CIRRELT

**CIRRELT-2024-28**

# The Integrated Agile Earth Observation Satellite Scheduling Problem

**Yure Rocha**
**Gulherme O. Chagas**
**Leandro C. Coelho**
**Anand Subramanian**

**August 2024**

Document de travail également publié par la Faculté des sciences de l'administration de l'Université Laval, sous le numéro FSA-2024-004

**Bureau de Montréal**
Université de Montréal
C.P. 6128, succ. Centre-Ville
Montréal (Québec) H3C 3J7
Tél : 1-514-343-7575
Télécopie : 1-514-343-7121

**Bureau de Québec**
Université Laval,
2325, rue de la Terrasse
Pavillon Palasis-Prince, local 2415
Québec (Québec) G1V 0A6
Tél : 1-418-656-2073
Télécopie : 1-418-656-2624

# The Integrated Agile Earth Observation Satellite Scheduling Problem

**Yure Rocha[1], Gulherme O. Chagas[2], Leandro C. Coelho[2,3,*], Anand Subramanian[1]**

1. Universidade Federal da Paraíba, Departamento de Sistemas de Computação, Centro de Informática, Rua dos Escoteiros s/n, Mangabeira, 58055-000, João Pessoa, Brazil
2. Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) and Department of Operations and Decision Systems, Université Laval, Québec, Canada
3. Canada Research Chair in Integrated Logistics, Université Laval, Québec, Canada

**Abstract.** Satellites are widely used for different purposes, such as monitoring deforestation, rising sea levels, and ecological changes. As a scarce resource, scheduling the activities performed by these spacecrafts is of paramount importance. This paper addresses the integrated agile Earth observation satellite scheduling problem (IAEOSSP) that employs a constellation of agile satellites to observe the Earth. The objective of the IAEOSSP is to maximize the profits associated with observed targets and the amount of data downloaded to ground stations. Critical aspects of the problem are considered, such as the onboard storage capacity, time windows, setup times, and energy dynamics. We propose a mixed-integer programming (MIP) formulation for the problem, as well as an improved MIP model that is computationally more tractable at the expense of possibly reducing the feasible search space. Moreover, we implement a MIP-based heuristic (MBH) to solve the proposed model with an efficient heuristic strategy for decreasing the size of the graph. Computational experiments evaluate the performance of the MBH not only on the IAEOSSP but also on a particular case involving conventional satellites denoted as constellation mission scheduling problem (CMSP). The results show that the MBH compares favorably against the improved MIP model for the IAEOSSP and against two existing algorithms for the CMSP.

**Keywords**: Integrated satellite scheduling, Earth observation satellite, Agile satellites, Conventional satellites, MIP-based heuristic.

**Acknowledgements.** This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) [grants 2021-00094], as well as the Brazilian research agencies CNPq [grants 406245/2021-5, and 309580/2021-8], and Paraíba State Research Foundation (FAPESQ) [grant 2021/3182]. We thank Dr. Doo-Hyun Cho for sharing the data and implementation associated with the article Cho et al. (2018).



---

* Corresponding author: leandro.coelho@fsa.ulaval.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2024

## 1. Introduction

*Earth observation satellites* (EOSs) are used to monitor and better understand land environments, water, and the atmosphere. They are equipped with different sensors, depending on the purpose of the observation. The data acquired by the satellites are processed to produce information that can serve various applications, including surveillance, weather forecasting, and agriculture. EOSs are strategic for both the public and private sectors. The Government of Canada, for example, has recently allocated over CA$1 billion to support *Earth observation* (EO) using the RADARSAT+ initiative (CSA, 2023), and the European Union estimates a global revenue rise of about €2.6 billion from 2023 to 2033 on the EO market (EUASP, 2024).

The satellite observation scheduling and the communication scheduling problems are two $\mathcal{NP}$-hard problems (Bensanna et al., 1996, Barbulescu et al., 2004) involving *low Earth orbit* (LEO) satellites, largely investigated in the literature. The integrated version of the satellite scheduling problem consists of both observation and communication (more specifically, download) activities simultaneously. The type of satellites further defines the problem: agile and non-agile (or conventional). *Agile* EOSs (AEOSs) have extra pitch and yaw mobility, as opposed to *conventional* EOSs (CEOSs), which can only maneuver on the roll axis (Lemaître et al., 2002). While the *integrated AEOS scheduling problem* (IAEOSSP) requires the selection and timetabling of observation and download tasks, the version with conventional satellites is a particular case where only the selection of tasks is considered.

The IAEOSSP we address considers a fleet (or constellation) of homogeneous agile satellites with maneuverability on the pitch and roll axes, a set of targets, and a set of homogeneous ground stations. Figure 1 illustrates the observation ranges of both angles.

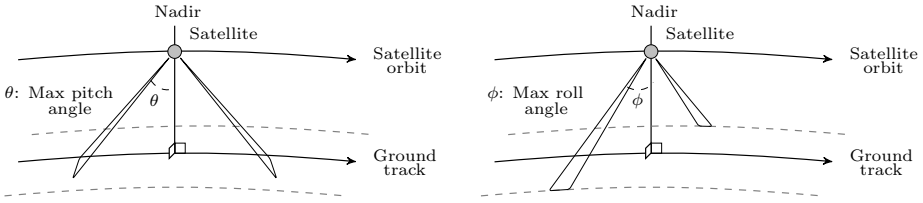


Figure 1: Maximum pitch and roll angles of a satellite.

Civil or military customers send target requests daily to the mission center, which assigns them priorities or rewards according to their importance. A request comprises the target coordinates and their length (or observation time), which can be completely imaged at a given time. *Observation time windows* (OTWs) and *download time windows* (DTWs) are then computed in a preprocessing step to determine when targets can be collected

and downloads to ground stations can be scheduled. This preprocessing also computes the rolling angle for the satellite according to its orbit and the target/ground station coordinates. The satellite must change its pitch and roll angles from one activity to the next, requiring setup times. A ground station also needs setup time to adjust its antenna between consecutive download operations.

After the image of a target is obtained, it is saved to the satellite onboard storage to be eventually downloaded to a ground station and sent to the mission center. Such observation and download operations consume energy, harvested by the satellites via solar panels in the sun zones. The objective of the problem is to select and schedule the observation tasks that maximize the rewards while downloading as much data as possible while satisfying time windows, storage, and energy constraints.

Figure 2 depicts a partial solution to the IAEOSSP. Let the energy harvested by the solar panels be sufficient to perform all selected tasks and let the maximum storage capacity of the satellite be 15 units of data. The satellite starts observing target 1 at time $t_1$. After 7.5 units of time, the target is acquired, and the satellite starts changing its pose to capture target 2 at time $t_2$. By the end of such operation, the storage has 15 units of data, downloaded to ground station $g$ at time $t_3$, after a setup time. Target 3 cannot be scheduled because the satellite has no storage capacity left. The partial solution finishes after the satellite captures target 4, which starts at $t_4$, and the satellite finishes the partial solution with 7.5 units of data stored.

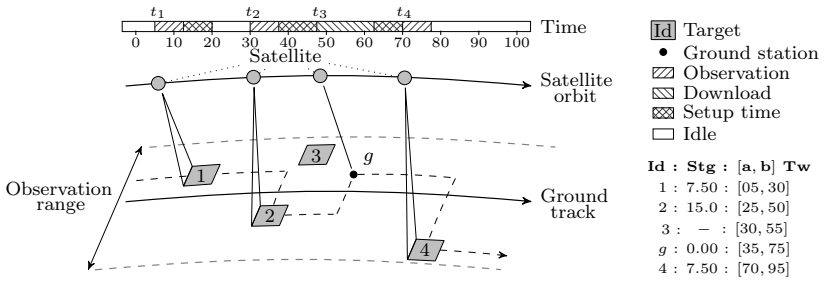


Figure 2: Example of a partial solution regarding time windows, setup times, changes in pose, and data storage.

The contributions of this paper are manifold and listed as follows.

- We propose a novel satellite scheduling problem with relevant real-world features: simultaneous scheduling of observation and download tasks (which are highly interconnected in practice), agile satellites, energy dynamics, and transition times for satellites and ground stations. The integrated scheduling makes the IAEOSSP a chal-

lenging problem. Moreover, agile satellites introduce further complexity by increasing image acquisition and data download opportunities.

- We formulate the IAEOSSP as a mixed-integer linear programming problem (MILP) on a directed, weighted, and connected graph.
- We provide an improved formulation in which variables and constraints associated with the setup times of the satellites are removed. Instead, we insert new constraints that rely on the triangle inequality to model transition times. Although such a scheme possibly discards feasible solutions for the sake of computational tractability, our computational experiments revealed that its implementation did not significantly affect the solution quality while keeping the runtimes considerably lower.
- We devise an efficient strategy that replaces arcs between distant nodes in the graph for new arcs that involve dummy nodes. The new arcs and dummy nodes can be inserted in a particular way that preserves the original solution space of the instance. The efficient strategy can reduce the number of arcs from $\mathcal{O}(n^2)$ down to $\mathcal{O}(n)$, where $n$ is the number of nodes in the graph.
- We develop a MIP-based heuristic (MBH) to solve the IAEOSSP. The MBH procedure allows for a controlled exploration of the solution space, achieved using neighborhood structures that fix and release the bounds of variables associated with the arcs in the graph. The heuristic obtained competitive solutions compared with the MIP solver applied to the IAEOSSP and existing algorithms for a related problem.

The remainder of this paper is organized as follows. Section 2 reviews the related literature. Section 3 contains the mathematical formulation of the problem. Section 4 details improvements achieved in the model by both the efficient strategy and an improved formulation. Section 5 presents the proposed algorithm and neighborhood structures. Section 6 describes the results of computational experiments on a new set of benchmark instances and from a similar problem from the literature. Finally, Section 7 concludes and provides future research directions.

## 2. Literature review

Many studies involving satellite scheduling have been conducted over the past few decades. They mostly differ in objective (e.g., to schedule observation tasks, communication activities, or both simultaneously), number of satellites and ground stations, and

constraints adopted. Most articles have studied the so-called *Earth observation satellite scheduling problem* (EOSSP), a simplified version of the IAEOSSP that considers only observation activities with conventional satellites. This section reviews the problems closely related to ours, namely involving agile satellites.

### *2.1. Earth observation satellite scheduling problem with agile satellites*

Agile satellites have mobility on up to three axes: pitch, roll, and yaw. Lemaître et al. (2002) is one of the seminal works to have adopted AEOSs in the scheduling of observations. They studied a general problem addressing, for example, spot targets (that can be imaged in a single shot), large polygonal areas (which cannot be photographed in a single shot), setup times, and meteorological uncertainties. They also investigated methods for a simplified version of the problem, including constraint programming and local search algorithms.

A related problem is further studied by Bianchessi et al. (2007), namely the multiple satellite and multiple orbit problem. The authors described a TS heuristic for solving the problem and a column generation-based procedure to determine upper bounds on the profit. They evaluated the performance of the heuristic approach on data sets provided by the Centre National d'Études Spatiales (CNES) in France.

The AEOSSP with time-dependent setup time was addressed by Liu et al. (2017). The authors mathematically formulated the problem and solved a simplified version of the model with CPLEX and constraint satisfaction programming, whereas the full version was solved by employing an adaptive large neighborhood search (ALNS) metaheuristic. Peng et al. (2019, 2020) considered, in addition to time-dependent setup time, time-dependent profit. Such class of problems associate the profit with the quality of the photographs, which mostly depends on the camera's angle. The authors applied dynamic programming-based methods to solve the problems in both articles. For a comprehensive review of AEOS scheduling problems (AEOSSPs), the reader is referred to Wang et al. (2020).

### *2.2. Integrated satellite scheduling problem*

Few works in the literature have addressed the integrated scheduling of observation and download tasks. Bianchessi and Righini (2008) explored a variant involving conventional satellites, large polygon areas, and different transmission channels, among others. The authors devised a constructive procedure with look-ahead and back-tracking features, designed to be executed daily with a planning horizon of a couple of days.

Sarkheyli et al. (2013) considered an integrated multi-satellite EOSSP with constraints such as power capacity and cloud coverage. They modeled the problem using graph coloring theory and employed a TS procedure to solve it. To validate their approach, the authors

conducted experiments with different algorithms, including a GA and a hill-climbing heuristic. Waiming et al. (2019) tackled a problem involving spot targets, relay satellites (for data transmission to ground stations), and ground stations equipped with multiple antenna resources. The problem was formulated as an MILP, and solved using a two-phase genetic annealing algorithm.

Hu et al. (2019) studied the EOS constellation imaging and downloading integrated scheduling problem (EOSCIDISP). They first presented a period-splitting mechanism to divide the planning horizon and reduce the number of variables and constraints. Then, they formulated the EOSCIDISP as an MILP, proposed a branch-and-price algorithm for solving it, and performed experiments on real-world and synthetic instances. Similarly, Zhang and Xing (2022) also proposed an MILP to an EOSSP variant, solved using an improved GA. They used two test cases to demonstrate the algorithm's performance.

The constellation mission scheduling problem (CMSP), proposed by Cho et al. (2018), adopts a fleet of conventional satellites. The CMSP considers energy constraints and aims to maximize the sum of profits associated with observed targets and the total amount of data downloaded. To solve it, the authors implemented two models: the first tries to find candidate time intervals for download operations, which are then used as constraints in the second model. Their solution approach was compared against a first-in-first-out (FIFO) heuristic on scenarios whose main parameters were derived from Terrasar-X and KOMPSAT-3A satellites. The data of this paper is one of the few to be made available in the literature, and we use this problem as a benchmark for our methods.

Regarding IAEOSSP-like variants, Wang et al. (2011) addressed a problem with large polygon areas in which the rewards associated with observations are computed with a piecewise linear function. Transition times between consecutive observation or download activities consider the differences in pitch and roll angles. Moreover, energy-related constraints are checked at the end of each orbit and accommodate observation tasks and changes in roll angles. Although the problem was initially formulated as a nonlinear model, the authors conducted preliminary experiments using ILOG CPLEX and ILOG CP on a simplified linear version. Finally, the original problem was solved using a deterministic priority-based conflict-avoided heuristic.

Chang et al. (2021) adopted satellites with mobility on yaw, roll, and pitch angles. Among the formulations they introduced, only the coordinated, integrated scheduling framework considers the scheduling of both observation and download tasks simultaneously. Their research incorporated energy consumption into a bi-objective function, but setup times were ignored for satellite download operations. To solve the problem, they

employed a bi-objective memetic algorithm, combining ALNS and GA, with additional destroy and repair operators to enhance performance.

It is evident that the literature lacks studies involving the integrated simultaneous scheduling of Earth observation tasks and download activities with agile satellites. Indeed, even when scheduling is investigated, the proposed models seem impractical. As a result, authors often neglect to properly incorporate important features of the problem, such as energy dynamics and setup times. Our work fills these gaps by formulating the challenging IAEOSSP as an MILP model, considering critical aspects of the problem. We also develop a MIP-based heuristic (MBH) to obtain high-quality solutions in a reasonable time.

## 3. Mathematical formulation

In this section, we provide a formal definition of the IAEOSSP. We first present the assumptions made to describe the problem. Section 3.1 introduces the mathematical notation utilized throughout this work. We propose a mixed-integer nonlinear programming model to address the IAEOSSP in Section 3.2, followed by its linearization in Section 3.3.

We consider the following assumptions: $(i)$ tasks cannot be preempted, that is, stopped or paused, once initiated; $(ii)$ targets can be collected with a single observation; $(iii)$ ground stations can serve at most one satellite at a time and, similarly, satellites can perform at most one operation (observation or download) at a time; $(iv)$ as in Wang et al. (2011), satellite setup times are a function of the changes in pitch and roll angles, and there are no overlaps between observation and download time windows; $(v)$ satellites are equipped with a single optical instrument; $(vi)$ the energy harvested by the solar panels in the sun zones during opportunity time windows is disregarded.

### *3.1. Notation*

Let $S = \{i_1, \ldots, i_m\}$ be the set of homogeneous satellites, $O = \{j_1, \ldots, j_o\}$ the set of observation spots, and $G = \{g_1, \ldots, g_k\}$ the set of homogeneous ground stations. We use the term target to refer to an observation spot or a ground station. Therefore, $T = G \cup O$ denotes the set of targets and $|T| = c$. Each target $\tau \in T$ is associated with a non-negative profit $\rho_\tau$ and a required duration $\delta_\tau$ for capturing or downloading the data. The planning horizon is evenly divided into $h$ days, for which $[0, H]$ represents its start and end times.

As a satellite continuously orbits the Earth, it can interact with a target multiple times throughout the day. Each potential interaction represents an opportunity time window where the target becomes visible to the satellite. We refer to an observation time window (OTW) for an observation spot and a download time window (DTW) for

a ground station. Hence, for each satellite $i \in S$ and each target $\tau \in T$, we define $\Gamma_i^\tau = \{(a_{i\tau}^1, b_{i\tau}^1), \ldots, (a_{i\tau}^{r_{i\tau}}, b_{i\tau}^{r_{i\tau}})\}$ as the set of opportunity time windows, where target $\tau \in T$ is visible for satellite $i$ to interact with (observe if $\tau \in O$ or download if $\tau \in G$) and $0 \leq a_{i\tau}^\gamma < b_{i\tau}^\gamma \leq H$ for $\gamma = 1, \ldots, r_{i\tau}$. Moreover, consider $\Gamma_i = \bigcup_{\tau \in T} \Gamma_i^\tau$ as the set of all time windows in which satellite $i \in S$ can observe or download, and $\Gamma = \bigcup_{i \in S} \Gamma_i$ as the set of all time windows, where $|\Gamma| = n$. In this work, we refer to each time window in $\Gamma$ as a task since they are opportunity time windows to execute an observation or download.

In Section 3.2, we will model the IAEOSSP considering each task in $\Gamma$ as a node in a directed, weighted, and connected graph. These nodes are denoted as task nodes, and we connect two task nodes of this graph by an arc if a satellite can perform one task after the other. Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{A}$ is the set of arcs. $\mathcal{V}$ also contains source and sink nodes, namely 0 and $n+1$, respectively. Therefore, $\mathcal{V} = \{0, p_1, \ldots, p_n, n+1\}$, with $|\mathcal{V}| = |\Gamma| + 2$. Moreover, let $\mathcal{V}' = \mathcal{V} \setminus \{0, n+1\}$ be the subset composed by only task nodes. Each node $p \in V$ is associated with the duration $(\delta_p)$ and the profit $(\rho_p)$ of the corresponding target $\tau \in T$, and with an opportunity time window $(a_{ip}, b_{ip})$ for satellite $i \in S$. Let $\delta_0 = \delta_{n+1} = 0$, $\rho_0 = \rho_{n+1} = 0$, $(a_{i0}, b_{i0}) = (0, 0)$, $(a_{in+1}, b_{in+1}) = (H, H)$, and $\varphi_0^i = \varphi_{n+1}^i = 0, \forall i \in S$.

Let $\mathcal{O}'$ and $\mathcal{D}'$ be the subsets of nodes of $\mathcal{V}$ that are associated with observation and download tasks of $\Gamma$, respectively. Consider $\mathcal{O}_i' \subseteq \mathcal{O}'$ and $\mathcal{D}_i' \subseteq \mathcal{D}'$ as sets of nodes whose associated tasks of $\Gamma$ can be performed by satellite $i \in S$. In addition, let sets $\mathcal{O}, \mathcal{O}_i$, $\mathcal{D}, \mathcal{D}_i$ be defined analogously to sets $\mathcal{O}'$, $\mathcal{O}_i'$, $\mathcal{D}'$, $\mathcal{D}_i'$, respectively, but with additional source and sink nodes $\{0, n+1\}$. We denote $\mathcal{V}_i' = \mathcal{D}_i' \cup \mathcal{O}_i'$ as the set of all tasks that can be performed by satellite $i \in S$, and $\mathcal{V}_i = \mathcal{V}_i' \cup \{0, n+1\} = \mathcal{D}_i \cup \mathcal{O}_i$. We also define $\mathcal{D}_i'^g \subseteq \mathcal{D}_i$ as download task nodes involving satellite $i \in S$ and ground station $g \in G$. $\mathcal{D}_i^g$ is denoted as $\mathcal{D}_i'^g \cup \{0, n+1\}$.

As depicted in Figure 1, a satellite may change its positioning to capture data from an observation target or transmit it to a ground station by adjusting both the roll and the pitch angles. The former, denoted by $\varphi_p^i$, is a parameter and represents the roll angle required by satellite $i \in S$ to execute task $p \in \mathcal{V}_i$. The latter, denoted by $\vartheta_p^i$, is variable since it depends on when task $p$ is scheduled within an available time window. Each of these angles is limited by a maximum value for the observation and the download tasks, defined as $\phi_{obs}$ and $\phi_{dwn}$ for the roll angle, and $\theta_{obs}$ and $\theta_{dwn}$ for the pitch angle.

Let $E_0$ and $L_0$ represent each satellite's initial energy charge and onboard storage, and $E_{max}$ and $L_{max}$ denote the maximum energy and data storage capacity, respectively. Satellites can obtain data from observation spots at a $\bar{L}_{obs}$ rate and transfer data to ground

stations at a $\bar{L}_{dwn}$ rate. Observations tasks consume $\bar{E}_{obs}$, and download ones consume $\bar{E}_{dwn}$ units of energy per unit of time. Moreover, $\epsilon^i_{pq}$ denotes the energy satellite $i \in S$ can obtain through its solar panels between tasks $p, q \in \mathcal{V}_i$. Table A.1 in Appendix A summarizes the mathematical notation adopted.

### *3.2. Mixed-integer nonlinear model*

We model the IAEOSSP by the mixed-integer nonlinear program (1)–(30). In this formulation, we use the following binary decision variables:

- $x^i_{pq}$: binary decision variable indicating whether tasks $p, q \in \mathcal{V}_i$ are performed consecutively by satellite $i \in S$. $x^i_{pq}$ only exists if task $q$ can be performed after $p$;
- $y^i_{pq}$: binary decision variable indicating whether tasks $p$ and $q$ ($p, q \in \mathcal{O}_i$ or $p, q \in \mathcal{D}_i$) are selected for satellite $i \in S$. Likewise, $y^i_{pq}$ is only created if task $q$ can be executed after task $q$. In addition, we only use $y^i_{pq}$ if setup time constraints are required for tasks $p$ and $q$;
- $z^{gii'}_{pq}$: binary decision variable denoting if download tasks $p \in \mathcal{D}^g_i$ and $q \in \mathcal{D}^g_{i'}$ are scheduled by different satellites $i, i' \in S$ for the same ground station $g \in G$. Variable $z^{gii'}_{pq}$ only exists if setup time constraints are required between these tasks.

Decision variables and constraints involving tasks $p, q \in \mathcal{V}$ presume $p \neq q$. Similarly, decision variables and constraints associated with satellites $i, i' \in S$ consider $i \neq i'$. Model (1)–(30) uses the following continuous variables:

- $d^i_p$: non-negative variable for the duration of task $p \in \mathcal{V}_i$ by satellite $i \in S$. If $p \in \mathcal{O}'_i$, $d^i_p = \delta_p$; if $p \in \mathcal{D}'_i$, $\delta_p \leq d^i_p \leq {}^{L_{max}}\!/\bar{L}_{dwn}$; otherwise, if $p \in \{0, n+1\}$, $d^i_p = 0$;
- $e^i_p$: non-negative variable indicating, at the end of task $p \in \mathcal{V}_i$, the energy level of satellite $i \in S$;
- $l^i_p$: non-negative variable indicating, at the end of task $p \in \mathcal{V}_i$, the storage level of satellite $i \in S$;
- $t^i_p$: non-negative variable for the start time of observation or download task $p \in \mathcal{V}'_i$ by satellite $i \in S$ within time window $(a_{ip}, b_{ip})$, i.e., $a_{ip} \leq t^i_p \leq b_{ip} - \delta_p$. If $p \in \{0, n+1\}$, $a_{ip} \leq t^i_p \leq b_{ip}$;
- $\vartheta^i_p$: continuous variable representing the pitch angle of a satellite $i \in S$ to perform task $p \in V_i$;

- $\Delta e_p^i$: non-negative variable representing the amount of energy satellite $i \in S$ has consumed to execute task $p \in V_i$;
- $\Delta l_p^i$: continuous variable which tracks the change in the onboard storage of satellite $i \in S$ for executing task $p \in V_i$;
- $\Delta s_{pq}^i$: non-negative variable for the setup time required by satellite $i \in S$ between two tasks $p, q \in \mathcal{V}_i$.

The IAEOSSP nonlinear formulation is presented next.

$$\max \sum_{i \in S} \sum_{p \in \mathcal{O}_i \setminus \{n+1\}} \sum_{q \in \mathcal{V}_i \setminus \{0\}} (\rho_p + \delta_p) x_{pq}^i - \frac{1}{\bar{L}_{dwn}} \sum_{i \in S} l_{n+1}^i \tag{1}$$

subject to

$$\sum_{p \in \mathcal{V}_i \setminus \{n+1\}} x_{pq}^i - \sum_{r \in \mathcal{V}_i \setminus \{0\}} x_{qr}^i = 0, \qquad i \in S, q \in \mathcal{V}_i' \tag{2}$$

$$\sum_{q \in \mathcal{V}_i \setminus \{0\}} x_{0q}^i = 1, \qquad i \in S \tag{3}$$

$$\sum_{p \in \mathcal{V}_i \setminus \{n+1\}} x_{pn+1}^i = 1, \qquad i \in S \tag{4}$$

$$\sum_{i \in S} \sum_{p \in \mathcal{O}_i^\tau} \sum_{q \in \mathcal{V}_i \setminus \{0\}} x_{pq}^i \leq 1, \qquad \tau \in T \tag{5}$$

$$y_{pq}^i + y_{qp}^i = \min \left\{ \sum_{r \in \mathcal{V}_i \setminus \{0\}} x_{pr}^i, \sum_{u \in \mathcal{V}_i \setminus \{0\}} x_{qu}^i \right\}, \qquad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i' \tag{6}$$

$$\vartheta_p^i = \theta_{obs} \left\{ 2 \frac{t_p^i - a_{ip}}{b_{ip} - a_{ip}} - 1 \right\}, \qquad i \in S, p \in \mathcal{O}_i \tag{7}$$

$$\vartheta_p^i = \theta_{dwn} \left\{ 2 \frac{t_p^i - a_{ip}}{b_{ip} - a_{ip}} - 1 \right\}, \qquad i \in S, p \in \mathcal{D}_i \tag{8}$$

$$\Delta s_{pq}^i = \omega_1 |\varphi_p^i - \varphi_q^i| + \omega_2 |\vartheta_p^i - \vartheta_q^i| + \beta_1, \qquad i \in S, p, q \in \mathcal{O}_i \tag{9}$$

$$\Delta s_{pq}^i = \omega_3 |\varphi_p^i - \varphi_q^i| + \omega_4 |\vartheta_p^i - \vartheta_q^i| + \beta_2, \qquad i \in S, p, q \in \mathcal{D}_i \tag{10}$$

$$y_{pq}^i (t_p^i + d_p^i + \Delta s_{pq}^i - t_q^i) \leq 0, \qquad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i \setminus \{0\} \tag{11}$$

$$z_{pq}^{gii'} + z_{qp}^{gi'i} = \min \left\{ \sum_{r \in \mathcal{V}_i \setminus \{0\}} x_{pr}^i, \sum_{u \in \mathcal{V}_{i'} \setminus \{0\}} x_{qu}^{i'} \right\}, \qquad g \in G, i, i' \in S, p \in \mathcal{D}_i'^g, q \in \mathcal{D}_{i'}'^g \tag{12}$$

$$z_{pq}^{gii'} (t_p^i + d_p^i + \Delta\sigma - t_q^{i'}) \leq 0, \qquad g \in G, i, i' \in S, p \in \mathcal{D}_i'^g, q \in \mathcal{D}_{i'}'^g \tag{13}$$

$$a_{ip} \leq t_p^i \leq b_{ip} - \delta_p, \qquad i \in S, p \in \mathcal{V}_i' \tag{14}$$

$$t_p^i + d_p^i \leq b_{ip}, \qquad i \in S, p \in \mathcal{D}_i' \tag{15}$$

$$\delta_p \leq d_p^i \leq \frac{L_{max}}{\bar{L}_{dwn}}, \qquad i \in S, p \in \mathcal{D}_i' \tag{16}$$

$$l_0^i = L_0, \qquad i \in S \tag{17}$$

$$\delta_p \leq l_p^i \leq L_{max}, \qquad i \in S, p \in \mathcal{O}_i' \tag{18}$$

$$0 \leq l_p^i \leq L_{max}, \qquad i \in S, p \in \mathcal{D}_i \quad (19)$$

$$\Delta l_p^i = \begin{cases} \bar{L}_{obs} d_p^i, & \text{if } p \in \mathcal{O}_i, \\ -\bar{L}_{dwn} d_p^i, & \text{otherwise.} \end{cases} \qquad i \in S, p \in \mathcal{V}_i \setminus \{0\} \quad (20)$$

$$x_{pq}^i (l_p^i + \Delta l_q^i - l_q^i) = 0, \qquad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i \setminus \{0\} \quad (21)$$

$$e_0^i = E_0, \qquad i \in S \quad (22)$$

$$0 \leq e_p^i \leq E_{max}, \qquad i \in S, p \in \mathcal{V}_i \quad (23)$$

$$\Delta e_p^i = \begin{cases} \bar{E}_{obs} d_p^i, & \text{if } p \in \mathcal{O}_i, \\ \bar{E}_{dwn} d_p^i, & \text{otherwise.} \end{cases} \qquad i \in S, p \in \mathcal{V}_i \setminus \{0\} \quad (24)$$

$$x_{pq}^i (e_q^i - \min\{E_{max}^i, e_p^i + \epsilon_{pq}^i - \Delta e_q^i\}) = 0, \qquad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i \setminus \{0\} \quad (25)$$

$$x_{pq}^i (t_p^i + d_p^i - t_q^i) \leq 0, \qquad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i \setminus \{0\} \quad (26)$$

$$\Delta l_p^i, \vartheta_p^i \in \mathbb{R}, d_p^i, e_p^i, l_p^i, t_p^i, \Delta e_p^i \in \mathbb{R}_{\geq 0}, \qquad \forall i \in S, \forall p \in \mathcal{V}_i \quad (27)$$

$$x_{pq}^i \in \{0,1\}, \Delta s_{pq}^i \in \mathbb{R}_{\geq 0}, \qquad \forall i \in S, \forall p, q \in \mathcal{V}_i \quad (28)$$

$$y_{pq}^i \in \{0,1\}, \qquad \forall i \in S, \forall p, q \in \mathcal{V}_i \quad (29)$$

$$z_{pq}^{gii'} \in \{0,1\}, \qquad \forall g \in G, \forall i, i' \in S, \forall p \in \mathcal{D}_i^g, \forall q \in \mathcal{D}_{i'}^g. \quad (30)$$

The objective function (1) maximizes the sum of profits associated with the observed targets and the amount of data downloaded. Note that the second term of the objective function disregards any remaining data storage at the sink node $n+1$. The flow conservation constraints (2)–(4) ensure the amount of flow into a node equals the amount of flow out of it for every node $q \in \mathcal{V}_i'$, except for the source and sink nodes. Constraints (5) set the maximum number of observations per target. In these constraints, $\mathcal{O}_i^\tau$ is the set of observation tasks related to target $\tau \in T$ that can be performed by satellite $i \in S$. The pitch angle of satellite $i \in S$ at the start of task $p \in O_i$ or $p \in D_i$ is defined by constraints (7) or (8), respectively. Setup times $\Delta s_{pq}^i$ of satellite $i \in S$ between observation tasks ($p, q \in O_i$) and download tasks ($p, q \in D_i$) are given by constraints (9) and (10), respectively. $\Delta s_{pq}^i$ are then employed on constraints (11) to ensure satellite $i \in S$ adjusts its pose between observation or download activities.

Constraints (12) model decision variables $z_{pq}^{gii'}$, used in constraints (13) to provide setup time constraints between two download tasks involving the same ground station. Constraints (14) ensure that the start time of task $p \in \mathcal{V}_i'$ is within its time window. As we consider $d_p^i \geq \delta_p$, the upper bound on $t_p^i$ can be narrowed by $\delta_p$ for both observation and download tasks. Constraints (15) and (16) set bounds on download durations. The initial on-board memory storage at the start of the planning horizon is defined by constraints (17), while constraints (18) and (19) enforce minimum and maximum storage limits for a satellite at the end of observation and download tasks, respectively. Similarly, constraints (22)

ensure the energy level at the beginning of the planning horizon, whereas constraints (23) establish its minimum and maximum values for observation and download tasks. Onboard memory storage and energy variations between two consecutive tasks are controlled by constraints (21) and (25), respectively. These constraints use $\Delta l_p^i$ and $\Delta e_p^i$, which are modeled by constraints (20) and (24), respectively. We link $x_{pq}^i$, $t_p^i$, and $t_q^i$ with constraints (26). Finally, constraints (27)–(30) define the domains of the variables.

### *3.3. Linearizations*

Let $\vartheta_{pq}^i = |\vartheta_p^i - \vartheta_q^i| = max\{\vartheta_p^i - \vartheta_q^i, \vartheta_q^i - \vartheta_p^i\}$. Constraints (9) are linearized as follows:

$$\Delta s_{pq}^i = \omega_1 |\varphi_p^i - \varphi_q^i| + \omega_2 \vartheta_{pq}^i + \beta_1, \qquad i \in S, p, q \in \mathcal{O}_i \tag{31}$$
$$\vartheta_{pq}^i \geq \vartheta_p^i - \vartheta_q^i, \qquad i \in S, p, q \in \mathcal{O}_i \tag{32}$$
$$\vartheta_{pq}^i \geq \vartheta_q^i - \vartheta_p^i, \qquad i \in S, p, q \in \mathcal{O}_i \tag{33}$$
$$\vartheta_{pq}^i \leq \vartheta_p^i - \vartheta_q^i + M_{pq}^1 (1 - y_{pq}'^i), \qquad i \in S, p, q \in \mathcal{O}_i \tag{34}$$
$$\vartheta_{pq}^i \leq \vartheta_q^i - \vartheta_p^i + M_{pq}^1 y_{pq}'^i, \qquad i \in S, p, q \in \mathcal{O}_i \tag{35}$$
$$y_{pq}'^i \in \{0, 1\}, \qquad \forall i \in S, \forall p, q \in \mathcal{O}_i \tag{36}$$
$$\vartheta_{pq}^i \in \mathbb{R}_{\geq 0}, \qquad \forall i \in S, \forall p, q \in \mathcal{O}_i. \tag{37}$$

Note that constraints (10) can be linearized similarly. Also, nonlinear constraints (6) and (12) are linearized by constraints (38)–(42), and (43)–(47), respectively, as follows:

$$y_{pq}^i + y_{qp}^i \leq 1 \qquad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i' \tag{38}$$
$$y_{pq}^i + y_{qp}^i \leq \sum_{r \in \mathcal{V}_i \setminus \{0\}} x_{pr}^i + \sum_{u \in \mathcal{V}_i \setminus \{0\}} x_{qu}^i \qquad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i' \tag{39}$$
$$y_{pq}^i + y_{qp}^i \geq \sum_{r \in \mathcal{V}_i \setminus \{0\}} x_{pr}^i + \sum_{u \in \mathcal{V}_i \setminus \{0\}} x_{qu}^i - 1 \qquad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i' \tag{40}$$
$$y_{pq}^i + y_{qp}^i \leq \sum_{r \in \mathcal{V}_i \setminus \{0\}} x_{pr}^i \qquad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i' \tag{41}$$
$$y_{pq}^i + y_{qp}^i \leq \sum_{u \in \mathcal{V}_i \setminus \{0\}} x_{qu}^i \qquad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i' \tag{42}$$
$$z_{pq}^{gii'} + z_{qp}^{gi'i} \leq 1, \qquad g \in G, i, i' \in S, p \in \mathcal{D}_i'^g, q \in \mathcal{D}_{i'}'^g \tag{43}$$
$$z_{pq}^{gii'} + z_{qp}^{gi'i} \leq \sum_{r \in \mathcal{V}_i \setminus \{0\}} x_{pr}^i + \sum_{u \in \mathcal{V}_{i'} \setminus \{0\}} x_{qu}^{i'}, \qquad g \in G, i, i' \in S, p \in \mathcal{D}_i'^g, q \in \mathcal{D}_{i'}'^g \tag{44}$$
$$z_{pq}^{gii'} + z_{qp}^{gi'i} \geq \sum_{r \in \mathcal{V}_i \setminus \{0\}} x_{pr}^i + \sum_{u \in \mathcal{V}_{i'} \setminus \{0\}} x_{qu}^{i'} - 1, \qquad g \in G, i, i' \in S, p \in \mathcal{D}_i'^g, q \in \mathcal{D}_{i'}'^g \tag{45}$$
$$z_{pq}^{gii'} + z_{qp}^{gi'i} \leq \sum_{r \in \mathcal{V}_i \setminus \{0\}} x_{pr}^i, \qquad g \in G, i, i' \in S, p \in \mathcal{D}_i'^g, q \in \mathcal{D}_{i'}'^g \tag{46}$$
$$z_{pq}^{gii'} + z_{qp}^{gi'i} \leq \sum_{u \in \mathcal{V}_{i'} \setminus \{0\}} x_{qu}^{i'}, \qquad g \in G, i, i' \in S, p \in \mathcal{D}_i'^g, q \in \mathcal{D}_{i'}'^g. \tag{47}$$

The remaining constraints can be linearized in the following way by employing Miller–Tucker–Zemlin (MTZ) constraints (Miller et al., 1960).

$$t_q^i \geq t_p^i + d_p^i + \Delta s_{pq}^i - M_{pq}^2(1 - y_{pq}^i), \quad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i \setminus \{0\} \tag{48}$$

$$t_q^{i'} \geq t_p^i + d_p^i + \Delta\sigma - M_{pq}^3(1 - z_{pq}^{gii'}), \quad g \in G, i, i' \in S, p \in \mathcal{D}_i'^g, q \in \mathcal{D}_{i'}'^g \tag{49}$$

$$l_q^i \geq l_p^i + \Delta l_q^i - M_{pq}^4(1 - x_{pq}^i), \quad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i \setminus \{0\} \tag{50}$$

$$l_q^i \leq l_p^i + \Delta l_q^i + M_{pq}^5(1 - x_{pq}^i), \quad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i \setminus \{0\} \tag{51}$$

$$e_q^i \geq min\{E_{max}^i, e_p^i + \epsilon_{pq}^i - \Delta e_q^i\} - M_{pq}^6(1 - x_{pq}^i), \quad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i \setminus \{0\} \tag{52}$$

$$e_q^i \leq min\{E_{max}^i, e_p^i + \epsilon_{pq}^i - \Delta e_q^i\} + M_{pq}^7(1 - x_{pq}^i), \quad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i \setminus \{0\} \tag{53}$$

$$t_q^i \geq t_p^i + d_p^i - M_{pq}^8(1 - x_{pq}^i), \quad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i \setminus \{0\}. \tag{54}$$

Nonlinear constraints (11), (13), (21), (25), and (26) are linearized by constraints (48), (49), (50)–(51), (52)–(53), and (54), respectively, where $M_{pq}^m$, $m \in \{1, ..., 8\}$, $p, q \in \mathcal{V}$, are sufficiently large numbers.

## 4. Model improvements

This section describes two improvements to the model just presented. Section 4.1 introduces the efficient strategy, and Section 4.2 provides an improved formulation to (1)–(30).

### *4.1. The efficient strategy*

This section details the efficient strategy, which is based on dummy tasks. To this end, we assume, without loss of generality, that the instances adopt a single satellite and tasks with non-overlapping time windows. Furthermore, as a single satellite is considered, variables $x_{pq}^i$ can be simply referred to as $x_{pq}$.

#### *4.1.1. Motivation and how the strategy works*

Figure 3a depicts a straightforward implementation using the decision variables $x_{pq}^i$ of the original model. It depicts an instance with four observation tasks, where the arcs (dashed and solid) represent all possible decisions. The solid arcs highlight the feasible solution sequence $0, 1, 2, 4, 5$. We refer to the solution space of this instance as every possible sequence of nodes starting at the source (0) and ending at the sink (5).

Figure 3b shows the same 4-task instance, now with additional dummy nodes. We modified the graph by removing arcs between distant nodes and introducing new arcs that involve these artificial nodes so that the original solution space of the instance is preserved. We use $f_{a:b}$ to indicate that an artificial task is associated with (or covers) all nodes between

$a$ and $b$, including both $a$ and $b$. Moreover, if $a = b$, $f_{a:b}$ becomes $f_a$. If a dummy node and a task are related, only one of them can be scheduled. In the figure, dummy $f_{1:2}$ is associated with tasks 1 and 2, and dummy $f_3$ covers task 3. The solution space remains unchanged because, for every node, one still has the option of either selecting it or not.


(a) Straightforward implementation.

(b) Implementation with dummy nodes.

Figure 3: The same solution space of a 4-task instance represented on different graphs.

In our efficient strategy, a dummy node and its associated tasks have arcs to the subsequent artificial node and the nodes covered by it. For example, in Figure 3b, nodes $f_{1:2}$, 1, and 2 are linked to node $f_3$ and 3. Since there is no dummy task covering node 4, nodes $f_3$ and 3 need to have arcs to tasks 4 and 5 (sink), to maintain the original solution space of the instance. Analogous to a task, the source node has arcs to nodes $f_{1:2}$, 1, and 2. The solution sequence $0, 1, 2, 4, 5$ is represented by the solid arcs.

Associating a dummy node with a single task minimizes the number of variables in the efficient implementation, as demonstrated in Appendix B. Figure 4 illustrates the strategy for a single satellite instance with six observation tasks. The straightforward version is not depicted, but it follows the same pattern as Figure 3a. Table 1 summarizes the number of outgoing arcs of every node for a 6-task instance according to both implementations.

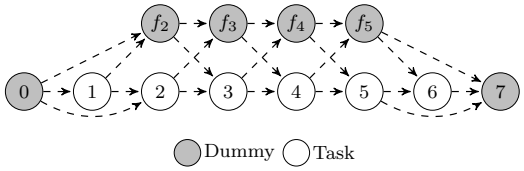


Figure 4: Efficient strategy for an instance with six tasks.

Table 1: Number of outgoing arcs for both implementations for an instance with six tasks.

| **Implementation** | **#Outgoing arcs** | | | | | | | | | | | | **Total** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Straightforward** | 7 | 6 | - | 5 | - | 4 | - | 3 | - | 2 | 1 | 0 | 28 |
| **Efficient** | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 0 | 22 |
| **Node** | 0 | 1 | 2 | 2 | 3 | 3 | 4 | 4 | 5 | 5 | 6 | 7 | |
| **Type** | source | task | dummy | task | dummy | task | dummy | task | dummy | task | task | sink | |

According to Table 1, the number $Nb^x(n)$ of $x_{pq}$ variables associated with the straightforward implementation of an instance with $n = 6$ tasks is: $7+6+5+\cdots+2+1+0 = 28$.

In fact, $Nb^x(n)$ is generally computed as follows:

$$Nb^x(n) = (n+1) + n + (n-1) + ... + 2 + 1 = \sum_{i=1}^{n+1} i = \frac{1}{2}(n^2 + 3n) + 1. \tag{55}$$

In Figure 4, tasks 2, 3, 4, and 5 (4 out of 6) have four outgoing arcs (to dummy nodes or other tasks). In addition, there are six more arcs: 5 at the beginning of the planning horizon (3 and 2 leaving nodes 0 (source) and 1) and one at the end (between nodes 6 and 7 (sink)). Hence, there exists $4 \times (6-2) + 6 = 22$ arcs. Starting with the 4-task instance, such a pattern repeats itself. As a result, the number of arcs $Nb^x_{eff}(n)$ for the efficient strategy can be computed as:

$$Nb^x_{eff}(n) = 4(n-2) + 6 = 4n - 2. \tag{56}$$

The straightforward implementation demands considerable computational effort due to the $\mathcal{O}(n^2)$ number of $x_{pq}$ variables as indicated by equation (55). With the efficient strategy, the number of $x_{pq}$ variables is $\mathcal{O}(n)$. Our efficient strategy yields the minimum number of $x_{pq}$ variables, as stated in Proposition 1.

**Proposition 1.** *(Optimality of the efficient strategy). For any number $n \geq 4$ of tasks, let $P(n)$ state that equation (56) amounts for the minimum number of arcs, e.g., $x_{pq}$ variables, while preserving the solution space associated with the instance.*

*Proof.* See Appendix B. □

A dummy node in model (1)–(30) has $d^i_p = \delta_p = \rho_p = 0$. Moreover, we consider the end of the time window of a dummy node to be equal to the largest time window end time of the tasks covered by it.

#### *4.1.2. Practical issue: setup time constraints*

Setup time constraints are only added for tasks associated with the same dummy or consecutive dummies. These constraints are not included when any of the tasks is dummy. In Figure 5, for example, consecutively selecting tasks $k-1$ and $k+1$ (which require setup time constraints) can yield an infeasible solution. For this reason, tasks that need setup time constraints must be covered by adjacent dummies.

#### *4.1.3. Theoretical issue: overlapping tasks*

Another problem arises when two or more tasks have overlapping time windows. Let that be the case of tasks $k-1$ and $k$ of Figure 5, which are covered by consecutive dummies. This implies that $k-1$ must always be scheduled before $k$. However, the efficient strategy

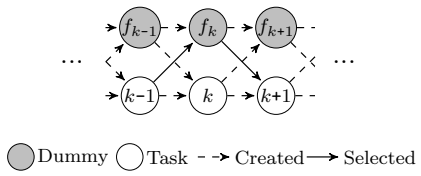


Figure 5: Transition time issues if tasks $k-1$ and $k+1$ are selected consecutively.

would eliminate feasible solutions if the overlap is large enough so that $k$ can be scheduled before $k-1$. To avoid this issue, all overlapping tasks should be covered by the same dummy node. While this is a theoretical concern, in practice, it can be ignored to significantly speed up the solver's execution at the risk of possibly discarding some feasible solutions.

### *4.2. An improved formulation*

Model (1)–(30) handles all relevant aspects of the problem, but demands considerable computational effort. Besides an important reduction in the number of $x_{pq}$ variables, it is also possible to reduce this model by removing the $y^i_{pq}$ variables and the associated constraints (6), (11), and (26). In order to model setup times, a few other constraints must be considered. The details addressed in this section concern the changes in the roll angles and are also valid for the changes in the pitch angles.

Let $\varphi^i_{1g_1} = |\varphi^i_1 - \varphi^i_{g_1}|$, $\varphi^i_{g_12} = |\varphi^i_{g_1} - \varphi^i_2|$, and $\varphi^i_{12} = |\varphi^i_1 - \varphi^i_2|$, depicted in Figure 6, be the changes in roll angles between tasks 1 and $g_1$, $g_1$ and 2, and 1 and 2, respectively, where 1 and 2 are observation tasks and $g_1$ is a download task. The triangle inequality of equation 57 holds and, as a consequence, the inequality of equation 58 also holds:

$$\varphi^i_{1g_1} + \varphi^i_{g_12} \geq \varphi^i_{12}, \tag{57}$$

$$\omega_1\varphi^i_{1g_1} + \omega_1\varphi^i_{g_12} \geq \omega_1\varphi^i_{12}. \tag{58}$$

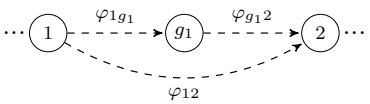


Figure 6: Setup times involving observation and download tasks.

Feasible solutions can still be obtained if one replaces the right-hand side of inequality (58) with the expression on its left-hand side on the corresponding setup time constraints. To ensure this condition holds for subsequences involving both observation and download tasks, the following additional constraints are required when consecutive tasks $p$ and $q$ are of different types:

$$\Delta s^i_{pq} = max\{\omega_1, \omega_3\}|\varphi^i_p - \varphi^i_q| + max\{\omega_2, \omega_4\}|\vartheta^i_p - \vartheta^i_q| + \frac{max\{\beta_1, \beta_2\}}{2}, \ i \in S, \begin{cases} p \in O_i, q \in D_i \text{ or} \\ p \in D_i, q \in O_i. \end{cases} \tag{59}$$

Note that the pitch angle is properly considered in constraints (59). Constraints (9)–(10) still have to be applied when both $p$ and $q$ are either observation or download tasks. Moreover, constraints (60) also have to be included:

$$x^i_{pq}(t^i_p + d^i_p + \Delta s^i_{pq} - t^i_q) \leq 0, \qquad i \in S, p \in \mathcal{V}_i \setminus \{n+1\}, q \in \mathcal{V}_i \setminus \{0\}. \tag{60}$$

We call this new model the *improved formulation*, which consists of model (1)–(5), (7)–(10), (12)–(25), (27)–(28), (30), and new constraints (59) and (60).

## 5. Proposed MIP-based heuristic

The main idea of the MIP-based heuristic (MBH) is to explore a reduced solution space associated with an instance of the problem in a controlled fashion. This control is achieved through neighborhood structures that set the lower and upper bounds of the variables before the MIP solver is invoked. A neighborhood is defined on a given solution by fixing most of the solution while allowing the MIP solver to reoptimize some parts of the solution and possibly obtain a better neighbor solution. For example, a neighborhood could be defined as a swap of a selected task with a non-selected task: the MIP model would then switch the value of one task variable from 1.0 to zero while choosing the best non-selected task from zero to 1.0. Our neighborhoods are more intricate, and we next describe the seven neighborhoods proposed, which define the set $\mathcal{N}$.

Algorithm 1 contains the pseudocode of the MBH procedure, which starts by allocating time for building an initial solution (line 2). Section 6.3.2 discusses the criterion for such division. Next, an initial solution is built (line 3). If the procedure to build an initial solution is not able to find a feasible solution within $time\_limit'^{(0)}$, an empty solution is set (an arc between the source and sink nodes with no intermediate tasks), which is also a feasible solution. The main loop of the algorithm (lines 5–8) runs $|\mathcal{N}|$ times, once for each neighborhood. The time left is proportionally distributed at each iteration among the remaining neighborhoods (line 6). The selected neighborhood $\mathcal{N}^k$ is applied (line 7), and we store both the solution and the runtime of the operator. The order in which we apply the neighborhoods is detailed in Section 6.3.3. We use the stored runtime to adjust the

remaining time for subsequent iterations (line 8). At the end of the MBH procedure, the best solution found is returned (line 9).

**Algorithm 1** MIP-based heuristic.

1: **procedure** MBH($\mathcal{N}, time\_limit$)
2: $time\_limit' \leftarrow$ Divide $time\_limit$ among the neighborhoods
3: $s, runtime \leftarrow$ **build_initial_solution**($time\_limit'^{(0)}$)
4: $rem\_time \leftarrow time\_limit - runtime$
5: **for** $k = 1$ **to** $|\mathcal{N}|$ **do**
6: $time\_limit' \leftarrow$ Divide $rem\_time$ among the remaining neighborhoods
7: $s, runtime \leftarrow \boldsymbol{\mathcal{N}^k}(s, time\_limit'^k)$
8: $rem\_time \leftarrow rem\_time - runtime$
9: **return** s

The neighborhoods control the computational effort of the MIP solver by fixing and releasing the bounds of the decision variables. More specifically, we change the bounds of the $x^i_{pq}$ arc variables by fixing them to their value in the current solution (either to zero by enforcing that the arc is not used or to one to enforce that the two tasks are performed), while leaving some carefully selected $x^i_{pq}$ variables free ($\{0, 1\}$), which can be used to optimize these variables and constraints of the model.

At each iteration of MBH, the bounds of all $x^i_{pq}$ arcs are initially fixed according to the incumbent solution, and then released based on the design of the current neighborhood. The seven neighborhood structures ($\mathcal{N}^1$–$\mathcal{N}^7$) are presented next.

### *5.1. Unscheduled targets neighborhood ($\mathcal{N}^1$)*

Each observation target is associated with a number of tasks (e.g., different observation windows) within the planning horizon. In the *unscheduled targets neighborhood*, we select the $\alpha\%$ of unscheduled targets with the fewest tasks. The idea is that these targets have few observation opportunities and might be more difficult to schedule. The tasks associated with these selected targets compose set $U$. The neighborhood then releases the bounds of arcs used in the incumbent solution and arcs between tasks in the incumbent solution and download tasks, dummy tasks, or tasks in $U$.

We use parameter $\kappa$ to implement multiple iterations of the unscheduled targets neighborhood, described in Algorithm 2. Two situations can happen between consecutive iterations. On the one hand, the solver might not improve the incumbent solution, in which case the bounds remain unchanged at the next iteration. This allows the solver to have extra time on harder cases. On the other hand, the solver might improve the incumbent solution by, for example, collecting some unobserved targets, in which case the set of free arcs change at the next iteration.

**Algorithm 2** Unscheduled targets neighborhood ($\mathcal{N}^1$).

1: **procedure** $\alpha$_$\kappa$_UNSCH_TARGETS($s, tl$)
2: $runtime \leftarrow 0$
3: **for** $k = 1$ **to** $\kappa$ **do**
4: $U \leftarrow$ {Tasks related with the $\alpha$% of unscheduled observation targets with fewest tasks}
5: Release the bounds of every arc used in the solution or between a task in the solution and a download task, a dummy task, or a task in $U$
6: $s, opt\_time \leftarrow$ **optimize**($s, tl$)
7: $runtime \leftarrow runtime + opt\_time$
8: **return** $s, runtime$

Figure 7 depicts an example of the procedure for an instance with a single satellite, $\alpha = 50\%$, and $\kappa = 1$. The solid arrows represent the incumbent solution, while the dashed black arrows connect nodes in the incumbent solution with the observation tasks related with 50% unscheduled targets. Download task $g$ and dummy task $f_4$ are connected to the incumbent solution by the dashed gray and the dotted arrows, respectively. Both nodes 2 and 5 are observation opportunities for the same target with id 4. Since target 4 is excluded from the $\alpha = 50\%$, the bounds of arcs involving tasks 2 and 5 remain fixed. Observation targets 1 and 4 are not considered unscheduled targets because the corresponding tasks 1 and 4, respectively, are part of the incumbent solution.

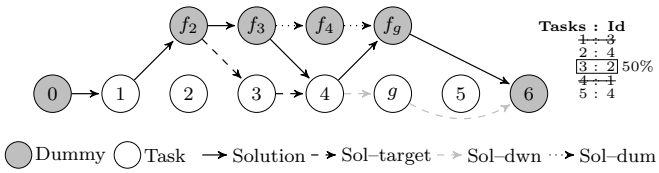


Figure 7: Unscheduled targets neighborhood ($\mathcal{N}^1$) for a single-satellite instance with $\alpha = 50\%$, and $\kappa = 1$.

### *5.2. Intra-satellite neighborhood ($\mathcal{N}^2$)*

The *intra-satellite neighborhood*, described in Algorithm 3, releases the bounds of arcs associated with one satellite per iteration. A full run of this operator takes $m$ iterations, one per satellite. At each iteration (lines 3–6), the bounds of $\frac{1}{m}$ of the total number of arcs are released (line 4), the solver is executed (line 5), and the runtime updated (line 6). The algorithm terminates by returning the incumbent solution and the runtime (line 7).

### *5.3. Multi-satellite neighborhood ($\mathcal{N}^3$)*

The *multi-satellite neighborhood* relies on divisions of the planning horizon into $\lambda$ intervals. At each iteration, the bounds of the arcs with tasks within the current interval are released, and the bounds of those with any task outside it are fixed. While the straightforward idea might be to use equal-length intervals, it can lead to unbalanced solver runs,

**Algorithm 3** Intra-satellite neighborhood ($\mathcal{N}^2$).

1: **procedure** INTRA_SAT($s, tl$)
2: $runtime \leftarrow 0$
3: **for** $i \in S$ **do**
4: Release bounds of every $x^i_{pq}$ such that the target associated with task $p$ is not scheduled to another satellite
5: $s, opt\_time \leftarrow$ **optimize**$(s, tl)$
6: $runtime \leftarrow runtime + opt\_time$
7: **return** $s, runtime$

where some intervals have more tasks and are harder to solve, and others have fewer tasks and are easier. To prevent this, intervals are set to have about the same number of tasks.

The neighborhood, illustrated in Algorithm 4, starts by dividing the tasks available to each satellite into $\lambda$ intervals (line 2). At each iteration of the main loop (lines 4–8), one interval is selected. Next, the bounds of arcs with tasks within the selected interval are released for each satellite (lines 5–6), representing $\frac{1}{\lambda}$ of the arcs. Subsequently, the solver is executed (line 7), and the runtime is updated (line 8). At the end, the solution and the runtime are returned (line 9).

**Algorithm 4** Multi-satellite neighborhood ($\mathcal{N}^3$).

1: **procedure** $\lambda$_MULTI_SATS($s, tl$)
2: $I \leftarrow$ Divide the tasks available to each satellite in $\lambda$ intervals. For a given satellite, all intervals have the same number of tasks
3: $runtime \leftarrow 0$
4: **for** $k = 1$ **to** $\lambda$ **do**
5: **for** $i \in S$ **do**
6: Release bounds of $x^i_{pq}$ whose $p$ and $q$ are within interval $I[i][k]$
7: $s, opt\_time \leftarrow$ **optimize**$(s, tl)$
8: $runtime \leftarrow runtime + opt\_time$
9: **return** $s, runtime$

### *5.4. Double-satellite neighborhood ($\mathcal{N}^4$)*

The *double-satellite neighborhood* considers every pair of satellites, unlike a single satellite as in the intra-satellite neighborhood. The procedure, presented in Algorithm 5, takes $\binom{m}{2} = \frac{m!}{2!(m-2)!}$ iterations (lines 3–7). At each iteration, the bounds of arcs associated with two satellites ($\frac{2}{m}$ of the arcs) are released (line 5). Next, the solver is executed (line 6), and the runtime is updated (line 7). The algorithm returns the incumbent solution and the runtime (line 8).

### *5.5. Variations of the neighborhood structures*

For large instances, even the neighborhoods we defined can still yield challenging MIPs to be optimized. Intending to continuously obtain small improvements in every call to the

**Algorithm 5** Double-satellite neighborhood ($\mathcal{N}^4$).

```
1: procedure DOUB_SATS(s, tl)
2:     runtime ← 0
3:     for i ∈ S do
4:         for i' > i ∈ S do
5:             Release bounds of every x^i_pq and x^i'_pq
6:             s, opt_time ← optimize(s, tl)
7:             runtime ← runtime + opt_time
8:     return s, runtime
```

MIP solver, we have also implemented variations of neighborhoods unscheduled targets, intra-satellite, and double-satellite that release the bounds of fewer variables per iteration.

- *download tasks* ($\mathcal{N}^5$) – a special case of unscheduled targets neighborhood in which $\alpha = 1$, $\kappa = 1$, and only the arcs used in the solution or between tasks in the solution and download tasks are released.

- $\mu$ *intra-satellite* ($\mathcal{N}^6$) – similar to the intra-satellite neighborhood but the tasks available to each satellite are further divided in $\mu$ intervals ($\mu \times m$ iterations). The bounds are released for the arcs in one interval ($\frac{1}{\mu m}$) at a time.

- $\pi$ *double-satellite* ($\mathcal{N}^7$) – differs from the double-satellite neighborhood because the satellites are further divided into $\pi$ intervals ($2\binom{m}{\pi}$ iterations). Bounds are released for arcs of each interval for two satellites, corresponding to $\frac{2\pi}{m}$ of bounds per iteration.

## 6. Computational experiments

This section describes the computational experiments conducted to assess the performance of the proposed methods. Both the model and the algorithm were implemented in C++ and compiled with *g++*, version 12.3.0. We used Gurobi 11.0.1 API for modeling and solving the mixed-integer linear programs. All tests were executed on a computer equipped with an Intel® Core™ i9-13900K processor with 32 threads at 3.0 GHz and 128 GB of RAM.

Cho et al. (2018) solved the *constellation mission scheduling problem* (CMSP) with a so-called *two-step binary linear programming* TSBLP model, whose performance was evaluated against a greedy *first-in first-out* (FIFO) greedy strategy. They implemented both solution approaches in Python and made the code available to us upon request. We evaluate the performance of the MBH procedure on the CMSP against both TSBLP and FIFO on the same machine. We also report the results of the MBH method on the IAEOSSP against the standalone MIP solver.

### *6.1. Benchmark instances*

To create the set of benchmark instances for both CMSP and IAEOSSP, we modified the data used in Cho et al. (2018) for the conventional version of the problem, designed based on real-world satellite missions. This dataset was composed of a single instance file with tasks for 1000 observation targets, 12 satellites, four ground stations, and a planning horizon of 7 days.

We changed the existing dataset by (*i*) rounding, to the nearest integer, the numbers associated with time windows start and end times; (*ii*) removing observation tasks whose time window overlaps with the time window of any download task; and (*iii*) following the authors of the original set, dividing the single large instance file into smaller instances. More specifically, we created 32 instances that differ in the number of observation targets (200, 400, 600, and 800), satellites (1, 3, and 6), ground stations (1, 2, and 4), and planning horizon (1, 2, and 3 days). For more information on the instances, as well as their parameters, we refer to Cho et al. (2018). Our instances and detailed results are available online at `https://github.com/yurerocha/IAEOSSP`.

### *6.2. Solving the CMSP*

The CMSP considers conventional satellites in which the values of the setup times of the satellites disregard the changes in the pitch angle. In fact, this is the main difference between the CMSP and the IAEOSSP. In view of this, one can adapt model (1)–(30) to solve the CMSP by removing constraints (7)–(10), and adding the following constraints:

$$\Delta s^i_{pq} = \omega_1 |\varphi^i_p - \varphi^i_q| + \beta_1, \qquad i \in S, p, q \in \mathcal{O}_i \tag{61}$$

$$\Delta s^i_{pq} = \omega_3 |\varphi^i_p - \varphi^i_q| + \beta_2, \qquad i \in S, p, q \in \mathcal{D}_i. \tag{62}$$

The new formulation introduced in this section is called the CMSP model. Hereafter, we refer to the improved formulation for the IAEOSSP, combined with the efficient strategy, as IAEOSSP model (Section 4.2). In a similar fashion, the improved formulation for the CMSP, combined with the efficient strategy, is referred to as CMSP model. In both the IAEOSSP formulation and the CMSP model, we ignore the issue discussed in Section 4.1.3; the impact of this decision is discussed in Section 6.6.

### *6.3. Parameter values and order of neighborhood exploration*

In this section, we discuss how the parameters were tuned for the MBH procedure. Section 6.3.1 provides some preliminary information regarding the tuning, whereas Section

6.3.2 explains the time limit allocation, and Section 6.3.3 concludes with the order in which neighborhoods are applied.

### *6.3.1. Preliminaries*

The benchmark instances were divided into eight groups according to the number of targets and satellites. To tune the parameters, we selected, from each group, the instance with the largest planning horizon and the smallest number of ground stations. Such instances seem more challenging because there are more targets, although there are fewer opportunities to download data.

We tune the MBH heuristic on the CMSP as this allows us to compare with the literature. The gaps are computed as follows: $gap = 100 \times \{[best(\text{FIFO, TSBLP}) - \text{MBH}]/best(\text{FIFO, TSBLP})\}$, where function $best(\text{FIFO, TSBLP})$ chooses the solution with the highest objective value among FIFO and TSBLP.

We build an initial solution with Gurobi parameters $SolutionLimit = 1$ and $TimeLimit = time\_limit_{init\_sol}$.

### *6.3.2. Time limit*

Except for neighborhoods unscheduled targets and download tasks, whose percentage of bounds released is not straightforward to compute (see Section 5), we divided the time limit proportionally among the neighborhoods according to their computational effort by iteration. Unless otherwise stated, we set a 1-hour time limit for the MBH procedure on the IAEOSSP and the CMSP formulations.

The time limit for an iteration of a move that releases the bounds of arcs associated with a single satellite is denoted as $t_{sat}$. This value is recomputed at every iteration of the MBH. The following time limits $tl$ were adopted for each iteration of the procedures: $tl = t_{sat}$ for the initial solution and neighborhoods unscheduled targets, intra-satellite, and download tasks; $tl = \frac{mt_{sat}}{\lambda}$ for multi-satellite; $tl = 2t_{sat}$ for double-satellite; $tl = \frac{t_{sat}}{\mu}$ for $\mu$ intra-satellite; and $tl = \frac{2t_{sat}}{\pi}$ for $\pi$ double-satellite.

### *6.3.3. Neighborhood structures*

We used an incremental approach, similar to the one described in Kramer and Subramanian (2019), to select the order in which to apply the neighborhood operators. These operators were sorted in non-descending order of the percentage of bounds released per iteration, considering instances with the largest number of satellites. A few exceptions exist to this rule, such as neighborhoods unscheduled targets and download tasks.

We begin the local search phase by applying the unscheduled targets neighborhood ($\mathcal{N}^1$) to the initial solution. This choice is motivated by the computational effort of the neighborhood, which can be controlled by $\alpha$, which limits the percentage of unscheduled targets with few opportunities to be observed. We adopted $\kappa = 2$ to limit the scope of the neighborhood (number of iterations).

Settings 1, 2, and 3 of Table 2 show the results of applying different values of $\alpha$ with a time limit of 600 seconds. The tuning starts with $\alpha = 70$ up to 100, in increments of 10 (setting 1). When applying this setting, we observe a gap to the solution from the literature of 2.43%. Next, we include $\alpha = 60$ before them, i.e., $\alpha = \{60, \ldots, 100\}$ (setting 2). This decreases the gap to 1.99%. Including $\alpha = 50$ before them (setting 3) worsens the average gap, so we select setting 2 and move on to the next neighborhood.

We now set the time limit to 3600 seconds and, at each time, add new operators after those applied in the previous settings. Setting 4 results from setting 2 with a time limit of 3600 seconds, which improves the gap to 1.84%. In setting 5, we include the download tasks neighborhood ($\mathcal{N}^5$), which is included after exploring similar neighborhoods to facilitate downloading the data, which corresponds to half of the objective of the problem. This setting achieves a gap of 0.58%. Setting 6 includes the intra-satellite neighborhood ($\mathcal{N}^6$) with $\mu = 2$, improving the gap to 0.26%. Setting 7 includes the intra-satellite neighborhood ($\mathcal{N}^2$) and improves the gap to 0.25%, while setting 8 adds the download tasks neighborhood again, after two similar neighborhoods were applied. The download tasks neighborhood brings the gap to $-0.09\%$, improving the literature results. Next, setting 9 includes the multi-satellite neighborhood ($\mathcal{N}^3$) with $\lambda = 4$ and improves the gap to $-0.34\%$. In setting 10, which incorporates the download tasks neighborhood again, the gap does not improve.

Setting 11 then includes the double-satellite ($\mathcal{N}^7$) with $\pi = 2$ and double-satellite ($\mathcal{N}^4$) neighborhoods, and improves the gap to $-0.65\%$. The former neighborhood is only applied to instances with fewer than 4 satellites for two reasons: first, applying the neighborhood double-satellite with $\pi = 2$ on instances with a large number of satellites takes a considerable number of iterations; and second, the neighborhood double-satellite takes few iterations on the small sized instances. The combination of the two neighborhoods balances the number of iterations. Setting 12 includes the download tasks neighborhood again, improving the gap to $-0.68\%$. Settings 13 and 14 include the multi-satellite neighborhoods ($\mathcal{N}^3$) with $\lambda = 2$ and $\lambda = 1$, improving the gap to $-0.82\%$ and $-0.89\%$, respectively. Finally, setting 15 tries to include the download tasks neighborhood again, but the gap degrades to $-0.83\%$, which makes setting 14 the best sequence of neighborhood operators.

Table 2: Results obtained for different settings of neighborhood structures.

| Setting | Neighborhoods ($\mathcal{N}$) | Time limit (s) | Gap (%) |
|---|---|---|---|
| 1 | $\mathcal{N}^1$: $\alpha = 70, \ldots, 100$, $\kappa = 2$ | 600 | 2.43 |
| 2* | $\mathcal{N}^1$: $\alpha = 60, \ldots, 100$, $\kappa = 2$ | 600 | 1.99 |
| 3 | $\mathcal{N}^1$: $\alpha = 50, \ldots, 100$, $\kappa = 2$ | 600 | 2.11 |
| 4* | Setting 2 + $tl = 3600$ | 3600 | 1.84 |
| 5* | Setting 4 + $\mathcal{N}^5$ | 3600 | 0.58 |
| 6* | Setting 5 + $\mathcal{N}^6$: $\mu = 2$ | 3600 | 0.26 |
| 7* | Setting 6 + $\mathcal{N}^2$ | 3600 | 0.25 |
| 8* | Setting 7 + $\mathcal{N}^5$ | 3600 | −0.09 |
| 9* | Setting 8 + $\mathcal{N}^3$: $\lambda = 4$ | 3600 | −0.34 |
| 10 | Setting 9 + $\mathcal{N}^5$ | 3600 | −0.34 |
| 11* | Setting 9 + $\mathcal{N}^7$: $\pi = 2 + \mathcal{N}^4$ | 3600 | −0.65 |
| 12* | Setting 11 + $\mathcal{N}^5$ | 3600 | −0.68 |
| 13* | Setting 12 + $\mathcal{N}^3$: $\lambda = 2$ | 3600 | −0.82 |
| 14* | Setting 13 + $\mathcal{N}^3$: $\lambda = 1$ | 3600 | −0.89 |
| 15 | Setting 14 + $\mathcal{N}^5$ | 3600 | −0.83 |

* indicates a selected setting

### *6.4. Comparison with the literature*

Table 3 compares the results of our MBH algorithm on the CMSP model against both the FIFO heuristic and the TSBLP model of Cho et al. (2018). In this table, "Obj" denotes the value of the objective function obtained by our solution approach, "Opt (%)" the optimality gap obtained by the commercial solver, whereas "Gap (%)" indicates the percentage gap (computed as specified in Section 6.3.1) between the best results found by the MBH and those from the two competing algorithms.

Table 3: Results obtained for the CMSP.

| Instance | | | | FIFO | CMSP | | TSBLP | MBH | |
|---|---|---|---|---|---|---|---|---|---|
| $c$ | $m$ | $k$ | $h$ | | Obj | Opt (%) | | Obj | Gap (%) |
| 200 | 1 | 1 | 1 | 1440 | 1640 | 64.38 | 1600 | 1640 | **−2.50** |
| 200 | 1 | 1 | 2 | 3290 | 3840 | 43.95 | 3730 | 3820 | **−2.41** |
| 200 | 1 | 2 | 1 | 1970 | 2190 | 27.54 | 2120 | 2190 | **−3.30** |
| 200 | 1 | 2 | 2 | 3910 | 4290 | 33.40 | 4150 | 4290 | **−3.37** |
| 200 | 3 | 1 | 1 | 4760 | 5360 | 50.37 | 5192 | 5360 | **−3.24** |
| 200 | 3 | 1 | 2 | 9050 | 10050 | 14.47 | 9910 | 10030 | **−1.21** |
| 200 | 3 | 2 | 1 | 6000 | 6540 | 22.37 | 6330 | 6540 | **−3.32** |
| 200 | 3 | 2 | 2 | 9870 | 10620 | 8.00 | 10480 | 10620 | **−1.34** |
| 400 | 1 | 1 | 1 | 1590 | 1850 | 143.24 | 1850 | 1850 | 0.00 |
| 400 | 1 | 1 | 2 | 3650 | 4460 | 104.37 | 4440 | 4460 | **−0.45** |
| 400 | 1 | 2 | 1 | 2370 | 2630 | 71.40 | 2580 | 2630 | **−1.94** |
| 400 | 1 | 2 | 2 | 4820 | 5490 | 67.55 | 5370 | 5490 | **−2.23** |
| 400 | 3 | 1 | 1 | 5340 | 6180 | 119.76 | 6012 | 6180 | **−2.79** |
| 400 | 3 | 1 | 2 | 11140 | 12890 | 68.61 | 12840 | 13010 | **−1.32** |
| 400 | 3 | 2 | 1 | 7140 | 7960 | 70.21 | 7860 | 7960 | **−1.27** |
| 400 | 3 | 2 | 2 | 12790 | 14620 | 47.63 | 14490 | 14610 | **−0.83** |
| **Average** | | | | 5571 | 6288 | 59.83 | 6185 | 6292 | **−1.97** |

| Instance | | | | FIFO | CMSP | | TSBLP | MBH | |
|---|---|---|---|---|---|---|---|---|---|
| $c$ | $m$ | $k$ | $h$ | | Obj | Opt (%) | | Obj | Gap (%) |
| 600 | 3 | 2 | 2 | 14220 | 16290 | 84.02 | 16230 | 16330 | **−0.62** |
| 600 | 3 | 2 | 3 | 18950 | 21040 | 65.34 | 21500 | 21340 | 0.74 |
| 600 | 3 | 4 | 2 | 16670 | 19780 | 50.62 | 19610 | 19630 | **−0.10** |
| 600 | 3 | 4 | 3 | 21850 | 24980 | 40.27 | 25070 | 25160 | **−0.36** |
| 600 | 6 | 2 | 2 | 24700 | 26470 | 34.04 | 29590 | 29610 | **−0.07** |
| 600 | 6 | 2 | 3 | 30950 | 33830 | 4.88 | 35180 | 35380 | **−0.57** |
| 600 | 6 | 4 | 2 | 28910 | 31250 | 13.54 | 32920 | 33180 | **−0.79** |
| 600 | 6 | 4 | 3 | 33670 | 34960 | 1.49 | 35330 | 35380 | **−0.14** |
| 800 | 3 | 2 | 2 | 16030 | 18090 | 109.25 | 18090 | 18370 | **−1.55** |
| 800 | 3 | 2 | 3 | 21550 | 22980 | 106.15 | 24320 | 24360 | **−0.16** |
| 800 | 3 | 4 | 2 | 18660 | 21740 | 76.07 | 21870 | 22190 | **−1.46** |
| 800 | 3 | 4 | 3 | 24760 | 27270 | 73.05 | 28690 | 28930 | **−0.84** |
| 800 | 6 | 2 | 2 | 27720 | 31970 | 48.70 | 34020 | 33870 | 0.44 |
| 800 | 6 | 2 | 3 | 35980 | 35400 | 34.29 | 42290 | 42710 | **−0.99** |
| 800 | 6 | 4 | 2 | 33480 | 35590 | 33.58 | 39460 | 39640 | **−0.46** |
| 800 | 6 | 4 | 3 | 40990 | 39590 | 20.08 | 45210 | 45280 | **−0.15** |
| **Average** | | | | 25568 | 27577 | 49.71 | 29336 | 29460 | **−0.44** |

As Table 3 shows, the MBH algorithm obtained the best results for 29 out of the 32 instances. In one of the three remaining instances, we achieved a solution with the same objective value as the TSBLP formulation. The average gaps for the other two

instances were 0.74% and 0.44%. Furthermore, considering the instances with up to 400 observation spots, our method improved the solutions found by −1.97% on average. For the instances with 600 and 800 observation targets, the MBH procedure also attained competitive solutions, with an average gap of −0.44%.

### *6.5. Results of the model with agile satellites*

Table 4 presents the results of the MBH method against the MIP solver on the IAEOSSP model. Column "Gap (%)" is computed as $gap = 100 \times \{[\text{Obj(model)} - \text{Obj(MBH)}]/\text{Obj(model)}\}$.

Table 4: Results obtained for the IAEOSSP formulation.

| **Instance** | | | | **IAEOSSP** | | **MBH** | |
|---|---|---|---|---|---|---|---|
| $c$ | $m$ | $k$ | $h$ | Obj | Opt (%) | Obj | Gap (%) |
| 200 | 1 | 1 | 1 | 1540 | 41.35 | 1540 | 0.00 |
| 200 | 1 | 1 | 2 | 3420 | 32.33 | 3420 | 0.00 |
| 200 | 1 | 2 | 1 | 1940 | 15.07 | 1940 | 0.00 |
| 200 | 1 | 2 | 2 | 3810 | 18.83 | 3810 | 0.00 |
| 200 | 3 | 1 | 1 | 4880 | 27.10 | 4880 | 0.00 |
| 200 | 3 | 1 | 2 | 9070 | 10.60 | 9070 | 0.00 |
| 200 | 3 | 2 | 1 | 5790 | 12.67 | 5790 | 0.00 |
| 200 | 3 | 2 | 2 | 9660 | 3.80 | 9660 | 0.00 |
| 400 | 1 | 1 | 1 | 1770 | 81.11 | 1770 | 0.00 |
| 400 | 1 | 1 | 2 | 4110 | 63.04 | 4110 | 0.00 |
| 400 | 1 | 2 | 1 | 2260 | 44.14 | 2260 | 0.00 |
| 400 | 1 | 2 | 2 | 4730 | 40.36 | 4730 | 0.00 |
| 400 | 3 | 1 | 1 | 5790 | 69.90 | 5790 | 0.00 |
| 400 | 3 | 1 | 2 | 11900 | 38.30 | 11940 | **−0.34** |
| 400 | 3 | 2 | 1 | 6980 | 40.84 | 6980 | 0.00 |
| 400 | 3 | 2 | 2 | 12930 | 27.84 | 12990 | **−0.46** |
| **Average** | | | | 5661 | 35.46 | 5668 | **−0.05** |

| **Instance** | | | | **IAEOSSP** | | **MBH** | |
|---|---|---|---|---|---|---|---|
| $c$ | $m$ | $k$ | $h$ | Obj | Opt (%) | Obj | Gap (%) |
| 600 | 3 | 2 | 2 | 14130 | 47.05 | 14280 | **−1.06** |
| 600 | 3 | 2 | 3 | 19010 | 37.59 | 19300 | **−1.53** |
| 600 | 3 | 4 | 2 | 16770 | 24.14 | 16910 | **−0.83** |
| 600 | 3 | 4 | 3 | 22110 | 18.30 | 22370 | **−1.18** |
| 600 | 6 | 2 | 2 | 26860 | 22.32 | 27570 | **−2.64** |
| 600 | 6 | 2 | 3 | 26420 | 29.98 | 33270 | **−25.93** |
| 600 | 6 | 4 | 2 | 29460 | 11.18 | 30270 | **−2.75** |
| 600 | 6 | 4 | 3 | 33910 | 3.37 | 34030 | **−0.35** |
| 800 | 3 | 2 | 2 | 15380 | 59.12 | 15790 | **−2.67** |
| 800 | 3 | 2 | 3 | 20850 | 52.10 | 21540 | **−3.31** |
| 800 | 3 | 4 | 2 | 18270 | 34.42 | 18610 | **−1.86** |
| 800 | 3 | 4 | 3 | 24210 | 31.16 | 24970 | **−3.14** |
| 800 | 6 | 2 | 2 | 26680 | 56.25 | 31190 | **−16.90** |
| 800 | 6 | 2 | 3 | 30890 | 49.10 | 36050 | **−16.70** |
| 800 | 6 | 4 | 2 | 32150 | 29.34 | 35910 | **−11.70** |
| 800 | 6 | 4 | 3 | 40580 | 13.39 | 42400 | **−4.48** |
| **Average** | | | | 24855 | 32.43 | 26529 | **−6.06** |

As Table 4 shows, our method clearly outperformed the MIP solver, finding solutions with the same or better objective values. The MBH approach improved the best-known solutions for 18 out of 32 instances. More specifically, it improved the results for two instances involving 200 and 400 observation spots, with an average gap reduction of 0.05%, and for all 16 instances involving 600 and 800 observation points, with an average gap improvement of 6.06%. Some improvements are significant, with a maximum improvement of more than 25% for an instance with 600 observation points. This is achieved by scheduling more observation tasks but also to downloading more data; this improved scheduling makes better use of the energy and storage capacities of the satellites and explores the solution space better.

### *6.6. Impact of formulations, efficient strategy, and overlapping tasks issue*

We conducted experiments using the MIP solver to determine the best formulation and assess the different strategies' impact on the CMSP model. More specifically, we evaluated

the performance of the CMSP with the straightforward formulation (S) of Section 3, the efficient strategy (E) of Section 4.1, and the improved formulation (I) of Section 4.2. When the efficient strategy is adopted, one can also consider the theoretical issue of overlapping tasks (O) of Section 4.1.3. Table 5 summarizes the four different models indicating which strategies are used in each one, whereas Table 6 provides the number of constraints and variables (continuous and binary) for an instance with 600 observation targets under the four models.

Table 5: The formulations/strategies.

| Names | Formulations | | Strategies | |
|---|---|---|---|---|
| | Straightforward | Improved | Efficient | Overlapping issues |
| CMSP-S | ✓ | | | |
| CMSP-I | | ✓ | | |
| CMSP-IEO | | ✓ | ✓ | ✓ |
| CMSP-IE | | ✓ | ✓ | |

Table 6: Number of constraints, continuous variables, and binary variables for an instance with 600 observation targets for different formulations/strategies.

| Components | # per model | | | |
|---|---|---|---|---|
| | CMSP-S | CMSP-I | CMSP-IEO | CMSP-IE |
| Constraints | 1,515,108 | 1,458,354 | 51,428 | 49,829 |
| Continuous variables | 2,933,253 | 2,933,253 | 124,621 | 121,727 |
| Binary variables | 1,660,613 | 1,458,354 | 51,428 | 49,829 |

The results with average gaps grouped by instances according to their corresponding number of observation targets are summarized in Table 7. As the table suggests, the CMSP model with the improved formulation (average gap of 6.68%) significantly improved upon the straightforward formulation (average gap of 41.27%). The remaining strategies also played essential roles; for example, the efficient strategy achieved an average reduction of 5.0%, and disregarding the overlapping issues reduced the average gap to 1.36%.

Table 7: Impact of the formulations/strategies.

| #Observations targets | Average gap (%) | | | |
|---|---|---|---|---|
| | CMSP–S | CMSP–I | CMSP–IEO | CMSP–IE |
| 200 | −2.55 | −2.68 | −2.68 | −2.59 |
| 400 | 0.07 | −1.25 | −1.03 | −1.23 |
| 600 | 71.76 | 4.51 | 3.30 | 3.04 |
| 800 | 95.81 | 26.16 | 7.14 | 6.21 |
| **Average** | 41.27 | 6.68 | 1.68 | 1.36 |

## 7. Concluding remarks and future work

This paper investigated the *integrated agile Earth observation satellite scheduling problem* (IAEOSSP). We have formulated the IAEOSSP as a mixed-integer linear programming (MILP) problem. Moreover, we have devised an improved formulation that reduces the computational effort by removing variables and constraints from the model while possibly losing feasible solutions; an efficient heuristic strategy that decreases the number or arcs from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$, where $n$ is associated with the number of opportunities for observing targets and downloading data; and a MIP-based heuristic (MBH) procedure that combines the MIP solver with a local search heuristic. To compare the MBH with existing algorithms, we solved a special case of the IAEOSSP, that considers conventional satellites, known as the *constellation mission scheduling problem* (CMSP).

The MBH method on the CMSP model outperformed a greedy algorithm and a two-step binary linear programming (TSBLP) formulation from the literature by achieving better solutions on 29 out of 32 instances. We also evaluated our heuristic method against the standalone MIP solver over the IAEOSSP model. In this case, both approaches performed well on the instances with 200 and 400 targets, whereas the MBH procedure obtained an average gap reduction of 6.06% for the remaining instances with up to 800 targets.

Prospective directions for further research include exploring area targets that cannot be acquired by a single observation, incorporating due dates for tasks, and modeling the energy consumption associated with changes in the satellites' pose.

### Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used ChatGPT in order to improve readability. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

### References

L. Barbulescu, J.-P. Watson, L. D. Whitley, and A. E. Howe. Scheduling space–ground communications for the air force satellite control network. *Journal of Scheduling*, 7:7–34, 2004.

E. Bensanna, G. Verfaillie, J. Agnèse, N. Bataille, and D. Blumstein. Exact and approximate methods for the daily management of an Earth observation satellite. In *Proceedings of the 4th International Symposium on Space Mission Operations and Ground Data Systems, Munich, Germany*, 1996.

N. Bianchessi and G. Righini. Planning and scheduling algorithms for the cosmo-skymed constellation. *Aerospace Science and Technology*, 12(7):535–544, 2008.

N. Bianchessi, J.-F. Cordeau, J. Desrosiers, G. Laporte, and V. Raymond. A heuristic for the multi-satellite, multi-orbit and multi-user management of Earth observation satellites. *European Journal of Operational Research*, 177(2):750–762, 2007.

Z. Chang, Z. Zhou, L. Xing, and F. Yao. Integrated scheduling problem for Earth observation satellites based on three modeling frameworks: an adaptive bi-objective memetic algorithm. *Memetic Computing*, 13(2):203–226, 2021.

D.-H. Cho, J.-H. Kim, H.-L. Choi, and J. Ahn. Optimization-based scheduling method for agile Earth-observing satellite constellation. *Journal of Aerospace Information Systems*, 15(11):611–626, 2018.

CSA. RADARSAT+: Over $1 billion for the future of satellite Earth observation, Oct 2023. URL https://www.asc-csa.gc.ca/eng/news/articles/2023/2023-10-18-radarsat-plus-over-1-billion-dollars-for-the-future-satellite-earth-observation.asp. Accessed: 2024-03-21.

EUASP. EUSPA EO and GNSS Market Report, Jan 2024. URL https://www.euspa.europa.eu/newsroom/news/new-euspa-eo-and-gnss-market-report. Accessed: 2024-03-21.

X. Hu, W. Zhu, B. An, P. Jin, and W. Xia. A branch and price algorithm for eos constellation imaging and downloading integrated scheduling problem. *Computers & Operations Research*, 104:74–89, 2019.

A. Kramer and A. Subramanian. A unified heuristic and an annotated bibliography for a large class of earliness–tardiness scheduling problems. *Journal of Scheduling*, 22(1): 21–57, 2019.

M. Lemaître, G. Verfaillie, F. Jouhaud, J.-M. Lachiver, and N. Bataille. Selecting and scheduling observations of agile satellites. *Aerospace Science and Technology*, 6(5):367–381, 2002.

X. Liu, G. Laporte, Y. Chen, and R. He. An adaptive large neighborhood search metaheuristic for agile satellite scheduling with time-dependent transition time. *Computers & Operations Research*, 86:41–53, 2017.

C. E. Miller, A. W. Tucker, and R. A. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7(4):326–329, 1960.

G. Peng, R. Dewil, C. Verbeeck, A. Gunawan, L. Xing, and P. Vansteenwegen. Agile Earth observation satellite scheduling: An orienteering problem with time-dependent profits and travel times. *Computers & Operations Research*, 111:84–98, 2019.

G. Peng, G. Song, L. Xing, A. Gunawan, and P. Vansteenwegen. An exact algorithm for agile Earth observation satellite scheduling with time-dependent profits. *Computers & Operations Research*, 120:104946, 2020.

A. Sarkheyli, A. Bagheri, B. Ghorbani-Vaghei, and R. Askari-Moghadam. Using an effective tabu search in interactive resources scheduling problem for leo satellites missions. *Aerospace Science and Technology*, 29(1):287–295, 2013.

Z. Waiming, H. Xiaoxuan, X. Wei, and J. Peng. A two-phase genetic annealing method for integrated Earth observation satellite scheduling problems. *Soft Computing*, 23:181–196, 2019.

P. Wang, G. Reinelt, P. Gao, and Y. Tan. A model, a heuristic and a decision support system to solve the scheduling problem of an Earth observing satellite constellation. *Computers & Industrial Engineering*, 61(2):322–335, 2011.

X. Wang, G. Wu, L. Xing, and W. Pedrycz. Agile Earth observation satellite scheduling over 20 years: Formulations, methods, and future directions. *IEEE Systems Journal*, 15 (3):3881–3892, 2020.

J. Zhang and L. Xing. An improved genetic algorithm for the integrated satellite imaging and data transmission scheduling problem. *Computers & Operations Research*, 139: 105626, 2022.