END-OF-STUDY INTERNSHIP INTERNSHIP REPORT

Integration of satellite constellation planning and allocation algorithms in a simulation platform

made by Jihanne EL HAOUARI

Internship from 28/02/2022 to 12/08/2022

Internship supervisors: Gauthier PICARD, Stéphanie ROUSSEL, Cédric PRALET
Academic supervisor: Daniel DELAHAYE

ONERA, Toulouse Research Center
2 Av. Edouard Belin, 31000 Toulouse

Master 2 RO
ENAC
Université Paul Sabatier

Acknowledgement

Foremost, I would like to express my sincere gratitude to my internship supervisors, Gauthier Picard, Stéphanie Roussel and Cédric Pralet, researchers at the Information Processing and Systems Department (DTIS) in ONERA for their help, their relevant ideas, their guidance, and their continuous support during the internship. The weekly exchanges as well as the debates and long discussions on the problems encountered have largely contributed to the progress of this project, and its successful completion.

My thanks also go to Jean-Loup Farges, head of the SYD team in the DTIS department, for allowing me to realize this internship at ONERA and to Claire Dabin, assistant in the DTIS department, for her availability and her great help to complete all the formalities, sometimes complicated, related to the internship.

I would also like to thank the other interns and Ph.D. students that I had the chance to meet during these 6 months for their welcome and also for the tarot breaks, at lunchtime. Sharing our subjects and problems was a source of constant enrichment. I am particularly grateful to Anouck Chan for her help and her pertinent advice throughout this internship.

Finally, I would like to express my gratitude to my pedagogical supervisor Daniel Delahaye and the whole pedagogical team of the SAT course at ENAC and of the RO master for their availability all year long and the adaptation to my internship.

Résumé

Le problème d'ordonnancement des satellites d'observation de la Terre est un problème d'optimisation complexe et difficile à résoudre. Il consiste à trouver le meilleur plan à envoyer aux satellites, afin d'optimiser le nombre d'images de la Terre prises en fonction des demandes des utilisateurs. Dans ce type de problème, les incertitudes météorologiques sont souvent très approximées ou parfois non considérées. Cependant, l'impact de la météo et surtout de la couverture nuageuse doit être pris en compte car il peut invalider les images réalisées. Il est donc nécessaire de résoudre le problème de planification plusieurs fois par jour afin de pouvoir prendre en compte les mises à jour régulières des prévisions météorologiques et d'envoyer régulièrement les plans aux satellites lorsqu'ils passent au-dessus d'une station sol.

Ce rapport étudie plusieurs approches pour trouver la planification optimale en utilisant des données et des prévisions météorologiques réelles. Deux modèles météorologiques différents, donnant des prévisions météorologiques à court terme ou à plus long terme, sont définis et utilisés dans le processus d'optimisation. Une première étude est faite sur la formulation d'un modèle MILP qui pourrait être utile pour trouver un algorithme d'optimisation adapté pour résoudre le problème. Cependant, l'approche principale développée consiste à utiliser un algorithme glouton pour effectuer la planification et un simulateur est développé pour évaluer les différentes méthodes proposées. Le travail se concentre ensuite sur la manière de classer les différentes opportunités d'observation disponibles pour prendre des images des points demandés par les utilisateurs. Ce classement est essentiel afin de savoir ce qui est interessant de mettre en entrée de l'algorithme glouton. Dans ce but, un algorithme de génération d'arbres de décision est implémenté et une métaheuristique de recuit simulé est utilisée pour optimiser l'arbre de décision.

Une comparaison est faite entre les méthodes développées et les solutions optimales obtenues lorsque la météo réelle est supposée être préalablement connue, en utilisant un outil de planification basé sur la Programmation par contraintes.

Keywords— Satellites d'observation de la Terre, Problème de planification, Incertitudes liées à la météo, Arbres de décision, Recuit simulé

Abstract

The Earth Observation Satellite (EOS) Scheduling problem is a complex and challenging optimization problem consisting in finding the best plan to send to the satellites, in order to optimize the number of images of the Earth taken according to the requests of ground users. In this kind of problem, weather uncertainties are often highly approximated or not considered. However, the impact of the weather and especially of the cloud cover needs to be taken into account as it may invalidate the images performed. There is, therefore, a need to solve the decision problem several times a day in order to be able to take into account the regular updates of the weather forecasts and to regularly send the plans to the satellites when they pass over a ground station.

This report investigates several approaches for finding the optimal schedule using real weather data and forecasts. Two different weather models, giving meteorological forecasts in the short term or the longer term, are defined and used in the optimization process A first study is made on the formulation of a MILP model that could be useful to find an adapted optimization algorithm to solve the problem. However, the main approach developed consists in using a greedy algorithm to make the planning and a simulator is developed to evaluate the different methods provided. The work then focuses on the way to classify the different observation opportunities available to take pictures of the requested points, in order to have an interesting input for the greedy algorithm. For this purpose, a decision tree algorithm is implemented and a Simulated Annealing (SA) metaheuristic is used to optimize the decision tree.

A comparison is made between the developed methods and optimal solutions obtained when the real weather is assumed to be known, using a constraint-based scheduling tool.

Keywords— Earth Observation Satellite (EOS), Scheduling problem, weather uncertainties, decision tree algorithm, Simulated Annealing (SA)

List of Figures

List of acronyms

AEOS Agile Earth Observation Satellite

AROME Application of Research to Operations at MEsoscale

ARPEGE Action de Recherche Petite Echelle Grande Echelle

BRKGA Biased Random Key Genetic Algorithm

CARPER Continuous Actively Scheduling Planning Execution and Replanning

CCP Chance Constraint Programming

CEOS Conventional Earth Observation Satellite

CEPMMT Centre Européen pour les Prévisions Météorologiques à Moyen Terme

DAG Directed Acyclic Graph

DCOP Distributed Constraint Optimization

DTIS Information Processing and Systems Department

Dyn DCOP Dynamic Distributed Constraint Optimization

EA Evolutionary Algorithms

ELECTRE Elimination and Choice Expressing Reality

EOS Earth Observation Satellite

GA Genetic Algorithms

GEFS Global Ensemble Forecast System

GFS Global Forecast System

GPHH Genetic Programming based Hyper Heuristic

IBMOLS Indicator-Based Multi-Objective Local Search

IG Information Gain

ILP Integer Linear Programming

ILS Iterative Local Search

LNS Large Neighborhood Search

MDP Markov Decision Process

MILP Mixed-Integer Linear Programming

ONERA Office National d'Études et de Recherches Aérospatiales

PBIL Population-Based Incremental Learning

POI Point Of Interest

R Reprogramming : number of observation opportunities left for one given request

SA Simulated Annealing

SACS Simulated Annealing Classifier System

SMIP Stochastic Mixed Integer Program

SYD Smart and decision-making systems

TCC Total Cloud Cover

TCC$_{asked}$ Total Cloud Cover threshold defined by the ground user for his request

TCC$_{forecast}$ Total Cloud Cover forecast using the M_0 or the M_1 model

TOPSIS Technique for Order Preference by Similarity to Ideal Solution

TS Time to Station : time between the start of the observation window and the previous ground station

TT Time Taken : time when the image is taken by a certain satellite

UARP Uncertain Capacity Arc Routing Problem

VRP Vehicle Routing Problem

TABLE OF CONTENTS

Introduction

This report was written at the end of an internship at the Office National d'Etudes et de Recherches Aérospatiales (ONERA), the French aerospace research center. This internship was carried out between February 28 and August 12, 2022, under the supervision of Gauthier Picard, Stéphanie Roussel and Cédric Pralet, researchers at the Information Processing and Systems Department (DTIS) in ONERA.

The exact topic of the end-of-study internship is: Integration of satellite constellation planning and allocation algorithms in a simulation platform. We refocused the subject at the beginning of the internship and the objective is then to solve a scheduling problem of Earth observation satellites taking into account meteorological uncertainties and to implement the associated optimization algorithms. Indeed, within the framework of a research project between ONERA and Airbus Defence and Space called LiChIE, ONERA is developing several techniques for managing satellite constellations, such as algorithms for scheduling observations in satellite constellations with exclusive orbit portions [1], algorithms for scheduling observations for several users and for planning downloads to ground stations, or even techniques for managing complex requests such as stereoscopic requests, periodic imaging requests, or one-shot simple observation requests [2].

In these kinds of problems, the uncertainties concerning the weather are sometimes not considered or rather approximated. However, these uncertainties, especially due to the presence of clouds, are very important during the scheduling since they cause the invalidity of many images taken by the satellites. It is therefore essential to take into account these weather uncertainties and this is the subject of my internship. The aim is to realize a new optimization model that will integrate existing satellite constellation planning and allocation models while taking into account the meteorological uncertainties. Then, solving methods will be presented for this problem and the associated algorithms will be implemented.

In what follows, the first and second part introduce the internship environment and the different detailed objectives of the end of the study project with the context of the internship and all the different problems raised. Then, the third section focuses on the state of art regarding Earth Observation Satellite Scheduling, and an overview of all the different techniques that can be used to solve such problems is made. Afterwards, section 4, 5 and 6 present the scheduling problem and the different approaches that have been investigated and then implemented to solve the defined optimization problem. Section 7 presents the simulator used to evaluate the different methods implemented. Finally, the last part of this report provides computational results obtained for each method.

1 Internship environment

ONERA (Office National d'Études et de Recherches Aérospatiales) is the French national aerospace research center. It is a public research organization that was founded in 1946. Placed under the supervision of the French Ministry of the Armed Forces, it has a budget of 234 million euros including contract business, grants for public service, and payroll.

ONERA has about 2,031 employees including 1,568 engineers and executives, 315 doctoral students and 27 qualified postgraduates, grouped in its eight major facilities in France (Figure 1), such as in Palaiseau, Châtillon, Meudon, Lille, Modane, Salon-de-Provence, Le Fauga-Mauzac and Toulouse (Figure 2).

This internship is done in the center of Toulouse at the address 2 avenue Edouard Belin.

Figure 1: Map of ONERA centers in France

Figure 2: Center of Toulouse [3]

1.1 The research center

The French aeronautics, space and defense research lab is a multi-disciplinary organization that brings expertise to government programs, both institutional and industrial. It is a world-ranking scientific center that aims to meet the aerospace challenges of the future and to contribute to the competitiveness of the aerospace industry.

Indeed, there are many fields of study in which ONERA is interested and multiple associated challenges (Figure 3).

The 12 ONERA challenges serving Defence, Aeronautics and Space

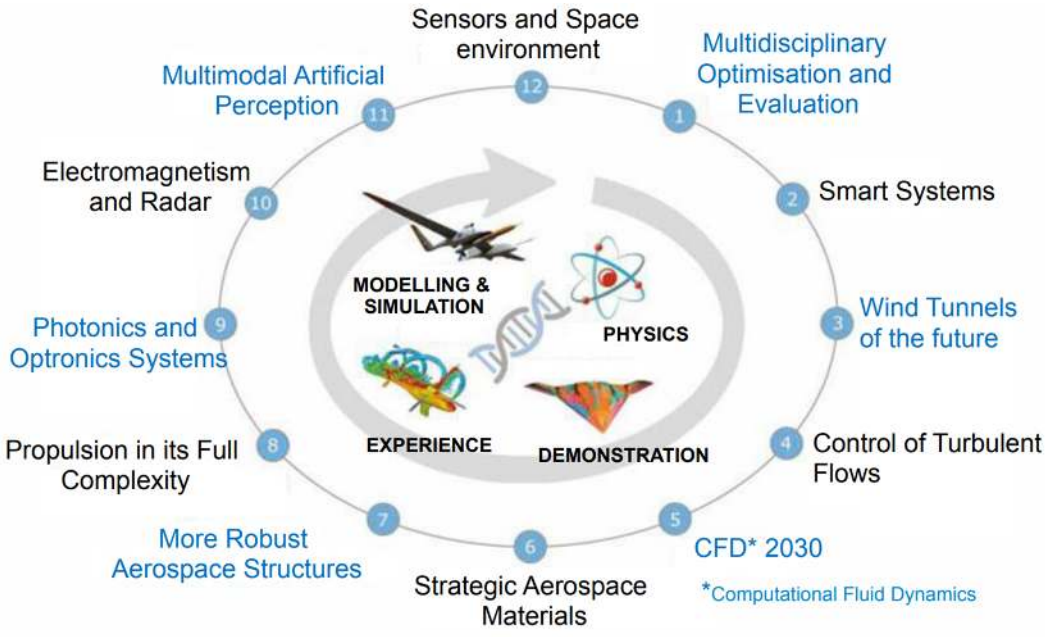

Figure 3: Diagram of the fields of study

The main mission of the research center is therefore to develop and guide aerospace research, to design all supporting means to implement the research, to disseminate and promote the outcome and to foster new initiatives in favor of aerospace research or industry. This is done by conducting or commissioning studies to the aerospace industry, by developing and implementing test and computational facilities and by ensuring the link with French and foreign organizations.

ONERA has indeed partnerships with laboratories like the CNES or the CNRS, with universities such as École Polytechnique, ENSTA ParisTech University, ISAE SUPAERO or ENAC. It is open to the international as it cooperates with international associations and networks like EREA or ESRE (the association or European Space Research Establishments), and has European contracts. It also collaborates with world-class companies and institutions such as JAXA (the Japanese Space Agency) or NASA aeronautics.

The research center is organized as follows:

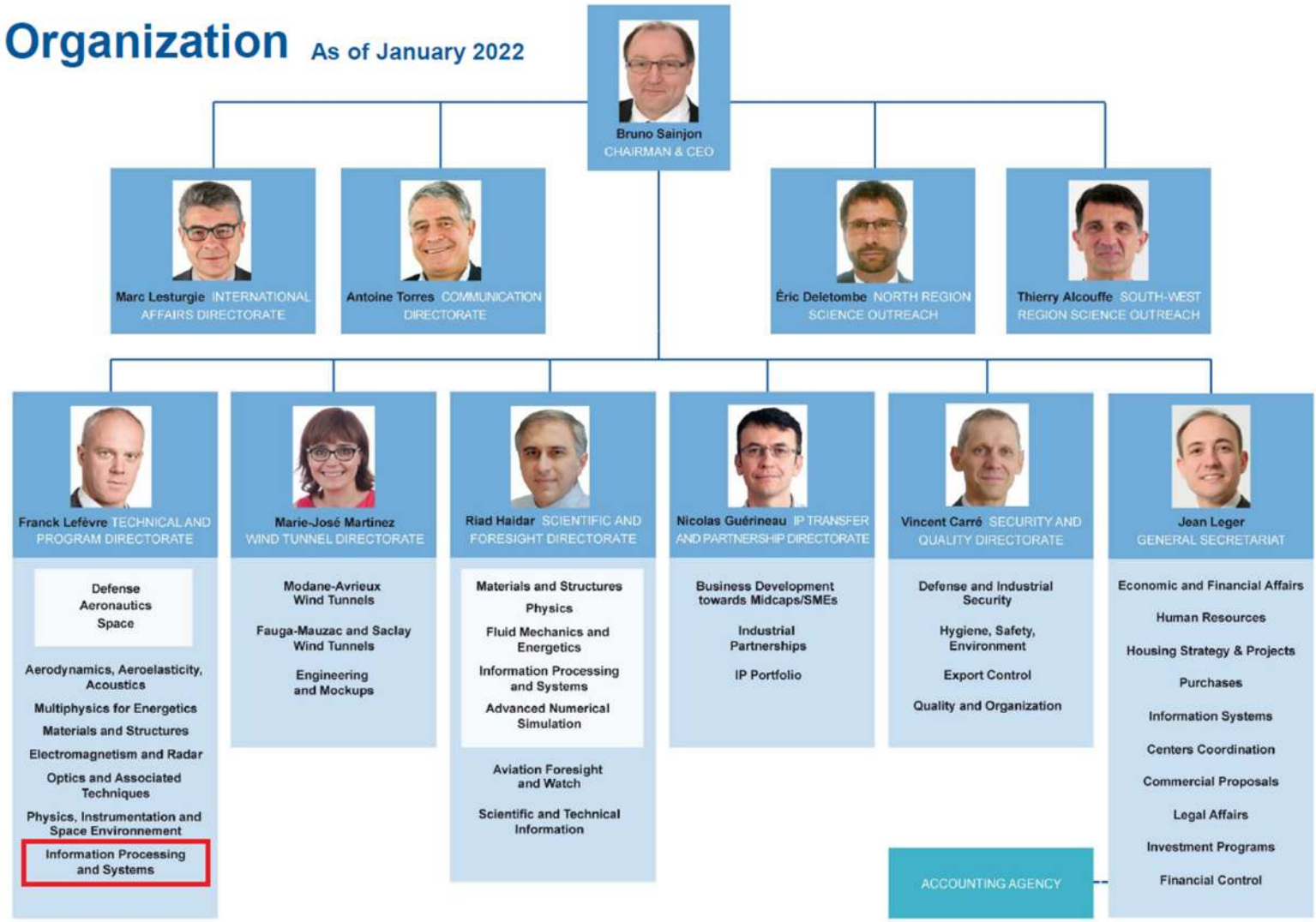

Figure 4: Organizational chart of the research center

Multiple departments exist at ONERA and each of them is interested in particular study topics. In Toulouse, five of these departments are present: the Electromagnetism and Radar department (DEMR), the Multi-physics department for energy (DMPE), the Optics and Associated Techniques department (DOTA), the department of Physics, Instrumentation, Environment and Space (DPHY) and the Information Processing and Systems department (DTIS).

The internship is done in the DTIS department.

1.2 Presentation of the DTIS department

The Information Processing and Systems department conducts studies and research to master the design, operations and autonomy of aerospace systems. The department implements its skills in the fields of aeronautics, space and defense with the main applications: aircraft (transport aircraft, fighter jets, UAVs, airships, helicopters, etc.), aerospace systems (air transport systems, launch vehicles, satellites, etc.), information systems (monitoring, localization systems, etc.) and defense systems (missiles, systems of systems, etc.). The DTIS carries out activities for the benefit of the industry and brings its expertise to the state services.

The department is composed of 16 research units which are located in three different sites in France: Palaiseau, Salon-de-Provence and Toulouse. The internship is performed in the SYD (Smart and decision-making systems) unit of this department in Toulouse. Part of this research unit is dedicated to the study and development of modeling frameworks and algorithms for the planning, scheduling and optimal allocation of resources in a context where the time and space aspects cannot be neglected. These optimization algorithms can be applied to Earth Observation Satellite constellations, air traffic, security analyses, or decision-making for robots autonomy.

1.3 Technical Environment

For this internship, I was supervised by three ONERA researchers, Gauthier Picard, Stéphanie Roussel and Cédric Pralet. I had weekly reunions with them to present my work. I also had meetings every month with Airbus Defence and Space to discuss the progress made regarding the LiChIE project.

2 A satellite constellation planning problem under weather uncertainties

2.1 Practical context

In recent years, there has been a large increase in the development of satellite constellations. Such constellations help to provide richer services and to satisfy many observation requests that can be formulated by the end-users. However, operating numerous Earth Observation Satellites (EOSs) can lead to complex combinatorial problems. The aim of some of the projects at ONERA is to develop models and solving methods in order to process and solve these combinatorial problems.

The BPI PSPC project "LiChIE" is a project supported by the French government in the context of the "Programme d'Invertissements d'Avenir". It is coordinated by Airbus Defence and Space and founded by BPI. This project concerns the next Airbus satellite constellation which should provide a better revisit rate of the different locations on Earth and a better quality of the observations made.

This project is done in collaboration with ONERA. The research center offers its expertise concerning the instrument production line and optimization missions. Many problems are tackled in this project such as the ones concerning ground mission planning, resource sharing between users, the allocation of orbit portions for exclusive users, and on-board planning methods.
Regarding the ground planning aspects, several problems must be tackled. Requests for observation of geographical areas are made and it is up to ONERA researchers to determine when to make the observations and with which satellites, but also when to dump the data and on which ground station. The goal is to optimize all these different criteria with the different types of requests (systematic, periodic, etc.) while making sure that the generation of the plans takes as little time as possible to be computed in order to be quickly sent to the satellites. As a solution, for each satellite, a plan is generated. These plans might need to be modified or generated again during the process to comply with the different requests. According to the figures that can be found in the literature, about 50% of the images taken are discarded because of clouds. Therefore, it is essential to handle weather uncertainties in order to avoid capturing cloud-full images.

2.2 Objectives

The objective of this End of Study Project is to provide a model and analyze several solution methods to take into account the cloud cover uncertainties for Earth Observation Satellites (EOS) scheduling.

This End of Study Project focuses on the development of different decision making methods to solve the problem. A first track consists in defining a Mixed-Integer Linear Programming (MILP) model to find a first solution. Later on, another method, using a decison tree generation is implemented to better solve the problem in terms of number of requests fulfilled, in terms of time at which the images are available to the user, and in terms of number of images failed.

Another objective of this internship is to characterize weather uncertainties using real weather data. To do so, meteorological forecasts are retrieved, and a method is developed in order to take those uncertainties into account in the scheduling. It can also be interesting to have a measurement of the inaccuracy of the forecasts in the optimization models.

3 State-of-the-art

In the literature, many articles have tackled problems concerning Earth Observation Satellite scheduling, whether considering uncertainties or not.

3.1 Earth Observation Satellite (EOS) scheduling

This section describes some defined problems tackled in the literature concerning EOS scheduling without real consideration of the uncertainties.

3.1.1 Some problems addressed at ONERA

Several challenges can be addressed regarding EOS constellation problems and the related applications. Many of these challenges are introduced in [4] and three main categories are targeted.
The first one focuses on the constellation design problems, and how to size a constellation and define the position of its satellites and their orbits, as well as the position of the ground station used to send plans to the satellites and to download the images taken. These problems are presented as a multi-objective problem where the objective is to optimize at the same time the number of times the satellites pass over target regions, the delay of revisit and the cost of the constellation. More precisely, the aim is to select the best subset of satellites, in some cluster, with the cheapest cost, in order to perform predefined tasks. The cost of the team consists of the aggregation of the costs of all the tasks that need to be performed.
The second category introduced is the offline operations. The aim is to compute offline plans for each satellite given an order book. The scheduling can be done under uncertainties, which can be meteorological, especially regarding cloud cover forecast, but which can also result in the limitation of the memory taken for each observation regarding the on-board memory and the compression hazards. One approach to solving this kind of problem may be to develop probabilistic Distributed Constraint Optimization Problems (DCOP) to solve the problem in a distributed way. Indeed, this is a technique, according to [5], where agents cooperate to solve a given problem by choosing values for a set of variables in a distributed way so that the cost associated to the constraints over the variables is minimized. The probabilistic aspect extends classical DCOPs and the outcome on the cost function is augmented with stochastic values.
The third category discussed is the online operations, where the problems are assumed to be dynamic. Indeed, satellites can be seen as dynamic systems that are deployed in a dynamic environment. Then the observations done by them can be rescheduled and different tasks can be added dynamically. Therefore, a revised plan needs to be pushed as soon as possible. It is thus possible to formulate these problems in a dynamic way, i.e. whose definition (variables, constraints, agents) evolves in time. Techniques exist and can be used to solve this kind of problem such as the Dynamic Distributed Constraint Optimization Problems (Dyn DCOP) technique or multi-agent plan repair techniques, that change some parts of an existing plan instead of rescheduling from the scratch. It is also possible to delay multi-agent online operations.
Finally, the paper [4] discusses the fact that an EOS constellation can be used or shared by several users and that it can be decided to allocate some orbit portions of some satellites to specific users.

The problem of orbit sharing and allocation is also tackled in [1]. It is indeed assumed that some orbit portions are fully controlled by some exclusive users and other users do not have exclusive access to any orbit portion. Each user (exclusive or not) has observation requests and it can be possible to ask to make observations in portions of orbit controlled by other users. Then, an exclusive user can agree to add non-exclusive requests inside his controlled orbit portion. This is done, for instance, when none of his observations are made or by

shifting its observations to allow the request to be accepted when it is advantageous. Moreover, in some cases, the exclusive user can have multiple windows and accept the request by moving one of its own observations inside another window that he owns. The aim is finally to find the best exclusive that can take the observation requests of non-exclusive users. The plan to be sent to the satellite will be recalculated or at least modified as the requests are added. Several constraints must be taken into account in the planning such as the one on the capacity of the satellites, the maximum number of observations during the orbit plan, the transition time between two given observations, the observation opportunities and the validity time window.
To solve this problem, two approaches are introduced. The first one is based on an auction approach and a Consensus-Based Bundles Algorithm (CBBA) has been implemented. This algorithm consists in making iterations between two phases, a building phase, where each bidder constructs a unique bundle of items, and a consensus phase, where marginal costs are calculated for each bidder to see if adding an observation request asked by non-exclusive users would improve the final solution or not. The best bid is kept in the end. The second approach is based on the DCOP which was presented earlier. Then, a greedy and fast algorithm is used to solve this problem. It provides a good solution while regarding realistic large-scale problems.

Another challenge studied at ONERA, with regards to EOS scheduling, is to develop a generic approach to deal with requests given by end-users that are more complex than what can be seen in the literature. In [2], request modes are used to manage the variety of requirements. These modes describe the link between observation requests, observation opportunities and downloading opportunities. Each mode of a request therefore represents a way for fulfilling this request. A reward is associated with each mode and it is used to measure the quality of the solution, taking into account some uncertainties in the reward. The modes are also used in literature in multi-mode Resource Constraint Scheduling Problems (RCSP) where the aim is to schedule a set of activities over a set of machines with limited capacity (like the satellites) knowing that there are several ways of performing an activity. In [2], two different approaches have been implemented such as the Mixed Integer Linear Programming (MILP) using CPLEX solver and an insertion procedure with a Large Neighborhood Search (LNS) algorithm that is based on iterative request insertions and plan repair. LNS is a technique that aims at finding a near-optimal solution by repeatedly transforming the current solution into a different one in the neighborhood of the current solution. At each step of the insertion procedure, a request is inserted or removed. This process is applied until the solution becomes infeasible. The solution is repaired by changing the order of the activities for the satellites or inserting new requests to the solution. It can be done, for instance in a greedy way, by choosing each time the request that offers the highest reward or by taking into account the load of the satellites and minimizing the height of the maximum peck of the satellite load profile. The aim is to finally obtain the best mode.

In this section, several problems regarding EOS scheduling problems have been studied and some solving methods have been presented. However, in all these approaches, aspects related to meteorological uncertainties are sometimes not taken into account or highly approximated and often only integrated in the rewards. This is not fully satisfying to provide a good scheduling in this case. Indeed, when the cloud cover is too important, the images of the ground taken by the satellites will not be exploitable since on these images, only the clouds will be visible and not the area targeted by the users.
It can be interesting to see what is done in the general in the literature concerning the EOS scheduling.

3.1.2 General literature review on Earth Observation Satellite (EOS) scheduling

Many other articles have tackled the problem of EOS scheduling in the literature. Indeed, in [6], authors focus on permutation-based approaches to solve an EOS scheduling problem. The observation opportunities are ordered in a certain way and each permutation between requests is evaluated using a greedy algorithm. The aim is to

find the best possible permutation that will allow to do as many observation requests as possible. A study on different permutation techniques is, therefore, made to compare the different algorithms that can apply to the problem such as the Simulated Annealing (SA) algorithm, the Hill Climbing algorithm, the Genetic Algorithm or the Iterated Sampling algorithm. It is assumed that a scheduler is efficient if at least 250 observations are made each day. Under the defined conditions, the SA algorithm with random swap provides very good solutions. The Simulated Annealing, as described in [7], is a metaheuristic method for global optimization problems. It is based on the physical phenomenon of annealing of materials. In metallurgy, there are two steps for physical annealing: first, the solid is heated to a very high temperature (until it glows) and then, it is slowly cooled to room temperature.

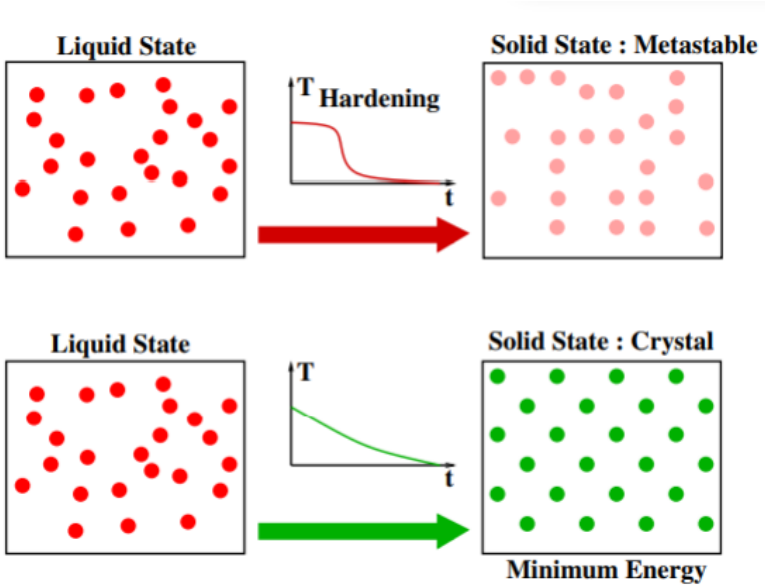

Figure 5: Visualization of the state of the material and how we can lower temperature to cool it

It can be seen in Figure 5 that when the temperature drops too quickly, the state ends in a metastable state whereas when the temperature drops slowly, the atoms in the solid are well organized. The aim is to make the material reach a solid state where all atoms are symmetrically organized. In the algorithm, the state-space point represents the different states of the solid and the function to maximize represents the energy of the solid. SA algorithm, as it can be seen in [7], is based on Monte-Carlo algorithms and is an adaptation of the Metropolis algorithm. The Monte-Carlo algorithm is a local search algorithm used to optimize a cost function. It converges to local optima whereas the Metropolis algorithm uses a criterion in order to escape from local maxima.
The operating context in the EOS scheduling problems can change. Indeed, in [8], it is considered that new requests can be sent to the scheduler. It is assumed that the schedule has already been done and iterative repairs are made in order to support continuous modification and to update the current working plan. The approach is implemented using a system called CASPER (Continuous Actively Scheduling Planning Execution and Replanning) where the planning is done continuously. In addition, another context where scheduling can be applied is when the satellites are of a different type.

There are different types of Earth observation satellites, whose plans must be programmed. The maneuverability of a satellite is defined using three axis: the roll, the yaw and the pitch axes (Figure 6).

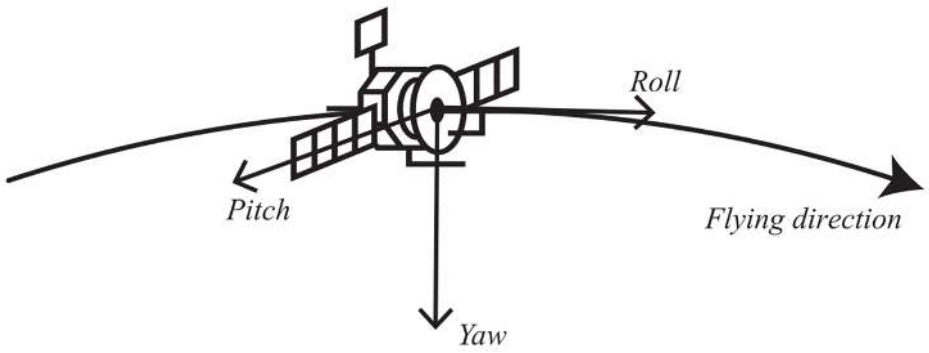

Figure 6: Schematic view of the manoeuvrability of a satellite [9]

A Conventional Earth Observation Satellite (CEOS) is a kind of satellite that cannot move toward all three axes, they only have one degree of freedom to acquire the different images. The Agile Earth Observation Satellites (AEOS) are different in this aspect as they have the possibility to maneuver in the three directions. It is indeed a new generation of satellites that enables us to have more maneuverability for image acquisitions as well as for transition between observations and efficient scheduling observations.
As illustrated in Figure 7, the AEOS can observe a target from three different start times that are placed inside an observation window using the agility on the pitch angle. Thanks to these satellites, the time window where a target is visible can be longer. An AEOS scheduling problem is therefore investigated in [10]. It is considered that each observation task has a profit and that the transition time and this profit are time-dependent. A weighted network model is defined where the aim is to find the optimal sequence of vertices with the highest sum of collected profits. In order to solve the problem, an Iterative Local Search (ILS) algorithm is used that combines a remove and an insertion procedure. Then, a bidirectional dynamic programming approach is proposed and is integrated to the ILS in order to evaluate the solution found with the procedures. After each insertion, the observation start times are optimized to improve the solution and the maximal collected profit of the task sequence is calculated. It has also been highlighted that this problem could be modeled as an orienteering problem with time-dependent travel time and profit or be extended to other combinatorial optimization problems such as the Vehicle Routing Problem (VRP) or job shop scheduling problem. Indeed, in this kind of problem, the machines can be associated with the satellites and the jobs with tasks asked by the users. A job shop scheduling problem with time window is introduced in [11] where each job has to start and finish within a given time window and the goal is not to maximize the profit but to minimize the cost. A difference between the two problems would be that in the job shop scheduling problem defined before, all jobs have to be finished whereas in EOS scheduling, it can be allowed not to plan all tasks.

The AEOS scheduling problem can also be formulated as a multi-objective optimization problem. Indeed, in [12], a multi-objective local search heuristic is developed for scheduling observations. For solving such a problem, Pareto fronts are computed. A Pareto front is defined as a set formed by the non-dominated points in the Pareto sense. It is assumed that a point x dominates y if x is as good as y for all objective and better than y for at least one objective, i.e.: $f_i(x) \leq f_i(y) \ \forall\, i \in \Omega \ \& \ \exists\, i_0 \mid f_{i_0}(x) < f_{i_0}(y)$. In [12], an Indicator-Based Multi-Objective Local Search (IBMOLS) is used to solve the problem. It is a generic algorithm that combines the use of a local search and a binary indicator. A random population of non-dominated solutions is first generated

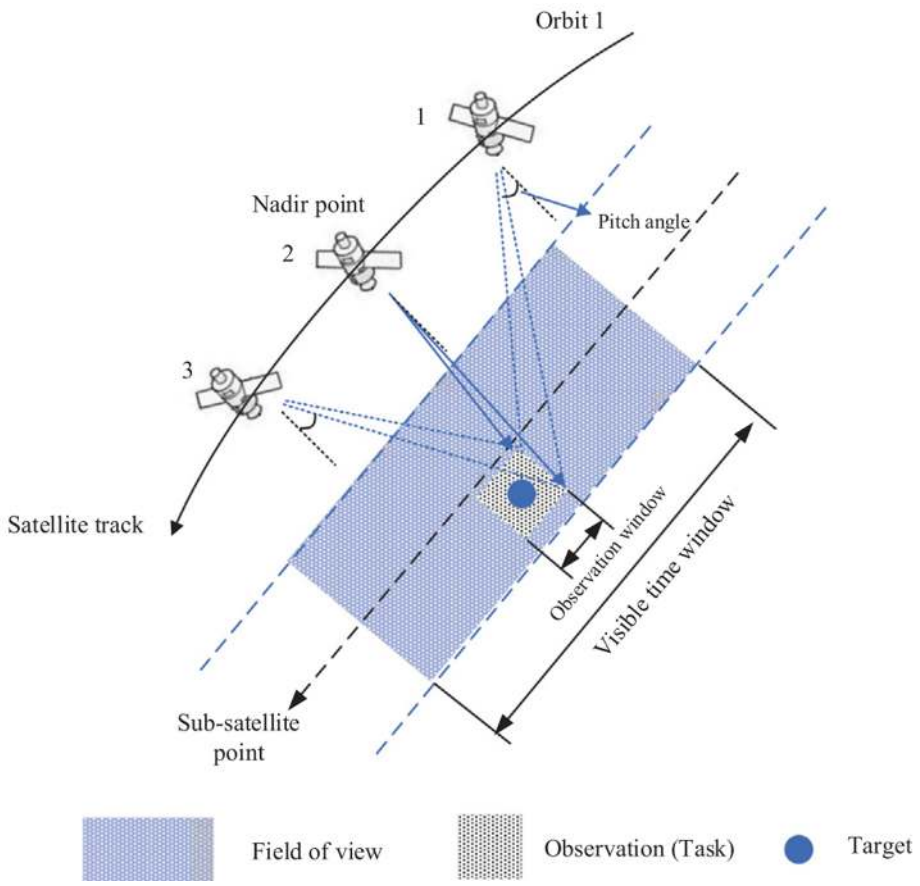

Figure 7: Using an Agile EOS to observe a target [10]

and stored, and then, for each iteration, a new population is generated using perturbations. This algorithm is compared to a Biased Random Key Genetic Algorithm (BRKGA) which combines a Genetic Algorithm (GA) and a concept of random key. A Genetic Algorithm is a metaheuristic that is part of the Evolutionary Algorithms (EA). In this kind of algorithm, the solutions are generated at each iteration based on biologically inspired operators such as selection, mutation and crossover. In [13], a GA is combined with a tabu search algorithm, which is a neighborhood search algorithm that prohibits the repetition of previous work and avoids local optima, to solve the multi-objective problem. Mission priority levels are associated with each target and the aim is to maximize the sum of the priority levels and the total mission revenues and to minimize the waiting time for missions that require urgent execution.

In [14], a new way of dealing with multi-objective optimization for observation scheduling is introduced. Indeed, the authors use a semi-Markov Decision Process (MDP) to formulate the satellite task scheduling problem. The aim is to enable us to decide which image should be collected to maximize the total reward over the planning horizon. It seems difficult to solve the problem with exact MDP algorithms since the state space grows exponentially with the number of images that need to be scheduled. A Forward Search algorithm was therefore implemented which is a simple and computationally efficient approach but it is more effective while planning with one single objective. In this light, a Monte-Carlo Tree Search approach was introduced to balance the multiple objectives while planning and showed real good performances for finding the best

reward. Another way of finding the optimal Pareto front is to use decision trees. In [15], a multi-objective genetic programming approach is proposed to develop Pareto optimal decision trees. A decision tree is set by following a sequence of simple tests to reach a final classification decision. According to the authors in [15], evolutionary computation heuristics are suitable for simultaneously evolving toward multiple alternative Pareto optimal solutions for multi-objective optimization problems. The fitness of an individual is assigned based on its relative non-dominance. The aim is then, to find non dominated decision trees that are Pareto optimal.

In what we have seen as far, the planning done still does not take into account the uncertainties, concerning new requests that may arrive in random ways, the volume of data collected by the satellites or even the weather, regarding the presence of clouds. This is, therefore, what we will focus on in what follows.

3.2 Scheduling under uncertainties

In the following section, the focus is made on the impact of the different uncertainties described above on scheduling, and then on how people in the literature manage to find an optimal scheduling taking into account the uncertainties.

3.2.1 Impact of uncertainties on general model

In the literature, different articles have tried to characterize the different uncertainties and to implement optimization algorithms that take them into account while scheduling. In [16], a review of the literature with regards to scheduling under uncertainty and on the modeling techniques is made. Scheduling approaches most commonly used consider the environment to be completely deterministic. However, in reality, these disturbances can drastically affect the planning. The article therefore introduces different scheduling approaches that can be found in the literature. For instance, proactive or robust approaches are widely used as they try to anticipate the uncertainty and then they design off-line schedules that are relatively insensitive to uncertainty. Reactive approaches are often used when the uncertainties are too large, the environment too highly perturbed. All decisions are, therefore, made in real time to adapt themselves to the environment. Hybrid methods are also present in the literature. These kinds of methods have two phases in order to support risks. Other approaches can be defined such as stochastic approaches or fuzzy approaches which are guided by the nature of the problem or the probabilistic approaches that define a probability of success and aim at building a schedule that maximizes this probability. This last approach has been widely used to handle weather forecasting uncertainties.

In the problems defined in the literature, uncertainties can affect different kinds of processes. In [17], an orienteering problem is solved in which the time and benefit are considered as uncertain variables based on uncertainty theory. This problem is represented as an uncertain complete graph where the vertices are the points that must be visited at most once and the weight on the edges represents the time between the point. A metaheuristic algorithm is designed using a iterative local search strategy similar to the one introduced in the section above in which first, an initial feasible route is generated and then, points are inserted before the endpoint. Perturbations are also designed in order to avoid local optima. In [18], a model of a personalized tourist trip is presented and in this example, it is the visit duration for each point that is this time uncertain. The problem is defined as a stochastic team-oriented problem with time windows and a Genetic Programming based Hyper Heuristic (GPHH) approach is implemented to automatically design different reactive solutions. This approach is also used in [19] to solve the Uncertain Capacity Arc Routing Problem (UARP) within a collaborative and multi-vehicle framework. The cooperative aspect is also included in [20] where the job shop scheduling problem is considered in a cooperative environment. The uncertainty is placed this time on the

nature of the jobs. They are not known in advance and appear randomly during the production process. The job release dates and due dates can change and new jobs need to be taken into account during the process. Moreover, the operation processing time can randomly vary and a machine can break down. Therefore, the production schedule is made in a reactive manner so that it can be adapted to all different kinds of solutions. The problem can be decomposed into different interdependent subproblems for each one of the machines. The objective is then to minimize the maximum lateness of all machines. In [20], negotiation between the machines seems to be an essential aspect to take into consideration.

The idea of minimizing the cost associated with lateness penalties is also taken up in [21] but with the addition of a resource maintenance aspect. A two-stage Stochastic Mixed Integer Program (SMIP) is used. Two different parameters are considered uncertain: the processing times of production and the duration of maintenance activities. To model these two uncertainties, probability distributions are used. A combination of Genetic Algorithms, which can efficiently explore the search space to find a near-optimal solution, and Monte-Carlo simulation models, that can predict the probability of different outcomes while using random variables such as in our case, are proposed to solve the problem. In [22], it is this time a probabilistic approach that is introduced to solve the problem of short-term maintenance scheduling under weather forecasting uncertainties. Indeed, the maintenance of offshore wind farms constructed near the sea is constrained by sea and wind conditions at every stage of the mission. The aim is therefore to have the next day's scheduled mission taking into account weather forecasts. Indeed, the decisions are informed by day-ahead and longer-term meteorological forecasts. The day before the maintenance operations, weather forecasts are consulted to determine whether the crew must prepare themselves to attempt the mission, and on a working day, the decision can be revised with the available updated forecasts and it is then decided whether the crew actually goes or not. In the probabilistic approach, a probability of success is predicted and compared to a ratio of the cost of attempting the mission over the loss if the mission is abandoned. In the end, this approach is compared to a deterministic approach that examines weather forecasts and verifies that the weather conditions are within the safety limit. Results show that this is unsatisfactory on marginal days since the risks and costs are not part of the decision-making.

The many different problems cited before can help to better understand the importance of characterizing uncertainties and they provide ways of solving applied issues such as the EOS scheduling problems under uncertainties. Indeed, in what was presented previously, it can be highlighted that many aspects of the different problem can be related to EOS scheduling problems.

3.2.2 Earth Observation Satellite (EOS) Scheduling under uncertainties

In practical satellite scheduling, the uncertainty, whether it is about the arrival of new acquisition requests during the mission, the real data volumes collected at execution time, the real state of the satellite or the presence of clouds, is inevitable and has to be handled.

In [23], an imaging order scheduler for an EOS is developed. The problem is formulated as an integer programming problem where a Lagrangian relaxation is used to decompose the problem into separable sub-problems. The success rate of each job during the scheduling time horizon is estimated using probabilities based on the Bernoulli distribution. A probability of success is also used to solve multi-satellite observation and data-downlink scheduling under weather uncertainties problems in [24]. A two-stage flow shop scheme is proposed for the observation and data transmission while considering weather uncertainties using probabilities of success under certain weather conditions. The problem is modeled as a MILP model with three objective functions which are the maximum completion time, the total weighted completion time and the overall weighted success rate in

terms of probability. It is solved using commercial solvers such as AMPL and CPLEX.

There are other ways that exist in the literature to formulate an EOS scheduling problem under uncertainties. Indeed, in [25], another modeling approach is implemented that also has a probabilistic aspect. The cloud coverage is formulated for each time window of observation as a stochastic event and the problem is formulated as a Chance Constraint Programming (CCP). Chance Constraint Programming is a formulation of a problem in which the objective function or the constraints are defined using probabilities. However, with this formulation, the probability in the chance constraints is hard to calculate and a feasible solution is hard to find. The feasible regions are generally non-convex and the number of forbidden sequences is exponential in the number of tasks. An approximation is therefore made to turn the CCP problem into an Integer Linear Programming (ILP) problem. A sample of realization is created using Monte-Carlo Simulation and a Branch and Cut algorithm is designed and solved using CPLEX. The cloud uncertainties are also formulated as stochastic events in many other studies such as in [26], [27] or in [28]. However, when in [25] the problem is formulated as a CCP and solved using CPLEX with a Branch and Bounds algorithm in [26], a Dantzig Wolfe decomposition is used and the problem is decomposed into a master problem and some sub-problems and then, solved using a Branch and Price algorithm with CPLEX. In [27], the model is again different as a Directed Acyclic Graph (DAG) formulation is presented. A graph is defined for each orbit and the tasks are represented as the nodes of the graphs. Some exact solution algorithms using a spatial Branch and Bound algorithm with the solver Couenne are implemented and then compared to heuristics such as column-based heuristics or knapsack-based ones. Such exact and inexact algorithms to solve the same kind of problem are also introduced in [28]. It is highlighted that it is harder for exact algorithms to handle large-scale problems and heuristics perform better for these problems. A comparison of different heuristics such as one based on the knapsack problem or another one that selects, for each orbit, all or only a certain number of the best solution columns and combines them to better optimize the objective function.

Other kinds of formulation of cloud coverage uncertainties are presented in the literature. Indeed, in [29], a budgeted uncertainty set is introduced to describe these uncertainties. Because of the uncertainties, the profit of a mission is bounded between the nominal profit minus or plus its deviation. The deviation depends on the accuracy of the weather forecast and an integer, called the budget. The budget is defined as the number of scheduled missions associated with a certain target that can deviate from the nominal value. A robust formulation is used and a column generation procedure is executed to find the optimal solution. In [9], the same budget parameter is used and the problem is this time applied to an agile satellite scheduling problem. A hybrid heuristic is used and combines the column generation procedure introduced previously and a SA heuristic. This algorithm is also used in [30] to solve the problem of multiple AEOS scheduling problem with cloud uncertainties. However, in this article, a CCP is used to describe the uncertainties. Thanks to an SA algorithm, an initial solution is generated and the approach allows to gradually approach the solution to maximize the entire observation profit.

The uncertainties with regards to the cloud cover can also be taken into account using a MDP Network. Indeed, in [31] a MDP framework is used to define a way to aggregate three different criteria such as the gain, the number of remaining feasibility opportunities and the meteorological forecast. The approach developed also uses probabilities such as the probability of realization of the photograph for a certain day d or probability of non-realization of the photograph the days after d and allows to define a kind of regret. For large instances, algorithms using Iterative Local Search schema provide good solutions whereas Branch and Bounds are more effective for smaller instances. The problem is also formalized as a MDP in [32] and in [33]. The acquisition scheduling approach is based on Deep Reinforcement Learning, which is a very suitable method to solve complex sequential decision making. Using Deep Learning improves the quality and better captures the uncertainty of cloud cover forecasts. Indeed, the latest cloud cover observations are ingested together with the prediction of

classical nowcasting techniques in order to estimate a probability of success for the observations. In this approach, an agent learns how to behave through trial-and-error interactions with a dynamic environment. The actions taken by the agent are defined by some policies. The aim is to find a policy that maximizes the expected reward over the finite horizon. Different kinds of heuristics have been developed and algorithms were implemented and assessed. MDP algorithms, as highlighted in [34], allow relying on known models of probabilistic transition and on reward learned beforehand or directly learning the optimal policy. However, in [34], a new objective is defined to make sure that the scheduler meets expected deadlines. It is then possible to dynamically alter how requests from the different users are prioritized and when a request priority is boosted, it can penalize the surrounded requests. For this kind of problem, the MDP formulation can not be used. Indeed, even though this formulation can handle large instances, it is not enough. Moreover, the MDP assumes that the transition evaluation between two different targets is quasi-immediate while it is not the case, especially in the problem targeted in [34]. The potential of Evolutionary Algorithms (EA) is investigated and several approaches are developed to optimize request priorities based on Local Search and Population-Based Incremental Learning (PBIL). A decoder in the optimization process is then introduced to help optimize the algorithm to produce good solutions from the start of the search. The choices are made in short term which implies that it is essential to take into account weather uncertainties. The study is done based on available historical weather data for a period of about 30 years. A future work considered in this study would be to use a multi-objective function. This is what is done in [35]. A multi-criteria decision-making model is therefore used in order to integrate a real-time scoring approach of imaging attempts, considering the aspect of cloud cover, customer priority, and image quality criteria in the objective function. The method explicitly evaluates the conflicting alternatives regarding the different three criteria defined. Two different established methods are discussed: one using an Elimination and Choice Expressing Reality (ELECTRE) model, which is a decision support and multi-criteria analysis method, and another using a Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) model. Real weather forecasts and corresponding observed cloud cover criteria are used to help score the quality of an image. This quality, therefore, depends on the cloud cover but also on the customer type and the sun elevation for instance. However, threshold values are defined for each criterion, such as for cloud cover uncertainties, to characterize the observation window to be used. There is therefore a weakness concerning the definition of these threshold values that can affect the final solution.

Reactive methods can also be used to deal with weather uncertainties and to manage to take them into account as best as possible. Indeed, two different approaches can be used to deal with uncertainties: proactive and reactive scheduling. According to [36], reactive scheduling procedures are required to repair the deviated or failed schedule during the execution to stabilize the schedule and obtain higher schedule profits. Different types of disruptions can occur during the implementation of EOS scheduling under cloud uncertainties: the tasks that fails because of the presence of cloud cover and that can not be repaired and the ones that fails but can be repaired. Furthermore, reactive scheduling can be done by completely rescheduling the observations from scratch at each iteration or by repairing the schedule. A replanning scheme is defined in [37]. A proactive schedule is first made based on a CCP model and then, cloud forecasting is continuously conducted. At each iteration, a replanning algorithm based on a rapid insertion strategy is used to recalculate the plan, taking into account the initial schedule and the cloud forecast. The amount of cloud cover is predicted using a predictive recurrent neural network and multiple satellite cloud images. It is considered that the cloud cover will not change drastically during the period of the visible time window. Moreover, the value of cloud percentage is calculated by linear interpolation of several cloud cover percentages. In the results, it can be seen that after 3 hours, the percentage error for the cloud forecast would be about 10 %. As a result, rescheduling errors are made as the forecast accuracy decreases.

3.3 Weather forecasting models

It is essential nowadays to forecast the weather with the highest possible accuracy. Indeed, weather forecasts enable us to better protect people from meteorological disasters, but it is also highly used in transportation and also for planning or scheduling for example in maintenance [22] or for satellite constellations. It is therefore crucial to have accurate forecasting models.

The atmospheric models aim at simulating the behavior of the atmosphere. They are based on several equations [38] such as the equations of motions, the equations of mass conservation, the perfect gas equations and those of thermodynamics. In addition to that, Earth observations are taken into account as time goes by in order to have a forecast which is as accurate as possible.

Different models have been developed. In global models, the whole Earth is meshed, the evolution of the real atmosphere is described using mathematical equations [39] and the atmosphere is regularly reanalyzed to have more accurate forecasts. Different models exist around the world [38] such as the GFS (Global Forecast System) model developed in the United States, the CEPMMT (Centre Européen pour les Prévisions Météorologiques à Moyen Terme) model, the ARPEGE model (Action de Recherche Petite Echelle Grande Echelle) that covers the entire Earth, or the AROME (Application of Research to Operations at MEsoscale) model which has a finer mesh and where the forecasts are done for France. The last two models are currently developed by Météo France.

Limited area models, or also called regional models, are established by multiple centers around the world. The mesh is made in restricted areas of the Earth. We therefore obtain a better and finer resolution with these models.

In another direction, there are two different kinds of weather forecasts that can be used: deterministic forecasts and ensemble forecasts. In the first one, the atmosphere is modeled as finely as possible and thus, the models provide more detailed forecasts. According to [39], it performs very well in the first six days but many errors are usually made afterward. Also, these deterministic models provide one single forecast and do not give any information on the level of confidence placed in the forecast. On the contrary, ensemble forecasts use several deterministic forecasts and need the models to be run several times. The different models use slightly different initial conditions, or sometimes slightly different equations, to produce different forecasts that would take into account the model errors. These ensemble models use a much larger mesh size and have a lower resolution but they become more accurate than the deterministic ones as time grows; this can be seen in Figure 8. The ensemble model used at Météo France is named PEARP and is the ARPEGE ensemble forecast. Another model that can be used is the American one, GEFS, the GFS ensemble model.

Multiple errors can impact the forecasts. Indeed, the model is based on initial conditions that are constructed from observation data and these observations are not precise enough. Other errors arise from the weather model [39]. The equations used do not replicate exactly all of the natural processes at work in the atmosphere and errors are made and they become more pronounced over time. Finally, there are edge errors that are due to the perturbation of the boundary conditions. All these errors are considered when forecasting but they can explain the forecast errors made. To reduce errors, some forecasts, the ensemble ones, are made from the multiple runs of the models. This allows us to have a more accurate forecast but also to estimate the uncertainties. Indeed,the models are run several times and at the end, the forecast given is the one that is the closest to all the others. There is therefore a probabilistic side to this method that can help to estimate uncertainties. These forecasts are made while using ensemble model.

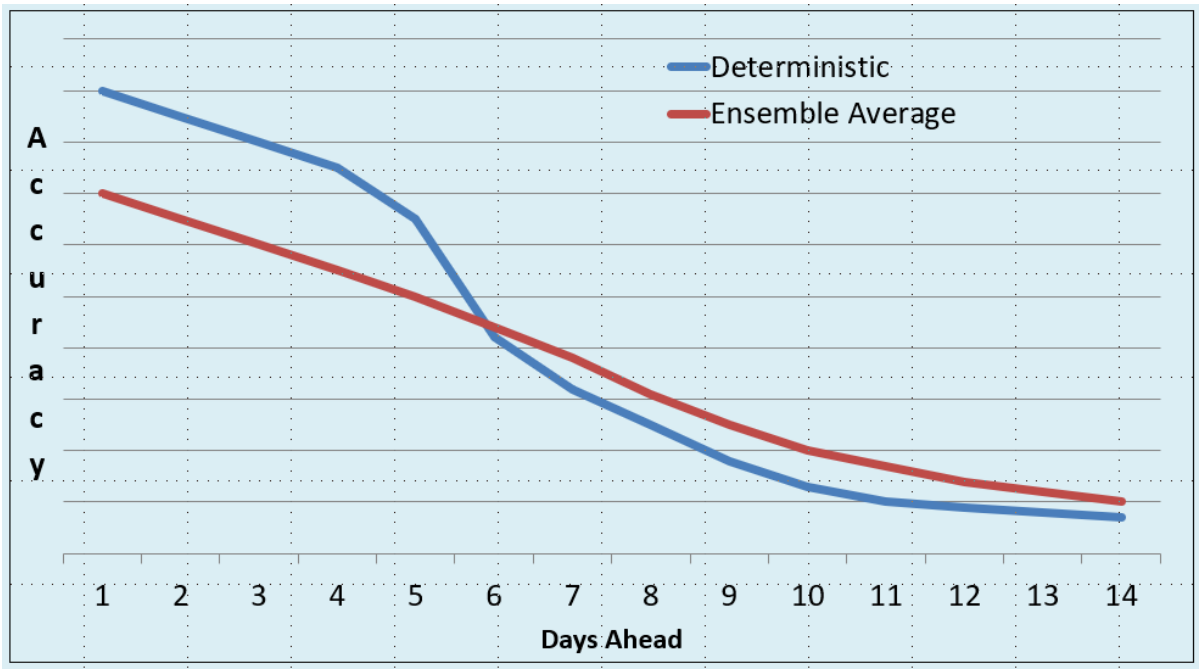

Figure 8: Schematic diagram of the typical accuracies of a deterministic forecast and an ensemble forecast at various lead times [39]

3.4 Analysis of the state-of-the-art

After what has been previously presented, it would seem interesting to use a probabilistic approach to model weather uncertainties. A first perspective for modeling the problem was, therefore, to define, from the uncertainties of the weather models, a probability of success that would be defined as the probability that the predicted Total Cloud Cover (TCC) in percentage is lower than the one requested by the user for a given observation target. This could be done using an ensemble model or if we had access to forecast uncertainties. However, the data using ensemble models are not freely available on the Météo France website nor are the uncertainties of the different forecasts. It can be, on the contrary, possible to find forecasts based on the American ensemble model named Global Ensemble Forecast System (GEFS), which is public. However, the runs are made each six hours and the forecast step is also of six hours. This would not be precise nor realistic enough in our situation. Moreover, our scheduling time is about four days and it was highlighted in [39] that deterministic models provide better forecasts that the ensemble ones for this period of time. In the view of these results, it seems difficult to define probabilities from real data. The model of our optimization problem will therefore not be done using probabilities. Moreover, the particularity of this study would be to use real weather data in the scheduling and therefore the method developed in [37] using neural network to predict the amount of cloud cover is interesting but will not be used to solve our problem. A new method needs to be found to model the meteorological forecast and to take into account the weather, and more specially the cloud cover, in a realistic way.

4 Description of the scheduling problem

The goal is to solve a scheduling problem applied to an Earth Observation Satellite (EOS) constellation. On the ground, there are multiple users that ask to have a shot of a target located on Earth, over a given time range. These pictures are taken using the satellites of an EOS constellation. The problem is to decide which satellite will be used to make which observation of the Earth. The best plan for all the observation requests asked by the users needs to be found while taking into account the cloud cover which can at any time invalidate the images taken.

4.1 Presentation of the problem

We have at our disposal a set of requests formulated by the ground users and a set of satellites that can perform some of the requests. For each request, a set of observation windows are defined for each satellite where it is possible to take a picture of the requested area. Figure 9 gives an example of what a given satellite can see of the ground and what shot can be made.

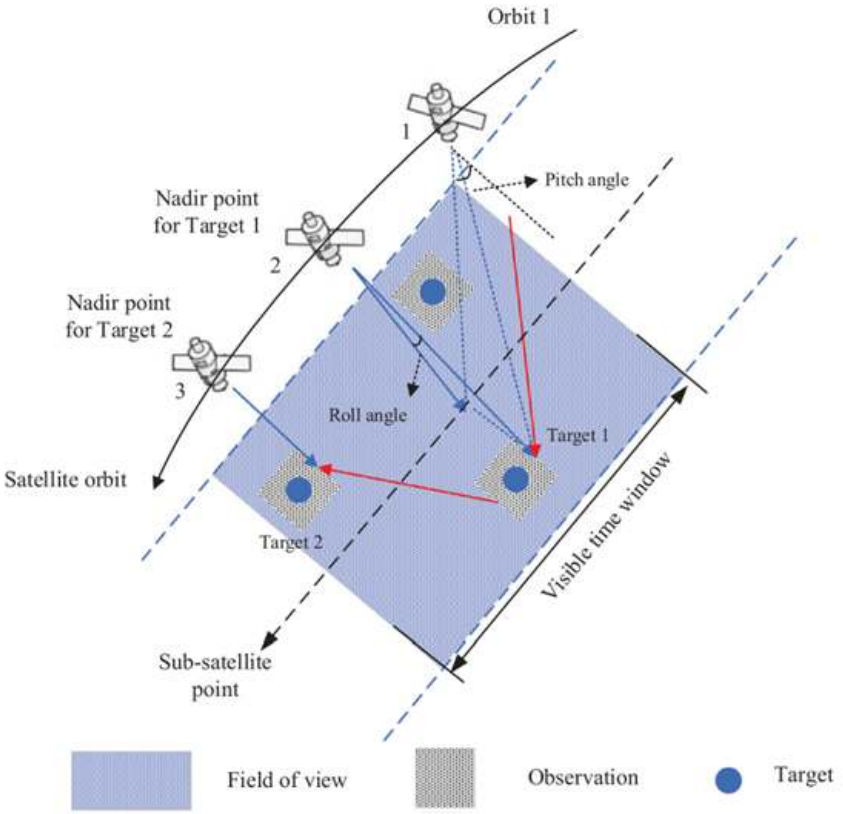

Figure 9: Representative diagram of the images taken by a given satellite [40]

The satellite is represented in its orbit at different times and what the satellite can see on the Earth is represented by a blue rectangle. In the figure, three targets are defined with the associated observation windows where it is possible for the satellite to take a picture. Each target is associated with one request asked by one ground user. We also define for each satellite a set of communication windows where it is possible for the ground station to send plans in order to specify to a satellite what request to make and when. These communication windows can

also be used to retrieve on the ground the pictures taken by the satellite. Moreover, for each request, a validity threshold is defined based on the cloud cover. Indeed, if the amount of cloud cover is too important, the image taken by the satellite will not be useful for the user. It is therefore important for the images taken to have a total cloud cover lower than the threshold defined by the user. This threshold can depend on the user.

4.2 System timeline

As for the system timeline, for all satellites in the constellation, multiple observation windows can be associated with the requests. Over the planning time frame, the satellites can move around the Earth multiple times and targeted area can be seen multiple times with one satellite. By adding all the observation windows of all the satellites, we obtain, thus, several observation opportunities associated with one given request.

A simple example of the system timeline is given in Figure 10.

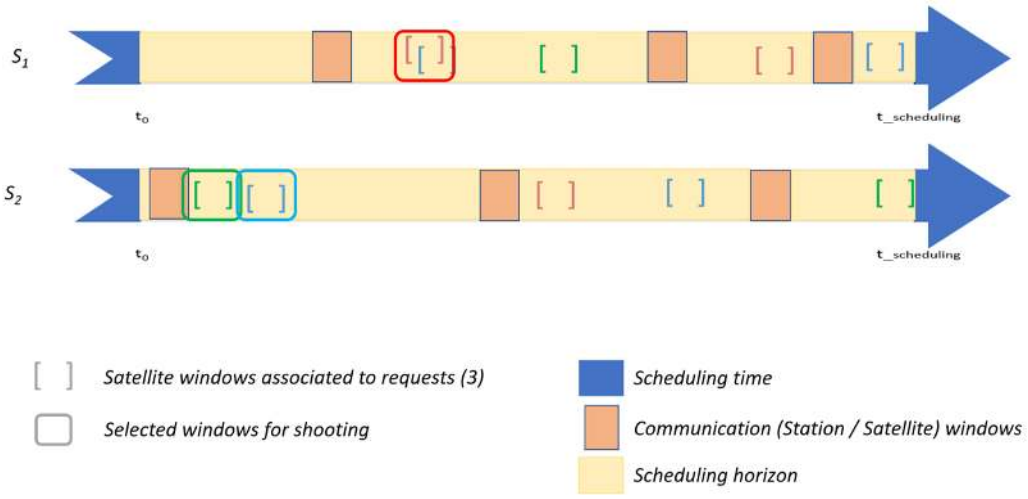

Figure 10: Example of the system timeline

In this example, two satellites (s_1 and s_2) and three requests (blue, red and green) are considered. The orange rectangles represents the communication windows between the satellite and the ground stations, the yellow part represents the scheduling horizon and in blue, we have the boundaries between which the planning is done. We can see that two windows are available to make the blue request with satellite s_1, however there are four observation opportunities for this request when considering the two satellites. Similarly, we can count three for the green and red requests.

For all requests, only one window per request can be selected to make the observation. A schedule is given in Figure 10 in which windows that are selected to make the observations are marked. In this plan, the scheduler tries to plan the requests as early as possible so that the ground users can quickly have their images available.

4.3 Impact of weather uncertainties

In the previous schedule, the cloud cover is not considered. However it is important to take into account its impact. Indeed, when a request is posted, the user sets a cloud cover threshold. Above this threshold, the image

is not considered to be interesting enough for the user and it will be invalidated.

In the previous example, the thresholds set by the users are assumed to be equal to 10% for all users. The Total Cloud Cover (TCC) is now defined for all observation windows and Figure 11 is obtained.

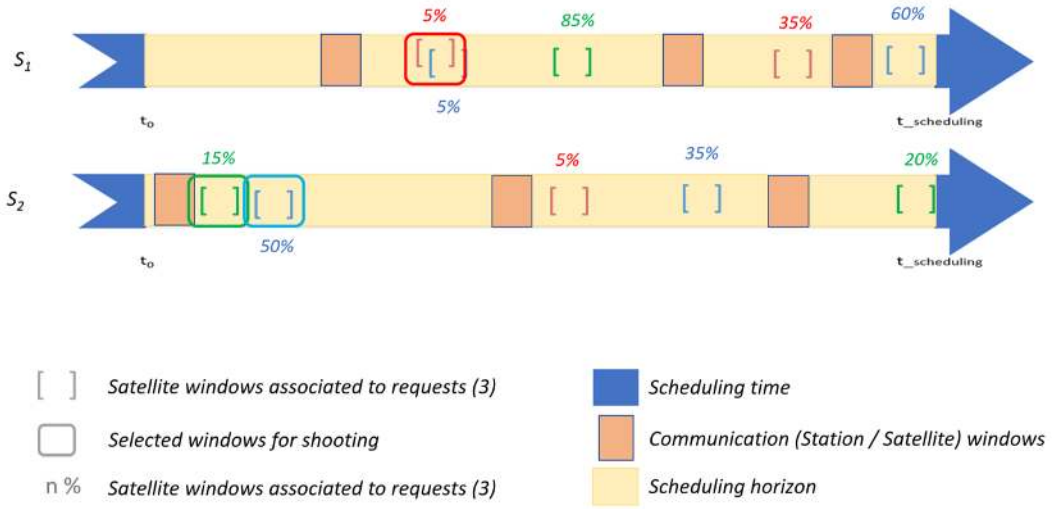

Figure 11: Example of the system timeline with associated cloud cover

We can see in this example that over the 3 images taken, only one will be validated when the images are brought down to Earth. If the cloud cover had been taken into account earlier, it would have been possible to plan 2 observations that would have been validated. Indeed, it would have been possible to plan the red request with satellite s_2 and the blue one with satellite s_1 using the first observation window.

It is therefore essential to consider the weather forecast before making a schedule. However, in order to do so, a weather model needs to be defined.

4.4 Definition of a weather model

As discussed before in Section 3.4, we do not have any formal nor precise models of the weather uncertainties. It is thus impossible to use probabilities to model the weather. Moreover, the approach we want to implement tries to stay as close as possible to the data whether it is real weather data or real forecast data.

Then, two different models of weather forecast are used.

The first one allows us to have short-term weather forecasts that would be very accurate in terms of time and space and is called M_0. For example, in our experiments, this model uses ground images that are taken by a geostationary satellite every Δ_0^{TS} minutes, where TS is the time slot. However, in a generic case, another way of defining the M_0 model could be used, with another Δ_0^{TS}. The weather information obtained using these images is assumed to be valid for a certain period of time defined by $\Delta_0^{validity}$.

The second model is called M_1 and provides a longer-term weather forecast. In this model, the forecast is completely recalculated every Δ_1^{run}. For one specific forecast, the time step is also fixed and is represented by Δ_1^{TS}.

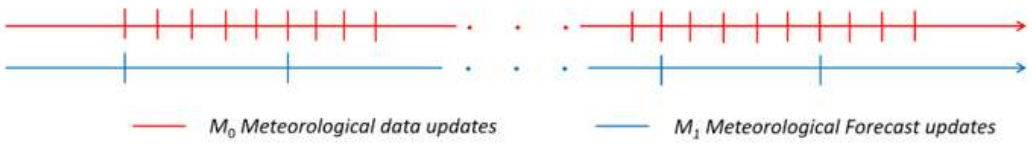

Figure 12: Presentation of the two models M_0 and M_1

As presented in Figure 12, model M_0 provides meteorological information more often than model M_1 and it is, by definition in our weather model, set to be more accurate. It is relevant to use M_0 to guide the planning process. Indeed, as this model is really accurate, when the TCC is obtained with it, it is possible to assume that the forecast is really close to the real weather. Therefore, when the TCC is obtained at the given time with this model, it will be important to plan the observation windows where the TCC is lower that the threshold set by the user; and on the contrary, when the TCC is above the threshold, the observation should not be made.

However, there are few observation windows where the TCC is given with the M_0 model and therefore the M_1 model is needed. The M_0 model will be used in all cases for the early validation and invalidation of the requests in order to be able to reschedule them as soon as possible. This can be seen in Figure 13. Indeed, in this figure, the validity time of M_0 is added. We can see that the scheduling can be done using M_0 only for three observation opportunities: the last blue window for satellite s_1 and the first green and red windows for satellite s_2. We can therefore say, regarding the threshold of 10 % set by the users, that, as M_0 is close to the real data, if the observations are planned in these blue or green windows, it is almost certain that the images taken will not be validated, while if an observation is made in the red window, the image taken is most likely to be validated.

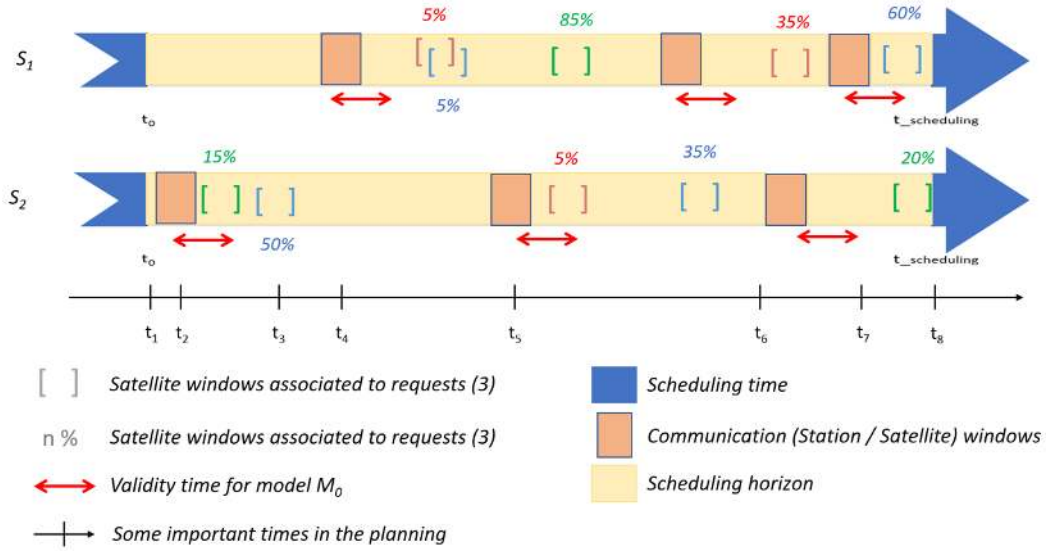

Figure 13: Presentation of the two models M_0 and M_1

4.5 Online planning architecture

The planning is done every Δ_0^{TS} minutes. There is, therefore, a recalculation from scratch of the plans to be sent to the satellites, and the time between two calculations is set to be equal to the one between two M_0 forecasts. The part defined between two communication windows is called a plan fragment. When a satellite passes over a ground station, the plan associated with the next plan fragment is sent to the satellite and cannot be modified later on.

In Figure 13, an example of an online planning architecture is given and some important times in the schedule are represented. In what follows, the notation $w^i_{r,s}$ is used to describe an observation opportunity where r is the request, which can either be *green*, *blue*, or *red*, s is the satellite, which can either be s_1 or s_2 and i corresponds to the ranking of the opportunity for a given satellite and request. For instance, for satellite s_1, the first red window is described by w^1_{red,s_1}, and for satellite s_2 the second blue observation opportunity is described by w^2_{blue,s_2}. In Figure 13, the initial time is named t_0. It represents the starting time of the scheduling. At this time, the communication windows are placed, the requests are posted and the associated observation windows are calculated. We have access to the forecast M_0 and M_1 and the updates over time. First, at the moment t_1, a first schedule is made and the only model available for all the opportunities is M_1. With this model, it is decided to schedule the following observation opportunities: w^1_{green,s_2}, w^1_{blue,s_2}, and w^1_{red,s_2}. We can see that if the first two opportunities are used, the images taken will not be validated whereas they are supposed to be according to M_1. Then, at time t_2, a new planning is made and at this time, M_0 is used to forecast the cloud cover for the opportunity w^1_{green,s_2}. According to M_0, this opportunity cannot be planned since the associated cloud cover is higher than the threshold. A new plan is calculated: w^2_{green,s_2}, w^1_{blue,s_2} and w^1_{red,s_2}. Afterward, this plan is sent to satellite s_2 and at this moment, nothing that has been scheduled in the following plan fragment can be modified. The image taken with the opportunity w^1_{blue,s_2} will not be validated. The following instant t_3 describes the moment when this image is taken. At this moment, we have access to M_0 forecast and the blue request can be reactivated and scheduled once again. A new plan is then calculated: w^2_{green,s_2}, w^1_{blue,s_1} and w^1_{red,s_2}. The non-validated image will still be made by the satellite s_1 but by doing this early reactivation with M_0, we will manage to find a new moment when the blue request can be fulfilled. At t_4 and t_5, the plan is still the same as before and the ground stations send it respectively to satellite s_1 and satellite s_2. The validation of the two images taken is done in instant t_6 and t_7 and in t_8 the last green request is performed with the observation opportunity w^2_{green,s_2}. Finally, four images will be made and only two will be validated. Finally, at the time $t_{scheduling}$ when the planning is complete, two requests out of three have been fulfilled.

The description of the online planning is presented in what follows:

t_0: *Starting time of the scheduling*

t_1: *Plan calculated with* M_1 *model:* w^1_{green,s_2} , w^1_{blue,s_2} , w^1_{red,s_2}

t_2: *Plan recalculated with* M_0 *model for opportunity* w^1_{green,s_2}: w^2_{green,s_2} , w^1_{blue,s_2} , w^1_{red,s_2}

t_3: *Plan recalculated using the early reactivation with* M_0: w^2_{green,s_2} , w^1_{blue,s_1} , w^1_{red,s_2}

t_4: *Sending the previous plan to satellite* s_1*, the following plan fragment cannot be modified*

t_5: *Sending the previous plan to satellite* s_2*, the following plan fragment cannot be modified*

t_6: *Validation of the blue request*

t_7: *Validation of the red request*

t_8: *The green request is failed*

$t_{scheduling}$: *Ending time of the scheduling*

5 A first scheduling approach

In what has been presented so far, the plans made were not always the best and the objective is then to find an approach that will allow us to have the best possible schedule in terms of number of validated requests, number of non-validated images and mean time when the data is available to the users. The first line of work is to properly define a model of the problem and to develop the associated optimization methods.

5.1 Single-shot optimization

A first way of optimizing the problem is to define a single-shot optimization using a Mixed Integer Linear Programming (MILP) approach.

5.1.1 Definition of a MILP model

The aim is to model the Earth Observation Satellite Constellation Scheduling problem taking into account weather forecasts. MILP models are widely used as they can allow to have a description of scheduling problems and they would help to find a proper solving method.

Space state

In the defined problem, it is assumed that we have at our disposal a set of satellites S with the associated maneuvering speed defined by $maneuvering\_speed_s \forall s \in S$. We also have a set of requests R asked by the users that need to be done with the satellites in S. For each request $r \in R$, we define W_r as the set of observation windows where it is possible to make the observation request. We define $W_s,\ s \in S$ as the set of observation windows associated with satellites. A set of windows can therefore be associated with a satellite and a request and is defined by $W_{r,s},\ \ \forall\ r \in R\ \forall\ s \in S$. We define as $RT_{r,w},\ \forall r\ \in R\ \forall w\ \in W_r$ the processing time of the request r in the window w.

We also have, for each satellite, a set of communication windows where the satellite can get new plans updated from the weather forecast. The requests are assumed to be simple (neither periodic nor stereoscopic). A request is mainly defined by a Point Of Interest (POI) on Earth where the target is located, a release date (to start observing the POI), a due date, and a cloud cover threshold asked by the user above which the image taken is considered as non-valid. The observation windows associated with the request are calculated so that they do not start before the release date nor end after the due date. For each observation window, the Total Cloud Cover (TCC) is obtained with the M_0 or M_1 model. If the M_0 cloud forecasts are available, then the TCC of the observation window will be that of M_0, otherwise, the TCC will be obtained with M_1.

Moreover, the simulation time is fixed and no window, whether they are for observation or communication, is defined after this time.

There are three sets of variables for the problem: $x_{r,w} \in \{0,1\},\ t_{r,w} \in [w_{r,w}^{start}, w_{r,w}^{end}]\ \forall r \in R, w \in W_r$ and $\beta_{w,w'} \in \{0,1\} \forall (r,r') \in R^2, (w,w') \in W_r \times W_r',\ r \neq r'$ defined as follow:

- $x_{r,w} = \begin{cases} 1 & \text{if observation request } r \text{ is made in window } w \\ 0 & \text{otherwise} \end{cases}$

- $t_{r,w} \in [w_{r,w}^{start}, w_{r,w}^{end}]\ \forall r \in R, w \in W_r$ the corresponding start time when request r is made in window w.
- $\beta_{w,w'} = \begin{cases} 1 & \text{if request } r \text{ is made in window } w \text{ before request } r' \text{ in window } w' \\ 0 & \text{otherwise} \end{cases}$

The β variables are used to sequence the different requests over each satellite. It has been presented in [41] that models using this kind of variables are much faster than the ones based on flow models and they give better results. However, constraints associated with those variables need to be defined.

Constraints

Constraints must be defined to ensure the feasibility of the defined problem and also to make sure that planning is feasible and consistent for the satellites.

Indeed, feasibility constraints should be formulated to ensure that if a request r is made in window w before request r' is made in window w' according to $\beta_{w,w'}$, then request r' must start after r. Moreover, for a given satellite, if r is made before r', then we need to ensure that the satellite as the time to make the first request and to change position to point to the new POI before making the second observation.

These constraints can be formulated as follows:

$$\begin{aligned}
&\beta_{w,w'} + \beta_{w',w} \geq x_{r,w} + x_{r',w'} - 1 && \forall s \in S,\ \forall (r,r') \in R^2,\ \forall (w,w') \in W_{r,s} \times W_{r',s}\ r \neq r'\ w \neq w' \\
&\beta_{w,w'} + \beta_{w',w} \leq x_{r,w} && \forall s \in S,\ \forall (r,r') \in R^2,\ \forall (w,w') \in W_{r,s} \times W_{r',s}\ r \neq r'\ w \neq w' \\
&\beta_{w,w'} + \beta_{w',w} \leq x_{r',w'} && \forall s \in S,\ \forall (r,r') \in R^2,\ \forall (w,w') \in W_{r,s} \times W_{r',s}\ r \neq r'\ w \neq w'
\end{aligned}$$

$$\begin{aligned}
t_{r,w} + RT_{r,w} + transitionTime(POI_r, POI_{r'}, maneuvering_speed_s) \leq t_{r',w'} + M(1 - \beta_{w,w'}) \\
\forall s \in S, (r,r') \in R^2,\ \forall (w,w') \in W_{r,s} \times W_{r',s}\ r \neq r'\ w \neq w'
\end{aligned}$$

where the function named *transitionTime* gives the rotation time between the two POIs and $RT_{r,w}$ is a parameter that represents the time needed to perform request r in window w. In this equation, M is defined to be a big M. It is a large coefficient that is chosen to be larger than any reasonable value that the expression on the left may take. M is set as as the total planning time.

The last constraint is that for each schedule, an observation request can only be made once. It can be defined as follows:

$$\sum_{w \in W_r} x_{r,w} \leq 1\ \forall r \in R$$

Objective function

The objective function can be described using rewards and/or regret aspects (of choosing the window w to make the observation request r for all windows and requests for instance).

For the reward, multiple criteria can be used to score an observation opportunity w. Indeed, the quality can be defined using the Total Cloud Cover (TCC) associated with the opportunity, depending on the model used (M_0 or M_1), but it can also be defined using a temporal quality. As a matter of fact, users appreciate to have their images as soon as possible. A quality measuring the cancellability of the opportunity is also defined. It can be used to deprogram an observation thanks to a later weather forecast. Finally, using a rescheduling quality is also important as it allows us to characterize the ability to reprogram an acquisition. There are therefore four different criteria that can be used to define the objective function of our MILP problem.

The final reward can then be a weighted sum of the four criteria, but this would imply to use a learning scheme to obtain the final function. Another way to define the reward is to use a lexicographic approach where priority is given to the quality of the observation. The last way would be to define a multi-objective reward function with the four criteria.

5.1.2 Possible optimization approaches and limitations

Multiple optimization approaches can be used to solve the problem. The final objective function could be a weighted sum of the four criteria, but this would imply to have an idea of which weight to put on each criterion. Another way to define the objective function is to use a lexicographic approach where priority is given to one criterion at a time. In these two ways of describing the objective function, a MILP solver such as CPLEX can be used. The last way would be to define a multi-objective reward function with the three criteria, which will lead us to find Pareto optimal solutions. This could be solved using Genetic Algorithms.

Multiple ways of defining the objective function can then be used. In the literature, different approaches of defining this kind of objective function, with multiple criteria, have been studied. According to [42], approaches using a weighted sum are presented as a simple method that highly depends on the weights put. It, therefore, needs prior knowledge. It would then be interesting to have a set of optimal trade-off designs between the different criteria. In [42], the aim is to propose distance-based data mining techniques to find the best decision rules while in [43], an incremental decision tree induction is used to build a decision model by analyzing data in short segments (local). Indeed, according to [15], decision tree techniques are efficient methods to build classification models to find an optimal order for the criteria. In this article, a multi-objective genetic programming approach is proposed to find Pareto optimal decision trees. The decision-maker can specify his preferences on the conflicting objectives. In [15], evolutionary heuristics have been applied to the problem since they are suitable for simultaneously evolving toward multiple alternative Pareto optimal solutions for multi-objective optimization problems and for generating accurate decision trees within reasonable periods. It might therefore be interesting the use decision trees and/or Genetic Algorithms to learn our objective function.

However, there are several limitations to this first MILP approach. Indeed, it can be complicated, in some cases, to mathematically formulate the objective function and no matter which form is used for it, it can be hard to find the optimal solution to the problem. Moreover, the four criteria defined in the objective function are used to know which observation opportunity is better than the other, but there are other global criteria that we have access to when the scheduling is over. These global criteria are not presented in the MILP since they are

calculated in the end of the scheduling process, but they are the ones that need to be optimized. These criteria are therefore presented in the following section.

5.2 Sequential optimization

The different four criteria defined in the objective function of the MILP model are local criteria. They are used in the scheduling process to determine which observation opportunity to schedule in priority. In these four criteria, the cloud cover is taken into account and can change over time during the scheduling process.

There are other criteria that we need to work on that are global. These criteria are defined according to the requirements of the users and are accessible at the end of the planning. There are three different global criteria: the number of validated requests, the mean data availability time that represents the mean time needed to obtain the images after the validation, and the number of non-validated images.

A difference is made between images and requests. Indeed, in the process, for one given request, multiple images can be taken. An image can be non-validated on the ground and then, the request will be sent once again to the scheduling engine to be planned again, and this process can be done multiple times. This is the reason why there are more images failed than requests failed. Moreover, an early replanning is done using model M_0. Indeed, at the moment the image is taken, model M_0 is used to see whether the image will be validated. This means that an image made can be invalidated with M_0 by the early, and the associated request will therefore be sent back to the scheduling engine to be made. Finally, during the final ground validation, the image non-validated by the early validation with M_0 can be validated. We will therefore have a gap between the number of images validated and the number of requests validated.

Finally, there are three global criteria :

- The number of requests fulfilled
- The number of non-validated images
- The mean time when the data is available

These global criteria are optimized using four local criteria :

- Total Cloud Cover using M_0 or M_1 (TCC)
- The Time Taken when the image is Taken (TT)
- The Time to Station between the end of the communication window and the moment when the image is taken (TS)
- The ability to Reprogram a request (R)

The objective is then to maximize the number of validated requests and minimize the mean time when the data is available and the number of non-validated images using the four local criteria.

5.3 Planning guidance

We have previously defined four local criteria and three global criteria that need to be optimized. The aim now is to know how to guide the planning with the four local criteria, in order to optimize the global criteria. Moreover, the choice of the local criteria is very important as they will be useful for the optimization process. In the literature, reinforcement learning models are sometimes used to have an idea of the decision problem on each cost in order to find a good decision rule on each cost to find the global optimum. An application is the slot machines problem. In this kind of problem, there are n slot machines whose results and probability law are not known in advance by the user. He can either use his knowledge and only use the machine that gave the best results so far, or he can choose to explore new machines to obtain better results.

This approach could be used to solve our problem. Using a decision-tree approach, as presented earlier, could also be interesting and could give good results. In any case, a new approach taking into account both local and global criteria needs to be found using the same space state and constraints defined in this section.

6 Development of a new scheduling approach

A new approach has been investigated to try to better take into account the weather uncertainties and to have a fast and sufficiently optimal planning considering the three global criteria previously defined.

6.1 Use of a greedy algorithm

Greedy algorithms are often used in the space domain to solve scheduling problems as they allow us to have an acceptable solution in a short time.

To be able to use this kind of algorithm, the observation opportunities associated with the requests of the ground users have to be sorted. The algorithm will plan the first opportunity, and if the second one is associated with a request that has not been planned yet, the second opportunity will be planned as well and the algorithm continues like this until the plans of the satellites are filled and it is therefore not possible to plan any opportunities anymore. This algorithm does not try to reschedule nor to use destruction and recovery operations to find the best scheduling, it only plans observations one by one.

An important work needs to be done considering the sorting of the observation opportunities as it is the input of the greedy planer. A pseudo-code of the implemented algorithm is given in what follows.

Algorithm 1 Greedy algorithm

Input *opps: activated observation opportunities sorted in the right way*
Output *plans: plans sent to the satellites with the associated scheduled observations*

```
1:  plans ← []
2:  if opps is not empty then
3:      firstOpp ← opps[0]
4:      while Opportunities can be added to the plans AND opps is not empty do
5:          if firstOpp can be planned then
6:              plans.add(firstOpp)
7:              opps.removeAllOpps(firstOpp)
8:               firstOpp ← opps[0]
9:          else
10:             opps.remove(firstOpp)
11:             firstOpp ← opps[0]
12:         end if
13:     end while
14: else
15:     Process is over
16: end if
```

where *opps.removeAllOpps*(*opp*) is a procedure that takes the request associated with the opportunity *opp*, put in argument, and removes all observation opportunities associated with this request from the list of opportunities named opps.

6.2 Scoring of observation opportunities

Multiple criteria can be used to score an observation opportunity w. The same four local critera as introduced before in the MILP model are used. The first quality of a given observation opportunity is therefore defined using the weather with the TCC associated. More precisely, this quality will be defined regarding the difference between TCC forecasted and the threshold: $TCC_{forecasted} - TCC_{asked}$. Model M_0 will be used for $TCC_{forecasted}$ when it is possible, so when the window is close to the ground station. Indeed, the M_0 forecasts do not stay valid for long and when a window is located too far from the station, the M_0 forecast is not valid anymore. The second quality defined is the one using the time when the observation is made. This allows us to send the requested images to the users as soon as possible. The third quality is the one measuring the cancellability. It is used to deprogram an observation thanks to a later weather forecast. The last quality is the rescheduling one. It depends on the ability to reprogram an acquisition and therefore on the number of observation opportunities left with an adequate TCC associated.
There are four different criteria that can be used to score an observation opportunity: the cloud cover named TCC, the time when the image is taken named TT (Time Taken), the time after the station name TS (Time to Station) and the Reprogramming R.

There are again different ways of calculated the final score. We can either use a weighted sum of the four criteria or a lexicographic approach. The lexicographic approach is the one that will be used and a ranking of the four criteria is made.

We are now able to score the different opportunities based on the four defined criteria. These criteria are ordered in a certain way, and then all opportunities must be ranked according to their score first according to the criterion number one, then the number two, then the number three and finally the number four. In what follows, some ways of ranking opportunities are investigated.

6.3 Classification of observation opportunities

In this part, the classification is made from the score obtained in the previous section.

6.3.1 Combination of the criteria

The classification is made using a lexicographic approach in order to learn which of the four criteria impacts the most the final solution. The aim is to find the best combination of the four local criteria that would allow us to find the optimal solution considering the three global criteria previously introduced.
A combination can be defined using one only criterion, but it can also use two, three or four criteria set in a lexicographical way. All combinations require to be tested. For each combination, the quality of the final solution is obtained by running the simulation.

6.3.2 Application to a toy example

A toy example has been implemented and is represented in Figures 14 and 15. The table defined in Annex 1 represents the entire problem.

In this example, the planning is made for two days, and the two days are independently planned. For each observation window, the TCC is represented in percentage of cloud cover and the max values represent, for each request (blue, red, and green), the threshold asked by the users. For each day, the observation opportunities and

the ground stations are calculated using a simulator and the weather data, whether it is real data or forecast used with M_0 and M_1 models, are randomly generated.

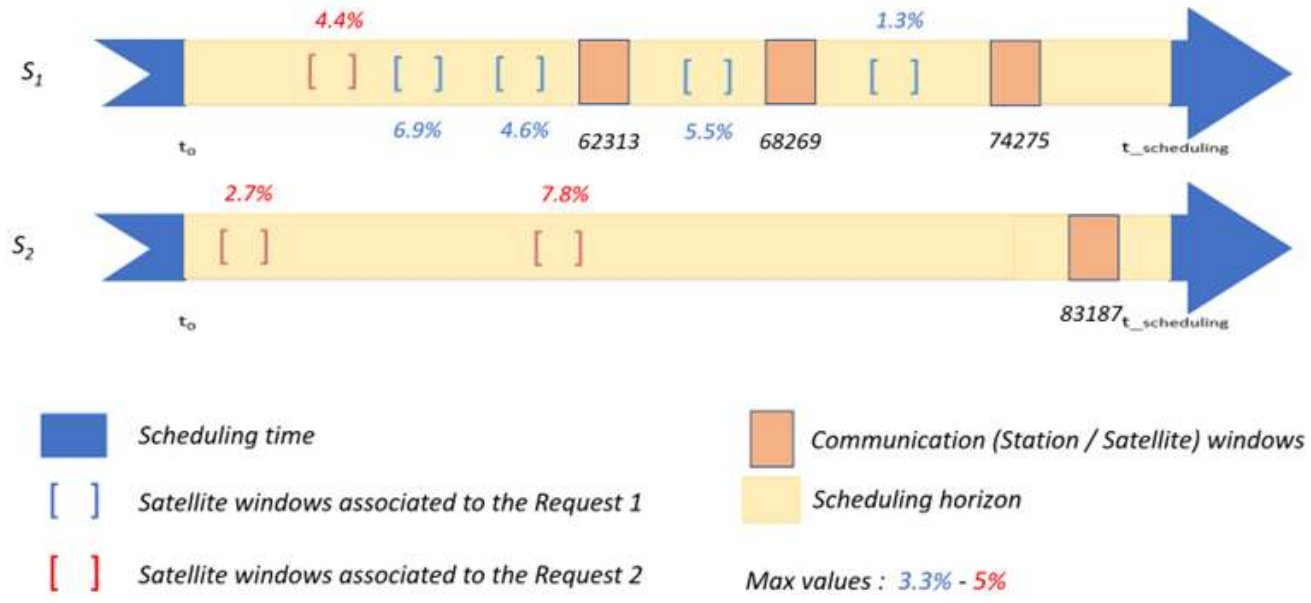

Figure 14: Presentation of the toy example for Day 1

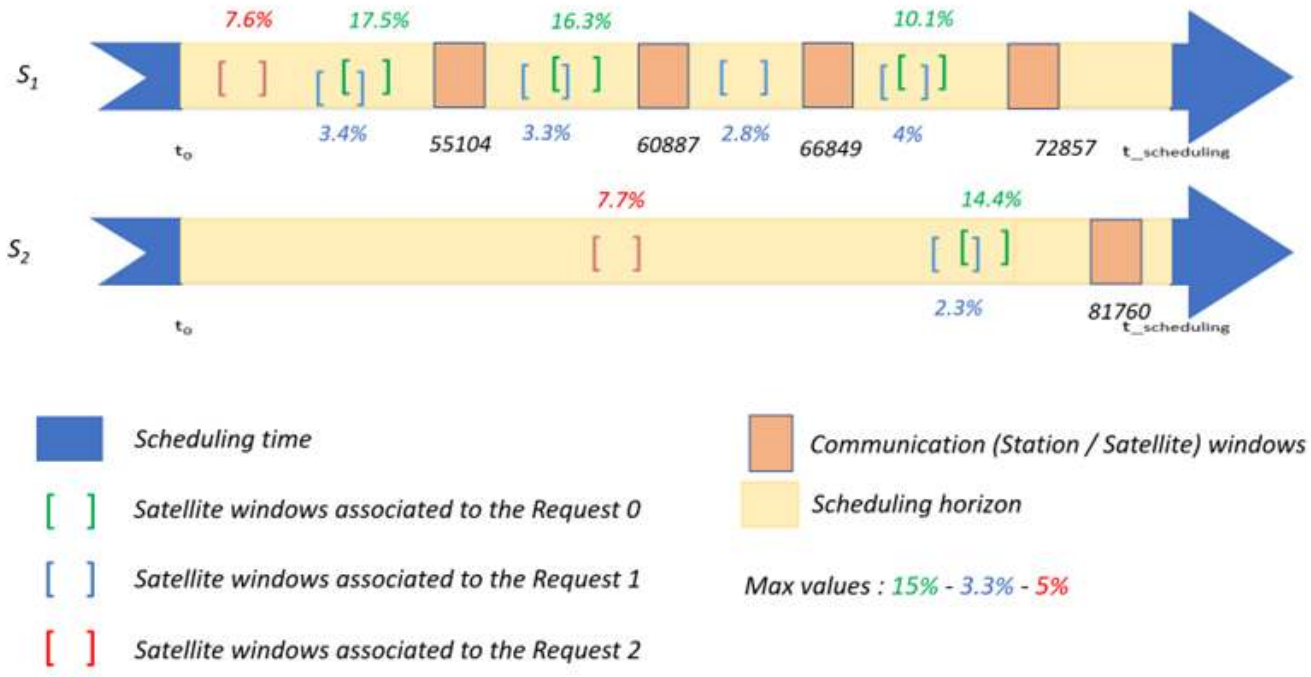

Figure 15: Presentation of the toy example for Day 2

The results are obtained for multiple combinations of the four local criteria and the most interesting ones are grouped in Table 1.

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
Max requests fulfilled TCC / TT / TS / R	5	6	**4**	4	1	2	18.77
Min images failed TCC / TT / TS / R	5	6	4	4	1	**2**	18.77
Min availability times TS / TCC / TT / R	5	6	2	2	3	4	**9.29**

Table 1: Results with different combinations of the greedy algorithm for the toy example

Each of the three combinations represented in the figure optimizes one of the three global criteria previously defined.

6.3.3 Limitations

It can be seen that some combinations work better to optimize the global criteria. Indeed, the combination TCC / TT / TS / R in this example allows to have a better maximization of the number of requests fulfilled and to minimize the number of images failed. However, the results using this combination in terms of mean availability time are very far from the one using the combination TS / TCC / TT / R that gives a better minimization.

It is interesting to have a view of what can be done by combining the four local criteria. However, even if we consider the three optimal combinations, nothing tells us that the results obtained are optimal. Another approach is therefore developed to try to find an optimal way of ranking the different observation opportunities.

6.4 Decision tree generation

Decision trees are well known in the literature and could be very useful in our situation. Indeed, they would allow us to learn a decision rule that could give us the optimal classification of observation opportunities, in terms of number of requests fulfilled, number of images non-validated and mean data availability times, considering the four local criteria.

6.4.1 Literature review

Different methods exist when it comes to constructing an optimal decision tree. Indeed, in the literature, multiple algorithms have been implemented to build them such as the ID3, the C4.5 or the CART algorithms. According to [44], the algorithms developed for learning decision trees are most often variations on a core algorithm that employs a top-down, greedy search through the space of possible decision trees. This approach was first implemented in the ID3 algorithm (Quinlan 1986) and then slightly modified in the C4.5 algorithm (Quinlan 1993). The CART was implemented afterwards and aims at finding the optimal classification and regression tree. In the ID3 algorithm, the decision trees are built from scratch and at each step, the instance attributes are evaluated using a statistical test to determine the best attribute among the candidates that better classifies alone the training example while growing the tree. In this algorithm, the Information Gain (IG) is used as the statistical test. The IG is calculated from the entropy. The entropy is a measure commonly used in information theory that aims to measure the homogeneity. It is used to characterize the purity or impurity of a given collection of training example and to quantify how much information there is in a random variable. The IG is calculated by comparing the entropy of the trained dataset before and after a transformation. The larger the IG is, the lower the entropy will be. The IG is used to measure the expected reduction in entropy after partitioning the example, and therefore, to measure the effectiveness of an attribute in classifying the training data. The aim of using the IG measure is to be able to generate decision trees for which the entropy is the lowest possible.

The ID3 algorithm was extended to address new issues such as the determination of how deep a decision tree should be, the choice of an appropriate attribute selection measure or the way to handle the attributes with differing costs. This resulted in a new system named C4.5.

These algorithms developed are widely used when it comes to building decision trees. However, a criticism of the top-down approach was made in [45]. Indeed, this approach is said to be local in nature and that each choice of attribute may be optimized but there is no guarantee that the resulting tree is in any sense optimal. According to this article, truly optimal solutions to the problem are in general infeasible while near-optimal solutions can be found in polynomial time using stochastic methods. The use of a Simulated Annealing (SA) metaheuristic is therefore proposed. The model defined is based on Markov chains and a Simulated Annealing Classifier System (SACS) algorithm is implemented using the Minimum Description Length Principle of Quinlan and Rivest as cost function. The idea is to minimize the amount of information necessary to encode the classification of all the data points. In this principle, the minimum information gives the best model. Results shows that SACS is capable of producing really accurate decision trees.

6.4.2 Application to the problem

A first idea would be to try to find an entropy that could be associated with our scheduling problem in order to define the IG by which the best decision tree is generated using an algorithm such as C4.5. However, the problem defined in here is slightly different from those that can be found in the literature. Indeed, in our situation, the aim is not to find the best decision tree with the lowest entropy associated but it is to classify in the best possible way the different observation opportunities in order to optimize the results obtained after the simulation regarding the three global criteria previously defined. Therefore, another method that can be applied to our problem should be employed. An example of a decision tree with our four criteria is given in Figure 17.

6.4.3 Use of a Simulated Annealing (SA) metaheuristic

The decision tree is build from a certain fixed order of the four local criteria and each node step is associated with a local criterion. At the beginning, as described in Figure 16, four steps are defined and linked together. All the opportunities are scored identically and their score is equal to 1.

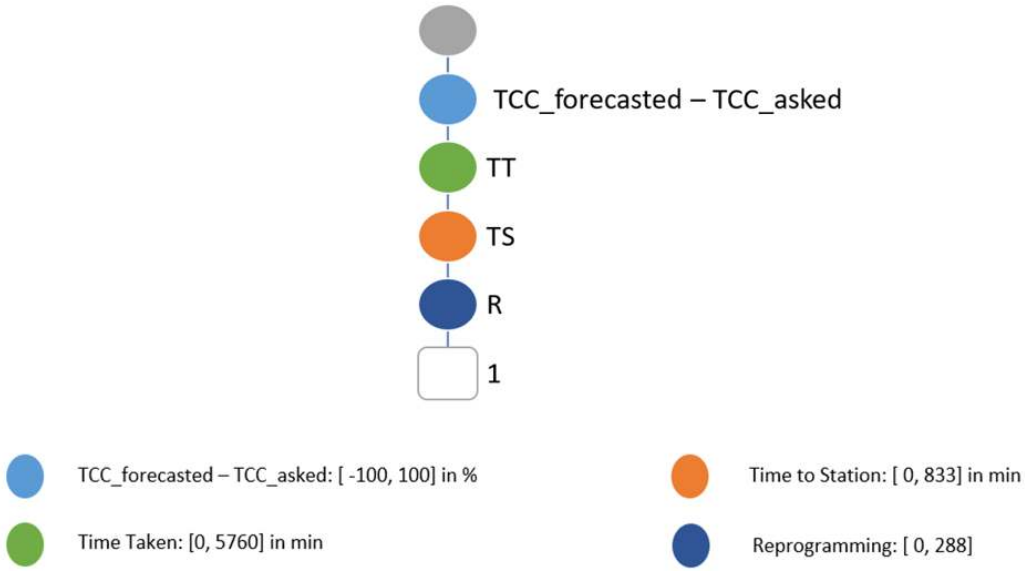

Figure 16: Representation of the initial aspect of the decision

Afterwards, split and merge operators are defined to update the tree and in the end, a decision tree is constructed. The opportunities are not scored the same way regarding the four criteria and the one with the lowest score will be ranked first. An example of a decision tree that we could have at the end of the generation is presented in Figure 17.

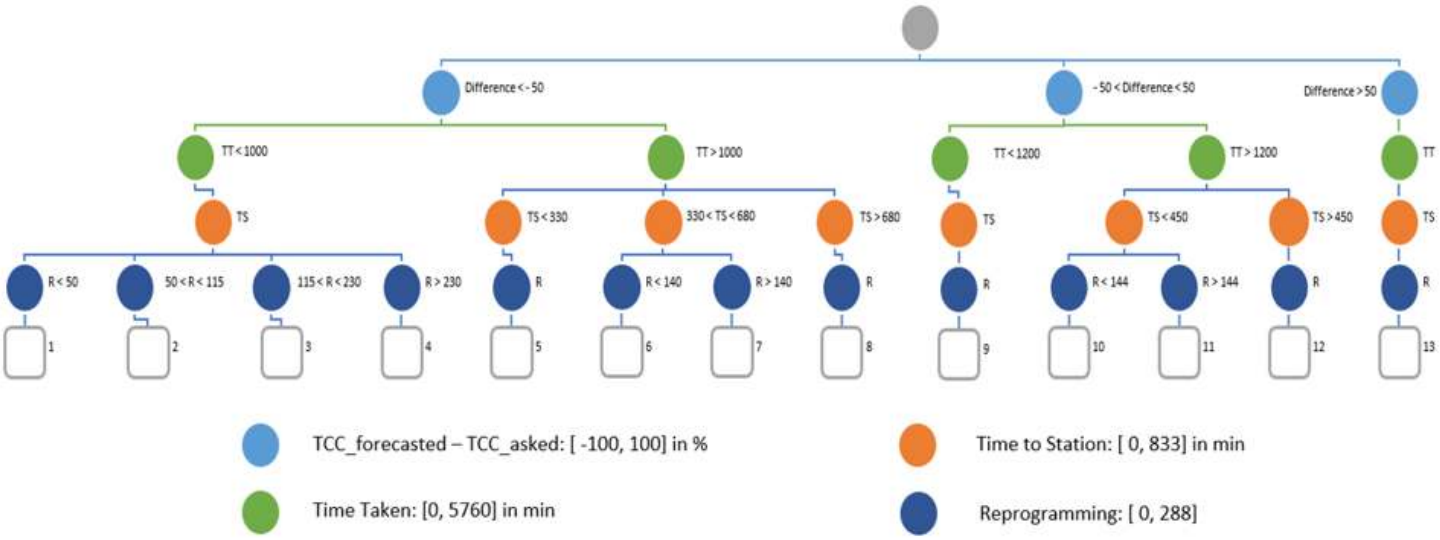

Figure 17: Example of a final decision tree

The split and merge operators used are local and can randomly affect any node in the tree. Indeed, the split operator affects one branch of a given node and it splits the branch in two by dichotomy. For each node, there is a fixed maximum number of splits that can be performed, depending on the assessed criterion. In addition, the splitting is guided in such a way that if there is no opportunity in a given branch, which means that if the branch does not allow to separate opportunities, this branch will not be split. This guidance is useful in order not to overload the memory for nothing.

Simulated Annealing algorithms are known to give very conclusive results from local operators. Such an algorithm has therefore been developed to find the best possible decision tree to optimize the three global criteria. In the SA process, the operator used each time is randomly chosen and the node and the branch on which the operator will be applied are randomly selected.

6.4.4 Application to the toy example

The toy example used here is the same as the one previously defined and represented in Annex 0. To generate the decision tree algorithm, the combination TCC / TT / TS / R is selected. The resolution time is of 16 seconds to obtain the decision tree.

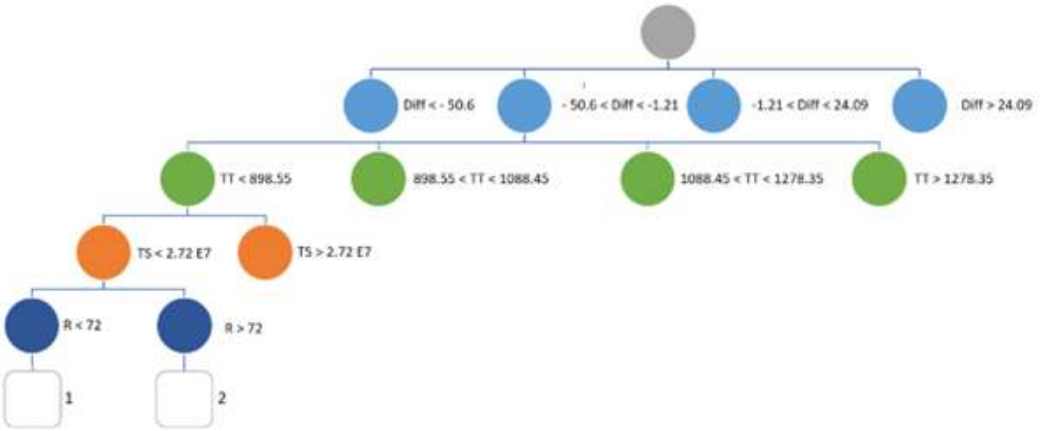

Figure 18: Representation of the decision tree obtained for the toy example

Figure 18 represents the tree that is generated for the toy example using the same notations previously introduced. In this example, there are no opportunities when the difference between the two TCCs is less than -50.1. This is why the notation starts afterward when the notation is between -50.6 and -1.21.

The following table, therefore, introduces the results obtained for this example while the tree is used:

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times
Decision tree	5	6	**4**	4	1	**2**	**17.77**

Table 2: Results with the decision tree for the toy example

In this small example, the optimal results were already obtained using the combination TCC / TT / TS / R, it is, therefore, consistent to retrieve the same results.

7 Simulation architecture

Two different methods, such as the one applying a greedy algorithm to the different combinations of the four criteria and the one generating a decision tree to classify the observation opportunities, have been introduced. The simulator used to evaluate them is presented in this section.

7.1 Presentation of the simulator

A simulator has been implemented to play different scenarios and to obtain the results regarding the three global criteria, the number of requests fulfilled, the number of images failed and the mean time when the data is available to the user, for the developed approaches. In this simulator, data, such as the one defining the targets, the weather information, the number of days over which the scheduling is made, or the satellite constellation, are loaded. From the given inputs, the simulator manages to find for each satellite all the observation opportunities possible to make all the given requests. At the end of the simulation, results are given in terms of the number of requests performed, the number of images performed, the number of requests fulfilled, the number of images fulfilled, the number of requests that either are not planned and/or are not validated, the number of images invalidated, and the mean data availability times. From this information, we can retrieve the three global criteria that we want to optimize the number of requests fulfilled, the number of images failed, and the mean time when the data will be available to the users.

In the simulator, as introduced before in Section 4.5, the time between two simulation steps is assumed to be equal to the one between two M_0 forecasts. At each simulation step, the plans to send to the satellites are recalculated from scratch with the updated weather information. It is therefore important to have an efficient allocation algorithm that returns a suitable and near-optimal plan in a relatively short time.

As for image validation, it is realized in two different time periods. Indeed, the same process is done as the one described in Section 4.5. When an image is taken, the early validation process is done, using the M_0 model to estimate whether the image will be validated. If the image, in the test, is considered to being valid, it will be sent once again to be planned. The second validation happens some time, about an hour, after the satellite has sent the image to a ground station. Therefore, in the calculation of the mean data availability time, it is necessary to take into account the time for the satellite to arrive at the next ground station and then a given fixed time before having the real validation on the ground.

A description of the processes that occur in the simulator is given in the following text and the different processes, such as the planning of a request or the sending of a plan to the satellites, happen over time and depend on the start times of the observation windows. Indeed, if a window is placed in the end, the image will be taken afterward and the two steps of validation will be done later.

7.2 Application to a toy example

The simulator is applied to the toy example introduced before in Figures 14 and 15. The description of the processes that occur in the simulator is given for the two days.

For the first day, the description is:

PLANNING OF THE REQUESTS
ACCEPT: Request 1 – Opportunity 1
ACCEPT: Request 2 – Opportunity 0

SENDING THE PLANS TO THE SATELLITES
COMMIT: Request 2 – Opportunity 0
COMMIT: Request 1 – Opportunity 1

EARLY VALIDATION USING THE M0 MODEL
ESTIMATE SUCCESS: Request 2 – Opportunity 0 -> Image validated
ESTIMATE SUCCESS: Request 1 – Opportunity 1 -> Image validated

VALIDATION OF THE IMAGES ON THE GROUND
ANALYSIS: Request 2 – Opportunity 0 -> request validated
ANALYSIS: Request 1 – Opportunity 1 -> image failed

SENDING THE REQUESTS FAILED TO THE SATELLITES
ACCEPT: Request 1 – Opportunity 3

EARLY VALIDATION USING THE M0 MODEL
ESTIMATE SUCCESS: Request 1 – Opportunity 3 -> image validated

VALIDATION OF THE IMAGES ON THE GROUND
ANALYSIS: Request 1 – Opportunity 3 -> image validated

For this day, the two defined requests are fulfilled, one image is failed. Three images and two requests are performed and the results are shown in Figure 19, providing a view of the two images taken where the requests are fulfilled.

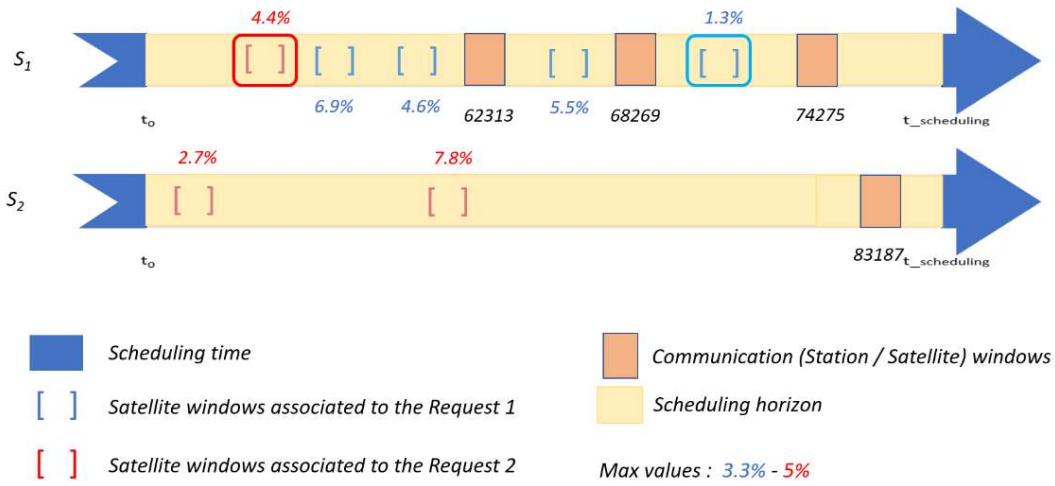

Figure 19: Representation of the simulation process for the toy example for Day 1

For the second day, the description is:

PLANNING OF THE REQUESTS
ACCEPT: Request 1 – Opportunity 1
ACCEPT: Request 2 – Opportunity 1
ACCEPT: Request 0 – Opportunity 2

SENDING THE PLANS TO THE SATELLITES
COMMIT: Request 2 – Opportunity 1
COMMIT: Request 1 – Opportunity 1

EARLY VALIDATION USING THE M0 MODEL
ESTIMATE SUCCESS: Request 1 – Opportunity 1 -> Image validated
ESTIMATE SUCCESS: Request 2 – Opportunity 1 -> Image failed

VALIDATION OF THE IMAGES ON THE GROUND
ANALYSIS: Request 1 – Opportunity 1 -> Request validated

SENDING THE PLANS TO THE SATELLITES
COMMIT: Request 0 – Opportunity 2

EARLY VALIDATION USING THE M0 MODEL
ESTIMATION SUCCESS: Request 0 – Opportunity 2 -> Image validated

VALIDATION OF THE IMAGES ON THE GROUND
ANALYSIS: Request 0 – Opportunity 2 -> Request validated
ANALYSIS: Request 2 – Opportunity 1 -> Image failed

For this day, two out of the three defined requests are fulfilled, one image is failed. Three images and three requests are performed and the results are shown in Figure 20, providing a view of the two images taken where the requests are fulfilled.

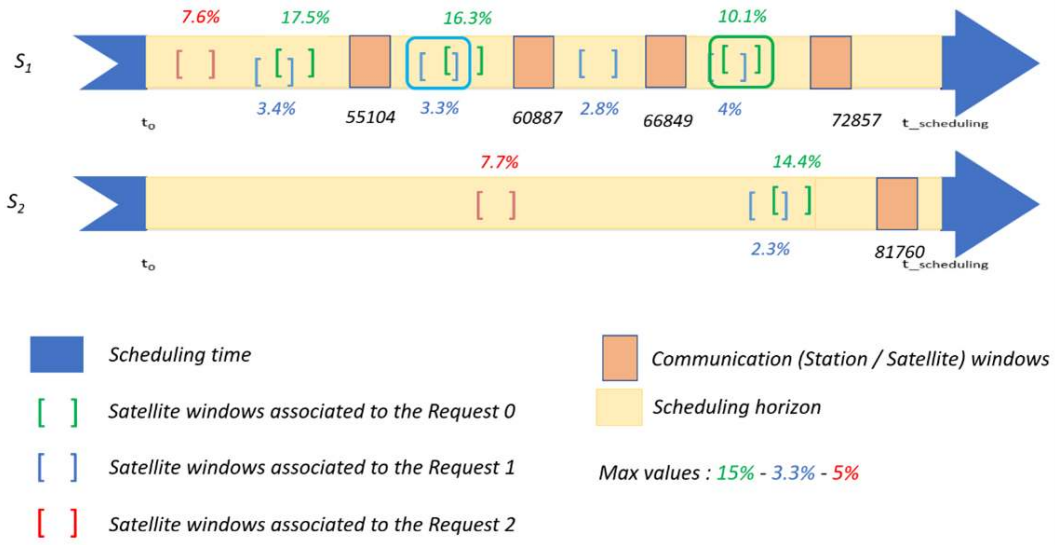

Figure 20: Representation of the simulation process for the toy example for Day 2

Therefore, in this example, for both days, we manage to have four requests fulfilled out of the five requests initially defined. Two images taken were failed and six images and five requests were performed. The mean time when the images are available is 18.77 hours.

8 Boundary calculation

In order to assess the results obtained, it is essential to compare them to bounds. Indeed, the different calculated bounds will allow us to have an idea of the optimal solution that can be found and therefore to assess the quality of the solutions found using the two developped methods previously presented.

Three different bounds are defined. The first one provides us with the maximum number of requests that can be made if the weather is not considered at all. In this case, it is assumed that the cloud cover is never significant and that the threshold is never exceeded, which is not realistic. The second bound allows us to know the maximum number of requests that could be done if the real weather was known in advance. In this case, it is assumed that the scheduler is prescient. For the two first described cases, the scheduling is made using an optimization solver named CP Optimizer and upper bounds are found. In the last calculated bound, it is assumed that the greedy scheduler is prescient.

8.1 Upper boundary calculation

In this section, we presents the method to calculate two different upper bounds using CP Optimizer solver.

8.1.1 CP Optimizer model

IBM ILOG CP Optimizer is a constraint-based scheduling tool. As the user manual describes, it is used to model and solve constraint satisfaction problems and optimization problems. The system first finds feasible solutions and then, uses powerful methods to improve them. The algorithm does not end until the solution found is proven to be optimal, unless other search limits are specified.

In our situation, the CP Optimizer solver is used to solve a scheduling problem. As inputs, we retrieve all the observation opportunities for all the requests of the ground users. All of these opportunities are associated in the model with intervals.

The model can therefore be encoded as follows.

Decision variables

The space decision variables correspond to interval variables that are associated with all the defined observation opportunities. In CP Optimizer, an interval variables are set to be optional, this means that they can either be selected or not by the scheduling process. Moreover, the start and the end of the different interval variables correspond to the start and the end to the opportunities associated. The duration of the interval variables corresponds to the time needed to complete the observation of the Point of Interest (POI) plus a transition time for the satellite to move from this POI to the next. This transition time is assumed to be constant.

Constraints

Two constraints are defined in the model. The first constraint aims at ensuring that a given request is not performed twice and the second one ensures that there is no overlap between the opportunities for a given

satellite. In CP Optimizer, these constraints are defined as follows:

forall(r in 0..nRequests) sum(opp in Opportunities: opp.request == r) presenceOf(itvs[opp]) <= 1
forall(s in 1..nSatellites) noOverlap(all(opp in Opportunities: opp.satellite == s) itvs[opp])

Objective function

There are two different approaches that can be implemented. A first approach consist in considering only the number of requests to do and the objective function associated consists of calculating the number of requests done. The aim is then to maximize this function. A second approach to investigate is the one that uses a lexicographic approach and two objective functions are defined. The first one is the same as before: the number of requests done and the second one is newly defined: the mean time when the data are available to users. The aim in this second approach is to first maximize the first objective function and then minimize the second.

In CP Optimizer, the first objective function is defined as follows:

maximize sum(opp in Opportunities) presenceOf (itvs [opp])

For the second lexicographic approach, the second objective function is:

minimize sum(opp in Opportunities) [presenceOf (itvs [opp]) × getAvailabilityTime (opp)]

where *getAvailabilityTime(opp)* is a function that return the mean time when the image will be available to the user if the observation is scheduled in the opportunity opp set as argument of the function.

For the first way of describing the objective function, the solver ends and manages to find an optimal solution. However, while using the second lexicographic way, a limit time is set and the solution found is not proven to be optimal.

The entire model is presented in Figure 21, the same way it is described in the mod file used by CP Optimizer.

8.1.2 Clear sky conditions

In this case, all the obtained observation windows are retrieved as input for the CP Optimizer model. The weather associated with the different windows is not considered insofar as all observation opportunities can be planned, regardless of the weather uncertainties.

8.1.3 Use of CP Optimizer with a known real weather

In this case, the same model is used in terms of constraints and objective functions, however, the space state is slightly different. Indeed, windows are still defined the same way but the inputs are different. A selection of the observation opportunities is made beforehand in order to keep only those that will be validated at the end of the simulation, thus those for which the real cloud cover is under the threshold defined by the user.

Calculating the upper bounds in these two cases is really important as it allows to see allows us to see whether it is the planning aspects that are constraining and that make us not plan all the requests, or the weather that has the most impact.

```
int nSatellites = ...;
int nRequests = ...;

tuple Opportunity {int request; int satellite; int opp; int wStart; int wEnd; int duration;   }

{Opportunity} Opportunities = ...;


// Variables

dvar interval itvs[opp in Opportunities] optional in opp.wStart..opp.wEnd size opp.duration;


// Objective function

maximize sum(opp in Opportunities) presenceOf(itvs[opp]);

minimize sum(opp in Opportunities) presenceOf ( itvs [opp] ) * getAvailabilityTime ( opp );


// Constraints

subject to {

        forall(r in 0..nRequests) sum(opp in Opportunities : opp.request == r) presenceOf(itvs[opp]) <= 1;

        forall(s in 1..nSatellites) noOverlap(all(opp in Opportunities : opp.satellite == s) itvs[opp]);

}
```

Figure 21: Model used in CP Optimizer

8.2 Greedy Algorithm with a known real weather

Another test can be found using the greedy algorithm by assuming that the real weather is known. The result is not optimal but it can allow us to see how far we are from the optimal solution. Two different greedy algorithms are implemented. All the opportunities are taken into consideration and this means that in the process, it will be possible to plan observation opportunities that will not be validated. In the second one, only the opportunities that will be validated are taken into consideration. A condition is therefore fixed to ensure that only the opportunities that have a real cloud cover lower than the threshold can be planned.

This prescient greedy algorithm does not provide an upper bound but it allows us to see how far the results obtained are from the upper bounds calculated with CP Optimizer. It therefore allows to evaluate the performance of the greedy strategy.

8.3 Application to a toy example

The three different upper bounds are calculed for the toy example.

The first results given here are the ones using the first way to describe the objective function. This is the case where the objective function is not lexicographical and only the number of requests done is maximized.

The following table (Table 3) gives the results obtained using CP Optimizer. The two cases are represented as *Without weather* when the weather is not considered, and as *With weather* when the weather is supposed to be known in the simulation.

	Total number of requests	Number of requests that can be done	Number of requests Done	Mean data times (in h)
Without weather	5	5	**5**	**21.46**
With weather	5	4	**4**	**22.29**

Table 3: Results with CP Optimizer using the toy example

We can see in this table that in the first case where the weather is not taken into account, all the requests are done. However, we already know, regarding the example and the thresholds set by the users, that one request can not be done because of the cloud cover. Therefore, in the second case, when the real cloud cover is assumed to be known, only four out of five requests are done. The mean time when the images can be sent back to the users are not the same.

The two following figures allow us to see which requests are made and the moment when there are made when the weather is not taken into account.

The following figure gives the results for Day 1:

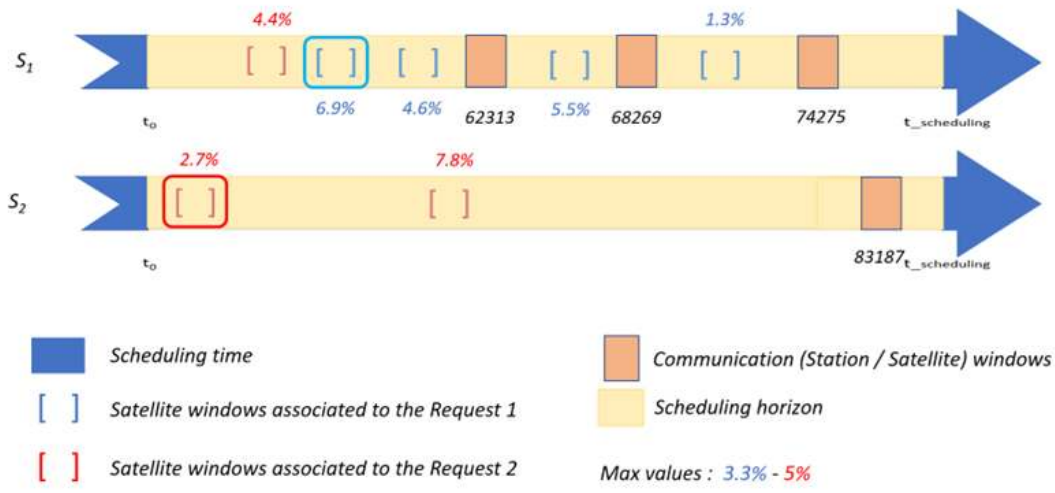

Figure 22: Representation of the results obtained using CP Optimizer for the toy example without weather considerations for Day 1

The following figure gives the results for Day 2:

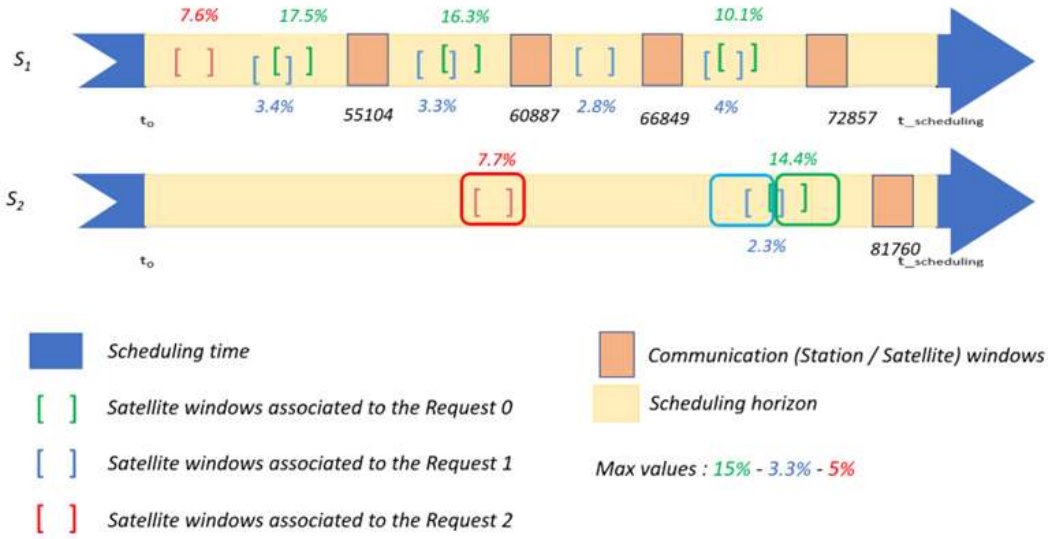

Figure 23: Representation of the results obtained using CP Optimizer for the toy example without weather considerations for Day 2

We can see that, in this case, the simulator only tries to schedule as many requests as possible and the moment when the requests are made is not considered at all. This way, for the second day, all requests are done using only the second satellite.

The two following figures allow us to see which requests are made and the moment they are performed when the weather is taken into account.

The following figure presents the results for Day 1:

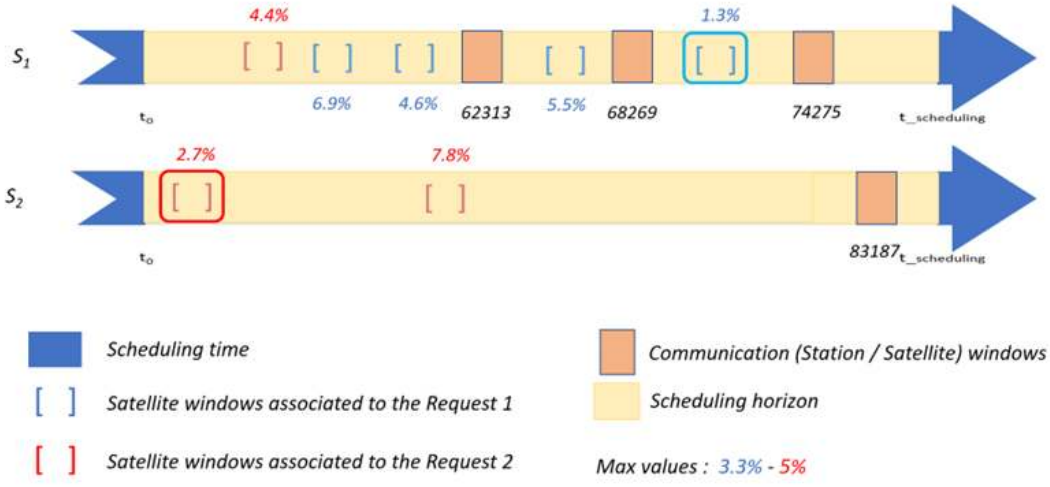

Figure 24: Representation of the results obtained using CP Optimizer for the toy example with weather considerations for Day 1

The following figure presents the results for Day 2:

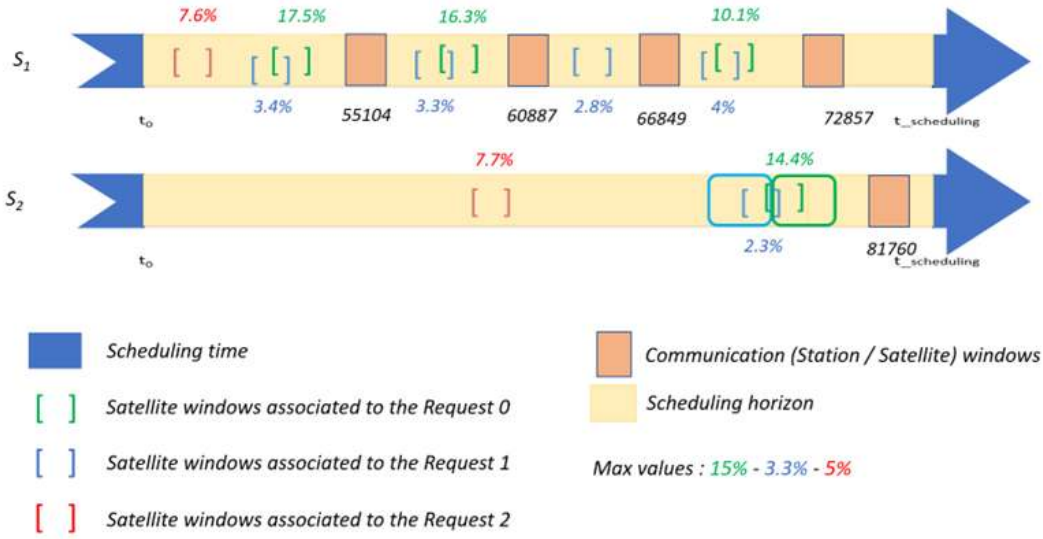

Figure 25: Representation of the results obtained using CP Optimizer for the toy example with weather considerations for Day 2

We can observe that in this case, when the cloud cover is considered, one request is not performed for Day 2. Indeed, for this request, there is no observation opportunity that is available with a proper associated cloud cover. The algorithm tries to make as many requests as possible, as long as the cloud cover associated to the observation opportunities remains below the threshold. The moment when the requests are done in this case is not important in the scheduling.

However, as we presented before, the ground users like to have their images available as soon as possible. A lexicographic objective function was, therefore, used to ensure that the images are available early.

	Total number of requests	Number of requests that can be done	Number of requests Done	Mean data times (in h)
Without weather	5	5	**5**	**16.31**
With weather	5	4	**4**	**18.77**

Table 4: Results with CP Optimizer using the toy example

In Table 4, we can see that, when a lexicographic approach is implemented, the mean time when the images are available for the users have been minimized. With this approach, users can have access to the requested images about five hours earlier, which is a great improvement.

The two following figures allow us to see which requests are made and the moment they are made in the first case, when the weather is not considered.

The following figure gives the results for Day 1:

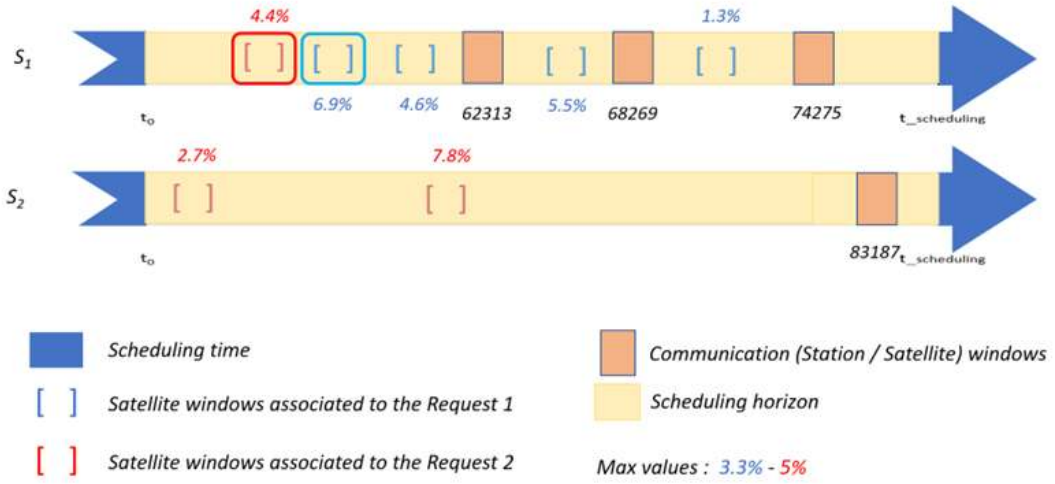

Figure 26: Representation of the results obtained using CP Optimizer for the toy example without weather considerations Day 1

The following figure gives the results for Day 2:

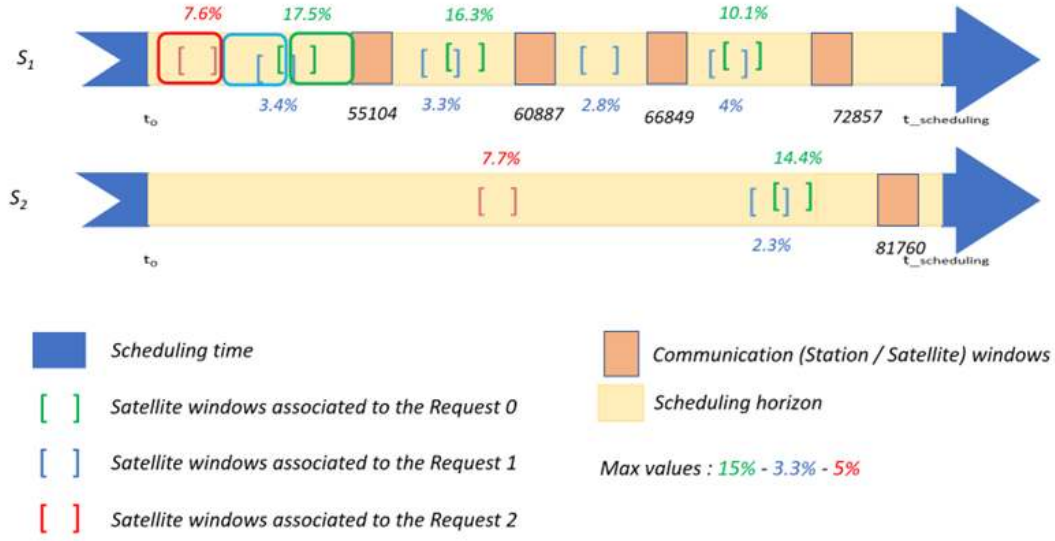

Figure 27: Representation of the results obtained using CP Optimizer for the toy example without weather considerations for Day 2

In this two observations scheduling, we can see that all requests are made and they are done as early as possible and each time close to the first communication window. Indeed, in the simulation process, the images taken are brought down to the ground as the satellites pass over the ground station and they are available to users some time later. Therefore, if we want to have the images as soon as possible, the images must be taken as soon as possible before the first communication windows.

In the previous scheduling, we have a view of which images are taken and when while the weather is not considered. In the two following figures, the weather is taken into account.

The following figure gives the results for Day 1:

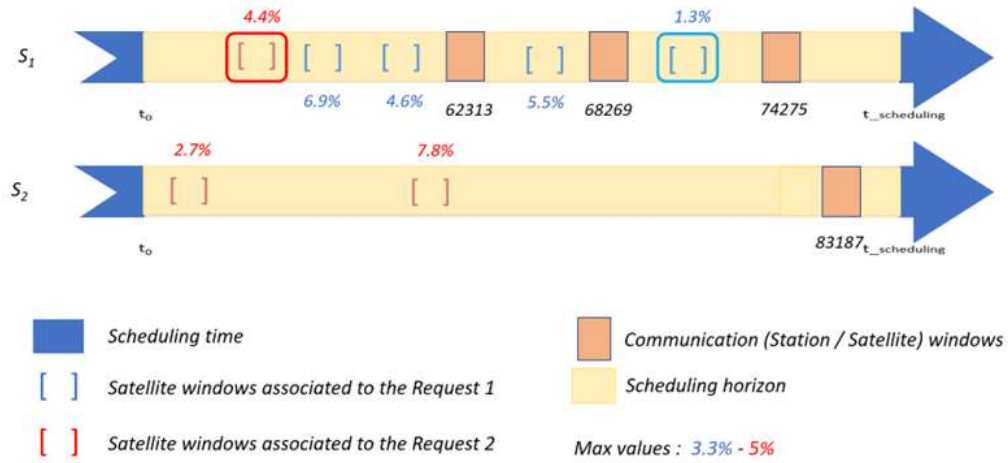

Figure 28: Representation of the results obtained using CP Optimizer for the toy example with weather considerations for Day 1

The following figure gives the results for Day 2:

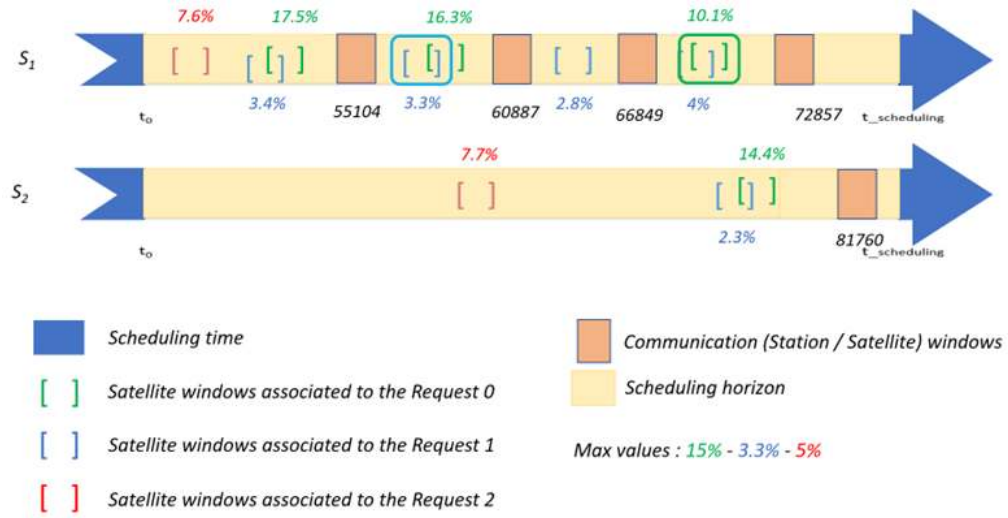

Figure 29: Representation of the results obtained using CP Optimizer for the toy example with weather considerations for Day 2

As seen previously, in the case where the approach is not lexicographic, no observations are planned for a request on the second day because of the weather. However, also in this case, the planned observation windows are close to the first communication windows when the weather allows it. This is done so that users can have their images as soon as possible.

All in all, we can conclude that when a lexicographic approach is used, the observations are planned earlier, regarding the presence of the communication windows.

9 Computational results

The following section introduces all the computational results obtained with the different developed methods.

9.1 Generation of 100 data sets

The targets are selected in an area where the weather can vary a lot. Therefore, 100 instances of the problem have been generated. For each generated instance, the position of the targets can change in the given area and the weather is different.

9.1.1 Definition of the area of interest

Figure 30 illustrates the location where the points of interest associated with the targets are located.

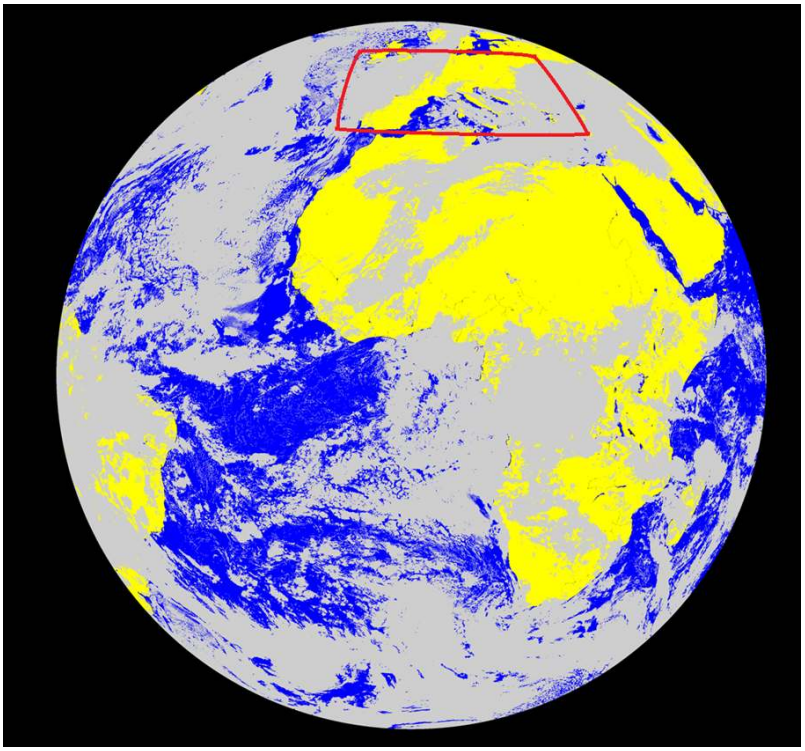

Figure 30: Area of interest where the target are selected

The selected area is around Europe. Indeed, it is a region where the climate is variable and therefore where the cloud cover can change a lot. There are indeed periods of clear and cloudy sky in this region, which can be very short or very long, and the climate is quite changeable between places. This will allow to test the adaptability of the planning algorithms developed to a weather that can change a lot.

9.1.2 Experimental protocol

For each generated instance, 100 targets, with 100 POI associated, are randomly selected from a set of 500 targets taken in the defined area (Figure 30). The planning is done over four days and each day is independent

of the other. The constellation is composed of 16 satellites with 8 inclined orbital planes with 2 satellites per plane. The inclination of the constellation is of 60 degrees. About 300 requests are defined for the four days where one request is associated with one target from the 100 selected for the generated instance. Each user can define its own TCC threshold and it ranges from 0 % to 20% for the maximum cloud cover. For the four days, the weather data, whether it is actual weather data or the forecast used in the M_0 and M_1 models, is either randomly generated or defined using real data. In both cases, the forecasts for model M_0 are made every 15 minutes and they are assumed to be valid for 30 minutes while for model M_1, the runs are made every 6 hours, and for one forecast, the time step is 3 hours. For each of the 500 targets, real weather data and real weather forecasts are retrieved from the month of March 2022. For each generated instance, weather data from four random days in March is therefore used for the 100 selected targets. The results obtained will be an average of the results for the 100 generated instances.

9.1.3 Results using random weather data generation

The first results are obtained when the weather forecasts in M_0 and M_1 models and the real weather associated with the targets are randomly generated. First of all, the real cloud cover is randomly generated from the TCC threshold asked by the users. Then, the forecasts for M_0 and M_1 are obtained from the real cloud cover by calculating, at a given time, the average between the real TCC at the previous time and the one at the next time. This is done on the given time horizon, which means that for model M_0, the two times introduced before are 15 minutes apart while for model M_1, they are 3 hours apart.

9.1.3.1 Boundary calculation

As described before, in Section 8, the calculation of an upper bound allows us to evaluate the accuracy and the relevance of the results obtained with the developed algorithms.

Table 5 illustrates the results obtained using CP Optimizer in the two cases defined in Section 8. In the results named *Without weather*, the weather is not considered at all. We can therefore see that over the 255 possible requests to make, at best, the solver manages to place 239.59 requests on average over the 100 generated instances. We can observe a difference between the total number of requests and the number of requests that can be done since sometimes, there are no observation opportunities for some targets. The POI is never seen by the satellites in the constellation which is something that can practically happen. In the results named *With weather*, the number of requests fulfilled is even lower since only the observation opportunities that will be validated are considered. Indeed, in that case, it is assumed that the real weather is known in advance and the only observation opportunities that are taken into account are the ones where the real cloud cover is under the threshold defined by the user. The resolution time for the CP Optimizer solver is about 65 seconds.

	Total number of requests	Number of requests that can be done	Number of requests Done	Mean data times (in h)
Without weather	299.81	255.0	**239.59**	**12.54**
With weather	299.81	136.36	**127.56**	**13.08**

Table 5: Results with CP Optimizer on average of the 100 generated instances

Two different objective functions were previously defined and both models are solved using the CP Optimizer solver. The first one aims at only maximizing the number of requests done while the second is lexicographic and maximizes first the number of requests done, and then it minimizes the mean time when the images are available for the users. However, the results are slightly the same. Indeed, in both cases, the number of requests done does not change and the mean data availability time is slightly different. This difference regarding the times cannot be seen as there are converted into hours, however, in Annex 2, the difference is more visible as time is converted into seconds. When the model is lexicographic, the solver does not terminate in a reasonable time and a time limit is therefore set first to one second and then to one minute in order to see how much the results change as the limit time increases. Doing so, the results regarding the time are slightly better and more minimized every time. However, while using the lexicographic model, the results obtained are not optimal as a time limit is set and the solver does not have time to prove the optimality of the solutions found.

Another test is done using a prescient greedy. The results are illustrated in Table 6.

	Requests performed	Requests fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
All opportunities	227.18	**120.79**	179.02	**106.39**	**16.891**
Validated opportunities	120.79	**120.79**	179.02	**0.0**	**16.891**

Table 6: Results with the prescient greedy algorithm on average over the 100 generated instances

Two cases were looked at. In the first one named *All opportunities*, all opportunities are selected. Therefore, in the greedy algorithm, the observation opportunities are initially classified and then chosen with the greedy one by one and placed when it is possible. It is therefore possible that some non-validated requests will be made. In the second case named *Validated opportunities*, the observation opportunities are selected beforehand and no opportunity destined to be invalidated will be selected.

Two different ways of ranking the observation opportunities are implemented. The first one only considers the real cloud cover while the second one considers lexicographically first the real cloud cover and then the time when the images are taken. However, with the greedy algorithm, the results obtained in these two ways of ranking the opportunities are identical.

9.1.3.2 First greedy approach

In this first approach, the greedy algorithm is applied to all possible combinations of the four local criteria previously described. The table representing the results for all combinations is given in Annex 2. In the following table, the main results, in terms of the maximum number of requests done, the minimum of images failed and the minimum mean data availability times, are given in Table 7. Each time the greedy algorithm is used, the resolution time is about 1.5 to 2 minutes.

For all the different combinations, it can be interesting to know how many times each model (M_0 or M_1) is used and how many times, for a given planning, the M_1 model is wrong. This means to calculate three different values for this model. First, we will calculate all the moments where an image taken is said to be fulfilled with the model but ends up being invalidated with the real weather. Then, we will calculate the moments where

the model estimates that an image is invalidated while the real weather would validate it, and finally, all the moment an image is assumed to be invalidated using the model but is still taken in the simulation by the greedy algorithm. Indeed, the opportunities are ordered and given as inputs for the greedy algorithm and this algorithm schedules all the opportunities that can be added to the plans of the satellites in the given order.

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
Max requests fulfilled R / TCC	207.98	234.65	**109.84**	109.98	189.97	124.67	20.072
Min images failed R / TCC / TS / TT	208.0	233.1	109.63	109.72	190.18	**123.38**	20.087
Min availability times TT / TCC / TS / R	186.39	250.53	88.53	89.09	211.28	161.44	**13.202**
Average values	203.09	236.1	**97.44**	97.67	202.37	**138.43**	**16.889**

Table 7: Table of the results obtained with the best combinations on average of the 100 generated instances

	Images M0	Images M1	Images fulfilled M1 failed real	Images fulfilled real failed M1	Images known failed M1
Average values	0.0	236.1	6.39	6.15	136.31

Table 8: Table of the results obtained considering the weather on average of the 100 generated instances

We can see in Table 8 that model M_0 is not valid long enough for the weather forecast of an observation to be used with it. Indeed, the M_0 forecast is available every 15 minutes and it is only valid for 30 minutes. This means that for the weather forecast associated with an observation window to be defined under the M_0 model, this window must be placed less than 30 minutes after the satellite passes over the ground station when the latest weather forecast updates are made. In our problem, the stations are located around the equator and are far away for the targeted POI. Model M_0 is therefore only used for the early reactivation of the non-validated requests. In this random example, the weather forecast does not seem to be far from reality. However, we notice that many observation opportunities that are not validated are still planned by the greedy algorithm. Indeed, by

using a greedy algorithm, it is assumed that as long as an observation can be done and if the observation is the next one to be done in the ranking, it will be planned no matter the weather forecasts.
It can be interesting to define a threshold below which it is not worth planning with the given observation opportunity.

In the following table, five thresholds have been defined: 60 %, 50 %, 40 % and 30%. This means that an observation opportunity will not be used in the planning if the TCC obtained using the forecast is above 60 %, 50 %, 40 % or 30% of the TCC asked by the user. The following table gives the results obtained with the combination TCC / TT / TS / R on average over the 100 generated instances and with the combination R / TCC / TT / TS that provides the best number of requests validated while using the four criteria.

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
Without threshold	186.4	216.79	**89.66**	89.88	210.15	**126.91**	**16.627**
60 %	112.18	117.56	**89.54**	89.76	210.27	**27.8**	**16.626**
50 %	110.06	114.46	**89.44**	89.66	210.37	**24.8**	**16.623**
40 %	106.99	110.63	**89.23**	89.44	210.58	**21.19**	**16.619**
30 %	103.68	106.44	**88.81**	89.01	211.0	**17.43**	**16.618**

Table 9: Results obtained using the greedy algorithm with a threshold with the combination: TCC / TT / TS / R

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
Without threshold	208.0	236.1	**109.82**	109.95	189.99	**126.15**	**20.071**
60 %	135.45	140.04	**109.76**	109.89	190.05	**30.15**	**20.075**
50 %	133.3	136.92	**109.66**	109.79	190.15	**27.13**	**20.076**
40 %	130.24	133.12	**109.46**	109.58	190.35	**23.54**	**20.079**
30 %	126.93	128.93	**109.04**	109.15	190.77	**19.78**	**20.092**

Table 10: Results obtained using the greedy algorithm with a threshold with the combination: R / TCC / TT / TS

In the two tables, the first line gives the results obtained at the end of the simulation without any threshold defined and afterwards, when the threshold is set, we can see that the number of non-validated images decreases drastically and the number of request fulfilled decreased slightly on average. In any case, the mean time when

the data is available remains roughly the same. It would be interesting to see, in a future work, how the results would vary if we applied decision tree generation while using a threshold and if the solutions found would be better.

9.1.3.3 Decision tree generation

In the previous part, the results obtained with all the different combinations are presented. Some combinations give better results than others regarding the three global criteria previously introduced. However, the classification found is not the optimal one. A good-quality decision tree is generated using a Simulated Annealing (SA) metaheuristic in order to find a near-optimal solution to the problem, regarding the three global criteria. Indeed, in this SA algorithm, the defined objective function is the sum / difference of the three previously normalized criteria with a higher weight putted on the criterion associated with the number of request fulfilled, which is assumed to be the most important one. The decision tree is generated from a combination. Two combinations of the four criteria were chosen: TCC / TT / TS / R and R / TCC / TT / TS. The second combination provides the best solution in terms of number of requests fulfilled.

The decision tree is learned from the 50 first generated instances. The learning is therefore done on the first part of the generated instances. The decision tree is generated using a Simulated Annealing (SA) algorithm. In this algorithm, the initial temperature is calculated after a prior heating process is performed. This process is done iteratively until the acceptance rate is close to a number set to 0.8. Afterward, in the algorithm, the temperature decreases iteratively using a geometrical law $c_{k+1} = \alpha c_k$ where α is set to 0.99. There are 200 iterations made at each step of the cooling process. The simulation time for generating this decision tree with the Simulated Annealing algorithm is of 17.98 hours.

In the following table, we have the results obtained for the 50 first generated instances.

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
1	233.3	286.02	**114.26**	114.58	182.06	**171.44**	**14.37**
2	233.3	288.0	**113.74**	114.06	182.58	**173.94**	**14.06**

Table 11: Results obtained using the SA algorithm on the 50 first generated instances with the two combinations
1: TCC / TT / TS / R 2: R / TCC / TT / TS

We can see that for these generated instances from which the learning process has been done, the results obtained are much better than the ones obtained on average with the different combinations and that we are getting close to the upper bounds.

However, in order to see if the resulting decision tree effectively ranks the observation opportunities, it is essential to apply it to the next 50 instances on which nothing has been learned. This is what is described in the following tables:

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
1	222.06	297.44	**106.12**	106.94	197.18	**190.5**	**13.85**
2	233.22	311.22	**111.26**	112.06	192.04	**199.16**	**13.57**

Table 12: Results obtained using the SA algorithm on the 50 last generated instances with the two combinations
1: TCC / TT / TS / R 2: R / TCC / TT / TS

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
1	189.24	220.54	**90.92**	91.18	212.38	**129.36**	**16.59**
2	211.06	240.2	**111.32**	111.46	191.98	**128.74**	**20.06**

Table 13: Results obtained using the greedy algorithm on the 50 last generated instances with the two combinations
1: TCC / TT / TS / R 2: R / TCC / TT / TS

The results obtained are less convincing than those found with the first 50 generated instances, which was expected. However, we obtain results that are better on average than the ones we have using only the greedy approach without decision tree, mainly with the combination TCC / TT / TS / R. Indeed, if we make a comparison using Table 13, we notice that for this combination the number of successful requests is much higher than what was found by the greedy algorithm alone and for both combinations, the number of failed images is higher but the average time of availability of data is much lower than that obtained by the greedy. The results obtained are quite interesting even though they are far from the defined upper bounds.

9.1.4 Weather forecasts

In what was done previously, the weather forecasts were randomly generated. However, it is important to see how the simulator responds when the weather forecast data is real and comes from some systems that may contain errors or are not precise enough.

9.1.4.1 Use of weather forecast on several scales

As previously defined, two scales are defined in the simulator. The first one is a short-term time scale and corresponds to the model M_0 while the second one is a longer-term time scale and it corresponds to the model M_1. In what follows, a presentation of the real data used for both models will be made.

Moreover, it is essential to know the real weather associated to the different observation opportunities in order to be able to do the validation of the images in the ground. We will therefore explain which data is used to

model the real weather.

The retrieval of all these weather data was quite complicated. Indeed, free data are not trivial to find. Moreover, the GRIB encoding format used to store weather data is not completely standardized, especially for the name of the attributes that we try to retrieve. It could take some time to determine what these attributes were. Moreover, it was sometimes necessary to make conversions to have the cloud cover associated with the targets at given latitude and longitude.

9.1.4.1.1 Retrieval of weather data for the M_0 model

The weather forecast data used for the M_0 model is given with Eumetsat. They are named Cloud Mask products and describe the scene type (either 'clear' or 'cloudy') on a pixel level. To retrieve all the weather data on the whole Earth would have required managing much too big data formats, a choice was therefore made to take the weather data only around the area where we define the POI for the targets (see Figure 31 and Figure 32 to zoom in on the selected area).

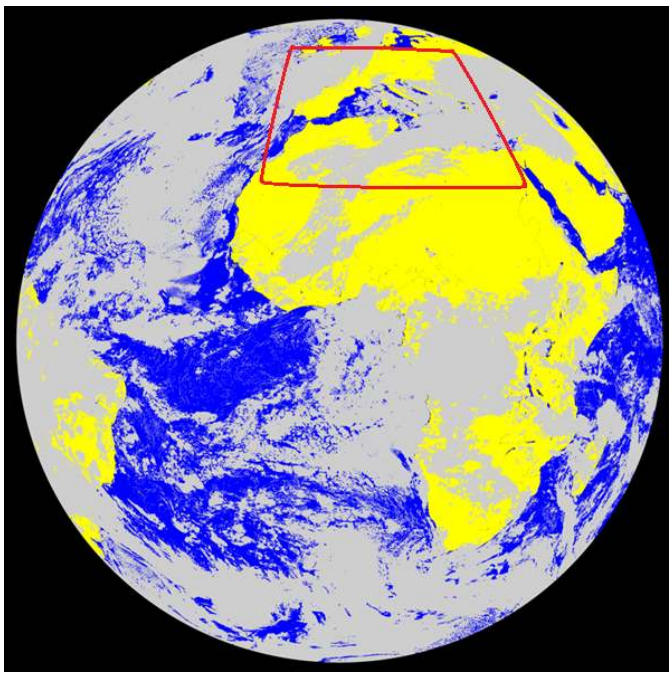

Figure 31: Area of interest where the forecast are retrieved

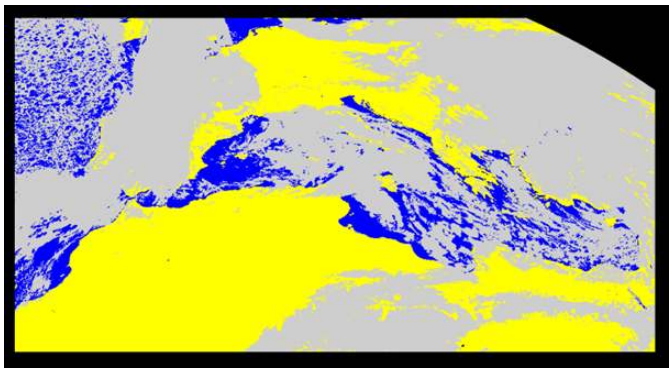

Figure 32: Flat map of the entire area of interest

The Eumetsat cloud cover mask data allows us to have the weather information every 15 minutes given with a cloud mask (the result is either 0 or 1). The spatial resolution is 0.041°. With Eumetsat, we obtain photographs of the Earth every 15 minutes made by a geostationary satellite, and the cloud cover mask is obtained with Eumetsat by making a calculation on the pixels of the images. Assuming that the weather data obtained remains valid for 30 minutes after the photo was taken, we can deduce that we have found here a short-term forecast model. It is indeed the data recovered for model M_0.

The data are taken for the month of March 2022. Figure 33 gives a visualization of the cloud mask as a function of time for one specific target.

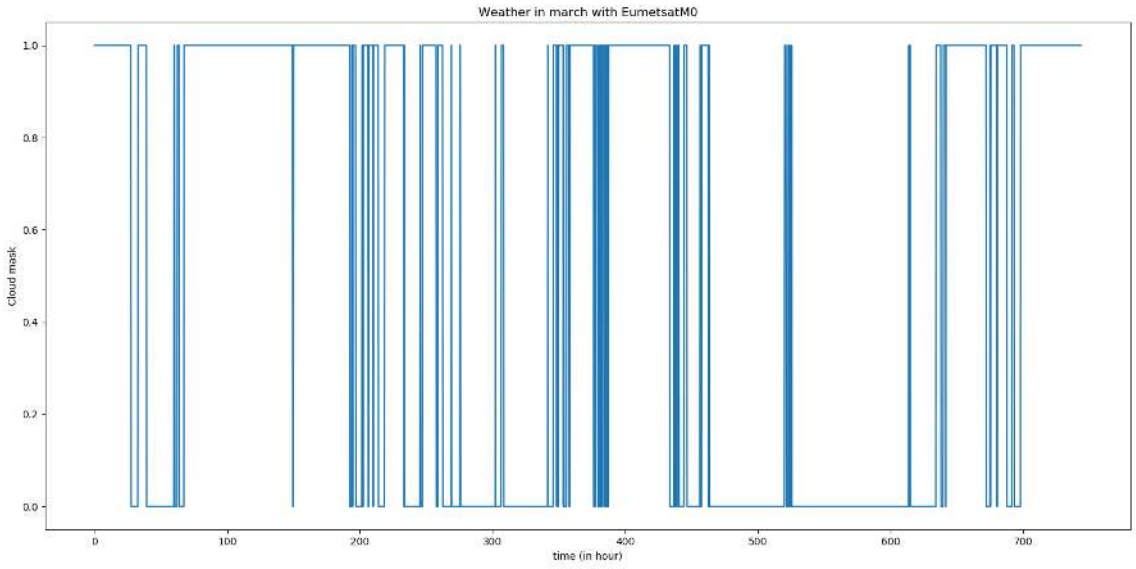

Figure 33: Cloud mask as a function of the time (in hours)

9.1.4.1.2 Retrieval of weather data for the M_1 model

The weather forecast data used for the M_1 model is given using GFS (Global Forecast System). It is total cloud cover data freely available on the GFS website. It is possible, with this product, to retrieve weather data for the entire Earth. The GFS total cloud cover data allows us to have the weather forecast every 3 hours with runs every 6 hours. The spatial resolution is 0.5°. Therefore, in this case, we have weather forecast data that are less spatially accurate, with a much less precise spatial resolution and larger meshes, and temporally with less frequent forecast updates. The weather forecast obtained is on the long term and for a given run, we have a possible forecast on the 4 days of planning. We can then deduce that we have found here a longer term forecast model. The GFS data will indeed be the recovered data used for model M_1.

The data are also taken for the month of March 2022. Figure 34 gives a visualization of the total cloud cover as a function of time.

In Figure 35, a comparison is made between the different runs for a given day in March. Indeed, for model M_1, the forecast is done every three hours and this forecast is recalculated every six hours. In Figure 35, we have a view of the weather forecast using the four runs for one specific day. At the beginning of the day, at midnight, *run 00* is used and gives forecast for the next 20 hours each three hours, then at 6 am, *run 06* gives the forecast for the next 20 hours every 3 hours, then by midday, we have access to the forecast using $run12$ and in the end, at 6 pm, we have the forecast using $run18$ for the same times.

We can see in the figure that, even if the forecasts are globally quite close, there are still sometimes significant differences between the forecasts obtained with two different runs available at the same time.

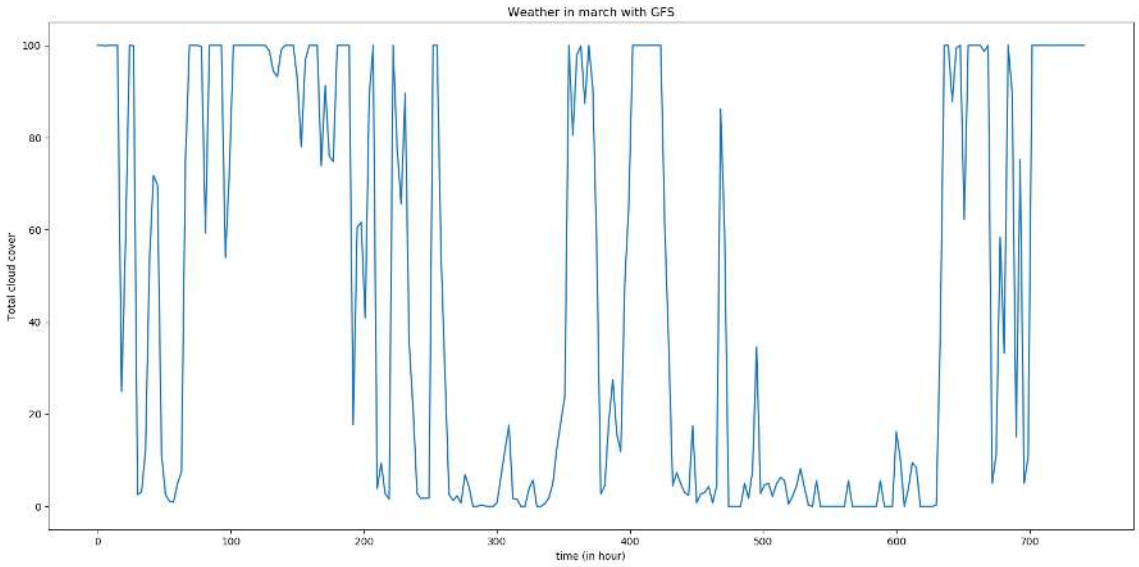

Figure 34: Cloud mask as a function of the time (in hours)

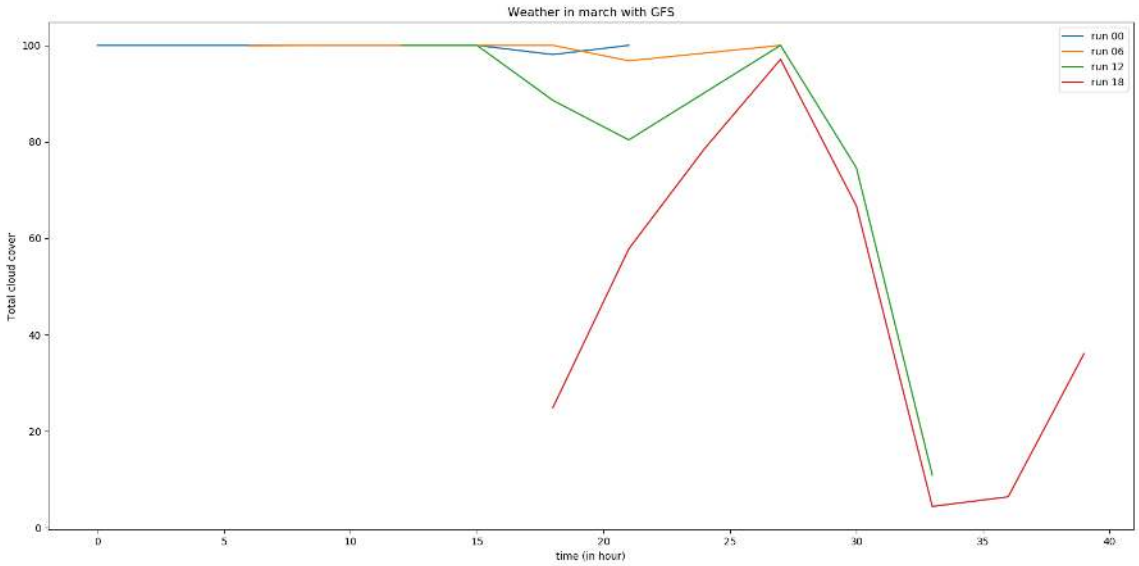

Figure 35: Comparison of different runs for a given day

We can conclude that the weather forecasts with GFS are not very accurate and this may influence the planning later.

9.1.4.2 Application to the reality

We have previously looked at how to retrieve real data that can be used for the M0 and M1 models, it is now important to look at how to retrieve data to simulate real weather. Indeed, in the simulation, when an image is taken by the satellite, it is sent back to the ground station and one hour later it will be validated. This validation is done using the real weather that we have when the image is taken. It is therefore highly important to retrieve real weather data in order to be able to make the validation or the invalidation of the images.

9.1.4.2.1 Obtaining real weather data

The first track was to use the data from ERA5 as it gives the real with in term of total cloud cover and it is freely available on the website. However, with ERA5, the forecast is made every hour and the spatial resolution was much larger than the one used with Eumetsat. Therefore, the decision was made to use the Eumetsat data to simulate real weather by making random draws around the chosen points using the set of pixels around these points

In figure 36, a comparison is made between the real weather obtained while using Era5 and the obtained while using Eumetsat with the random draws.

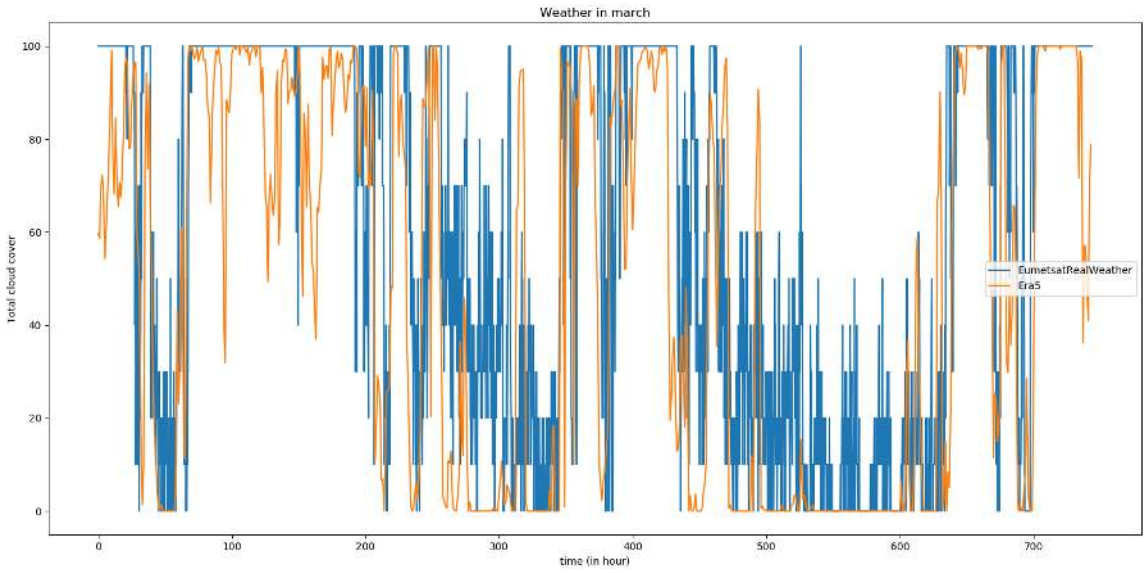

Figure 36: Comparison of the real weather obtained while using Era5 or Eumetsat

In the figure, we can see that there is a difference between the two total cloud cover over the month of March, however, the weather information is still quite similar.

9.1.4.2.2 Comparison of the different weather information

In the following figure (Figure 37), a comparison is made between the real weather obtained with Eumetsat using random draws and the weather forecast used in the two models M_0 and M_1 with Eumetsat and GFS.

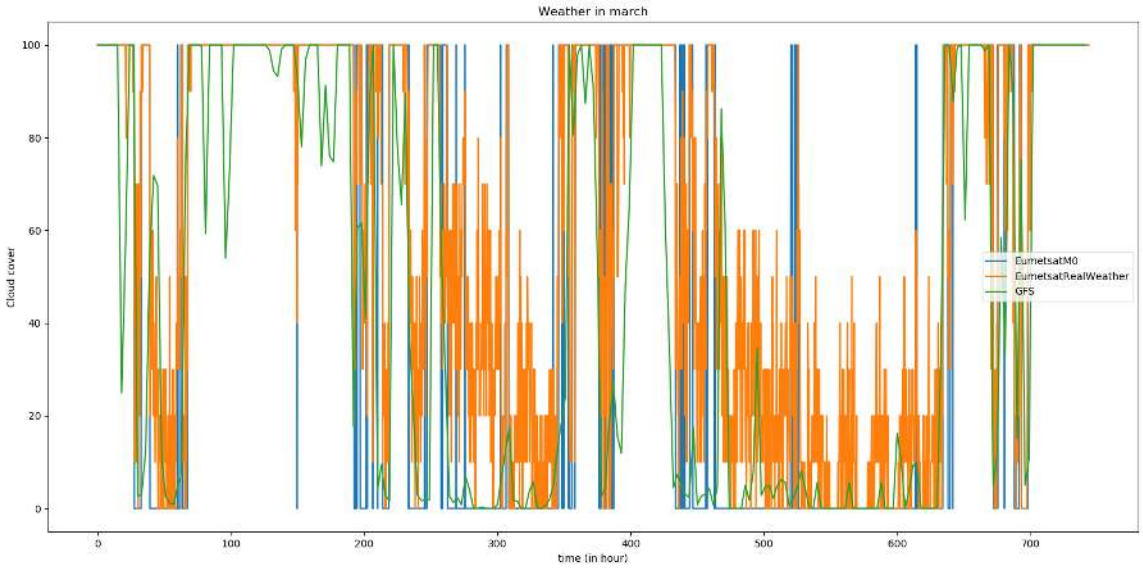

Figure 37: Comparison of the weather information obtained for the real weather and for the two models

We can see in the figure that the results obtained for the three weather information are globally roughly similar over the month of March, however there are sometimes large discrepancies, especially with the GFS data where the accuracy is the worse.

9.1.5 Results using real weather data

The recovery of weather data is an important part of the work. It allows to apply the developed algorithms to a real and visible case: the earth observation satellite scheduling in Europe in March 2022. This is indeed a month where the weather varies a lot from one day to another and this allows us to see how the simulator reacts to such variations. Weather data is, therefore, different from the one used in the previous generated instance. However, everything else, such as the location of the targets, is the same as what we had before.

9.1.5.1 Boundary calculation

As presented previously, a calculation of upper bounds is made to evaluate the relevance of the methods developed.

Table 14 illustrates the results obtained using CP Optimizer solver. The resolution time to solve the problem with CP Optimizer is 72 seconds for the 100 generated instances and the 4 days.
This table represents the results obtained when the model is not lexicographic and when the solver maximizes the number of requests performed. In the case where the weather is not considered, entitled *Without weather*, the difference between the total number of requests and the number of requests that can be done is the same as the one in the case where weather data is randomly generated. However, in the case where the weather is

	Total number of requests	Number of requests that can be done	Number of requests Done	Mean data times (in h)
Without weather	299.81	255.0	**239.59**	**12.54**
With weather	299.81	132.3	**124.65**	**14.19**

Table 14: Results with CP Optimizer on average of the 100 generated instances

taken into account, this difference is a bit larger while using real weather data. This can be understandable since the weather forecast data randomly generated is slightly closer to the actual weather, also generated in the first case. The results obtained in the lexicographic case are the same while we consider the number of requests performed. However, as described in Annex 3, the mean times when the data is available are slightly smaller in the lexicographic case and they decrease as the limit time assigned to the solver increases.

Another bound we can look at is the one when the greedy algorithm is assumed to be prescient and to know the real weather. As previously introduced in the random case, two cases are looked at: *All Opportunities* and *Validated Opportunities*. The results are illustrated in Table 15.

	Requests performed	Requests fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
All opportunities	227.74	**118.17**	181.64	**109.57**	**16.40**
Validated opportunities	118.4	**118.4**	181.41	**0.0**	**16.41**

Table 15: Results with the prescient greedy algorithm on average of the 100 generated instances

9.1.5.2 First greedy approach

In this first greedy approach, the greedy algorithm is applied once again to all possible combinations of the four local criteria previously described in the case where real weather data are retrieved. The table representing the results for all combinations is given in Annex 3. The main results, in terms of maximum number of requests done, minimum number of images failed and minimum mean data availability times, are given in Table 16. We can see that the optimal combinations in this case are different from the ones obtained in the case where weather data is randomly generated. The weather has, therefore, a real impact on the planning and the choice of combinations. For all the different combinations, it can be interesting to know, as seen previously, how many times each model (M_0 or M_1) is used and how many times, for a given planning, the M_1 model is wrong. The results are presented, in average, in Table 17.

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
Max requests fulfilled TCC	182.27	199.77	**72.18**	72.19	227.63	127.58	16.31
Min images failed TCC / TS / TT / R	178.3	192.96	68.39	68.4	231.42	**124.56**	16.67
Min availability times TT / TCC / R	186.24	223.41	69.29	69.38	230.52	154.03	**13.32**
Average values	194.6	214.15	**70.09**	70.13	229.72	**147.06**	**16.35**

Table 16: Table of the results obtained with the best combinations on average of the 100 generated instances

	Images M0	Images M1	Images fulfilled M1 failed real	Images fulfilled real failed M1	Images known failed M1
Average values	0.0	217.21	19.74	23.99	151.33

Table 17: Table of the results obtained considering the weather on average of the 100 generated instances

We can see that model M_0 is still not valid long enough for the weather forecast of an observation to be used with it. Model M_0 is therefore only used for the early reactivation of the non-validated requests. In this case, described as number 2, the forecast obtained with model M_1 seems to be quite far from the actual weather. Indeed, the number of errors made with the M_1 model is higher than the one in the first case where the data are generated in a random way, named case number 1. This can explain the difference between the number of fulfilled requests in case number 2, about 72 requests, and the number in the previous case (number 1) about 110 requests. Moreover, in case number 2, we are performing more images that are assumed, according to M_1, to be invalidated, about 151 images, than in the previous case, about 136 images. However, it can be noticed that the number of images performed that failed using M_1 but are then validated using the real weather is larger than the one of images performed that are validated using M_1 and then failed with the real weather.

As previously done, it can be interesting to define a threshold below which it is not worth planning with the given observation opportunity. Table 18 shows the results obtained with the four threshold defined earlier: 60 % , 50 % , 40 % and 30 % . The results are given with two combinations on average over the 100 generated

instances: TCC / TT / TS / R and TS / TCC / TT / R that provides the best number of requests validated while using the four criteria.

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
Without threshold	176.76	205.97	**68.06**	68.09	231.75	**137.88**	**15.727**
60 %	74.96	76.74	**51.77**	51.77	248.04	**24.97**	**15.018**
50 %	74.23	75.99	**51.48**	51.48	248.33	**24.51**	**15.016**
40 %	73.34	75.04	**51.12**	51.12	248.69	**23.92**	**14.990**
30 %	72.32	74.0	**50.66**	50.66	249.15	**23.34**	**14.936**

Table 18: Results obtained using the greedy algorithm with a threshold with the combination: TCC / TT / TS / R

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
Without threshold	212.92	222.76	**71.84**	71.85	227.97	**150.91**	**17.6**
60 %	87.29	88.41	**56.6**	56.61	243.21	**31.8**	**17.654**
50 %	86.57	87.7	**56.26**	56.27	243.55	**31.43**	**16.604**
40 %	85.64	86.8	**55.92**	55.93	243.89	**30.87**	**16.581**
30 %	84.5	85.63	**55.45**	55.46	244.36	**30.17**	**16.587**

Table 19: Results obtained using the greedy algorithm with a threshold with the combination: TS / TCC / TT / R

In the two table, we can see that the number of non-validated images decreases drastically when a threshold is made. However, the number of requests fulfilled also decreases a lot in this case. This can be understandable regarding the number of requests that are validated with the real weather but are considered non-validated by M_1 model. Indeed, the threshold is defined from the forecasts of the M_1 model and all the images that were previously expected to be validated with the real weather are no longer taken into account in the scheduling because of the M_1 threshold.

9.1.5.3 Decision tree generation

In this part, a near-optimal decision tree is generated the same way as described before. This decision tree (in case number 2) is generated from a defined combination. Two combinations of the four criteria are chosen:

a first one that is the same as the one used in case number 1 where weather data is randomly generated: TCC / TT / TS / R and the one that gives the maximum number of requests fulfilled: TS / TCC / TT / R. The resolution time for generating this decision tree with the SA algorithm is 23.36 hours, larger than the resolution time obtained in the case where the data are randomly generated.

The decision tree is also learned here from the 50 first generated instances. In Table 20, we have the results obtained for the 50 first generated instances.

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
1	206.46	234.32	**78.92**	78.92	217.4	**155.4**	**15.17**
2	207.08	238.52	**78.18**	78.22	218.14	**160.3**	**14.60**

Table 20: Results obtained using the SA algorithm on the 50 first generated instances with the two combinations
1: TCC / TT / TS / R 2: TS / TCC / TT / R

We can see that for these generated instances from which the learning process has been done, the results obtained are much better than the one obtained on average with the different combinations. However, we are still far from the upper bound of about 125 defined using CP Optimizer. The results obtained using a decision tree in the random generation provides better results of about 114 for the number of requests fulfilled.

In the same way as described above, it is essential to apply the decision tree generated to the next 50 instances on which nothing has been learned in order to see if the resulting decision tree effectively ranks the observation opportunities.

This is what is described in the following table:

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
1	229.92	262.9	**88.74**	88.78	214.56	**174.12**	**14.22**
2	216.88	253.94	**83.46**	83.5	219.84	**170.44**	**13.90**

Table 21: Results obtained using the SA algorithm on the 50 last generated instances with the two combinations
1: TCC / TT / TS / R 2: TS / TCC / TT / R

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in h)
1	179.44	208.1	**72.02**	72.04	231.28	**136.06**	**15.72**
2	216.12	225.54	**76.44**	76.44	226.86	**149.1**	**17.53**

Table 22: Results obtained using the greedy algorithm on the 50 last generated instances with the two combinations
1: TCC / TT / TS / R 2: TS / TCC / TT / R

The results obtained are much more convincing than those found with the first 50 generated instances. Indeed, about ten additional requests are satisfied using the last 50 generated instances, which may be strange since the tree has not learn anything from these last 50 generated instances.

In any case, we find better results using the random generation. This can be explained by the fact that the M_1 model used is not accurate enough or close to reality. However, in both case, we manage to find results that are much better using the decision-tree than the ones using only the greedy algorithm alone. Indeed, if we make a comparison between the two algorithms for both cases, each time, about 15 more requests are fulfilled when using the decision tree, the number of images failed increases but the mean time when the data is available for the users is shorter.

Either way, for bother the random generation case and the real data generation case, the combination that gives the best results between the two combinations tested in the decision-tree generation is TCC / TT / TS / R.

9.1.6 Conclusion

To conclude this section, the results obtained using all the different methods are presented in two different cases: a first one where the weather data is randomly generated and a second one where the weather data is retrieved using free cloud forecast. As we can see in the different tables, the results obtained using decision tree generation are better than the ones using the first greedy approach. However, we are still far from the upper bound defined using CP Optimizer.

One reason for this could be that surbooking is not used and therefore requests that are destined to be non-validated can be frozen for a long time before being rescheduled using the M_0 model. Doing an early reschedule presents a great advantage since it allows to send back to the scheduler the requests that are known to be non-validated instead of waiting for the invalidation much later. However, if there were surbooking mecanisms in the process, one could perhaps consider getting closer to the results obtained with greedy prescient by scheduling several times the same requests earlier. Moreover, the algorithm used is a very simple greedy algorithm which does not use any destruction/repair operators and thus has a low insertion power. Indeed, this algorithm places the observations to be made as they are given to it. However, the problem with the results does not seem to come from the use of this greedy strategy since the results obtained with the greedy prescient are very close to the optimum. Errors can therefore come from the reprogramming factor but also from the M_1 model which, in the case where real data are used, is quite far from the real weather, not very accurate and gives many errors.

An example of the behavior of the different algorithms is given in the following figures. This behavior is given for one specific generated instance on which the decision tree has not learned anything. We chose the instance 51. The two graphs are used to compare the performance of the four algorithms: the first greedy approach, the prescient greedy approach with all opportunities, the prescient greedy approach with validated opportunities and the decision tree approach. It allows us to have a view of the behavior of the four algorithms over the entire simulation in terms of the number of requests fulfilled and then, the number of non-validated images.

We can see, with these two figures, Figure 38 and 39, that the best results are obtained using the prescient greedy using the validated opportunities and then the one using all the opportunities, which is normal. Then, the generation tree approach provides much better results for this instance in terms of number of requests fulfilled but poorer results regarding the number of non-validated requests. Globally, the four behaviors are quite similar regarding the simulation time and therefore, the moments during the simulation when some requests are validated and some images are non-validated.

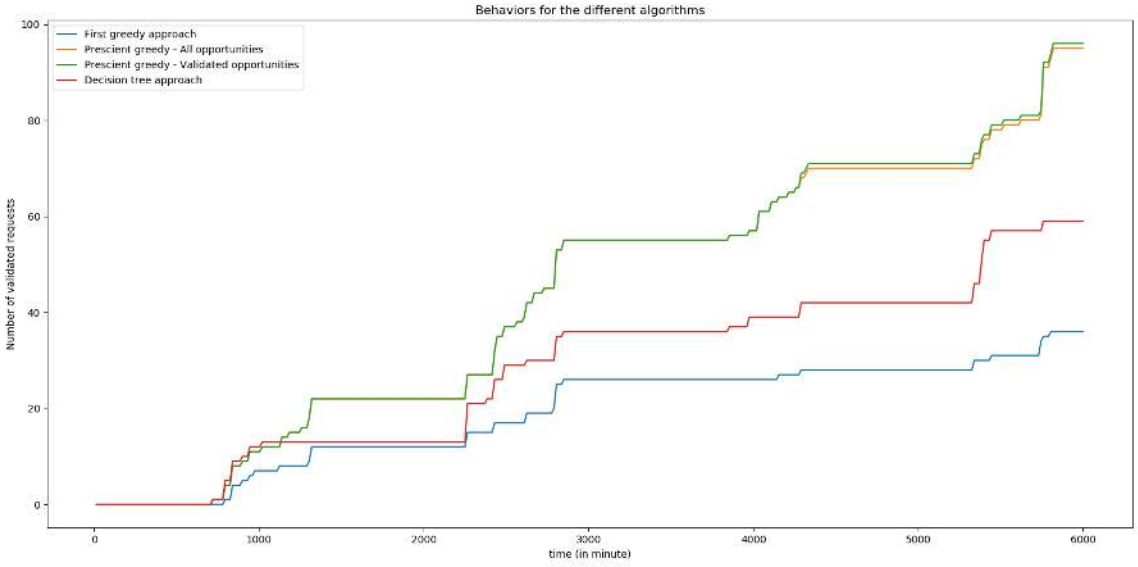

Figure 38: Comparison of the four approaches in terms of number of fulfilled request as a function of the simulation time

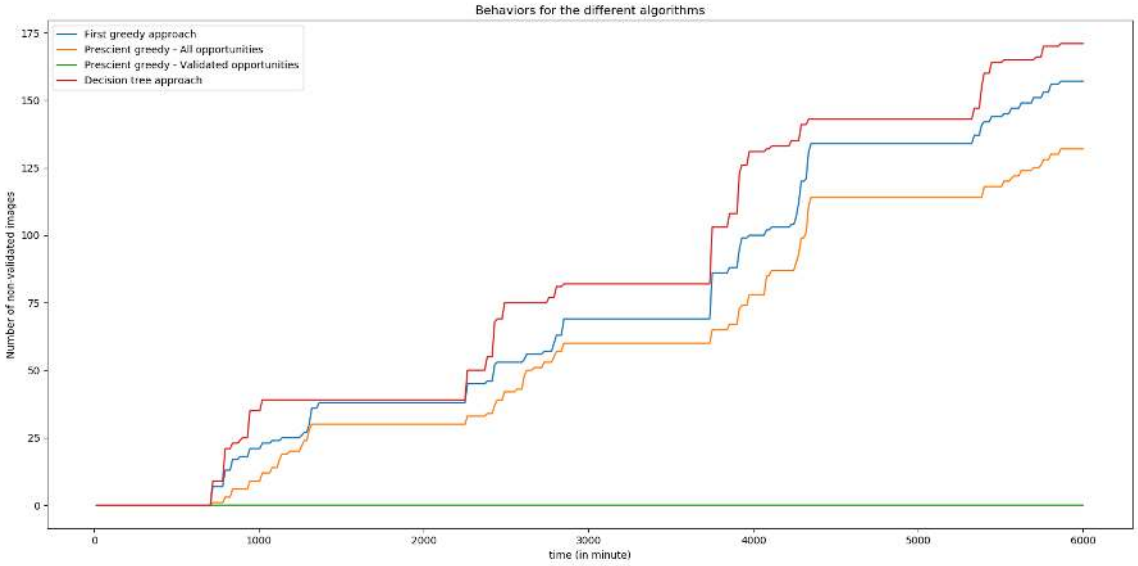

Figure 39: Comparison of the four approaches in terms of number of failed images as a function of the simulation time

Conclusion

In this rapport, we have presented the importance of taking into account weather uncertainties in the planning for solving Earth Observation Scheduling (EOS) problems. Indeed, the presence of cloud cover can affect the validity of an image taken when its percentage is higher than the threshold requested by the user. The image will therefore have to be taken a second time with adequate cloud cover in order for the request associated to be validated for the user.

This report presents a way of taking into account these weather uncertainties by defining to different models: a first one M_0 on the short term and a second one M_1 on a longer term. Our planning is therefore based on these weather models.

In order to solve the EOS scheduling problem, the objective and characteristics of the optimization problem of the internship have been defined. A state-of-the-art of various approaches that has already been implemented in the literature concerning the scheduling of Earth observation satellites under meteorological uncertainties has been done. There are several types of models that can be defined and several ways to express the objective function have been presented. Then, the simulation of the problem has been introduced. Two different approaches have been implemented in order to, from four local criteria - such as the Total Cloud Cover (TTC) obtained using M_0 or M_1 model, the time (TT) when the image is taken, the time (TS) between the end of the communication station and the moment when the image is taken, and the ability to reprogram (R) an request later - optimize three global criteria - such as the number of requests fulfilled, the number of images failed and the mean time when the data is available to the user -. The first implemented approach only considers using a greedy algorithm with different combinations of the four different criteria while the second one uses a generated decision tree to classify the different observation opportunities to perform the requests. The decision tree is generated using a Simulated Annealing (SA) algorithm. The evaluation of the approaches is done using CP Optimizer, in order to have an upper bound of the number of requests that can be done. A prescient greedy algorithm is also used in order to assess the performance of a greedy strategy.

The last part of this report presents all the computational results we manage to obtained during the end-of-study internship. We can see that the approach using the decision tree provides better results than the one using the combination of the four criteria in terms of number of requests fulfilled and a good comprise regarding the number of requests fulfilled, the number of images failed and the mean time when the data is available for the users. However, we are still far from both the upper bound found using CP optimizer and the results using the prescient greedy. Solutions can therefore be found and new approaches implemented in order to find better results and get closer to the optimal bounds.

There are new prospects at sight on which it would be possible to work. A new study could be made on the subject in order to find these better results. A first way to do so would be to try to find a new M_1 model in the available weather data that could be more precise, or try to use M_0 more often in the scheduling and not only while doing the early rescheduling. However, that would imply to change the location of the ground station and to put them closer to the area where the requests are made, impacting the robustness of the approach. Another way could be to modify slightly the greedy algorithm. Instead of planning from scratch, it could be interesting to determine a first plan, and then, to use destruction and repair operators to modify the plan as the simulation progresses. The greedy algorithm could also be modified to have a better power of insertion of the different observation opportunities in the plans of the satellites. Another way to resolve better the issue could be to add a step before the one with the decision tree generation where we could estimate probabilities of success for

the opportunities. The TCC criterion would become a weather-dependent probability of success, and for the Rescheduling criterion, it will not be calculated using the number of weather-valid observations that can be made afterwards the given opportunity anymore, but will be calculated based on the probability of success. A different approach that people could work on could be to use a learning process that would determine which threshold to use for the cloud cover and then to apply the decision tree to the observation opportunities associated with the threshold. Also, in the defined process, the early reprogramming with M_0 is made only when we arrive at the moment when the image is taken. It could be interesting to use a surbooking process that would reschedule the observation earlier when we know that it will be not validated. Furthermore, a question arises about where and when the classification with the generated decision tree is valid. Indeed, in this report, the results are made for the region of Europe during the month of March. However, it could be interesting to see if the generated tree can be applied to a different region on Earth and how much the results change depending on the moment of the year. It would be interesting to see if on decision tree can be generated for all the Earth and for one period of time or if several decision trees should be generated for different regions on Earth and for specific moment in the year.

There are many things that can be done and many approaches that can be tested to try to solve the problem. This rapport presents a part of the possible approaches that can be implemented.

This internship allowed me to develop many skills, both technical and social, and to complete my engineering training. It also allowed me to apply different concepts seen in the courses of the Operational Research Master (Master Master Recherche Opérationnelle), in particular for the formulation of an optimization problem using a MILP model and the development of stochastic resolution methods. My courses at ENAC also allowed me to better understand what I could read in the literature on EOS scheduling. This internship taught me how to use new solving methods, such as constraint programming with the CP Optimizer solver or decision tree generation with a SA algorithm. This internship, therefore, brought me a lot professionally but also personally, as it allowed me to work with researchers for more than five months and to learn a lot more about research and how it can be applied in industry. With this experience and having now a clearer idea of the reality of the profession of researcher, I would like to continue in research and specialize in optimization applied to the space sector.

Annex

Annex 1: Tables of results defining and solving the toy example

Presentation of the situation

The situation is presented in the following table.

Annex 2: Tables of results with random weather data generations

Tables of results for boundary calculation

In the following table, the results are obtained using CP Optimizer solver by maximizing the number of requests.

	Total number of requests	Number of requests that can be done	Number of requests Done	Mean data times (in s)
Without weather	299.81	255.0	239.59	45134.97
With weather	299.81	136.36	127.56	47098.74

Table 23: Results with CP Optimizer on average of the 100 generated instances while maximizing the number of requests

In the following table, the results are obtained using CP Optimizer solver by using the lexicographic modeling with a time limit set to one second.

	Total number of requests	Number of requests that can be done	Number of requests Done	Mean data times (in s)
Without weather	299.81	255.0	239.59	45066.1
With weather	299.81	136.36	127.56	47077.64

Table 24: Results with CP Optimizer on average of the 100 generated instances while using the lexicographic approach with a time limit of one second

In the following table, the results are obtained using CP Optimizer solver by using the lexicographic modeling with a time limit set to one minute.

	Total number of requests	Number of requests that can be done	Number of requests Done	Mean data times (in s)
Without weather	299.81	255.0	239.59	45063.7
With weather	299.81	136.36	127.56	47077.64

Table 25: Results with CP Optimizer on average of the 100 generated instances while using the lexicographic approach with a time limit of one minute

Tables of results for the different combinations using the greedy approach

The four local criteria are combined in 65 ways. The following table gives a association between the described combination and the associated number.

Name of the combination	Number associated
Total Cloud Cover (TCC)	1
Time Taken (TT)	2
Time Taken to get	3
Time to Station (TS)	4
Reprogramming (R)	5
TCC / TT	6
TCC / TS	7
TCC / R	8
TT / TCC	9
TT / TS	10
TT / R	11
TS / TCC	12
TS / TT	13
TS / R	14
R / TCC	15
R / TT	16
R / TS	17
TCC / TT / TS	18
TCC / TT / R	19
TCC / TS / TT	20
TCC / TS / R	21
TCC / R / TT	22
TCC / R / TS	23
TT / TCC / TS	24
TT / TCC / R	25
TT / TS / TCC	26
TT / TS / R	27
TT / R / TCC	28
TT / R / TS	29
TS / TCC / TT	30
TS / TCC / R	31
TS / TT / TCC	32
TS / TT / R	33
TS / R / TCC	34
TS / R / TT	35

Name of the combination	Number associated
R / TCC / TT	36
R / TCC / TS	37
R / TT / TCC	38
R / TT / TS	39
R / TS / TCC	40
R / TS / TT	41
TCC / TT / TS / R	42
TCC / TT / R / TS	43
TCC / TS / TT / R	44
TCC / TS / R / TT	45
TCC / R / TT / TS	46
TCC / R / TS / TT	47
TT / TCC / TS / R	48
TT / TCC / R / TS	49
TT / TS / TCC / R	50
TT / TS / R / TCC	51
TT / R / TCC / TS	52
TT / R / TS / TCC	53
TS / TCC / TT / R	54
TS / TCC / R / TT	55
TS / TT / TCC / R	56
TS / TT / R / TCC	57
TS / R / TCC / TT	58
TS / R / TT / TCC	59
R / TCC / TT / TS	60
R / TCC / TS / TT	61
R / TT / TCC / TS	62
R / TT / TS / TCC	63
R / TS / TCC / TT	64
R / TS / TT / TCC	65

Table 26: Association used to describe the combinations

Using the association made in Table 26, the results of the simulation for each combination are given in the following table.

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in s)
1	186.43	215.31	89.62	89.83	210.19	125.48	60001.83
2	186.39	249.78	89.04	89.6	210.77	160.18	47638.39
3	186.47	248.49	89.08	89.65	210.73	158.84	47637.23
4	224.01	231.99	102.43	102.49	197.38	129.5	63898.43
5	236.26	250.45	109.32	109.35	190.49	141.1	72176.16
6	186.43	216.82	89.62	89.84	210.19	126.98	59862.7
7	186.4	213.56	89.41	89.58	210.4	123.98	60388.05
8	186.37	215.16	89.61	89.83	210.2	125.33	60392.2
9	186.42	250.45	88.51	89.07	211.3	161.38	47528.57
10	186.6	250.53	89.3	89.83	210.51	160.7	47685.45
11	186.38	249.9	89.94	90.54	209.87	159.36	47754.64
12	223.99	231.97	102.43	102.49	197.38	129.48	63899.49
13	223.99	231.97	102.43	102.49	197.38	129.48	63898.43
14	224.01	231.99	102.43	102.49	197.38	129.5	63899.49
15	207.98	234.65	109.84	109.98	189.97	124.67	72260.94
16	208.03	266.07	109.36	109.58	190.45	156.49	71903.43
17	229.65	237.16	107.97	107.98	191.84	129.18	72497.46
18	186.4	216.79	89.66	89.88	210.15	126.91	59860.02
19	186.4	216.76	89.69	89.91	210.12	126.85	59861.05
20	186.4	213.56	89.41	89.58	210.4	123.98	60388.05
21	186.4	213.56	89.41	89.58	210.4	123.98	60388.05
22	186.37	216.53	89.59	89.8	210.22	126.73	60384.73
23	186.35	213.41	89.41	89.58	210.4	123.83	60652.99
24	186.39	250.53	88.53	89.09	211.28	161.44	47525.93
25	186.39	250.41	88.59	89.15	211.22	161.26	47527.64
26	186.6	250.53	89.3	89.83	210.51	160.7	47685.45
27	186.6	250.53	89.3	89.83	210.51	160.7	47685.45
28	186.42	249.9	89.99	90.58	209.82	159.32	47732.34
29	186.5	250.1	89.95	90.53	209.86	159.57	47761.71
30	223.99	231.97	102.43	102.49	197.38	129.48	63899.49
31	223.99	231.97	102.43	102.49	197.38	129.48	63899.49
32	223.99	231.97	102.43	102.49	197.38	129.48	63898.43
33	223.99	231.97	102.43	102.49	197.38	129.48	63898.43
34	223.99	231.97	102.43	102.49	197.38	129.48	63899.49
35	223.99	231.97	102.43	102.49	197.38	129.48	63899.49

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in s)
36	207.98	236.03	109.82	109.95	189.99	126.08	72255.63
37	208.0	233.1	109.63	109.72	190.18	123.38	72313.47
38	208.0	265.99	109.4	109.62	190.41	156.37	71905.04
39	208.12	266.15	109.35	109.57	190.46	156.58	71896.71
40	229.63	237.14	107.97	107.98	191.84	129.16	72497.46
41	229.63	237.14	107.97	107.98	191.84	129.16	72497.46
42	186.4	216.79	89.66	89.88	210.15	126.91	59860.02
43	186.39	216.75	89.69	89.91	210.12	126.84	59860.25
44	186.4	213.56	89.41	89.58	210.4	123.98	60388.05
45	186.4	213.56	89.41	89.58	210.4	123.98	60388.05
46	186.35	216.51	89.59	89.8	210.22	126.71	60383.92
47	186.35	213.41	89.41	89.58	210.4	123.83	60652.99
48	186.39	250.53	88.53	89.09	211.28	161.44	47525.93
49	186.38	250.51	88.58	89.14	211.23	161.37	47527.71
50	186.6	250.53	89.3	89.83	210.51	160.7	47685.45
51	186.6	250.53	89.3	89.83	210.51	160.7	47685.45
52	186.43	249.97	89.94	90.52	209.87	159.45	47738.09
53	186.5	250.1	89.95	90.53	209.86	159.57	47761.71
54	223.99	231.97	102.43	102.49	197.38	129.48	63899.49
55	223.99	231.97	102.43	102.49	197.38	129.48	63899.49
56	223.99	231.97	102.43	102.49	197.38	129.48	63898.43
57	223.99	231.97	102.43	102.49	197.38	129.48	63898.43
58	223.99	231.97	102.43	102.49	197.38	129.48	63899.49
59	223.99	231.97	102.43	102.49	197.38	129.48	63899.49
60	208.0	236.1	109.82	109.95	189.99	126.15	72255.63
61	208.0	233.1	109.63	109.72	190.18	123.38	72313.47
62	208.02	266.08	109.4	109.62	190.41	156.46	71905.04
63	208.12	266.15	109.35	109.57	190.46	156.58	71896.71
64	229.63	237.14	107.97	107.98	191.84	129.16	72497.46
65	229.63	237.14	107.97	107.98	191.84	129.16	72497.46

Table 27: Table of the results obtained for each combination on average of the 100 generated instances

For all the different combinations, it can be interesting to know how many times each model (M_0 or M_1) is used and how many times, for a given planning, the M_1 model is wrong. The following table provides the results for every combination. The same association is used.

	Images M0	Images M1	Images fulfilled M1 failed real	Images fulfilled real failed M1	Images known failed M1
1	0.0	215.31	6.23	5.09	124.34
2	0.0	249.78	5.54	7.4	162.04
3	0.0	248.49	5.55	7.41	160.7
4	0.0	231.99	5.76	6.63	130.37
5	0.0	250.45	7.59	6.11	139.62
6	0.0	216.82	6.27	5.06	125.77
7	0.0	213.56	6.29	4.87	122.5
8	0.0	215.16	6.28	5.08	124.13
9	0.0	250.45	5.5	7.39	163.27
10	0.0	250.53	5.58	7.48	162.6
11	0.0	249.9	5.62	7.39	161.3
12	0.0	231.97	5.76	6.63	130.35
13	0.0	231.97	5.76	6.63	130.35
14	0.0	231.99	5.76	6.63	130.37
15	0.0	234.65	8.65	5.01	121.03
16	0.0	266.07	8.26	5.32	153.55
17	0.0	237.16	7.08	6.07	128.17
18	0.0	216.79	6.27	5.06	125.7
19	0.0	216.76	6.29	5.06	125.62
20	0.0	213.56	6.29	4.87	122.56
21	0.0	213.56	6.29	4.87	122.56
22	0.0	216.53	6.28	5.05	125.5
23	0.0	213.41	6.31	4.85	122.37
24	0.0	250.53	5.5	7.38	163.32
25	0.0	250.41	5.53	7.4	163.13
26	0.0	250.53	5.58	7.48	162.6
27	0.0	250.53	5.58	7.48	162.6
28	0.0	249.9	5.61	7.42	161.13
29	0.0	250.1	5.69	7.41	161.29
30	0.0	231.97	5.76	6.63	130.35
31	0.0	231.97	5.76	6.63	130.35
32	0.0	231.97	5.76	6.63	130.35
33	0.0	231.97	5.76	6.63	130.35
34	0.0	231.97	5.76	6.63	130.35
35	0.0	231.97	5.76	6.63	130.35

	Images M0	Images M1	Images fulfilled M1 failed real	Images fulfilled real failed M1	Images known failed M1
36	0.0	236.03	8.65	4.98	122.41
37	0.0	233.1	8.69	4.79	119.48
38	0.0	265.99	8.23	5.33	153.47
39	0.0	266.15	8.26	5.31	153.63
40	0.0	237.14	7.08	6.07	128.15
41	0.0	237.14	7.08	6.07	128.15
42	0.0	216.79	6.27	5.06	125.7
43	0.0	216.75	6.29	5.06	125.6
44	0.0	213.56	6.29	4.87	122.56
45	0.0	213.56	6.29	4.87	122.56
46	0.0	216.51	6.28	5.05	125.48
47	0.0	213.41	6.31	4.85	122.37
48	0.0	250.53	5.5	7.38	163.32
49	0.0	250.51	5.53	7.39	163.23
50	0.0	250.53	5.58	7.48	162.6
51	0.0	250.53	5.58	7.48	162.6
52	0.0	249.97	5.63	7.42	161.24
53	0.0	250.1	5.69	7.41	161.29
54	0.0	231.97	5.76	6.63	130.35
55	0.0	231.97	5.76	6.63	130.35
56	0.0	231.97	5.76	6.63	130.35
57	0.0	231.97	5.76	6.63	130.35
58	0.0	231.97	5.76	6.63	130.35
59	0.0	231.97	5.76	6.63	130.35
60	0.0	236.1	8.65	4.98	122.48
61	0.0	233.1	8.69	4.79	119.48
62	0.0	266.08	8.23	5.33	153.56
63	0.0	266.15	8.26	5.31	153.63
64	0.0	237.14	7.08	6.07	128.15
65	0.0	237.14	7.08	6.07	128.15

Table 28: Table of the results obtained with the simulation considering the weather

Annex 3: Table of results with real weather data

Tables of results for boundary calculation

In the following table, the results are obtained using CP Optimizer solver by maximizing the number of requests.

	Total number of requests	Number of requests that can be done	Number of requests Done	Mean data times (in s)
Without weather	299.81	255.0	239.59	45134.97
With weather	299.81	132.3	124.65	51070.62

Table 29: Results with CP Optimizer on average of the 100 generated instances while maximizing the number of requests

In the following table, the results are obtained using CP Optimizer solver by using the lexicographic modeling with a time limit set to one second.

	Total number of requests	Number of requests that can be done	Number of requests Done	Mean data times (in s)
Without weather	299.81	255.0	239.59	45066.1
With weather	299.81	132.3	124.65	50996.41

Table 30: Results with CP Optimizer on average of the 100 generated instances while using the lexicographic approach with a time limit of one second

In the following table, the results are obtained using CP Optimizer solver by using the lexicographic modeling with a time limit set to one minute.

	Total number of requests	Number of requests that can be done	Number of requests Done	Mean data times (in s)
Without weather	299.81	255.0	239.59	45063.68
Without weather	299.81	132.3	124.65	50996.41

Table 31: Results with CP Optimizer on average of the 100 generated instances while using the lexicographic approach with a time limit of one minute

Tables of results for the different combinations using the greedy approach

This association in the following table are the same as defined before in Table 26. The results for each combination are given in the following table.

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in s)
1	182.27	199.77	72.18	72.19	227.63	127.58	58732.26
2	186.18	223.15	69.24	69.32	230.57	153.83	48119.42
3	186.27	219.7	68.76	68.85	231.05	150.85	47876.02
4	212.95	222.78	71.82	71.83	227.99	150.95	63356.46
5	220.71	233.24	71.0	71.01	228.81	162.23	66448.4
6	176.74	205.9	68.13	68.16	231.68	137.74	56622.19
7	178.3	192.96	68.39	68.4	231.42	124.56	60016.01
8	181.4	198.58	70.01	70.03	229.8	128.55	61536.0
9	186.28	223.45	69.26	69.35	230.55	154.1	47960.79
10	186.41	223.72	69.24	69.33	230.57	154.39	48171.78
11	186.11	223.18	69.41	69.5	230.4	153.68	48163.73
12	212.92	222.76	71.84	71.85	227.97	150.91	63359.24
13	212.93	222.76	71.82	71.83	227.99	150.93	63356.46
14	212.95	222.78	71.82	71.83	227.99	150.95	63356.46
15	188.31	205.32	71.25	71.27	228.56	134.05	65083.73
16	195.35	232.6	70.41	70.51	229.4	162.09	61526.86
17	215.67	224.51	71.16	71.16	228.65	153.35	68133.48
18	176.76	205.97	68.06	68.09	231.75	137.88	56618.68
19	176.75	205.9	68.18	68.21	231.63	137.69	56621.15
20	178.3	192.96	68.39	68.4	231.42	124.56	60016.01
21	178.3	192.96	68.39	68.4	231.42	124.56	60016.01
22	180.77	208.94	68.49	68.53	231.32	140.41	62072.85
23	179.29	193.68	68.1	68.11	231.71	125.57	62063.18
24	186.22	223.42	69.19	69.28	230.62	154.14	48106.33
25	186.24	223.41	69.29	69.38	230.52	154.03	47960.59
26	186.41	223.72	69.24	69.33	230.57	154.39	48171.78
27	186.41	223.72	69.24	69.33	230.57	154.39	48171.78
28	186.29	223.38	69.39	69.48	230.42	153.9	48012.58
29	186.43	223.62	69.49	69.58	230.32	154.04	48185.85
30	212.92	222.76	71.84	71.85	227.97	150.91	63359.24
31	212.92	222.76	71.84	71.85	227.97	150.91	63359.24
32	212.92	222.76	71.84	71.85	227.97	150.91	63359.24
33	212.93	222.76	71.82	71.83	227.99	150.93	63356.46
34	212.92	222.76	71.84	71.85	227.97	150.91	63359.24
35	212.93	222.76	71.82	71.83	227.99	150.93	63356.46

	Requests performed	Images performed	Requests fulfilled	Images fulfilled	Requests failed / not done	Images failed	Mean data availability times (in s)
36	188.57	216.53	70.65	70.69	229.16	145.84	65428.4
37	186.63	200.83	69.93	69.94	229.88	130.09	65429.21
38	195.57	232.9	70.44	70.54	229.37	162.36	61555.37
39	195.63	233.09	70.36	70.46	229.45	162.63	61524.99
40	215.64	224.49	71.18	71.18	228.63	153.31	68134.12
41	215.65	224.49	71.16	71.16	228.65	153.33	68133.48
42	176.76	205.97	68.06	68.09	231.75	137.88	56618.68
43	176.83	206.03	68.14	68.17	231.67	137.86	56603.89
44	178.3	192.96	68.39	68.4	231.42	124.56	60016.01
45	178.3	192.96	68.39	68.4	231.42	124.56	60016.01
46	180.81	209.01	68.43	68.47	231.38	140.54	62065.1
47	179.29	193.68	68.1	68.11	231.71	125.57	62063.18
48	186.22	223.42	69.19	69.28	230.62	154.14	48106.33
49	186.23	223.43	69.29	69.38	230.52	154.05	48108.13
50	186.41	223.72	69.24	69.33	230.57	154.39	48171.78
51	186.41	223.72	69.24	69.33	230.57	154.39	48171.78
52	186.32	223.42	69.43	69.52	230.38	153.9	48153.42
53	186.43	223.62	69.49	69.58	230.32	154.04	48185.85
54	212.92	222.76	71.84	71.85	227.97	150.91	63359.24
55	212.92	222.76	71.84	71.85	227.97	150.91	63359.24
56	212.92	222.76	71.84	71.85	227.97	150.91	63359.24
57	212.92	222.76	71.84	71.85	227.97	150.91	63359.24
58	212.92	222.76	71.84	71.85	227.97	150.91	63359.24
59	212.92	222.76	71.84	71.85	227.97	150.91	63359.24
60	188.64	216.65	70.66	70.68	229.17	145.97	65425.29
61	186.63	200.83	69.93	69.94	229.88	130.89	65429.21
62	195.54	232.87	70.41	70.51	229.4	162.36	61563.87
63	195.63	233.09	70.36	70.46	229.45	162.63	61524.99
64	215.64	224.49	71.18	71.18	228.63	153.31	68134.12
65	215.64	224.49	71.18	71.18	228.63	153.31	68134.12

Table 32: Table of the results obtained for each combination on average of the 100 generated instances

The following table provides the results for every combination considering the weather.

	Images M0	Images M1	Images fulfilled M1 failed real	Images fulfilled real failed M1	Images known failed M1
1	0.0	199.77	22.48	18.53	123.63
2	0.0	223.15	13.05	26.45	167.23
3	0.0	219.7	13.0	26.23	164.08
4	0.0	222.78	18.04	28.65	161.56
5	0.0	233.24	24.0	25.43	163.66
6	0.0	205.9	19.95	19.52	137.31
7	0.0	192.96	22.56	18.19	120.19
8	0.0	198.58	23.62	18.57	123.5
9	0.0	223.45	12.97	26.49	167.62
10	0.0	223.72	12.99	26.41	167.81
11	0.0	223.18	13.13	26.35	166.9
12	0.0	222.76	18.04	28.66	161.53
13	0.0	222.76	18.04	28.65	161.54
14	0.0	222.78	18.04	28.65	161.56
15	0.0	205.32	29.44	18.47	123.08
16	0.0	232.6	27.41	19.92	154.6
17	0.0	224.51	22.68	26.75	157.42
18	0.0	205.97	19.93	19.5	137.45
19	0.0	205.9	19.93	19.51	137.27
20	0.0	192.96	22.56	18.19	120.19
21	0.0	192.96	22.56	18.19	120.19
22	0.0	208.94	23.07	19.38	136.72
23	0.0	193.68	23.7	18.17	120.04
24	0.0	223.42	12.99	26.5	167.65
25	0.0	223.41	12.96	26.46	167.53
26	0.0	223.72	12.99	26.41	167.81
27	0.0	223.72	12.99	26.41	167.81
28	0.0	223.38	13.11	26.22	167.01
29	0.0	223.62	13.04	26.28	167.28
30	0.0	222.76	18.04	28.66	161.53
31	0.0	222.76	18.04	28.66	161.53
32	0.0	222.76	18.04	28.66	161.53
33	0.0	222.76	18.04	28.65	161.54
34	0.0	222.76	18.04	28.66	161.53
35	0.0	222.76	18.04	28.65	161.54

	Images M0	Images M1	Images fulfilled M1 failed real	Images fulfilled real failed M1	Images known failed M1
36	0.0	216.53	29.01	19.26	136.09
37	0.0	200.83	29.47	18.07	119.49
38	0.0	232.9	27.44	19.89	154.81
39	0.0	233.09	27.46	19.81	154.98
40	0.0	224.49	22.68	26.76	157.39
41	0.0	224.49	22.68	26.75	157.4
42	0.0	205.97	19.93	19.5	137.45
43	0.0	206.03	19.9	19.47	137.43
44	0.0	192.96	22.56	18.19	120.19
45	0.0	192.96	22.56	18.19	120.19
46	0.0	209.01	23.05	19.37	136.86
47	0.0	193.68	23.7	18.17	120.04
48	0.0	223.42	12.99	26.5	167.65
49	0.0	223.43	12.96	26.47	167.56
50	0.0	223.72	12.99	26.41	167.81
51	0.0	223.72	12.99	26.41	167.81
52	0.0	223.42	13.14	26.25	167.01
53	0.0	223.62	13.04	26.28	167.28
54	0.0	222.76	18.04	28.66	161.53
55	0.0	222.76	18.04	28.66	161.53
56	0.0	222.76	18.04	28.66	161.53
57	0.0	222.76	18.04	28.66	161.53
58	0.0	222.76	18.04	28.66	161.53
59	0.0	222.76	18.04	28.66	161.53
60	0.0	216.65	29.0	19.28	136.25
61	0.0	200.83	29.47	18.07	119.49
62	0.0	232.87	27.43	19.89	154.82
63	0.0	233.09	27.46	19.81	154.98
64	0.0	224.49	22.68	26.76	157.39
65	0.0	224.49	22.68	26.76	157.39

Table 33: Table of the results obtained with the simulation considering the weather

References

[1] Gauthier Picard. "Auction-based and Distributed Optimization Approaches for Scheduling Observations in Satellite Constellations with Exclusive Orbit Portions". In: *International Workshop on Planning and Scheduling for Space (IWPSS'21)*. virtuel, United States, July 2021. URL: https://hal.archives-ouvertes.fr/hal-03247432.

[2] Samuel Squillaci, Stéphanie Roussel, and Cédric Pralet. "Managing complex requests for a constellation of Earth observing satellites". In: *Association for the Advancement of Artificial Intelligence (www.aaai.org)* (2021).

[3] In: URL: https://fr.m.wikipedia.org/wiki/Fichier:ONERA_Toulouse.jpg.

[4] Gauthier Picard et al. "Autonomous Agents and Multiagent Systems Challenges in Earth Observation Satellite Constellations". In: *International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021)*. Proceedings of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2021). Londres, United Kingdom, May 2021, pp. 39–44. DOI: 10.5555/3463952.3463961. URL: https://hal.archives-ouvertes.fr/hal-03181968.

[5] Ferdinando Fioretto, Enrico Pontelli, and William Yeoh. "Distributed Constraint Optimization Problems and Applications: A Survey". In: *J. Artif. Int. Res.* 61.1 (Jan. 2018), pp. 623–698. ISSN: 1076-9757.

[6] Al Globus et al. "A Comparison of Techniques for Scheduling Earth Observing Satellites." In: Jan. 2004, pp. 836–843.

[7] Daniel Delahaye, Supatcha Chaimatanan, and Marcel Mongeau. ""Simulated annealing: From basics to applications"". eng. In: *Handbook of Metaheuristics* (2019), pp. 1–35.

[8] Steve Chien et al. "Using Iterative Repair to Increase the Responsiveness of Planning and Scheduling for Autonomous Spacecraft". In: (Jan. 2000).

[9] Xinwei Wang et al. "Robust scheduling for multiple agile Earth observation satellites under cloud coverage uncertainty". In: *Computers Industrial Engineering* 156 (2021), p. 107292. ISSN: 0360-8352. DOI: https://doi.org/10.1016/j.cie.2021.107292. URL: https://www.sciencedirect.com/science/article/pii/S0360835221001960.

[10] Guansheng Peng et al. "Agile Earth Observation Satellite Scheduling: an Orienteering Problem with Time-Dependent Profits and Travel Times". In: *Computers Operations Research* 111 (June 2019). DOI: 10.1016/j.cor.2019.05.030.

[11] Li Qingsong and Sun Hong. "A Network Flow Model for Job-Shop Scheduling Problem with Time-Windows". In: *Intelligent Computation Technology and Automation, International Conference on* 1 (May 2010), pp. 200–202. DOI: 10.1109/ICICTA.2010.46.

[12] Panwadee Tangpattanakul, Nicolas Jozefowiez, and Pierre Lopez. "A multi-objective local search heuristic for scheduling Earth observations taken by an agile satellite". In: *European Journal of Operational Research* 245 (Mar. 2015). DOI: 10.1016/j.ejor.2015.03.011.

[13] Jintian Cui and Xin Zhang. "Application of a Multi-Satellite Dynamic Mission Scheduling Model Based on Mission Priority in Emergency Response". In: *Sensors (Switzerland)* 19 (Mar. 2019). DOI: 10.3390/s19061430.

[14] Duncan Eddy and Mykel Kochenderfer. "Markov Decision Processes For Multi-Objective Satellite Task Planning". In: Mar. 2020, pp. 1–12. DOI: 10.1109/AERO47225.2020.9172258.

[15] Huimin Zhao. "A multi-objective genetic programming approach to developing Pareto optimal decision trees". In: *Decision Support Systems* 43 (Apr. 2007), pp. 809–826. DOI: 10.1016/j.dss.2006.12.011.

[16] Tarek Chaari et al. "Scheduling under uncertainty: Survey and research directions". In: *2014 International Conference on Advanced Logistics and Transport (ICALT)*. 2014, pp. 229–234. DOI: 10.1109/ICAdLT.2014.6866316.

[17] Jian Wang et al. "An Iterated Local Search Algorithm for The Uncertain Orienteering Problem". In: *2019 IEEE 3rd Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*. 2019, pp. 1065–1070. DOI: 10.1109/IMCEC46724.2019.8983877.

[18] Deepak Karunakaran, Yi Mei, and Mengjie Zhang. "Multitasking Genetic Programming for Stochastic Team Orienteering Problem with Time Windows". In: *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*. 2019, pp. 1598–1605. DOI: 10.1109/SSCI44817.2019.9002804.

[19] Jordan MacLachlan et al. "Genetic Programming Hyper-Heuristics with Vehicle Collaboration for Uncertain Capacitated Arc Routing Problems". In: *Evolutionary Computation* (Nov. 2019), pp. 1–29. DOI: 10.1162/evco_a_00267.

[20] Cyril Briand, Samia Ourari, and B. Bouzouiai. "A Cooperative Approach for Job Shop Scheduling under Uncertainties." In: vol. 176. Jan. 2008, pp. 5–15.

[21] Javad Seif, Mohammad Dehghanimohammadabadi, and Andrew Yu. "Integrated preventive maintenance and flow shop scheduling under uncertainty". In: *Flexible Services and Manufacturing Journal* 32 (Dec. 2020). DOI: 10.1007/s10696-019-09357-4.

[22] J. Browell, I. Dinwoodie, and D. McMillan. "Forecasting for day-ahead offshore maintenance scheduling under uncertainty". English. In: *Proceedings of the European Safety and Reliability (ESREL) Conference, 2016*. European Safety and Reliability Conference 2016, ESREL 2016 ; Conference date: 25-09-2016 Through 29-09-2016. University of Strathclyde, June 2016, pp. 1–8. URL: http://esrel2016.org/.

[23] Da-Yin Liao. "Imaging Order Scheduling of An Earth Observation Satellite". In: *IEEE Transactions on Systems Man and Cybernetics Part C (Applications and Reviews)* 37 (Aug. 2007), pp. 794–802.

[24] Yiyong Xiao et al. "A two-stage flow-shop scheme for the multi-satellite observation and data-downlink scheduling problem considering weather uncertainties". In: *Reliability Engineering System Safety* 188 (2019), pp. 263–275. ISSN: 0951-8320. DOI: https://doi.org/10.1016/j.ress.2019.03.016. URL: https://www.sciencedirect.com/science/article/pii/S0951832018311451.

[25] Jianjiang Wang, Erik Demeulemeester, and Dishan Qiu. "A pure proactive scheduling algorithm for multiple earth observation satellites under uncertainties of clouds". In: *Computers Operations Research* 74 (2016), pp. 1–13. ISSN: 0305-0548. DOI: https://doi.org/10.1016/j.cor.2016.04.014. URL: https://www.sciencedirect.com/science/article/pii/S0305054816300843.

[26] Jianjiang Wang et al. "Expectation and SAA Models and Algorithms for Scheduling of Multiple Earth Observation Satellites Under the Impact of Clouds". In: *IEEE Systems Journal* PP (Jan. 2020), pp. 1–12. DOI: 10.1109/JSYST.2019.2961236.

[27] Jianjiang Wang et al. "Exact and Heuristic Scheduling Algorithms for Multiple Earth Observation Satellites Under Uncertainties of Clouds". In: *IEEE Systems Journal* 13.3 (2019), pp. 3556–3567. DOI: 10.1109/JSYST.2018.2874223.

[28] Jianjiang Wang et al. "Exact and inexact scheduling algorithms for multiple earth observation satellites under uncertainties of clouds". In: *Available at SSRN 2634934* (2015).

[29] Xinwei Wang et al. "Robust Earth Observation Satellite Scheduling With Uncertainty of Cloud Coverage". In: *IEEE Transactions on Aerospace and Electronic Systems* PP (Oct. 2019), pp. 1–1. DOI: 10.1109/TAES.2019.2947978.

[30] Chao Han et al. "Simulated annealing based heuristic for multiple agile satellites scheduling under cloud coverage uncertainty". In: (Mar. 2020).

[31] Eric Bensana et al. "Dealing with Uncertainty when Managing an Earth Observation Satellite". In: Jan. 1999, pp. 120–124.

[32] Adrien HADJ-SALAH et al. "Schedule Earth Observation satellites with Deep Reinforcement Learning". In: *IWPSS 2019*. Berkeley, United States, July 2019. URL: https://hal.archives-ouvertes.fr/hal-02352095.

[33] Adrien HADJ-SALAH et al. "Towards operational application of Deep Reinforcement Learning to Earth Observation satellite scheduling". working paper or preprint. Aug. 2020. URL: https://hal.archives-ouvertes.fr/hal-02925740.

[34] Guillaume Povéda et al. "Evolutionary approaches to dynamic earth observation satellites mission planning under uncertainty". In: July 2019, pp. 1302–1310. ISBN: 978-1-4503-6111-8. DOI: 10.1145/3321707.3321859.

[35] Alex Elkjær Vasegaard et al. "Multi Criteria Decision Making for the Multi-Satellite Image Acquisition Scheduling Problem". In: *Sensors* 20 (Feb. 2020), p. 1242. DOI: 10.3390/s20051242.

[36] Wang Jianjiang, Hu Xuejun, and He Chuan. "Reactive scheduling of multiple EOSs under cloud uncertainties: Model and algorithms". In: *Journal of Systems Engineering and Electronics* 32.1 (2021), pp. 163–177. DOI: 10.23919/JSEE.2021.000015.

[37] Gu Yi et al. "Mission Re-planning For Multiple Agile Earth Observation Satellites based on Cloud Coverage Forecasting". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* PP (Dec. 2021), pp. 1–1. DOI: 10.1109/JSTARS.2021.3135529.

[38] Météo-Contact. "Les modèles météorologiques ". In: URL: https://www.meteocontact.fr/pour-aller-plus-loin/les-modeles-meteo.

[39] Jan Dutton. "The Difference Between Deterministic and Ensemble Forecasts ". In: World Climate Service, Dec. 2021. URL: https://www.worldclimateservice.com/2021/10/12/difference-between-deterministic-and-ensemble-forecasts/.

[40] Guansheng Peng et al. "An Exact Algorithm for Agile Earth Observation Satellite Scheduling with Time-Dependent Profits". In: *Computers Operations Research* 120 (2020), p. 104946. ISSN: 0305-0548. DOI: https://doi.org/10.1016/j.cor.2020.104946. URL: https://www.sciencedirect.com/science/article/pii/S0305054820300630.

[41] Camila Mejıa Quintero, M. Jaramillo, and Juan Carlos Agudelo Agudelo. "Heuristic and exact solution strategies for the Team Orienteering Problem". In: 2016.

[42] Catarina Dudas et al. "Integration of data mining and multi-objective optimization for decision support in production system development". In: *International Journal of Computer Integrated Manufacturing* 27 (July 2013), pp. 824–839. DOI: 10.1080/0951192X.2013.834481.

[43] Hang Yang, Simon Fong, and Yain-Whar Si. "Multi-objective Optimization for Incremental Decision Tree Learning". In: *Data Warehousing and Knowledge Discovery*. Ed. by Alfredo Cuzzocrea and Umeshwar Dayal. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 217–228.

[44] Tom M. Mitchell. *Machine Learning*. New York: McGraw-Hill, 1997. ISBN: 978-0-07-042807-2.

[45] James F. Lutsko and Bart Kuijpers. "Simulated annealing in the construction of near-optimal decision trees". In: *Selecting Models from Data*. Ed. by P. Cheeseman and R. W. Oldford. New York, NY: Springer New York, 1994, pp. 453–462.