Multi-Satellite Observation Tasks Dispatching and Scheduling

Romain Barrault
DTIS, ONERA, Université de Toulouse
Toulouse, France
romain.barrault@onera.fr

Cédric Pralet
DTIS, ONERA, Université de Toulouse
Toulouse, France
cedric.pralet@onera.fr

Gauthier Picard
DTIS, ONERA, Université de Toulouse
Toulouse, France
gauthier.picard@onera.fr

Eric Sawyer
CNES
Toulouse, France
eric.sawyer@cnes.fr

ABSTRACT

A standard problem in the field of Earth observation is the scheduling of the observations of an agile satellite constellation. Given a set of end-user requests over Points Of Interest (POIs), it consists in selecting observations among the candidate ones, attributing each of them to a satellite, and defining the sequence of observations planned for each satellite given operational constraints. The latter are related to the visibility windows available to observe the POIs and the time-dependent maneuvers required to reorient the observation instrument between two POIs. They result in a highly combinatorial problem that must be solved in a restricted amount of time. To solve such a complex problem, we propose an approach that combines matheuristics (mathematical programming-based heuristics) to filter the observation tasks and metaheuristics to schedule them. Firstly, we solve a Sequential Ordering Problem for each satellite to get a giant tour visiting all the visible POIs. From this giant tour, we exploit a Linear Programming Model to compute the best observation set under several tour length constraints. Finally, we schedule the selected observations based on a Large Neighborhood Search. This three-step method substantially improves the solution quality when compared to a baseline scheduling approach.

KEYWORDS

Earth Observation Satellites, Constellation, Planning, Matheuristics, MILP, Large Neighborhood Search

ACM Reference Format:
Romain Barrault, Cédric Pralet, Gauthier Picard, and Eric Sawyer. 2026. Multi-Satellite Observation Tasks Dispatching and Scheduling. In *Appears at the International Workshop on Autonomous Agents and Multi-Agent Systems for Space Applications (MASSpace-26). Held as part of the Workshops at the 25th International Conference on Autonomous Agents and Multiagent Systems., Paphos, Cyprus, May 2026*, IFAAMAS, 9 pages.

1 INTRODUCTION

Earth observation constellations have become essential assets for a wide range of applications, from climate monitoring and disaster management to urban planning and security. The increasing number of satellites and the growing demand for timely, high-resolution data lead to a challenging planning and scheduling problem: each satellite must be assigned a sequence of observation tasks that respects strict temporal windows, resource constraints, and orbital dynamics while maximizing the overall scientific and economic value of the acquired imagery. This Multi-Satellite Earth Observation Scheduling Problem (MSEOSP) is intrinsically hard, as it can be formulated as a time-dependent Team Orienteering Problem with Time Windows (TD-TOP-TW) [10], where both the transition times and the profit of each observation vary with the exact acquisition time. Consequently, the problem is highly combinatorial (NP-hard), and highly sensitive to the interplay between multiple satellites, making exact solution methods impractical for realistic instance sizes and motivating the development of dedicated heuristic and metaheuristic methods.

In this paper, we propose a novel approach combining a matheuristic approach to select a subset of the (large) set of candidate observations for a constellation, and a metaheuristic scheduler, based on Large Neighborhood Search (LNS) [17, 18], to schedule the selected observations. The idea of the selector is to exploit the locations of the POIs and a simplified transition model in a Sequential Ordering Problem (SOP) solver in order to generate a so-called *giant tour* that then constrains a MILP aiming to select a set of observations maximizing the total reward. On this point, we note that giant tours are already used for vehicle routing problems [6], team orienteering problems [5], and even team orienteering problems with time windows [1]. In these related works, a unique giant tour visiting a subset of customers is split to get one tour per vehicle. Given that each satellite can see different POIs, the giant tour approach we propose is different: we build several giant tours (one per vehicle) taking into account some precedence constraints, and then select sub-tours of these giant tours to try and satisfy the temporal constraints. To further restrict which observations are selected based on the SOP ordering, we also consider sub-tours for sub-time frames to generate capacity constraints. Once a selection is provided by the selector, we use a LNS to schedule the observations, depending on their time windows and transition times.

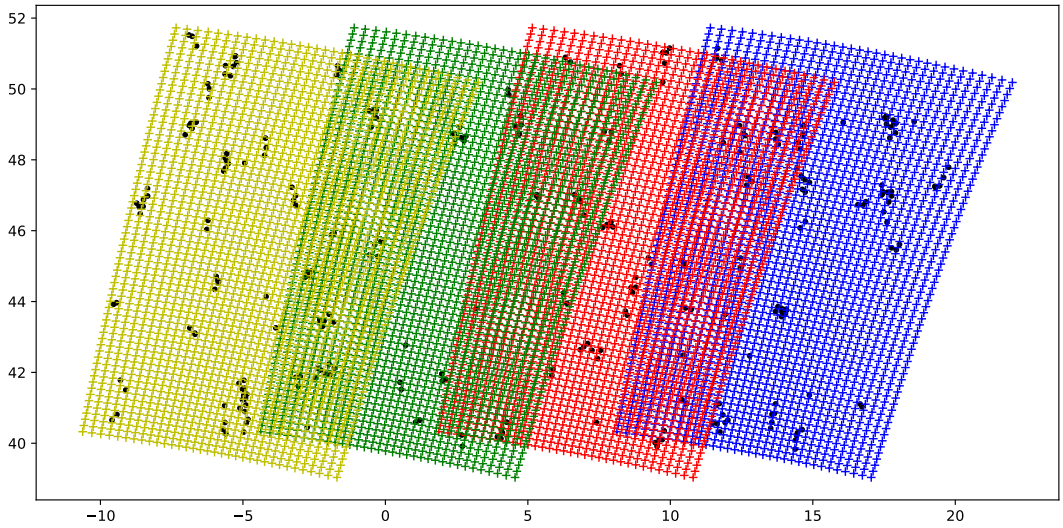

Figure 1: Sample problem instance represented in a Geographic Coordinates System: x-axis is latitude, y-axis is longitude. Black dots correspond to POIs. The colored meshings illustrate the areas successively overflight by four different satellites evenly separated on the same orbit, with a shift of the area overflight between two successive satellites due to the rotation of Earth.

The remainder of the paper is structured as follows. Section 2 retraces previous and current works on similar problems in the literature. Section 3 presents the core concepts and formalizes the problem. Section 4 overviews the global approach we follow. Next sections zoom in the main building blocks of our method: a matheuristic selector combining sequential ordering problem solving and an integer linear program that selects candidate observations to schedule, in Section 5, and a metaheuristics scheduler based on a large neighborhood search aimed to schedule the selected observations, in Section 6. Our approach is experimentally evaluated on a constellation composed of four agile low-orbit satellites, on scenarios where the meteorological conditions (cloud coverage) impact the quality of the observations. We compare our approach to a baseline LNS running for 20 minutes in Section 7. Finally, Section 8 concludes the paper with perspectives.

2 BACKGROUND

The Multi-Satellite Earth Observation Scheduling Problem is a Time-Dependent Team Orienteering Problem with Time Windows and Time-Dependent Profit (TD-TOP-TW-TDP). The latter is an extension of TD-TOP-TW, where the goal is to design a set of routes for a fleet of vehicles to maximize the total collected profit while respecting a total time budget, time windows at visited locations, and time-dependent travel times, while maximizing the sum of time-dependent profits (*e.g.*, profit increasing with service time) [14]. Due to its computational complexity, exact methods, such as Mixed Integer Linear Programming (MILP) formulations, are typically only viable for small-scale instances [14]. For larger, more realistic scenarios, metaheuristics are the dominant solution approach. Effective metaheuristic techniques include Variable Neighborhood Search (VNS) [14], Iterated Local Search (ILS) [11], and nature-inspired algorithms like the Hybrid Artificial Bee Colony (HABC) algorithm, which are designed to balance exploration and exploitation of the search space [20]. These methods often employ tailored local search operations and insertion heuristics to efficiently handle the dynamic nature of travel times and the profit gained, which can depend on the vehicle's arrival and service time at a location [11]. Exact approaches based on Dynamic Programming (DP) and extensions of algorithms like A^* search have also been adapted for related time-dependent shortest path and TSP problems with time windows, often using concepts like partially time-expanded networks for improved performance [9].

In the space domain, Earth Observation Satellite Scheduling Problems require the maximization of total priority/profit from observation requests while respecting time windows (target visibility periods) and intricate operational constraints (e.g., memory capacity, power availability, slew angle/maneuver time, and non-overlap of tasks) [7]. The core techniques mirror those for the TD-TOP-TW, centered around Mixed-Integer Linear Programming (MILP) for optimal solutions on small instances, and metaheuristics for large-scale real-world problems. Reinforcement learning techniques have also recently proven to be competitive in the context of multi-satellite scheduling [16, 19]. For large constellations or long planning horizons, decomposition methods and hybrid metaheuristics are crucial, often combining Local Search (like Iterated Local Search or Simulated Annealing) with greedy construction heuristics and calls to Constraint Programming (CP) solvers for solving sub-problems[2, 12]. MILP formulations often represent the problem using a time-space network or as a variation of the Traveling Salesman Problem with Time Windows (TSPTW) or Orienteering Problem [15]. The next section details the problem's features.

3 PROBLEM DEFINITION

We consider the problem of scheduling tasks for a low orbit agile EOS constellation. In this problem, observation requests over

specific POIs are submitted to the system. Each satellite of the constellation overflies different candidate POIs along its orbit, but corresponding areas may overlap, implying a need to assign each observation to only one satellite (see Fig. 1). The observations also come up with an individual reward which depends on a cloud cover forecast and the time at which each satellite can observe the corresponding POI (time-dependent profit). The purpose is to compute the best constellation schedule, that means the set of individual schedules for each satellite which maximizes the total sum of rewards collected by the constellation.

3.1 Input data: satellite observation areas

We define our case study in order to measure the efficiency of the allocation part of our method without drastically increasing the problem complexity. Firstly, we define input data related to a single agile satellite that can point left, right, forward, or backward thanks to gyroscopic actuators. For this satellite, we consider a time horizon $[0, H_0]$ where $H_0 = N \cdot \delta_T$ is defined from a timestep δ_T and a number steps N. Then, we determine the Earth portion visible by the satellite during time frame $[0, H_0]$. This portion is centered on the satellite ground track and bounded in two dimensions: along track because of the restricted time horizon, and across track because of a maximum angle α_{max} beyond which observation quality is considered as too low. Any point within this area may be a candidate POI for the satellite (also called a target). For each target, we can also compute a Time Window (TW) during which the satellite can point to the target while having a visibility angle less than α_{max} (the farther the POI from the satellite ground track, the smaller the time window). Finally, each observation is associated with an individual reward in interval $[0.2, 1]$. This reward is obtained from a decreasing function of the local cloud cover forecast provided by real weather data.

Once the data for a single satellite is computed, we derive data for an entire EOS constellation with the following process. We define a set of satellites $\mathcal{S}$ along the same orbit, separated by a duration Δ_T. Basically, Δ_T is the time required for a satellite to reach the same orbital position as its predecessor. In our case, we consider a unique orbital plane and four satellites evenly spaced on this plane ($\Delta_T = 25$ minutes in the experiments). Satellite number s is considered over time frame $[s \times \Delta_T, s \times \Delta_T + H_0]$. From the set of Earth portions visible for the satellites, we get a set of potential candidate POIs.

3.2 Input data: candidate POIs within the observation areas

For each satellite s, we generate a set of N_s candidate POIs within its area, using some meshing to avoid having very close candidate POIs. The ith candidate POI for satellite s is referred to as $P_{s,i}$, and the visibility window of this POI by satellite s is denoted by $W_{s,i} = [Ostart_{s,i}, Oend_{s,i}]$. For this POI, we also compute an individual reward $R_{s,i}$. Note that in our settings, each candidate POI p may be viewed by several satellites, that is we may have $p = P_{s,i}$ for one satellite s and $p = P_{s',i'}$ for another satellite s'. However, due to the time-dependent cloud cover, the observation reward for p may depend on the satellite (*i.e.*, $R_{s,i}$ not necessarily equal to $R_{s',i'}$). In the following, we denote Π the set of positions for the POI visible by at least one satellite ($\Pi = \cup_{s \in \mathcal{S}, i \in [1..N_s]} P_{s,i}$). At this point, the set of candidate POIs makes up for an instance of our problem, as represented in Figure 1.

3.3 Optimization problem

As an input, we consider a set of POIs Π defining the so-called *Order Book*, and for each satellite s the set of candidate POIs ($\{P_{s,i} \mid i \in [1..N_s]\}$), their rewards ($\{R_{s,i} \mid i \in [1..N_s]\}$), and their time windows ($\{W_{s,i} \mid i \in [1..N_s]\}$). An additional input data comes from the time-dependent maneuvers between successive observations, in order for each satellite to point its observation instrument towards the right directions. We denote by $tt_{s,i,j,t}$ the duration of the maneuver by satellite $s \in \mathcal{S}$ when maneuvering from observation number $i \in [1..N_s]$ to observation number j, for a maneuver starting at time t. This transition function is time-dependent since it depends on the current position of the considered satellite on its orbit.

Definition 1. The *Multi-Satellite Earth Observation Scheduling Problem* (MSEOSP), for $\zeta = |\mathcal{S}|$ satellites, consists in finding ζ sequences of observations $\sigma_s = [\sigma_{s,1}, \ldots, \sigma_{s,K_s}]$ such that

- each candidate observation in Π appears at most once over all sequences in σ_s;
- for each satellite $s \in S$, all observations are performed during the available time window ($W_{s,i}$), while respecting the required transition times; formally, if number i is the first element of σ_s, the earliest start time for the corresponding observation is $t_{s,i} = Ostart_{s,i}$; if i and j are two consecutive elements in σ_s, then we have $t_{s,j} = \max(Ostart_{s,j}, t_{s,i} + tt(s, i, j, t_{s,i})$; finally, the transition constraints are satisfied if and only if condition $t_{s,i} \leq Oend_{s,i}$ is satisfied for every observation number i belonging to σ_s;
- the total collected reward $\sum_{s \in S, i \in \sigma_s} R_{s,i}$ is maximized.

The problem is usually over-constrained, that is it is usually not possible to observe all the candidate POIs.

4 GLOBAL SEARCH APPROACH

To solve the optimization problem presented before, we exploit a two-step approach involving a matheuristic that computes an estimation of the optimal solution of our problem given coarse-grain constraints, and a metaheuristic that adapts this solution given the detailed constraints. The global search architecture is illustrated in Figure 2.

4.1 Step 1: resolution of a Sequential Ordering Problem

First, for each satellite s, we compute a relevant observation order containing all the POIs it is able to see, in order to build a giant tour for s. Several methods can be used to get such a giant tour. A baseline method is the Earliest Start First (ESF, also referred to as *Earliest Due Date* [8]) heuristic that returns observations sorted by increasing window start times ($Ostart_{s,i}$), so that the satellite tries to observe any POI as soon as it is visible. In practice, this may lead to very bad solutions where along its orbit, the satellite is moving very often between the left and right parts of its ground track. This is why we exploit another method based on the resolution of a problem known as the *Sequential Ordering Problem* (SOP).

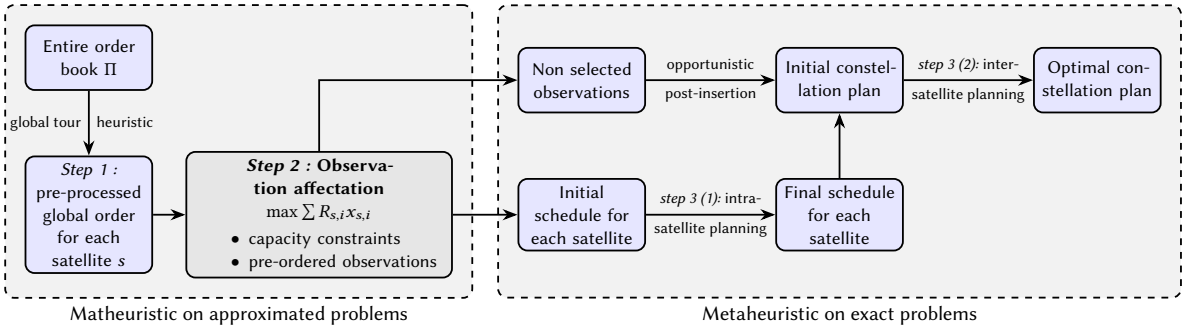

Figure 2: Constellation scheduling algorithm workflow

Definition 2. Given a weighted directed graph $G = (V, E)$ where we denote by $(c_e)_{e \in E}$ the set of costs over the edges of the graph, and a set of *precedence constraints* $\mathcal{P} \subset E$, the *Sequential Ordering Problem (SOP)* consists in the following optimization problem :

$$\min_{\sigma \subset E} \sum_{e \in \sigma} c_e \tag{1}$$

$$\text{s.t.} \quad \sigma = [(\sigma_1, \sigma_2), (\sigma_2, \sigma_3) ... (\sigma_{|V|-1}, \sigma_{|V|})] \tag{2}$$

$$\forall i \in [1, ..|V|], \sigma_i \in V \tag{3}$$

$$\forall v \in V, \exists! \, i \in [1, ..|V|] \; s.t. \; \sigma_i = v \tag{4}$$

$$\forall (v, v') \in \mathcal{P}, indexOf(v, \sigma) < indexOf(v', \sigma) \tag{5}$$

This can be viewed as an *Asymmetric Traveling Salesman Problem* with added precedence constraints implementing the impossibility for certain nodes to follow other ones in the resulting sequence (constraint (5)). There is however no time window for visiting the nodes, nor time-dependency on the transition costs, hence this is an approximated version of our problem.

4.2 Step 2: matheuristic to dispatch POIs to satellites

The SOPs solved provide us with a giant tour for each satellite. We then exploit these giant tours in a MILP model that assigns observations to satellites while taking into account the following constraints.

- Each POI may be assigned to at most one satellite.
- Given giant tour G_s for satellite s, the total cost associated with a selection of observations O_s for s corresponds to the total transition time (or cost) of the sub-tour of G_s induced by O_s. At this search step, this constraint allows us to get free of the decisions concerning POI observation ordering.
- The total time (or total cost) consumed by an observation sequence must be lower than a pre-determined value which is the duration between the earliest start time and the latest end time of candidate POI observations (overload checking, as in constraint programming methods). We also add local constraints on the observation sequence cost (more details later on this point).

To compute the total cost of a sequence while using only linear constraints in a MILP, we approximate the time-dependent transition times as time-independent transition times based on a specific policy, such as an *optimistic policy* considering the minimum transition duration among a set of possible discrete start times for the corresponding maneuver, or a *pessimistic policy* considering the maximum transition duration.

4.3 Step 3: metaheuristic part

At the end of the previous step, we obtain a good subset of candidate POIs assigned to each satellite. At that time, we resort to existing scheduling algorithms described in [3] to (1) plan observations without changing the assigned satellite first, and (2) then consider exchanging acquisitions between the satellites. The purpose is to first exploit the MILP result as a warm start for observation assignment, and then improve for the best the total plan quality. Basically, the existing scheduler introduced by Barrault et al. generates an initial plan thanks to a greedy method called `GreedySch` that iteratively inserts observations in the current plan. In this scheduler, the next observation to be inserted is selected based on heuristic values that depend on both the individual observation rewards and the additional maneuver durations required to insert observations. To improve the resulting plan, we resort to a *Large Neighborhood Search* algorithm, namely `LNS to schedule`, which iteratively destroys and repairs the current plan based on these same heuristic values. In the first part of the process, this algorithm is computed at the same time locally on all satellites, and in the second part a single `LNS to schedule` is performed to destroy and repair the plans in a multi-satellite context. This is done during a pre-determined amount of time, and in the end, the plan that got the best reward sum is returned.

5 MATHEURISTICS FOR SOLVING A RELAXED ROUTING PROBLEM WITH CAPACITIES

To quickly produce a first good quality solution, we resort to a *matheuristic* [4], that is to a technique that exploits mathematical programming techniques (MILP techniques in our case) to get heuristic solutions. The MILP model proposed in this context attempts to quickly produce good quality solutions by going beyond greedy heuristics that fail having a global view of the problem. Before detailing this MILP model, we present how we compute a global tour over all visible POIs from each satellite's perspective, based on the resolution of sequential ordering problems. The global tours

obtained are then exploited in a MILP model that avoids considering scheduling decisions and focuses on pure observation selection and assignment and then we refer to the obtained sequences to select and attribute observations from the order book with respect to satellite capacity constraints.

5.1 Sequential Ordering Problem (SOP)

In our process, the SOP solved can be seen as a problem relaxation where the decisions are focused on the routing part, that is on the transition times between observations, while the time windows available to perform the observations are (almost) completely discarded. This SOP is solved thanks to the LKH solver presented in [13]. For a satellite s that sees POIs numbered from 1 to N_s, as we are only looking for a sequence of observations and not a tour over them, we add a fictitious node numbered 0 which makes up for both the departure and the ending of our sequence. This gives us a weighted directed graph $G = ([0..N_s], E)$ where the edges are weighted by an approximation of the transition times between the observations. A key part in the problem definition is the establishment of a set of mandatory precedence constraints $\mathcal{P}_s$ between the observations of satellite s. More precisely, for two distinct observation numbers $i, j \in [1..N_s]$, if the maneuver towards j starting from i at time $Ostart_{s,i}$ arrives after the end of the window of j (after $Oend_{s,j}$), then j must precede i in the resulting sequence, that is

$$(Ostart_{s,i} + tt(s, i, j, Ostart_{s,i}) > Oend_{s,j}) \rightarrow ((j, i) \in \mathcal{P}_s) \quad (6)$$

The previous rule may however lead to both $(j, i) \in \mathcal{P}_s$ and $(i, j) \in \mathcal{P}_s$ for certain observation couples with very narrow and overlapping time windows. In this case, to avoid precedence cycles, only one of the two precedence constraints is kept in $\mathcal{P}_s$. We then compute all edge weights with the optimistic transition times policy.

Executing LKH over the SOP built provides us with an observation sequence that contains all the observations, satisfies the precedence constraints, and minimizes the total tour cost. Figure **??** displays the kind of sequence returned, for a satellite whose ground track traverses the center of the area from the top to the bottom. We can see on the figure that in the solution found, the satellite limits large maneuvers going backward with regards to its ground track. This is quite natural since backward maneuvers, requiring the satellite to move counter to its orbital motion, are usually longer for such a system.

5.2 MILP Model

From now on, for satellite s, the index $i \in [1..N_s]$ of a candidate observation corresponds to position of this observation in the order returned by the SOP solver. Moreover, $P_{s,i}$ still denotes the i-th candidate observation point of satellite s and Π denotes the set of candidate POIs over all the satellites. To select a subset of the candidate observations and assign each of them to a satellite, we implement the following MILP that takes into account some global capacity constraints related to the time windows. This model is detailed thereafter.

$$\text{maximize} \sum_{s \in \mathcal{S}} \sum_{i \in [1..N_s]} R_{s,i} \cdot x_{s,i} \quad (7)$$

s.t.

$$\forall s \in \mathcal{S}, \forall i \in [1..N_s],\ x_{s,i} = \sum_{j>i} next_{s,i,j} \quad (8)$$

$$\forall s \in \mathcal{S}, \forall i \in [1..N_s],\ x_{s,i} = \sum_{j<i} next_{s,j,i} \quad (9)$$

$$\forall s \in \mathcal{S},\ \sum_{j>0} next_{s,0,j} = 1 \quad (10)$$

$$\forall s \in \mathcal{S},\ \sum_{j \leq N_s} next_{s,j,N_s+1} = 1 \quad (11)$$

$$\forall p \in \Pi,\ \sum_{s \in \mathcal{S},\, i \in [1..N_s] \text{ s.t. } P_{s,i}=p} x_{s,i} \leq 1 \quad (12)$$

$$\forall s \in \mathcal{S}, \forall w \in \mathcal{W}_s,\ \sum_{(i,j) \in U} next_{s,i,j} \cdot T_{s,i,j} \leq Dur_w \quad (13)$$

where $U = \{(i, j) \in [1..N_s]^2 \mid i < j,\ W_{s,i} \subseteq w \vee W_{s,j} \subseteq w\}$

$$\forall s \in \mathcal{S}, \forall i \in [1..N_s], x_{s,i} \in \{0, 1\} \quad (14)$$

$$\forall s \in \mathcal{S}, \forall i, j \in [0..N_s + 1] \text{ s.t. } i < j,\ next_{s,i,j} \in \{0, 1\} \quad (15)$$

For each satellite $s \in \mathcal{S}$ and observation point $p = P_{s,i} \in \Pi$, $x_{s,i}$ is a Boolean variable that takes value 1 if and only if p is selected in the observation sequence of satellite s. Variables $next_{s,i,j}$ are Boolean variables equal to 1 if and only if:

- points $P_{s,i}$ and $P_{s,j}$ are both selected for satellite s,
- for each $k \in]i; j[$, $P_{s,k}$ is not selected for satellite s, that is $P_{s,j}$ is the next point selected after $P_{s,i}$ in the observation sequence returned by the SOP solver.

The objective function defined in Equation (7) corresponds to the total reward collected. Constraints (8) and (9) are standard flow constraints ensuring that each selected observation has a unique predecessor and successor in the sequence of selected observations. The constraints use two additional fictitious nodes: one node numbered 0 for the start of the observation plan, and on node numbered $N_s + 1$ for its end. Constraints (10) and (11) are particular cases for the start and end of the sequence. Constraint (12) models that each POI cannot be observed by more than one satellite.

Constraints (13) are capacity constraints over a set of specific time windows. More precisely, in these constraints, $T_{s,i,j}$ is the same approximated transition time between $P_{s,i}$ and $P_{s,j}$ as explained in Section 4. We also denote by $\mathcal{W}_s$ a set of time windows $w = [Wstart_w, Wend_w]$, of duration Dur_w, over which we attempt to prevent overloads in terms of transition times for satellite s. Set $\mathcal{W}_s$ is built as follows.

- The first added window is the larger one $\tilde{w}$:
  $\tilde{w} = [Wstart_{\tilde{w}}, Wend_{\tilde{w}}] = [\min_i Ostart_{s,i}, \max_i Oend_{s,i}]$ so that the resulting constraint concerns all observations for satellite s.
- We generate other windows iteratively. For this, we initiate an integer $Div = 2$. We create $nCand$ candidate windows of length $l = \frac{Wend_{\tilde{w}} - Wend_{\tilde{w}}}{Div}$. We set $nCand = 5$ after some experiments. The windows generated are spread in $\tilde{w}$, which means that they start at one of the timestamps $Wstart_{\tilde{w}} + (k - 1) \cdot step$ for $k \in [1..nCand]$, where $step = \frac{Wend_{\tilde{w}} - l - Wstart_{\tilde{w}}}{nCand}$. Among all these windows w, we keep only those verifying $\exists i \in [1..N_s]$, $W_{s,i} \subseteq w$. If there are still windows satisfying

this property, we increment Div and compute this step again, otherwise we stop the window generation process.

The set of capacity constraints given in Equation (13) limits the sum of transition times performed on numerous time windows.

In the end, the MILP obtained solves a time-independent version of the MSEOSP introduced in Section 1, using a pre-determined order and without taking into account detailed time windows. Solving this MILP gives us both an assignment of observations to satellites (based on the $x_{s,i}$ variables) and for each satellite, a pre-determined order on the selected set of observations (based on the $next_{s,i,j}$ variables) by following the path from node 0 to node $N_s + 1$. Of course, following the order provided by the global tour can be sub-optimal, but the goal is to define a MILP that can be quickly solved to get a first heuristic solution.

6 METAHEURISTIC FOR THE DETAILED SCHEDULING

In this section, we describe the greedy search and the Large Neighborhood Search (LNS) metaheuristic used to compute "actual" MSEOSP schedules over the satellites, where "actual" means that no input data of the MSEOSP is approximated from this point. More specifically, the time windows and the exact time-dependent transition times are both taken into account to get actually feasible solutions. Both the greedy search and LNS exploit operators that update the satellite plans at the level of a single satellite (Section 6.1), or at the level of the entire constellation (Section 6.2).

6.1 Intra-satellite Moves

When the previously described MILP model returns all observations selected for a given satellite, there is a need to obtain an actual schedule satisfying all the MSEOSP constraints. This part describes two different ways to do so.

6.1.1 Selection-based Greedy Repair Heuristic. This heuristic is described by algorithm 1. It starts either from an empy satellite schedule or one that is simply not full of observations, and fills this schedule according to a greedy policy that considers only the observations selected for the satellite in the MILP solution. The best observation to insert at any time is the one with the minimum insertion penalty. For this, the algorithm tests the addition of each unplanned observation at each possible position in the current schedule and computes an associated insertion penalty. The penalty value obtained for the insertion of observation $i \in [1..N_s]$ between two observations j and k in schedule σ_s is computed as:

$$\frac{tt(s,j,i,t_{s,j}) + tt(s,i,k,t_{s,i}) - tt(s,j,k,t_{s,j})}{R_{s,i}}$$

where $t_{s,j}$ describes the current observation time for j in σ_s and $t_{s,i} = t_{s,j} + tt(s,j,i,t_{s,j})$ gives the observation time of i after insertion (without considering the time window of i). The penalty value therefore takes into account both the temporal cost for inserting each acquisition at any position and the reward provided by the insertion. The downside of this method is that it ignores the formerly computed global order on $\sigma_s \cup \overline{\sigma_s}$. At any time during the process, the feasibility of an observation plan is verified using a `Earliest Start heuristic`.

Algorithm 1 Greedy Repair Heuristic

Require: `current schedule` σ_s
`unplanned observations` $\overline{\sigma_s}$

```
full ← false
while !full do
    p̃ ← ∞
    full ← true
    for acq ∈ σ̄_s do
        for position ≤ len(σ_s) do
            if insertable(i, position, σ_s) then
                full ← false
                p ← penalty(acq, position, σ_s)
                if p ≤ p̃ then
                    p̃ ← p
                    updateBestAcq(acq, position)
    if !full then
        insertBestAcq(σ, bestAcq, bestPosition)
        removeAcq(bestAcq, σ̄)
```

6.1.2 Tour-based Greedy Repair Heuristic. The second greedy heuristic is very similar to the former one except that it only considers schedules respecting the global order induced by the values obtained for the $next_{s,i,j}$ variables at the level of the matheuristic (cf. Section 5). In this so-called *Tour-based Greedy Repair Heuristic*, a single insertion position is tested at each step for each unplanned observation. This approach considers a restricted set of schedules but is must faster.

Randomized Greedy Destroy Heuristic. Once the greedy search process produced a full solution σ_s for each satellite s, the LNS part of the algorithm applies *destroy* and *repair* operations to iteratively enhance the schedule quality for s. The destroy operator used in this purpose removes K observations from the current schedule, with K randomly chosen between 1 and $|\sigma_s|/3$. At each destroy step, one observation is randomly removed from the schedule, with a higher removal probability for observations having a higher presence penalty. Here, we use the same penalty criterion as before: each observation in the current schedule has a presence penalty based on its reward and the time gained in case of removal.

6.2 Inter-satellites Moves

When all satellite schedules are optimized, the last part of the algorithm destroys and repairs the current solution at the constellation level. Also, any observation selected by the MILP solver that did not make it in the best schedule for the corresponding satellite after intra-sallite moves is considered as unworthy and will remain unused for the rest of the algorithm. Two ideas are exploited to try and get better rewards or smaller transitions: first, it can be relevant to reassign an observation to another satellite able to perform it, and second there may be some place left to insert observations not selected initially in the solution derived from the MILP. To tackle these two points, we define an Inter-Satellite Greedy Repair Heuristic (ISGRH) and a Randomized Greedy Destroy Heuristic (RGDH). These two heuristics are very similar to their intra-satellite counterpart. The only difference is that they browse all satellite current schedules before inserting or removing any observation, which

Table 1: Algorithm versions results on 72 instances: optimality gaps (%) at the end of any step. The best value per column is highlighted in gray. If bv is the best known reward on instance I with all algorithms, and algorithm A finds reward v on instance I at the end of step s, then the optimality gap (%) of A on I at the end of s is $100 \cdot (bv - v)/bv$.

Algorithm	initial solution				post intra-moves				post inter-moves			
	max	median	mean	min	max	median	mean	min	max	median	mean	min
Baseline (20min)	16.4	5.29	6.11	1.41	n/a	n/a	n/a	n/a	10.5	0.792	1.57	0
Baseline (1min)	16.4	5.29	6.11	1.41	n/a	n/a	n/a	n/a	10.5	1.35	1.85	0
ESF-SelGH	15.6	9.38	9.81	4.14	15.0	6.71	7.12	3.25	4.37	1.44	1.52	0
ESF-TourGH	24.0	14.8	14.8	8.66	15.5	6.81	7.13	3.25	7.91	1.60	1.87	0
LKH-SelGH	15.1	8.16	8.52	2.78	11.0	5.12	5.17	0.831	5.46	0.699	1.11	0
LKH-TourGH	20.2	10.2	10.7	2.35	10.3	4.79	5.07	0.892	6.80	0.754	1.17	0

implies that the lowest or highest penalty is computed over a wider set of observations and a wider set of insertion positions. Empirically, the inter-satellite moves and the insertion of observations not selected by the MILP allow us to enhance the final schedule quality, especially when problem approximations lead to some erroneous decisions at the MILP level. The counterpart of this scheduling process is that unlike intra-satellite moves, these computations cannot be performed in parallel. ISGRH and RGDH are also used to define our baseline LNS method. In this context, they start computations from an empty constellation schedule, and all observations can be inserted or removed since no MILP computation is involved. We have therefore presented all components of our algorithm shown in Figure 2. The next part describes its performances.

7 EXPERIMENTS

7.1 Instances

The instance set considers 36 different POI distributions. For each of them, 300 observation points are chosen in the area covered by the four satellites, as in Figure 1. In addition, in order to test our algorithms on various realistic study cases, we designed three instance archetypes in terms of POI local density :

(1) *dense* ones, where there are a handful of POI hubs with a high local density,
(2) *sparse* ones, where most POIs are isolated from one another,
(3) *mixed* ones, as a mix of both of these types, where POIs are either isolated or part of a very narrow hub.

To compute observation rewards, 24 real cloud cover scenarios from year 2024 are retrieved [1]. For a cloud cover scenario, the reward $R_{s,i}$ associated with an observation is equal to one in case of no cloud cover over point $P_{s,i}$, 0.2 for a full cloud cover, and quadratically decreasing in between. More precisely, the resulting reward function is a piecewise constant version of the latter one with 5 possible values. In the end, we obtain 36 different POI distributions and 24 cloud cover scenarios (corresponding to 1^{st} and 15^{th} of each month of 2024). To limit the size of the instance set, we consider only 2 cloud cover scenarios per POI distribution (and each cloud cover scenario is applied to exactly 3 POI distribution scenarios). This results in a total of 72 problem instances. Finally, visibility time windows are defined according to a maximum observation angle α_{max} set to 30° in accordance with operational needs.

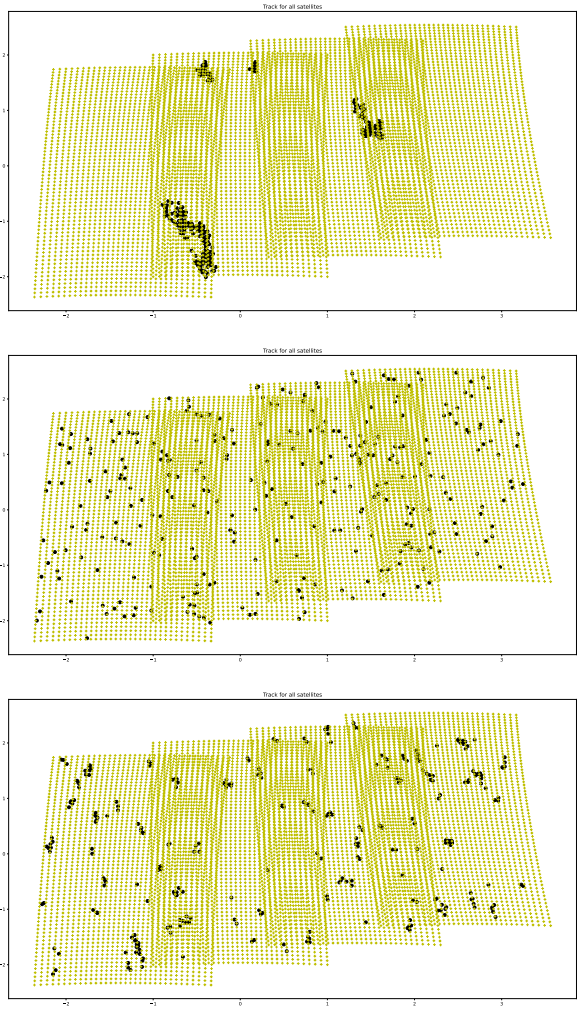

Figure 3: 2D representations of instance examples for each archetype described in 7.1. From top to bottom : *dense*, *sparse* and *mixed*. Black dots correspond to POIs and each yellow grid displays the Earth area covered by a single satellite.

[1] https://cds.climate.copernicus.eu/datasets/reanalysis-era5-single-levels

7.2 Experimental Setup

Firstly, two policies were implemented to return a global route over all satellite visible observations: the basic ESF heuristic that visits observation following their window start times, and the LKH heuristic that solves an SOP. Secondly, once the MILP solver returns a satellite-observation assignment, each detailed satellite schedule can be built based either on the Selection-based Greedy Repair Heuristic (denoted SelGH) or on the Tour-based Greedy Repair Heuristic (denoted TourGH). This leads to four different algorithm variants : ESF-SelGH, ESF-TourGH, LKH-SelGH, and LKH-TourGH.

The maximum CPU time for each algorithmic variant is set to 1 minute. A maximum CPU time of 40 seconds is dedicated to the matheuristic (including ESF or LKH, plus the resolution of the MILP model that is stopped when the optimality gap is less than 1%). The observation assignment problem is solved thanks to CPLEX solver, and the matheuristic usually converges much faster, typically in 5 seconds. Once the matheuristic produces a solution, at time t, the remaining time (1min minus t) is equally split between the intra-satellite LNS and the inter-satellite LNS. In the latter part, inter-satellite LNS operates with the possibility to insert observations that were not selected in the observation affectation process. Empirically, it allows us to enhance the final schedule's quality in case of unprecise choices in the latter part because of problem approximations. Thanks to problem decomposition, intra-satellite LNS is performed simultaneously over each satellite by multi-threading. All schedulers are implemented in Java and executed on 20-core Intel(R) Xeon(R) CPU ES-2660 v3 @ 2.60GHz, 62GB RAM, Ubuntu 18.04.5 LTS, with an OpenJDK 11.0.9 JVM.

The four algorithmic variants are compared to a baseline *LNS* algorithm (denoted Baseline) whose operators are ISGRH and RGDH.

7.3 Result Analysis

Performance metrics are shown in Table 1, where we display the rewards' optimality gap : (i) after initial greedy schedule generation for each satellite, that means with inter-satellite SelGH for Baseline, and with the corresponding heuristic over each satellite's observation affectation after MILP solving for the other algorithms, (ii) after intra-satellite LNS, (iii) after inter-satellite LNS. This gap is computed with reference to the best known reward for each instance. There are several results to point out. First, LKH-based algorithms perform better than ESF-based ones. Unfortunately, computing an initial schedule that respects the order obtained in the MILP solution gives worse results in most cases, since the initial solution reward computed with TourGH is lower than with SelGH, though it is less marked with LKH. A weakness in the algorithm architecture is that it requires intra-satellite scheduling to be competitive with a greedy heuristic that provides a solution very quickly, as shown in Figure 4. However, the overall performance of our LKH-based algorithms is very satisfying: after opportunistic post-insertion of non selected observations and inter-satellite scheduling, LKH-SelGH and LKH-TourGH beat the baseline algorithm when using the same amount of computation time. Our results also show that the instance archetype only slightly matters, as shown in Table 2.

Moveover, in terms of solution quality, LKH-SelGH and LKH-TourGH are competitive with the baseline algorithm running during 20 minutes. In our algorithms, the SOP solving takes several seconds per satellite. However, both LKH-SelGH and LKH-TourGH are able to outperform the baseline once in the inter-satellite phase.

Instance type	dense	sparse	mixed
Baseline (20min)	0.870	0.784	0.792
Baseline (1min)	1.05	1.53	1.15
LKH-SelGH	0.654	1.16	0.620
LKH-TourGH	0.685	1.15	0.485

Table 2: Median value of algorithms' final result depending on the instance type. The best value per instance type is highlighted in gray.

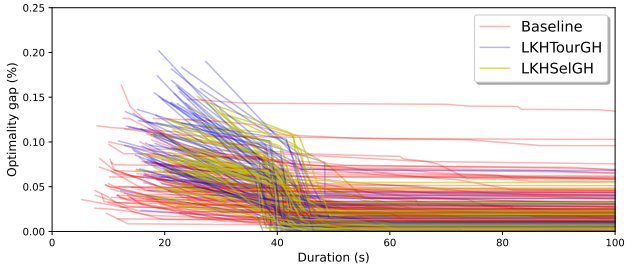

Figure 4: Optimality gaps on each instance depending on CPU time for Baseline and LKH-based algorithms. One line corresponds to a single instance.

8 CONCLUSION

This paper proposes a novel approach combining matheuristics and metaheuristics for solving EOS constellation scheduling problems. The main idea is to heuristically compute, for each satellite, a route visiting visible POIs, based on the resolution of SOPs and a multi-satellite MILP that makes some problem approximations. After that, we reconstruct actually feasible schedules for the satellites, based on some called intra-satellite and inter-satellite moves. Through schedule destroy and repair operations, the algorithm obtained is able to outperform an efficient LNS algorithm in the same amount of time.

One of the bottlenecks remains the quality of the solutions produced by the MILP and the impact of some problem approximations. Indeed, our results show that the method should not be too committed to follow the pre-determined satellite-observation assignments and observation orders. One of the next steps is to build a version of our current MILP where transition costs would be artificially increased, and then to solve the initial problem while keeping the previous satellite-observation assignment decisions. The aim would be to build a more robust assignment algorithm. In this direction, we could try and learn the parameters of such an over-constrained model to get the best results. We also believe that a Constraint Programming approach would be relevant in replacement of the matheuristic part, and we would consider comparing these different approaches.

REFERENCES

[1] Youcef Amarouche, Rym Nesrine Guibadj, Elhadja Chaalal, and Aziz Moukrim. 2020. Effective neighborhood search with optimal splitting and adaptive memory for the team orienteering problem with time windows. *Computers & Operations Research* 123 (2020), 105039. https://doi.org/10.1016/j.cor.2020.105039

[2] Valentin Antuori, Damien T. Wojtowicz, and Emmanuel Hebrard. 2025. Solving the Agile Earth Observation Satellite Scheduling Problem with CP and Local Search. In *31st International Conference on Principles and Practice of Constraint Programming (CP 2025) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 340)*, Maria Garcia de la Banda (Ed.). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 3:1–3:22. https://doi.org/10.4230/LIPIcs.CP.2025.3

[3] Romain Barrault, Cédric Pralet, Gauthier Picard, and Eric Sawyer. 2025. Hybridizing Machine Learning and Optimization for Planning Satellite Observations. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research: 22nd International Conference, CPAIOR 2025, Melbourne, VIC, Australia, November 10–13, 2025, Proceedings, Part I* (Melbourne, VIC, Australia). Springer-Verlag, Berlin, Heidelberg, 68–85. https://doi.org/10.1007/978-3-031-95973-8_5

[4] Marco Antonio Boschetti and Vittorio Maniezzo. 2022. Matheuristics: using mathematics for heuristic design. *4OR* 20, 2 (2022), 173–208.

[5] Hermann Bouly, Duc-Cuong Dang, and Aziz Moukrim. 2008. A Memetic Algorithm for the Team Orienteering Problem. In *Applications of Evolutionary Computing*, Mario Giacobini, Anthony Brabazon, Stefano Cagnoni, Gianni A. Di Caro, Rolf Drechsler, Anikó Ekárt, Anna Isabel Esparcia-Alcázar, Muddassar Farooq, Andreas Fink, Jon McCormack, Michael O'Neill, Juan Romero, Franz Rothlauf, Giovanni Squillero, A. Şima Uyar, and Shengxiang Yang (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 649–658.

[6] Diego Cattaruzza, Nabil Absi, and Dominique Feillet. 2016. The Multi-Trip Vehicle Routing Problem with Time Windows and Release Dates. *Transportation Science* 50, 2 (2016), 676–693.

[7] Xiaoyu Chen, Gerhard Reinelt, Guangming Dai, and Andreas Spitz. 2019. A mixed integer linear programming model for multi-satellite scheduling. *European Journal of Operational Research* 275, 2 (2019), 694–707. https://doi.org/10.1016/j.ejor.2018.11.058

[8] W. Dullaert, G.K. Janssens, K. Sirensen, and B. Vernimmen. 2002. New heuristics for the fleet size and mix vehicle routing problem with time windows. *Journal of the Operational Research Society* 53, 11 (2002), 1232 – 1238. https://doi.org/10.1057/palgrave.jors.2601422 Cited by: 49.

[9] Romain Fontaine, Jilles Dibangoye, and Christine Solnon. 2023. Exact and anytime approach for solving the time dependent traveling salesman problem with time windows. *European Journal of Operational Research* 311, 3 (2023), 833–844. https://doi.org/10.1016/j.ejor.2023.06.001

[10] Ander Garcia, Pieter Vansteenwegen, Olatz Arbelaitz, Wouter Souffriau, and Maria Teresa Linaza. 2013. Integrating public transportation in personalised electronic tourist guides. *Computers & Operations Research* 40, 3 (2013), 758–774. https://doi.org/10.1016/j.cor.2011.03.020 Transport Scheduling.

[11] Damianos Gavalas, Charalampos Konstantopoulos, Konstantinos Mastakas, Grammati Pantziou, and Nikolaos Vathis. 2014. Efficient Heuristics for the Time Dependent Team Orienteering Problem with Time Windows. In *Applied Algorithms*, Prosenjit Gupta and Christos Zaroliagis (Eds.). Springer International Publishing, Cham, 152–163.

[12] Al Globus, James Crawford, Jason Lohn, and Anna Pryor. 2004. A comparison of techniques for scheduling earth observing satellites. In *Proceedings of the 16th Conference on Innovative Applications of Artifical Intelligence* (San Jose, California) *(IAAI'04)*. AAAI Press, 836–843.

[13] Keld Helsgaun. 2017. An Extension of the Lin-Kernighan-Helsgaun TSP Solver for Constrained Traveling Salesman and Vehicle Routing Problems. https://doi.org/10.13140/RG.2.2.25569.40807

[14] M. Khodadadian, A. Divsalar, C. Verbeeck, A. Gunawan, and P. Vansteenwegen. 2022. Time dependent orienteering problem with time windows and service time dependent profits. *Computers & Operations Research* 143 (2022), 105794. https://doi.org/10.1016/j.cor.2022.105794

[15] Minkeon Lee, Seunghyeon Yu, Kybeom Kwon, Myungshin Lee, Junghyun Lee, and Heungseob Kim. 2024. Mixed-Integer Linear Programming Model for Scheduling Missions and Communications of Multiple Satellites. *Aerospace* 11, 1 (2024). https://doi.org/10.3390/aerospace11010083

[16] Bohua Li, Ming Chen, Lining Xing, Yingguo Chen, and Yingwu Chen. 2025. Optimizing time-dependent multi-agile Earth observation satellite scheduling problem using deep Q-learning and ensemble heuristics. *Information Sciences* 712 (2025), 122140. https://doi.org/10.1016/j.ins.2025.122140

[17] David Pisinger and Stefan Ropke. 2019. *Large Neighborhood Search*. Springer International Publishing, Cham, 99–127. https://doi.org/10.1007/978-3-319-91086-4_4

[18] Cédric Pralet. 2023. Iterated Maximum Large Neighborhood Search for the Traveling Salesman Problem with Time Windows and its Time-dependent Version. *Computers & Operations Research* 150 (2023), 106078. https://doi.org/10.1016/j.cor.2022.106078

[19] Lili Ren, Xin Ning, and Zheng Wang. 2022. A competitive Markov decision process model and a recursive reinforcement-learning algorithm for fairness scheduling of agile satellites. *Computers and Industrial Engineering* 169 (2022). https://doi.org/10.1016/j.cie.2022.108242 Cited by: 28.

[20] Vincent F. Yu, Parida Jewpanya, Shih-Wei Lin, and A.A.N. Perwira Redi. 2019. Team orienteering problem with time windows and time-dependent scores. *Computers & Industrial Engineering* 127 (2019), 213–224. https://doi.org/10.1016/j.cie.2018.11.044