2 to the drone "×" for instance. Note that setting variables x_d^r does not affect the variables y_p^r. Consequently, just redundant integer solutions are eliminated.

3.2.5 Complete formulation

Let $T_{\max}$ be the 3D mapping completion time, i.e., the makespan, calculated as the maximum processing time obtained from the swarm of drones. The variable $T_{\max}$ is computed by the group of constraints

$$T_{\max} \geq \sum_{r \in R, p \in P} g_{dp}^r \lambda_p \qquad \forall d \in \bar{D} \tag{28}$$

where $\sum_{r \in R, p \in P} (g_{dp}^r \lambda_p)$ computes the required time to process all the photos assigned to drone $d \in \bar{D}$.

Finally, the pCAPsac formulation is expressed by the following MILP.

$$\begin{aligned}
\min_{x,y} \quad & T_{\max} && && (29)\\
\text{s.t.} \quad & (1) - (28) && && \\
& x_d^r, y_p^r, g_{dp}^r \in \{0,1\} && \forall r \in R, \forall p \in P, \forall d \in \bar{D} && (30)\\
& w_{ij}^{hd} \in \{0,1\} && \forall (i,j) \in A, \forall (h,d) \in D \times \bar{D} && (31)\\
& \phi^{hd} \geq 0, z^{hd} \in \{0,1\} && \forall (h,d) \in D \times \bar{D} && (32)\\
& \alpha_c^r, \beta_c^r \in \{0,1\} && \forall c \in C, \forall r \in R && (33)\\
& \gamma_\ell^r, \omega_\ell^r \in \{0,1\} && \forall \ell \in L, \forall r \in R && (34)\\
& u_{ij} \geq 0 && \forall (i,j) \in A. && (35)
\end{aligned}$$

The objective function (29) minimizes the makespan of the whole 3D mapping procedure. The domain constraints are given by (30)–(35). As $|R| = |\bar{D}|$, the total number of constraints in the model is $O(|P| \cdot |\bar{D}|^2)$ as well as the number of its variables.

3.3 Region-based CAPsac

The *CAPsac* problem can be addressed by explicitly considering the set $\mathcal{S}$ of all feasible rectangular subsets of photos, such that each element of $\mathcal{S}$ corresponds to a possible rectangular sub-region to be 3D-reconstructed. It is important to remark that the cardinality of $\mathcal{S}$ is polynomial and bounded by $O(|C|^2|L|^2)$, which is $O(|P|^4)$ in the worst scenario:

Proposition 1 *Given a set $\mathcal{S}$ comprising all feasible rectangular sub-regions to a CAPsac instance. The $|\mathcal{S}|$ is bounded by $O(|C|^2|L|^2)$, which is $O(|P|^4)$ in the worst case.*

Proof. As in Section 3.2.2, any feasible hyperectangle $S \in \mathcal{S}$ is defined by a tuple (α^S, β^S, γ^S, ω^S) of latitudes and longitudes corresponding to the left, right, bottom, and top borders of S, respectively, with $\alpha^S, \beta^S \in C$ and $\gamma^S, \omega^S \in L$, and such that $\alpha^S \leq \beta^S$ and $\gamma^S \leq \omega^S$. Therefore, $\mathcal{S} = C \times C \times L \times L$. Since $1 \leq |C| \leq |P|$ and $1 \leq |L| \leq |P|$, $|\mathcal{S}|$ is bounded by $O(|P|^4)$. □

In particular, the photos are commonly spread across the target region in a grid pattern to fulfil photo footprint overlapping constraints [67]. Consequently, $|C|$ and $|L|$ are usually far smaller than $|P|$, and hence, $|C|^2 \cdot |L|^2$ is in practice usually significantly smaller than $|P|^4$.

Let $\mathcal{S}_p$ be the collection of rectangular subsets $S \in \mathcal{S}$ which cover photo $p \in P$. For each set $S \in \mathcal{S}$, denote t^S the photo processing time of S, and μ_S^{hd} the amount of data to transfer from drone $h \in D$ to the drone $d \in \bar{D}$ if S is selected. Let q_d^S be the binary variables equal to 1 if S is allocated to drone $d \in \bar{D}$. Finally, let us denote o^S the auxiliary binary variable which is equal to 1 if S is selected, and 0 otherwise.

The region-based formulation of the $CAPsac$ is expressed as follows:

$$\min_{q,o} \quad T_{\max} \tag{36}$$

$$\text{s.t. } T_{\max} \geq \sum_{S \in \mathcal{S}} t^S q_d^S \qquad \forall d \in \bar{D} \tag{37}$$

$$\hat{T} \cdot \phi^{hd} \geq \sum_{S \in \mathcal{S}} \mu_S^{hd} q_d^S \qquad \forall (h,d) \in D \times \bar{D} \tag{38}$$

$$\sum_{d \in \bar{D}} q_d^S \geq \sigma o^S \qquad \forall S \in \mathcal{S} \tag{39}$$

$$\sum_{S \in \mathcal{S}_p} o^S \geq 1 \qquad \forall p \in P \tag{40}$$

$$\sum_{S \in \mathcal{S}} o^S = m \tag{41}$$

$$z^{hd} \leq \sum_{S \in \mathcal{S}} \mu_S^{hd} q_d^S \qquad \forall (h,d) \in D \times \bar{D} \tag{42}$$

$$(21) - (26)$$

$$o^S, q_d^S \in \{0,1\} \qquad \forall S \in \mathcal{S}, \forall d \in \bar{D} \tag{43}$$

$$w_{ij}^{hd} \in \{0,1\} \qquad \forall (i,j) \in A, \forall (h,d) \in D \times \bar{D} \tag{44}$$

$$\phi^{hd} \geq 0, z^{hd} \in \{0,1\} \qquad \forall (h,d) \in D \times \bar{D} \tag{45}$$

$$u_{ij} \geq 0 \qquad \forall (i,j) \in A. \tag{46}$$

The objective function (36) minimizes the makespan $T_{\max}$, which is computed by constraints (37). Constraints (38) limit the networking delay for each photo transmission traffic demand. Constraints (39) impose that the selected subsets in $\mathcal{S}$ are assigned to σ drones which can do the 3D reconstruction. The set of constraints (40) ensures that each photo $p \in P$ is covered at least once, and constraint (41) defines the number of selected subsets to m, i.e., the number of drones which can perform the 3D reconstruction. The transmission rates are defined by (42) and (21)–(26), following MMF rate allocation, as explained in Section 3.2. Finally, domain constraints are given in (43)–(46).

The cardinality of $\mathcal{S}$ is bounded by $O(|P|^4)$ (Proposition 1). Therefore, the number of constraints in the formulation is bounded by $O(|P|^4)$ due to the amount of constraints (39). The number of variables is bounded by $O(|D| \cdot |P|^4)$ due to the number of variables q_d^S.

4 NP-Hardness of the CAPsac

The proof comes from a reduction of the decision version of the *unweighted Geometric Set-Covering Problem - GSCP*, whose objective is to assert, for a finite set of points $P' = \{p_1, p_2, ..., p_n \in \mathbb{R}^d\}$ and a finite collection $\mathcal{S}'$ of subsets of P', if there exists a covering for the points P' composed by at most $k < |\mathcal{S}'|$ sets of $\mathcal{S}'$, i.e., if there exists a $\mathcal{C} \subset \mathcal{S}'$ such that $\cup_{c \in \mathcal{C}} c = P'$ and $|\mathcal{C}| \leq k$. The collection $\mathcal{S}'$ is induced by a fixed polytope $\mathcal{T}$, that is, $\mathcal{S}'$ is formed by the points covered by the distinct placements of $\mathcal{T}$ over the coordinates of the points P'. The decision problem is NP-Complete even when $\mathcal{T}$ is a fixed square [70] or a fixed circumference [71].

Proposition 2 *Given an instance I' of the $GSCP$, there exists a polynomial-time transformation from I' to an instance I of the $rCAPsac$.*

Proof. Consider an instance of the $GSCP$ with P' points, a collection $\mathcal{S}'$ of subsets of P' induced by a fixed square of length s, and a positive integer $k < |\mathcal{S}'|$. The instance of the $rCAPsac$ is created on polynomial-time as follows.

Let the set of photos P and their location be equal to the set of points P' (i.e., $P = P'$), and consider a set of k drones which can do the 3D reconstruction, i.e., $|D| = |\bar{D}| = k$. The communication network $T = (N, A)$ is a random tree whose links $(i, j) \in A$ have infinite capacity. Therefore, the transmission times, the transmission rates ϕ^{hd}, and the photos storage location θ_{dp} can be dismissed. Thus, the latest completion time $T_{\max}$ is defined only by the photo processing times of the drones. Concerning the reliability factor, it is made equal to 1 ($\sigma = 1$). Indeed, the collection $\mathcal{S}'$ does not have all possible rectangular spatial-convex sets, being the $\mathcal{S} \setminus \mathcal{S}'$ missing spatial-convex sets obtained in polynomial-time by inspecting the tuples in $C \times C \times L \times L$. Finally, the photo processing times of the spatial-convex sets will be either 1 or $+\infty$. For $S \in \mathcal{S}'$, $t^S = 1$, while for the remaining $S \in \mathcal{S} \setminus \mathcal{S}'$ $t^S = +\infty$. □

Proposition 3 The $CAPsac$ answers the $GSCP$.

Proof. Consider $rCAPsac(P, D, T, \mathcal{S})$ a routine which solves the $CAPsac$ by the $rCAPsac$ formulation (Section 3.3). Let its optimal solution be comprised by the optimal set Q^* of variables q_d^S and the optimal completion time $T^*_{\max}$. Given a solution of an instance of the $CAPsac$ created from an instance of the $GSCP$, evaluating $T^*_{\max}$ is enough to answer the $GSCP$. If $T^*_{\max} = 1$, reply *yes*. Otherwise, reply *no*. For an optimal solution whose $T^*_{\max} = 1$, the covering $\mathcal{C}$ of the P', with $|\mathcal{C}| \leq k$, can be extracted from Q^*. □

Theorem 1 The $CAPsac$ is NP-Hard.

Proof. Given the propositions 2 and 3, one can state that the $GSCP$ is no harder than the $CAPsac$. Since the $GSCP$ is NP-complete, the $CAPsac$ is NP-Hard. □

5 Computational experiments

Our experimental analysis assessed (i) the effectiveness of the ordering inequalities and the branching strategies for the pCAPsac formulation; (ii) the performance of formulation pCAPsac; (iii) the sensitivity of the pCAPsac formulation with respect to both reliability factor σ and maximum allowed transmission time $\hat{T}$. We used CPLEX v12.8 as general-purpose linear programming solver. All experiments were carried exploiting a single core on a machine powered by an Intel E5-2683 v4 Broadwell 2.1GHz with 20Gb of RAM, and running the CentOS Linux 7.5.1804 OS.

We do not present computational experiments for formulation rCAPsac. As demonstrated in Section 3.3, the number of variables of that formulation is bounded by $O(|P|^4)$, which can rapidly increase. Column Generation (CG) strategy has been extensively applied to formulations with a massive number of variables [72]. The great advantage of employing CG is to solve the Linear Program (LP) continuous relaxation considering only a relevant subset of variables. Such LP with reduced number of variables is called the *restricted master problem* (RMP). The subset of relevant variables is iteratively created as needed by solving the so-called *pricing subproblem* (PS). Usually, a CG iteration comprises: i) solving the current RMP to obtain current primal optimal solution and its associated dual variables, and ii) optimizing the PS to find new variables with negative reduced costs (when considering minimization problems). The CG terminates when the PS does not find any variable with negative reduced cost, i.e., when the optimality of the current RMP has been proved. Preliminary experiments (restricted to $\sigma = 1$ and $\hat{T} = +\infty$) were performed applying a vanilla column generation on the $rCAPsac$. Unfortunately, $rCAPsac$ has proved highly degenerate requiring several CG iterations to prove optimality. Therefore, its performance for solving CAPsac was largely inferior to that of using the pCAPsac formulation. Finally, note that the above degeneragy is a common CG drawback that leads the resulting algorithms (and codes) to be difficult to tune and somehow delicate, thus incompatible for the applied context we deal with.

5.1 Tested instances and computational settings

The instances, which were constructed from realistic data, comprise two scenarios:

i *Unweighted*: all photos require the same amount of processing time λ.

ii *Weighted*: each photo $p \in P$ requires a certain amount of processing time λ_p.

The λ_p are acquired from the equivalent unweighted case: $\lceil |P| \times 0.1 \rceil$ groups of nine adjacent photos are randomly selected, and then, for each of those groups, a single λ_p is drawn from a normal distribution ($\mu = 26.72$ seconds and $\sigma = 5.0$) and attributed to all photos of that group. Changing the photo-processing time in that fashion allows to represent the 3D reconstruction of distinct complex objects in the region of interest. The name of the instances follows the notation X-PYYD$Z\%\bar{D}WW$ where "X" is "u" for the unweighted instances and "w" for the weighted instances, "YY" stands for the number of photos in the instance, "Z" specifies the number of drones in the swarm, and "WW" informs the percentage of drones that can do 3D reconstruction. The number of drones able to perform 3D reconstruction, called $|\bar{D}|$, is always equal to $\lfloor Z \times \frac{WW}{100} \rfloor$. The characteristics of the tested instances are listed in Table 2.

Table 2: Characteristics of the tested instances.

Photos(P)	200, 400
Drones(D)	5, 7, 10
%3D-capable drones(%$\bar{D}$)	50%, 70%, 90%

The tables presented hereafter report the instance employed at each row (column "Instance"), the corresponding formulation (column "Form."), the dual gap (in percentage) wrt the optimal solution found at the root node (column "gap_0"), the number of cuts added by CPLEX at the root node (column "cuts"), the number of nodes explored by the CPLEX's branch-and-cut method (column "Nodes"), and the dual gap (in percentage) wrt the optimal solution (best known, see below) at the end of the branch-and-cut enumeration (column "gap"). CPU times spent in the solution of the root node and by the branch and cut algorithm are also reported (column "*sec.*").

Note that dual gaps are computed with respect to the best upper bound solution found whenever the optimal solutions are not obtained by CPLEX within one day of execution. These situations are represented in the tables by the symbol '$*$'.

5.2 pCAPsac experiments

This section evaluates the performance of the proposed pCAPsac formulation. It investigates the effectiveness of valid inequalities (16)–(19), whose aim is, for any possible sub-region, to clean the search space from all the infeasible boundary configuration. Also, the experiments analyze how different branching priorities can influence the branch-and-cut method. All the experiments of this section are for $\sigma = 1$ and $\hat{T}$=$+\infty$.

5.2.1 pCAPsac performance

The pCAPsac formulation using the $\overline{BC_0}$ constraints (14), called "PB:$\overline{BC_0}$", and the pCAPsac formulation using the BC_0 constraints (15), named "PB:BC_0", are compared in Tables 3 and 4.

The results in both Tables 3 and 4 clearly show that the "PB:$\overline{BC_0}$" solves faster than "PB:BC_0" formulation (T=-2.877 and p-val=0.008 via "PB:$\overline{BC_0}$" vs. "PB:BC_0" paired t-test [73]).

We can observe that the dual gaps at the root node are equal to zero for the unweighted instances whenever the number of photos is divisible by the number of drones that can do the 3D reconstruction. Consequently, the objective function value of the optimum solution coincides with the dual bound already at the root node for instances "u-P200D5%$\bar{D}$90", "u-P200D7%$\bar{D}$70", "u-P400D5%$\bar{D}$90", "u-P400D7%$\bar{D}$70", "u-P200D10%$\bar{D}$50" and "u-P400D10%$\bar{D}$50".

Table 3: CPLEX results when solving unweighted instances for the "$PB{:}\overline{BC_0}$" and the "$PB{:}BC_0$" formulations.

Instance	Form.	Root Node			Branch-and-Cut		
		gap$_0$	cuts	sec.	Nodes	gap	sec.
u-P200D5%$\bar{D}$70	$PB{:}\overline{BC_0}$	4.76	28	1.47	182	0.00	15.81
	$PB{:}BC_0$	4.76	110	2.84	300	0.00	33.83
u-P200D7%$\bar{D}$50	$PB{:}\overline{BC_0}$	4.76	17	1.25	395	0.00	27.48
	$PB{:}BC_0$	4.76	62	1.14	383	0.00	19.70
u-P400D5%$\bar{D}$70	$PB{:}\overline{BC_0}$	1.23	28	3.33	386	0.00	87.16
	$PB{:}BC_0$	1.23	74	3.57	264	0.00	57.94
u-P400D7%$\bar{D}$50	$PB{:}\overline{BC_0}$	1.23	27	3.42	556	0.00	156.15
	$PB{:}BC_0$	1.23	50	3.16	1178	0.00	181.14
u-P200D5%$\bar{D}$90	$PB{:}\overline{BC_0}$	0.00	9	2.16	433	0.00	24.73
	$PB{:}BC_0$	0.00	31	3.21	41	0.00	13.98
u-P200D7%$\bar{D}$70	$PB{:}\overline{BC_0}$	0.00	23	2.14	371	0.00	27.49
	$PB{:}BC_0$	0.00	22	4.07	175	0.00	24.74
u-P400D5%$\bar{D}$90	$PB{:}\overline{BC_0}$	0.00	4	5.94	631	0.00	440.53
	$PB{:}BC_0$	0.00	81	4.09	121	0.00	44.60
u-P400D7%$\bar{D}$70	$PB{:}\overline{BC_0}$	0.00	71	5.04	790	0.00	141.43
	$PB{:}BC_0$	0.00	84	5.06	1631	0.00	696.21
u-P200D10%$\bar{D}$50	$PB{:}\overline{BC_0}$	0.00	81	3.63	6591	0.00	1036.81
	$PB{:}BC_0$	0.00	95	2.72	18840	0.00	3141.16
u-P400D10%$\bar{D}$50	$PB{:}\overline{BC_0}$	0.00	38	9.99	4363	0.00	1737.28
	$PB{:}BC_0$	0.00	127	10.64	16216	0.00	7200.00
u-P200D7%$\bar{D}$90	$PB{:}\overline{BC_0}$	1.96	24	5.17	8448	0.00	1354.41
	$PB{:}BC_0$	1.96	47	9.60	13835	1.96	7200.00
u-P400D7%$\bar{D}$90	$PB{:}\overline{BC_0}$	*1.96	130	12.28	12362	*1.96	7200.00
	$PB{:}BC_0$	*1.96	84	13.56	9063	*1.96	7200.00

5.2.2 Ordering inequalities effectiveness

The effect of adding all the ordering inequalities (16)–(19) into the "$PB{:}\overline{BC_0}$" formulation is analyzed in Tables 5 and 6. The inclusion of the ordering inequalities is identified by the "$+Ord.$" in the formulation name.

The valid inequalities (16)–(19) eliminate infeasible boundaries in the continuous solution space whereas not necessarily excluding the continuous optimum solution. Consequently, these inequalities are not guaranteed to increase the dual bound obtained. In fact, gap_0 was never improved in our experiments after adding the ordering inequalities. Nevertheless, the insertion of (16)–(19) improved the CPU time required to reach the optimum solution of 11 out to 19 instances solved to optimality (considering 2h of execution). In fact, paired t-tests show significant improvements (T= 1.907 p-val= 0.034) by adding them into the "$PB{:}\overline{BC_0}$" formulation, except for instances "P400D7%$\bar{D}$70". However, when limited to weighted cases, there is no significant improvement on reducing the enumeration CPU time. For these cases, 11 seconds of improvement is obtained when comparing the average computing CPU time of non-inserting against inserting constraints (16)–(19). Finally, we observed that the number of cuts added by CPLEX at the root node increased considerably when the ordering inequalities were employed.

5.2.3 Branching priority

Different branching priorities for the selection of the boundary assignment (i.e., α_c^r, β_c^r, γ_ℓ^r, and ω_ℓ^r) and the photo assignment (i.e., y_p^r) variables were also explored in formulation "$PB{:}\overline{BC_0}{+}Ord.$" (simply denoted "$PB{:}\overline{BC_0}$" at this section) and the results reported in Tables 7 and 8. The distinct branching priorities are denoted as "$b{>}y$" and "$y{>}b$". The the default option of CPLEX is identified by the

Table 4: CPLEX results when solving weighted instances for the “$PB{:}\overline{BC_0}$” and the “$PB{:}BC_0$” formulations.

Instance	Form.	Root Node			Branch-and-Cut		
		gap$_0$	cuts	sec.	Nodes	gap	sec.
w-P200D5%$\bar{D}$70	$PB{:}\overline{BC_0}$	3.36	28	1.40	226	0.00	19.13
	$PB{:}BC_0$	3.36	19	2.14	450	0.00	45.84
w-P200D7%$\bar{D}$50	$PB{:}\overline{BC_0}$	3.67	22	1.34	961	0.00	50.73
	$PB{:}BC_0$	3.67	40	1.27	1057	0.00	65.62
w-P400D5%$\bar{D}$70	$PB{:}\overline{BC_0}$	0.86	9	2.57	483	0.00	97.08
	$PB{:}BC_0$	0.86	45	3.19	910	0.00	148.94
w-P400D7%$\bar{D}$50	$PB{:}\overline{BC_0}$	2.02	44	2.92	1008	0.00	191.33
	$PB{:}BC_0$	2.02	22	3.47	930	0.00	121.78
w-P200D5%$\bar{D}$90	$PB{:}\overline{BC_0}$	2.96	13	2.37	5080	0.00	748.98
	$PB{:}BC_0$	2.96	42	4.76	24186	0.00	5199.65
w-P200D7%$\bar{D}$70	$PB{:}\overline{BC_0}$	2.69	58	2.55	4227	0.00	453.18
	$PB{:}BC_0$	2.69	21	3.95	18204	0.00	4191.87
w-P400D5%$\bar{D}$90	$PB{:}\overline{BC_0}$	1.26	19	4.85	3161	0.00	1425.80
	$PB{:}BC_0$	1.26	51	5.29	18195	1.24	7200.00
w-P400D7%$\bar{D}$70	$PB{:}\overline{BC_0}$	0.68	34	4.97	4529	0.00	1805.83
	$PB{:}BC_0$	0.68	13	4.63	19444	0.68	7200.00
w-P200D10%$\bar{D}$50	$PB{:}\overline{BC_0}$	1.65	15	5.42	32327	1.65	7200.00
	$PB{:}BC_0$	1.65	63	3.62	42035	1.65	7200.00
w-P400D10%$\bar{D}$50	$PB{:}\overline{BC_0}$	*3.38	129	10.59	9416	*3.38	7200.00
	$PB{:}BC_0$	*3.38	82	8.88	7364	*3.38	7200.00
w-P200D7%$\bar{D}$90	$PB{:}\overline{BC_0}$	*3.81	1	8.28	54078	*3.81	7200.00
	$PB{:}BC_0$	*3.81	71	6.17	19819	*3.81	7200.00
w-P400D7%$\bar{D}$90	$PB{:}\overline{BC_0}$	*3.18	15	9.42	13316	*3.18	7200.00
	$PB{:}BC_0$	*3.18	83	17.06	10581	*3.18	7200.00

absence of those notations. The “$b{>}y$” is used when the boundary assignment variables are given higher priority over the photo assignment variables, which still have higher priority over the remaining variables. The “$y{>}b$” refers to the opposite case, that is, when photo assignment variables are rather branched over boundary assignment variables.

The adoption of different branching priorities improves the CPU times to solve some instances. In fact, the “$y{>}b$” strategy achieves better or equivalent CPU times in 10 out to 11 unweighted instances solved to optimality (within 2h of execution). Regarding weighted cases, the “$b{>}y$” strategy results in better or equivalent CPU times in 5 out to 8 instances solved to optimality (considering 2h of execution). However, paired t-tests do not show significant improvement for both “$b{>}y$” (T= 0.889 p-val= 0.191) or “$y{>}b$” (T=1.295 p-val= 0.103) strategies considering overall cases.

5.3 Sensitivity of the pCAPsac formulation

This section evaluates the sensitivity of pCAPsac formulation with respect to both reliability factor σ and to maximum transmission time allowed $\hat{T}$.

5.3.1 Reliability factor sensitivity

Tables 9 and 10 report results for various values of σ, ranging from 1 to $|\bar{D}|-1$. The subset of instances used in this experiment consists of those for which CPLEX was able to solve within 2h of execution the associated problem with $\sigma = 1$. Besides, the communication constraints concerning $\hat{T}$ were relaxed, i.e. $\hat{T}{=}{+}\infty$.

Table 5: CPLEX results when solving unweighted instances for the "$PB{:}\overline{BC_0}$" and the "$PB{:}\overline{BC_0}{+}Ord.$" formulations.

Instance	Form.	Root Node			Branch-and-Cut		
		gap$_0$	cuts	sec.	Nodes	gap	sec.
u-P200D5%$\bar{D}$70	$PB{:}\overline{BC}_0$	4.76	28	1.47	182	0.00	15.81
	$PB{:}\overline{BC}_0$+Ord.	4.76	242	1.15	345	0.00	19.02
u-P200D7%$\bar{D}$50	$PB{:}\overline{BC}_0$	4.76	17	1.25	395	0.00	27.48
	$PB{:}\overline{BC}_0$+Ord.	4.76	189	1.46	239	0.00	11.45
u-P400D5%$\bar{D}$70	$PB{:}\overline{BC}_0$	1.23	28	3.33	386	0.00	87.16
	$PB{:}\overline{BC}_0$+Ord.	1.23	183	3.38	500	0.00	91.56
u-P400D7%$\bar{D}$50	$PB{:}\overline{BC}_0$	1.23	27	3.42	556	0.00	156.15
	$PB{:}\overline{BC}_0$+Ord.	1.23	188	4.31	707	0.00	133.70
u-P200D5%$\bar{D}$90	$PB{:}\overline{BC}_0$	0.00	9	2.16	433	0.00	24.73
	$PB{:}\overline{BC}_0$+Ord.	0.00	227	1.99	359	0.00	22.60
u-P200D7%$\bar{D}$70	$PB{:}\overline{BC}_0$	0.00	23	2.14	371	0.00	27.49
	$PB{:}\overline{BC}_0$+Ord.	0.00	226	2.33	699	0.00	45.39
u-P400D5%$\bar{D}$90	$PB{:}\overline{BC}_0$	0.00	4	5.94	631	0.00	440.53
	$PB{:}\overline{BC}_0$+Ord.	0.00	267	5.00	49	0.00	35.02
u-P400D7%$\bar{D}$70	$PB{:}\overline{BC}_0$	0.00	71	5.04	790	0.00	141.43
	$PB{:}\overline{BC}_0$+Ord.	0.00	175	6.23	7011	0.00	6665.71
u-P200D10%$\bar{D}$50	$PB{:}\overline{BC}_0$	0.00	81	3.63	6591	0.00	1036.81
	$PB{:}\overline{BC}_0$+Ord.	0.00	312	4.99	3546	0.00	556.97
u-P400D10%$\bar{D}$50	$PB{:}\overline{BC}_0$	0.00	38	9.99	4363	0.00	1737.28
	$PB{:}\overline{BC}_0$+Ord.	0.00	46	6.35	990	0.00	219.40
u-P200D7%$\bar{D}$90	$PB{:}\overline{BC}_0$	1.96	24	5.17	8448	0.00	1354.41
	$PB{:}\overline{BC}_0$+Ord.	1.96	130	8.80	521	0.00	49.52
u-P400D7%$\bar{D}$90	$PB{:}\overline{BC}_0$	*1.96	130	12.28	12362	*1.96	7200.00
	$PB{:}\overline{BC}_0$+Ord.	*1.96	50	17.94	14897	*1.96	7200.00

Tables 9 and 10 show that initial dual gaps are largely affected by σ. Those large dual gaps result from the increase of the optimum solution values with σ not accompanied by the increase in the dual bounds obtained at the root node. This is illustrated in Figure 5 and Figure 6, where the changes in the optimal objective function value (named "$T_{\max}$") given the increase of σ are presented for unweighted instances "u-P200D7%$\bar{D}$70", "u-P200D10%$\bar{D}$50", "u-P200D7%$\bar{D}$90", and for weighted instances "w-P200D7%$\bar{D}$70", "w-P200D10%$\bar{D}$50", "w-P200D7%$\bar{D}$90". Those instances were selected to include distinct values of $|\bar{D}|$.

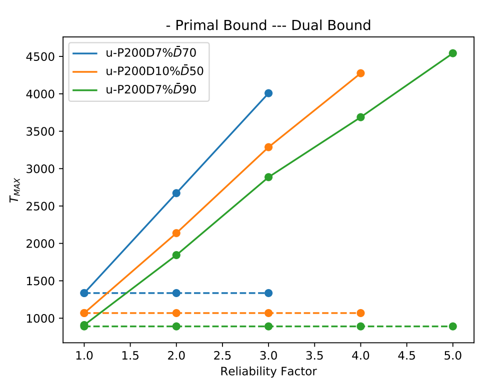

Figure 5: Primal bound and dual bound on increasing σ for unweighted instances.

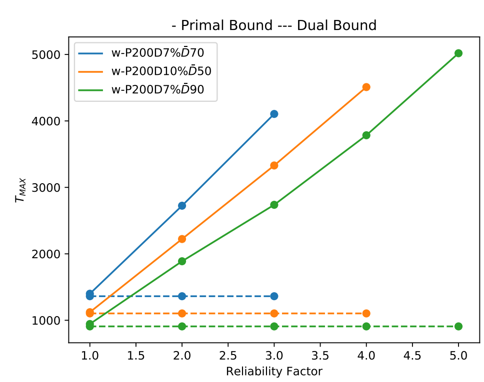

Figure 6: Primal bound and dual bound on increasing σ for weighted instances.

Table 6: CPLEX results when solving weighted instances for the "$PB{:}\overline{BC_0}$" and the "$PB{:}\overline{BC_0}{+}Ord.$" formulations.

Instance	Form.	Root Node			Branch-and-Cut		
		gap$_0$	**cuts**	**sec.**	**Nodes**	**gap**	**sec.**
w-P200D5%$\bar{D}$70	$PB{:}\overline{BC_0}$	3.36	28	1.40	226	0.00	19.13
	$PB{:}\overline{BC_0}$+Ord.	3.36	93	1.25	300	0.00	16.87
w-P200D7%$\bar{D}$50	$PB{:}\overline{BC_0}$	3.67	22	1.34	961	0.00	50.73
	$PB{:}\overline{BC_0}$+Ord.	3.67	111	1.49	549	0.00	31.52
w-P400D5%$\bar{D}$70	$PB{:}\overline{BC_0}$	0.86	9	2.57	483	0.00	97.08
	$PB{:}\overline{BC_0}$+Ord.	0.86	190	3.48	414	0.00	79.99
w-P400D7%$\bar{D}$50	$PB{:}\overline{BC_0}$	2.02	44	2.92	1008	0.00	191.33
	$PB{:}\overline{BC_0}$+Ord.	2.02	264	3.40	974	0.00	241.08
w-P200D5%$\bar{D}$90	$PB{:}\overline{BC_0}$	2.96	13	2.37	5080	0.00	748.98
	$PB{:}\overline{BC_0}$+Ord.	2.96	84	2.30	3427	0.00	427.78
w-P200D7%$\bar{D}$70	$PB{:}\overline{BC_0}$	2.69	58	2.55	4227	0.00	453.18
	$PB{:}\overline{BC_0}$+Ord.	2.69	87	3.01	3736	0.00	514.63
w-P400D5%$\bar{D}$90	$PB{:}\overline{BC_0}$	1.26	19	4.85	3161	0.00	1425.80
	$PB{:}\overline{BC_0}$+Ord.	1.26	442	6.14	3048	0.00	1532.36
w-P400D7%$\bar{D}$70	$PB{:}\overline{BC_0}$	0.68	34	4.97	4529	0.00	1805.83
	$PB{:}\overline{BC_0}$+Ord.	0.68	628	6.68	9429	0.00	6031.09
w-P200D10%$\bar{D}$50	$PB{:}\overline{BC_0}$	1.65	15	5.42	32327	1.65	7200.00
	$PB{:}\overline{BC_0}$+Ord.	1.65	80	5.55	19491	1.65	7200.00
w-P400D10%$\bar{D}$50	$PB{:}\overline{BC_0}$	*3.38	129	10.59	9416	*3.38	7200.00
	$PB{:}\overline{BC_0}$+Ord.	*3.38	194	7.03	10625	*3.38	7200.00
w-P200D7%$\bar{D}$90	$PB{:}\overline{BC_0}$	*3.81	1	8.28	54078	*3.81	7200.00
	$PB{:}\overline{BC_0}$+Ord.	*3.81	57	10.48	23206	*3.81	7200.00
w-P400D7%$\bar{D}$90	$PB{:}\overline{BC_0}$	*3.18	15	9.42	13316	*3.18	7200.00
	$PB{:}\overline{BC_0}$+Ord.	*3.18	226	13.09	9753	*3.18	7200.00

5.3.2 Maximum transmission time sensitivity

The sensitivity analysis of formulation "PB" to parameter $\hat{T}$ is performed by decreasing its values progressively (the value of σ is fixed to 1 in this set of experiments). The first value of $\hat{T}$ tested corresponds to the allocated communication time between the drones when no time limit is imposed for their communication, i.e., $\hat{T}{=}{+}\infty$. From that value, $\hat{T}$ is decreased by 0.5 seconds until formulation "PB" becomes infeasible. Figure 7 and Figure 8 present results for instances "u-P200D5%$\bar{D}$70" and "w-P200D5%$\bar{D}$70". For them, the first value of $\hat{T}$ is 48 seconds, being decreased down to 23.5 when both problems become infeasible. The reported number of nodes explored by CPLEX branch-and-cut method and the optimum objective function value are obtained within 2h of computing time.

The decreasing of $\hat{T}$ tends to reduce the number of nodes explored in the branch-and-cut enumeration whereas the objective function value increases until the problem becomes infeasible. For example, in Figure 7, the optimum objective function value increases from 1870.4 to 2939.2 starting at $\hat{T} = 33.5$. The problem becomes infeasible for $\hat{T}$ smaller than 24s.

Table 7: CPLEX results when solving unweighted instances for the "$PB{:}\overline{BC}_0$", "$PB{:}\overline{BC}_0{-}b{>}y$" and "$PB{:}\overline{BC}_0{-}y{>}b$" branching priority strategies.

Instance	Form.	Branch-and-Cut		
		Nodes	gap	sec.
u-P200D5%$\bar{D}$70	$PB{:}\overline{BC}_0$	345	0.00	19.02
	$PB{:}\overline{BC}_0{-}b{>}y$	529	0.00	25.46
	$PB{:}\overline{BC}_0{-}y{>}b$	178	0.00	15.05
u-P200D7%$\bar{D}$50	$PB{:}\overline{BC}_0$	239	0.00	11.45
	$PB{:}\overline{BC}_0{-}b{>}y$	744	0.00	33.73
	$PB{:}\overline{BC}_0{-}y{>}b$	232	0.00	11.47
u-P400D5%$\bar{D}$70	$PB{:}\overline{BC}_0$	500	0.00	91.56
	$PB{:}\overline{BC}_0{-}b{>}y$	1993	0.00	255.79
	$PB{:}\overline{BC}_0{-}y{>}b$	179	0.00	50.86
u-P400D7%$\bar{D}$50	$PB{:}\overline{BC}_0$	707	0.00	133.70
	$PB{:}\overline{BC}_0{-}b{>}y$	1011	0.00	207.03
	$PB{:}\overline{BC}_0{-}y{>}b$	202	0.00	68.28
u-P200D5%$\bar{D}$90	$PB{:}\overline{BC}_0$	359	0.00	22.60
	$PB{:}\overline{BC}_0{-}b{>}y$	471	0.00	41.63
	$PB{:}\overline{BC}_0{-}y{>}b$	72	0.00	7.06
u-P200D7%$\bar{D}$70	$PB{:}\overline{BC}_0$	699	0.00	45.39
	$PB{:}\overline{BC}_0{-}b{>}y$	1096	0.00	110.24
	$PB{:}\overline{BC}_0{-}y{>}b$	393	0.00	25.89
u-P400D5%$\bar{D}$90	$PB{:}\overline{BC}_0$	49	0.00	35.02
	$PB{:}\overline{BC}_0{-}b{>}y$	812	0.00	215.49
	$PB{:}\overline{BC}_0{-}y{>}b$	49	0.00	35.16
u-P400D7%$\bar{D}$70	$PB{:}\overline{BC}_0$	7011	0.00	6665.71
	$PB{:}\overline{BC}_0{-}b{>}y$	1330	0.00	235.25
	$PB{:}\overline{BC}_0{-}y{>}b$	430	0.00	59.71
u-P200D10%$\bar{D}$50	$PB{:}\overline{BC}_0$	3546	0.00	556.97
	$PB{:}\overline{BC}_0{-}b{>}y$	2231	0.00	217.96
	$PB{:}\overline{BC}_0{-}y{>}b$	1328	0.00	144.06
u-P400D10%$\bar{D}$50	$PB{:}\overline{BC}_0$	990	0.00	219.40
	$PB{:}\overline{BC}_0{-}b{>}y$	1022	0.00	107.68
	$PB{:}\overline{BC}_0{-}y{>}b$	922	0.00	199.24
u-P200D7%$\bar{D}$90	$PB{:}\overline{BC}_0$	521	0.00	49.52
	$PB{:}\overline{BC}_0{-}b{>}y$	18419	0.00	3891.44
	$PB{:}\overline{BC}_0{-}y{>}b$	2466	0.00	557.22
u-P400D7%$\bar{D}$90	$PB{:}\overline{BC}_0$	14897	*1.96	7200.00
	$PB{:}\overline{BC}_0{-}b{>}y$	30016	*1.96	7200.00
	$PB{:}\overline{BC}_0{-}y{>}b$	7749	*1.96	7200.00

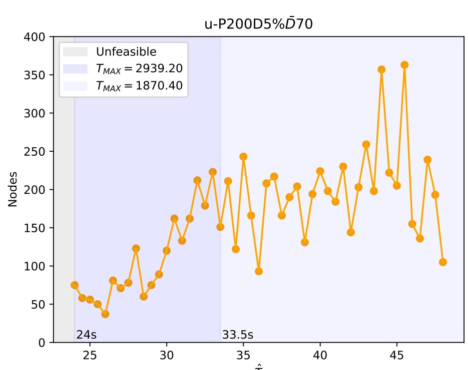

Figure 7: Number of nodes explored by CPLEX on varying $\hat{T}$ for the instance u-P200D5%$\bar{D}$70.

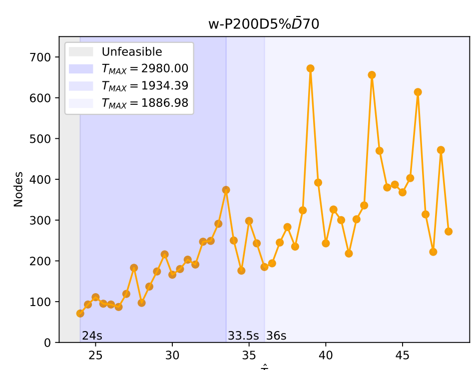

Figure 8: Number of nodes explored by CPLEX on varying $\hat{T}$ for the instance w-P200D5%$\bar{D}$70.

Table 8: CPLEX results when solving weighted instances for the "$PB{:}\overline{BC_0}$", "$PB{:}\overline{BC_0} - b{>}y$" and "$PB{:}\overline{BC_0} - y{>}b$" branching priority strategies.

Instance	Form.	Branch-and-Cut		
		Nodes	gap	sec.
w-P200D5%$\bar{D}$70	$PB{:}\overline{BC}_0$	300	0.00	16.87
	$PB{:}\overline{BC}_0{-}b{>}y$	620	0.00	38.68
	$PB{:}\overline{BC}_0{-}y{>}b$	279	0.00	20.40
w-P200D7%$\bar{D}$50	$PB{:}\overline{BC}_0$	549	0.00	31.52
	$PB{:}\overline{BC}_0{-}b{>}y$	600	0.00	28.76
	$PB{:}\overline{BC}_0{-}y{>}b$	327	0.00	29.57
w-P400D5%$\bar{D}$70	$PB{:}\overline{BC}_0$	414	0.00	79.99
	$PB{:}\overline{BC}_0{-}b{>}y$	1918	0.00	330.83
	$PB{:}\overline{BC}_0{-}y{>}b$	299	0.00	80.89
w-P400D7%$\bar{D}$50	$PB{:}\overline{BC}_0$	974	0.00	241.08
	$PB{:}\overline{BC}_0{-}b{>}y$	602	0.00	94.42
	$PB{:}\overline{BC}_0{-}y{>}b$	315	0.00	126.37
w-P200D5%$\bar{D}$90	$PB{:}\overline{BC}_0$	3427	0.00	427.78
	$PB{:}\overline{BC}_0{-}b{>}y$	4281	0.00	409.38
	$PB{:}\overline{BC}_0{-}y{>}b$	3833	0.00	560.64
w-P200D7%$\bar{D}$70	$PB{:}\overline{BC}_0$	3736	0.00	514.63
	$PB{:}\overline{BC}_0{-}b{>}y$	5879	0.00	663.39
	$PB{:}\overline{BC}_0{-}y{>}b$	4680	0.00	679.89
w-P400D5%$\bar{D}$90	$PB{:}\overline{BC}_0$	3048	0.00	1532.36
	$PB{:}\overline{BC}_0{-}b{>}y$	2189	0.00	925.85
	$PB{:}\overline{BC}_0{-}y{>}b$	3146	0.00	2052.07
w-P400D7%$\bar{D}$70	$PB{:}\overline{BC}_0$	9429	0.00	6031.09
	$PB{:}\overline{BC}_0{-}b{>}y$	1397	0.00	605.48
	$PB{:}\overline{BC}_0{-}y{>}b$	3341	0.00	2097.93
w-P200D10%$\bar{D}$50	$PB{:}\overline{BC}_0$	19491	1.65	7200.00
	$PB{:}\overline{BC}_0{-}b{>}y$	46502	1.08	7200.00
	$PB{:}\overline{BC}_0{-}y{>}b$	25588	1.65	7200.00
w-P400D10%$\bar{D}$50	$PB{:}\overline{BC}_0$	10625	*3.38	7200.00
	$PB{:}\overline{BC}_0{-}b{>}y$	10981	*3.38	7200.00
	$PB{:}\overline{BC}_0{-}y{>}b$	8520	*3.38	7200.00
w-P200D7%$\bar{D}$90	$PB{:}\overline{BC}_0$	23206	*3.81	7200.00
	$PB{:}\overline{BC}_0{-}b{>}y$	33427	*3.81	7200.00
	$PB{:}\overline{BC}_0{-}y{>}b$	29215	*3.81	7200.00
w-P400D7%$\bar{D}$90	$PB{:}\overline{BC}_0$	9753	*3.18	7200.00
	$PB{:}\overline{BC}_0{-}b{>}y$	20758	*3.18	7200.00
	$PB{:}\overline{BC}_0{-}y{>}b$	9417	*3.18	7200.00

Table 9: CPLEX results when solving unweighted instances for the PB formulation with $\sigma \in \{1, \ldots, |\bar{D}| - 1\}$.

Instance	σ	Root Node			Branch-and-Cut		
		gap$_0$	**cuts**	**sec.**	**Nodes**	**gap**	**sec.**
u-P200D5%$\bar{D}$70	1	4.76	242	1.15	345	0.00	19.02
	2	50.62	752	10.21	1113	0.00	138.18
u-P200D7%$\bar{D}$50	1	4.76	189	1.46	239	0.00	11.45
	2	50.62	1053	9.16	902	0.00	101.71
u-P400D5%$\bar{D}$70	1	1.23	183	3.38	500	0.00	91.56
	2	50.25	667	2.90	1912	0.00	539.52
u-P400D7%$\bar{D}$50	1	1.23	188	4.31	707	0.00	133.70
	2	50.25	366	2.80	3070	0.00	1723.51
u-P200D5%$\bar{D}$90	1	0.00	227	1.99	359	0.00	22.60
	2	50.00	228	3.47	14578	0.00	5685.71
	3	66.67	194	2.46	2392	0.00	1014.65
u-P200D7%$\bar{D}$70	1	0.00	226	2.33	699	0.00	45.39
	2	*50.00	119	3.71	16579	*33.33	7200.00
	3	66.67	402	3.52	2209	0.00	1150.97
u-P400D5%$\bar{D}$90	1	0.00	267	5.00	49	0.00	35.02
	2	*50.00	349	6.82	3573	*33.33	7200.00
	3	66.67	656	5.65	2631	0.00	4593.69
u-P400D7%$\bar{D}$70	1	0.00	175	6.23	7011	0.00	6665.71
	2	*50.00	403	6.59	3019	*49.50	7200.00
	3	*66.67	749	7.69	3275	*11.11	7200.00
u-P200D10%$\bar{D}$50	1	0.00	312	4.99	3546	0.00	556.97
	2	50.00	522	8.13	11659	16.67	7200.00
	3	*67.48	202	9.37	9759	*59.32	7200.00
	4	75.00	2419	967.58	1145	26.00	7200.00

Table 10: CPLEX results when solving weighted instances for the PB formulation with $\sigma \in \{1, \ldots, |\bar{D}| - 1\}$.

Instance	σ	Root Node			Branch-and-Cut		
		gap$_0$	**cuts**	**sec.**	**Nodes**	**gap**	**sec.**
w-P200D5%$\bar{D}$70	1	3.36	93	1.25	300	0.00	16.87
	2	50.71	831	9.81	1663	0.00	172.46
w-P200D7%$\bar{D}$50	1	3.67	111	1.49	549	0.00	31.52
	2	51.44	670	11.53	4013	0.00	665.60
w-P400D5%$\bar{D}$70	1	0.86	190	3.48	414	0.00	79.99
	2	50.34	480	2.74	1652	0.00	891.45
w-P400D7%$\bar{D}$50	1	2.02	264	3.40	974	0.00	241.08
	2	50.29	566	3.15	2819	0.00	1557.25
w-P200D5%$\bar{D}$90	1	2.96	84	2.30	3427	0.00	427.78
	2	50.00	293	3.73	12045	32.56	7200.00
	3	*67.03	289	4.70	20671	*1.09	7200.00
w-P200D7%$\bar{D}$70	1	2.69	87	3.01	3736	0.00	514.63
	2	50.00	428	3.89	6632	0.00	2647.32
	3	*66.82	382	3.03	25358	*0.46	7200.00
w-P400D5%$\bar{D}$90	1	1.26	442	6.14	3048	0.00	1532.36
	2	*50.01	381	4.95	3415	*48.87	7200.00
	3	*66.86	430	10.48	2994	*11.63	7200.00
w-P400D7%$\bar{D}$70	1	0.68	628	6.68	9429	0.00	6031.09
	2	*50.00	387	7.68	2605	*49.52	7200.00
	3	*66.78	650	6.11	2089	*33.02	7200.00

6 Conclusion

A swarm of drones (UAVs) can be used to automate a wide range of missions, from surveillance to search and rescue, from 3D mapping to telecommunication enhancement. While UAVs are typically responsible for the mission phases related to data collection — thanks to their flying capabilities and to the availability of embedded sensors — most of the data processing is offloaded to dedicated machines (virtual or bare-metal) placed in the cloud. However, when the communication bandwidth between the swarm and the cloud is limited, an ad-hoc cloud established on top of the UAVs' computing resources (and those of other elements available in the area) can be leveraged to replace the cloud and keep data processing local.

For the purpose of optimizing the use of such ad-hoc cloud infrastructure powered by the swarming UAVs, we introduced a new optimization problem, namely the Covering-Assignment Problem for swarm-powered ad-hoc clouds - CAPsac, based on a real-life use-case in the emergency management field: swarm-powered distributed 3D reconstruction for humanitarian emergency response. After having established the relationship between the general problem and the specific use-case, we presented the NP-Hardness proof of the CAPsac and described two MILP formulations for it.

Given a set of geo-positioned aerial pictures (data) that are subject to geolocation/clustering constraints, CAPsac minimizes the 3D mapping (data-processing phase) completion time by jointly computing: (i) the optimal covering of photos (workload configuration), and (ii) the optimal assignment of photographed sub-regions (workload assignment) to UAVs (computing elements). Besides being a way to provide optimal solutions for the problem, our integrated decision model contrasts with the *decompose-then-allocate* and the *allocate-then-decompose* paradigms usually seen in (both the cloud computing optimization and) the multi-robot task allocation literature. Finally, modeling CAPsac in this way is flexible and amendable to take into account any other additional ground computing elements connected to the swarm itself.

In order to assess the proposed formulations, a series of computational experiments was conducted with a set of unweighted and weighted realistic benchmark instances available online (https://github.com/ds4dm/CAPsac). The experiments revealed that the photo-based formulation "PB" was more efficient by using ordering inequalities that remove from the feasible continuous search space those sub-regions whose boundaries are not regular (e.g. left boundary at the right of a right boundary). However, the different branching priority strategies and row generation methods have not proven to yield a performance gain while solving "PB". Column Generation was employed in the region-based formulation "RB", but the presence of highly degenerate optimums led to long execution times.

Finally, the sensitivity analysis of the formulation "PB" showed that it becomes more difficult to solve as the reliability factor σ increases. Tests with varying values for the maximum allowed transmission time $\hat{T}$ also presented a slight gain of performance as $\hat{T}$ approaches a limit when the problem becomes infeasible.

References

[1] Y. Tan and Z. yang Zheng, Research advance in swarm robotics, Defence Technology, 9(1):18–39, 2013.

[2] F. Nex and F. Remondino, Uav for 3d mapping applications: a review, Applied Geomatics, 6(1):1–15, 2014.

[3] J. A. J. Berni, P. J. Zarco-Tejada, L. Suarez, and E. Fereres, Thermal and narrowband multispectral remote sensing for vegetation monitoring from an unmanned aerial vehicle, IEEE Transactions on Geoscience and Remote Sensing, 47(3):722–738, March 2009.

[4] G. Grenzdörffer, A. Engel, and B. Teichert, The photogrammetric potential of low-cost uavs in forestry and agriculture, The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XXXVII(B1):1207–1214, 2008.

[5] J. Martínez-de Dios, L. Merino, F. Caballero, A. Ollero, and D. Viegas, Experimental results of automatic fire detection and monitoring with uavs, Forest Ecology and Management, 234(1):S232, 2006.

[6] F. Chiabrando, F. Nex, D. Piatti, and F. Rinaudo, Uav and rpv systems for photogrammetric surveys in archaelogical areas: two tests in the piedmont region (italy), Journal of Archaeological Science, 38(3):697–710, 2011.

[7] K. Lambers, H. Eisenbeiss, M. Sauerbier, D. Kupferschmidt, T. Gaisecker, S. Sotoodeh, and T. Hanusch, Combining photogrammetry and laser scanning for the recording and modelling of the late intermediate period site of pinchango alto, palpa, peru, Journal of Archaeological Science, 34(10):1702–1712, 2007.

[8] M. Oczipka, J. Bemmann, H. Piezonka, J. Munkabayar, B. Ahrens, M. Achtelik, and F. Lehmann, Small drones for geo-archaeology in the steppes: locating and documenting the archaeological heritage of the orkhon valley in mongolia, in Remote Sensing for Environmental Monitoring, GIS Applications, and Geology IX, U. Michel and D. L. Civco, Eds., vol. 7478, International Society for Optics and Photonics. SPIE, 2009, pp. 53–63.

[9] F. Rinaudo, F. Chiabrando, A. M. Lingua, and A. Spanò, Archaeological site monitoring: Uav photogrammetry can be an answer, ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XXXIX-B5:583–588, 2012.

[10] G. J. J. Verhoeven, Providing an archaeological bird's-eye view – an overall picture of ground-based means to execute low-altitude aerial photography (laap) in archaeology, Archaeological Prospection, 16(4):233–249, 2009.

[11] W. Hartmann, S. Tilch, H. Eisenbeiss, and K. Schindler, Determination of the uav position by automatic processing of thermal images, ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XXXIX-B6:111–116, 2012.

[12] M. Manyoky, P. Theiler, D. Steudler, and H. Eisenbeiss, Unmanned aerial vehicle in cadastral applications, ISPRS-international archives of the photogrammetry, remote sensing and spatial information sciences, XXXVIII-1/C22:57–62, 2011.

[13] U. Niethammer, S. Rothmund, M. James, J. Travelletti, and M. Joswig, Uav-based remote sensing of landslides, International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 38(Part 5):496–501, 2010.

[14] J. G. Smith, J. Dehn, R. P. Hoblitt, R. G. LaHusen, J. B. Lowenstern, S. C. Moran, L. McClelland, K. A. McGee, M. Nathenson, P. G. Okubo, J. S. Pallister, M. P. Poland, J. A. Power, D. J. Schneider, and T. W. Sisson, Volcano monitoring, in Geological Monitoring. Geological Society of America, 2009, pp. 273–305.

[15] C. Zhang, An uav-based photogrammetric mapping system for road condition assessment, ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XXXVII-B5:627–632, 2008.

[16] T.-Y. Chou, M.-L. Yeh, Y. C. Chen, and Y. H. Chen, Disaster monitoring and management by the unmanned aerial vehicle technology, in International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 2010, 38(7B):137–142.

[17] R. Haarbrink and E. Koers, Helicopter uav for photogrammetry and rapid response, in 2nd Int. Workshop "The Future of Remote Sensing", ISPRS Inter-Commission Working Group I/V Autonomous Navigation, vol. 1. Citeseer, 2006.

[18] P. Molina, I. Colomina, T. Victoria, J. Skaloud, W. Kornus, R. Prades, and C. Aguilera, Searching lost people with uavs: The system and results of the close-search project, in International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 39, no. EPFL-CONF-182482, 2012, pp. 441–446.

[19] P. Zarco-Tejada, R. Diaz-Varela, V. Angileri, and P. Loudjani, Tree height quantification using very high resolution imagery acquired from an unmanned aerial vehicle (uav) and automatic 3d photo-reconstruction methods, European Journal of Agronomy, 55:89–99, 2014.

[20] J. Bendig, A. Bolten, S. Bennertz, J. Broscheit, S. Eichfuss, and G. Bareth, Estimating biomass of barley using crop surface models (csms) derived from uav-based rgb imaging, Remote Sensing, 6(11):10 395–10 412, 2014.

[21] R. A. Díaz-Varela, R. De la Rosa, L. León, and P. J. Zarco-Tejada, High-resolution airborne uav imagery to assess olive tree crown parameters using 3d photo reconstruction: Application in breeding trials, Remote Sensing, 7(4):4213–4232, 2015.

[22] A. Otto, N. Agatz, J. Campbell, B. Golden, and E. Pesch, Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey, Networks, 0(0), 2018.

[23] W. P. Coutinho, M. Battarra, and J. Fliege, The unmanned aerial vehicle routing and trajectory optimisation problem, a taxonomic review, Computers & Industrial Engineering, 120:116–128, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0360835218301815

[24] P. B. Sujit and D. Ghose, Search using multiple uavs with flight time constraints, IEEE Transactions on Aerospace and Electronic Systems, 40(2):491–509, April 2004.

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

[26] H. Oh, H.-S. Shin, S. Kim, A. Tsourdos, and B. A. White, Cooperative Mission and Path Planning for a Team of UAVs. Dordrecht: Springer Netherlands, 2015, pp. 1509–1545.

[27] P. Lanillos, S. K. Gan, E. Besada-Portas, G. Pajares, and S. Sukkarieh, Multi-uav target search using decentralized gradient-based negotiation with expected observation, Information Sciences, 282:92–110, 2014.

[28] S. K. Gan and S. Sukkarieh, Multi-uav target search using explicit decentralized gradient-based negotiation, in Robotics and Automation (ICRA), 2011 IEEE International Conference on. IEEE, 2011, pp. 751–756.

[29] X. Ji, X. Wang, Y. Niu, and L. Shen, Cooperative search by multiple unmanned aerial vehicles in a nonconvex environment, Mathematical Problems in Engineering, 2015:1–19, 2015.

[30] Z. Tang and U. Ozguner, Motion planning for multitarget surveillance with mobile sensor agents, IEEE Transactions on Robotics, 21(5):898–908, Oct 2005.

[31] S. Karaman and G. Inalhan, Large-scale task/target assignment for uav fleets using a distributed branch and price optimization scheme, IFAC Proceedings Volumes, 41(2):13 310 – 13 317, 2008, 17th IFAC World Congress.

[32] M. Niccolini, M. Innocenti, and L. Pollini, Multiple uav task assignment using descriptor functions, IFAC Proceedings Volumes, 43(15):93–98, 2010.

[33] A. Viguria, I. Maza, and A. Ollero, Distributed service-based cooperation in aerial/ground robot teams applied to fire detection and extinguishing missions, Advanced Robotics, 24(1–2):1–23, 2010.

[34] H. Choi, Y. Kim, and H. Kim, Genetic algorithm based decentralized task assignment for multiple unmanned aerial vehicles in dynamic environments, International Journal Aeronautical and Space Sciences, 12(2):163–174, 2011.

[35] A. Barrientos, J. Colorado, J. d. Cerro, A. Martinez, C. Rossi, D. Sanz, and J. Valente, Aerial remote sensing in agriculture: A practical approach to area coverage and path planning for fleets of mini aerial robots, Journal of Field Robotics, 28(5):667–689, 2011.

[36] S. Moon, E. Oh, and D. H. Shim, An integral framework of task assignment and path planning for multiple unmanned aerial vehicles in dynamic environments, Journal of Intelligent & Robotic Systems, 70(1):303–313, Apr 2013.

[37] S. Moon, D. H. Shim, and E. Oh, Cooperative Task Assignment and Path Planning for Multiple UAVs. Dordrecht: Springer Netherlands, 2015, pp. 1547–1576.

[38] M. Turpin, N. Michael, and V. Kumar, Capt: Concurrent assignment and planning of trajectories for multiple robots, The International Journal of Robotics Research, 33,(1):98–112, 2014.

[39] J. J. Enright, E. Frazzoli, M. Pavone, and K. Savla, UAV Routing and Coordination in Stochastic, Dynamic Environments. Dordrecht: Springer Netherlands, 2015, pp. 2079–2109.

[40] A. Sadeghi and S. L. Smith, Heterogeneous task allocation and sequencing via decentralized large neighborhood search, Unmanned Systems, 5(02):79–95, 2017.

[41] D. Zorbas, L. D. P. Pugliese, T. Razafindralambo, and F. Guerriero, Optimal drone placement and cost-efficient target coverage, Journal of Network and Computer Applications, 75:16–31, 2016.

[42] P. Ladosz, H. Oh, and W.-H. Chen, Trajectory planning for communication relay unmanned aerial vehicles in urban dynamic environments, Journal of Intelligent & Robotic Systems, 89(1):7–25, Jan 2018.

[43] L. Caraballo, J. Díaz-Báñez, I. Maza, and A. Ollero, The block-information-sharing strategy for task allocation: A case study for structure assembly with aerial robots, European Journal of Operational Research, 260(2):725–738, 2017.

[44] A. Grancharova, E. I. Grøtli, D.-T. Ho, and T. A. Johansen, Uavs trajectory planning by distributed mpc under radio communication path loss constraints, Journal of Intelligent & Robotic Systems, 79(1):115–134, 2015.

[45] S. Koulali, E. Sabir, T. Taleb, and M. Azizi, A green strategic activity scheduling for uav networks: A sub-modular game perspective, IEEE Communications Magazine, 54(5):58–64, May 2016.

[46] S. Xu, K. Doğançay, and H. Hmam, Distributed pseudolinear estimation and uav path optimization for 3d aoa target tracking, Signal Processing, 133:64–78, 2017.

[47] M. A. Gomarasca, Elements of Photogrammetry. Dordrecht: Springer Netherlands, 2009, ch. 3, pp. 79–121.

[48] W. Linder, Digital Photogrammetry. Springer Berlin Heidelberg, 2016.

[49] P. Doherty, J. Kvarnström, P. Rudol, M. Wzorek, G. Conte, C. Berger, T. Hinzmann, and T. Stastny, A collaborative framework for 3d mapping using unmanned aerial vehicles, in International Conference on Principles and Practice of Multi-Agent Systems. Springer, 2016, pp. 110–130.

[50] G. Loianno, Y. Mulgaonkar, C. Brunner, D. Ahuja, A. Ramanandan, M. Chari, S. Diaz, and V. Kumar, A swarm of flying smartphones, in Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on. IEEE, 2016, pp. 1681–1688.

[51] J. Schmiemann, H. Harms, J. Schattenberg, M. Becker, S. Batzdorfer, and L. Frerichs, A distributed online 3d-lidar mappin system, ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLII-2/W6:339–346, 2017.

[52] K. Kobayashi, H. Shishido, Y. Kameda, and I. Kitahara, Method to generate disaster-damage map using 3d photometry and crowd sourcing, in Big Data (Big Data), 2017 IEEE International Conference on. IEEE, 2017, pp. 4397–4399.

[53] S. Golodetz, T. Cavallari, N. A. Lord, V. A. Prisacariu, D. W. Murray, and P. H. Torr, Collaborative large-scale dense 3d reconstruction with online inter-agent pose optimisation, arXiv preprint arXiv:1801.08361, 2018.

[54] S. Milani and A. Memo, Impact of drone swarm formations in 3d scene reconstruction, in 2016 IEEE International Conference on Image Processing (ICIP). IEEE, Sep. 2016, pp. 2598–2602.

[55] W. Chen, B. Liu, H. Huang, S. Guo, and Z. Zheng, When UAV Swarm Meets Edge-Cloud Computing: The QoS Perspective, IEEE Network, 33(2):36–43, mar 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8675170/

[56] M. Hamdaqa, M. M. Sabri, A. Singh, and L. Tahvildari, Adoop: MapReduce for Ad-hoc Cloud Computing, Proceedings of the 25th Annual International Conference on Computer Science and Software Engineering, pp. 26–34, 2015. [Online]. Available: http://dl.acm.org/citation.cfm?id=2886444.2886449

[57] A. Malhotra, S. K. Dhurandher, and B. Kumar, Resource allocation in multi-hop Mobile Ad hoc cloud, in 2014 Recent Advances in Engineering and Computational Sciences (RAECS). IEEE, mar 2014, pp. 1–6. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6799578

[58] I. Yaqoob, E. Ahmed, A. Gani, S. Mokhtar, M. Imran, and S. Guizani, Mobile ad hoc cloud: A survey, Wireless Communications and Mobile Computing, pp. 421–430, 2016. [Online]. Available: http://doi.wiley.com/10.1002/wcm.2709

[59] CBC/Radio-Canada. (2020) Émission découverte: Drones humanitaires. [Online]. Available: https://ici.radio-canada.ca/tele/decouverte/site/segments/reportage/139538/drones-humanitaires

[60] W. Gay, Raspberry Pi Hardware Reference, 1st ed. USA: Apress, 2014.

[61] L. G. Gianoli, April 2020, private communication.

[62] A. Khamis, A. Hussein, and A. Elmogy, Multi-robot task allocation: A review of the state-of-the-art, in Cooperative Robots and Sensor Networks 2015. Springer, 2015, pp. 31–51.

[63] I. Bekmezci, O. K. Sahingoz, and S. Temel, Flying Ad-Hoc Networks (FANETs): A survey, Ad Hoc Networks, 11(3):1254–1270, may 2013.

[64] L. Massoulié and J. Roberts, Bandwidth sharing: Objectives and algorithms, in INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, vol. 3. IEEE, 1999, pp. 1395–1403.

[65] D. P. Bertsekas, R. G. Gallager, and P. Humblet, Data networks. Prentice-Hall International New Jersey, 1992, vol. 2.

[66] G. P. McCormick, Computability of global solutions to factorable nonconvex programs: Part i—convex underestimating problems, Mathematical programming, 10(1):147–175, 1976.

[67] M. Pepe, L. Fregonese, and M. Scaioni, Planning airborne photogrammetry and remote-sensing missions with modern platforms and sensors, European Journal of Remote Sensing, 51(1):412–436, 2018.

[68] D. Nace and M. Pióro, Max-min fairness and its applications to routing and load-balancing in communication networks: a tutorial, IEEE Communications Surveys & Tutorials, 10(4), 2008.

[69] E. Amaldi, A. Capone, S. Coniglio, and L. G. Gianoli, Network optimization problems subject to max-min fair flow allocation, IEEE Communications Letters, 17(7):1463–1466, 2013.

[70] R. J. Fowler, M. S. Paterson, and S. L. Tanimoto, Optimal packing and covering in the plane are np-complete, Information Processing Letters, 12(3):133–137, 1981.

[71] D. S. Johnson, The np-completeness column: An ongoing guide, Journal of Algorithms, 3(2):182–195, 1982.

[72] G. Desaulniers, J. Desrosiers, and M. M. Solomon, Eds., Column Generation. Springer US, 2005. [Online]. Available: https://doi.org/10.1007/b135457

[73] R. Jain, The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling. New York: Wiley, 1991.