Building Flexible Download Plans for Agile Earth-Observing Satellites

A. Maillard, G. Verfaillie, C. Pralet, J. Jaubert, T. Desmousceaux

Building Flexible Download Plans for Agile Earth-Observing Satellites

Adrien Maillard[1,2,3], Gérard Verfaillie[1], Cédric Pralet[1],
Jean Jaubert[2], Thierry Desmousceaux[3]

[1]ONERA - The French Aerospace Lab, FR-31055 Toulouse, France
e-mail : surname.name@onera.fr

[2]CNES, Toulouse, France
e-mail : surname.name@cnes.fr

[3]Airbus Defence and Space, Toulouse, France
e-mail : surname.name@astrium.eads.net

Abstract

We consider the problem of downloading observations for a next-generation agile Earth-observing satellite. The goal is to schedule file downloads during ground reception station visibility windows while minimizing information age and promoting the fair sharing of the satellite between users. It is a complex scheduling problem with constraints ranging from unsharable resources to time-dependent processing times. Usually, planning and scheduling are done on the ground but in our case, data volumes are unpredictable and we propose to share decision-making between the ground and the onboard software, which allows us to take decisions knowing the data volume onboard while keeping a fair level of predictability on the ground. We develop several levels of flexibility, a decision-making architecture and compare these approaches on realistic scenarios.

1 Introduction

Earth-observation satellites are equipped with optical instruments to produce images of the Earth. These images are stored on board and then downloaded when the satellite can communicate with a ground reception station. The use of sophisticated onboard compression algorithms makes the amount of data resulting from an observation very variable. Until now, time-stamped data download plans are built on the ground, sent to the satellite and executed on board without any change. The current way of dealing with the data volume uncertainty is to consider maximum volumes. This makes the data download plans always consistent but also very sub-optimal since data volumes are often lower than maximum. On one hand, satellites are not continuously accessible by a ground station and generated volumes are known only on board and difficult to estimate. This leads to think that decisions about downloads should be made on board. On the other hand, predictability is a key aspect when considering critical systems and users who have made a request for an image may want to know when data will be downloaded.

In this work, which is part of a joint study between CNES, ONERA and Airbus Defence and Space, whose aim is to define a new generation of agile Earth-observing satellites (following the currently operational Pléiades satellites), we consider a decision-making process that is said to be *flexible* because decisions leading to an executable download plan are shared between ground and board which allows us to make decisions knowing the data volume onboard while keeping a fair level of predictability on the ground. We consider several forms of flexibility in scheduling : temporal, resource assignment, execution mode [1]. We do not consider flexibility in observation plans because the available computing power and time is not sufficient onboard to tackle some of the needed calculations. We assume that an observation plan is computed beforehand on the ground.

The remainder of this paper is organised as follows. First, we present details of the downloading problem including its scheduling constraints and the optimization criterion. Then, looking at some of the uncertainties we face, we express our motivation for flexible scheduling. We describe the decision-making architecture and several flexible scheduling approaches which differ in their degree of sharing between board and ground. We then compare these flexible approaches on realistic 24-hour scenarios provided by our industrial partner.

2 Data Download Planning Problem

We consider a next-generation low-orbit agile Earth-observing satellite. The satellite is *agile*, meaning that it is able to move quickly around its gravity center along its three axes while moving along its orbit, thanks to gyroscopic actuators. It is equipped with a body-mounted optical instrument. To observe a ground area with the instrument, the satellite must be pointed to it.

Figure 1 shows how data is acquired, encrypted, stored and downloaded. The data download problem that we consider is a hard scheduling problem with constraints ranging from unsharable resources to time-dependent processing times and specific constraints such as data encryption. In this part, we describe informally the constraints of the problem. The same problem has been described in [9].

2.1 Problem description

Data acquisition Each acquisition activates a subset of detector lines, allowing a multi-frequency observation and resulting in a set of data files, each one recorded in one memory bank. The size of one file depends on the compression rate applied.

Data downloading Data downloading can use several concurrent emission channels. Each file can be downloaded using one channel. Interrupting a file download is not acceptable. The data downloading rate is a piecewise constant function of the satellite-station distance. Due to the movement of the satellite on its orbit and of the Earth on itself, the satellite-station distance evolves and hence the download duration of a file depends on the time at which download starts. Files resulting from an acquisition can be downloaded in any order using any channels, but must be all downloaded within one visibility window. Channels and memory banks are unsharable resources. This means that, if two files are recorded on the same memory bank or use the same channel, their downloads cannot overlap.

Data encryption Data is encrypted before download. One encryption key is associated with each user. There is one physical encryption component for each download channel, allowing data associated with several users to be downloaded concurrently. The operation of changing a key on a component (to switch user) is immediate but needs to be written as an entry in a global key change table. The number of entries it is possible to record in this table is limited. Making changes in this table takes a small amount of time, typically 2 seconds and requires stopping all downloading activities.

Download windows Data downloading is only possible within visibility windows. Moreover, depending on the user, only some stations (and thus some visibility windows) are allowed for data download. We assume that a visibility window may involve several download windows, each one with an associated key change table. Hence, between any two successive download windows, a minimum time is necessary to reset the key change table. Moreover, if the successive download windows are associated with two different stations, moving from the first to the second takes some time to point the download antenna towards the new station. This transition duration depends on the time at which transition starts, due to the movement of the satellite on its orbit and on itself and to the movement of the Earth on itself.

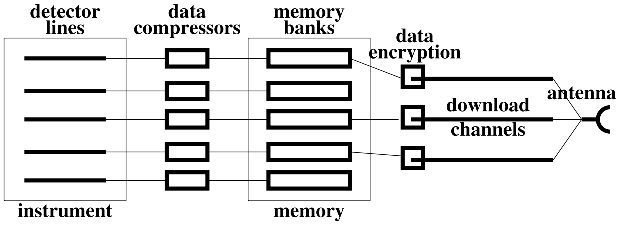


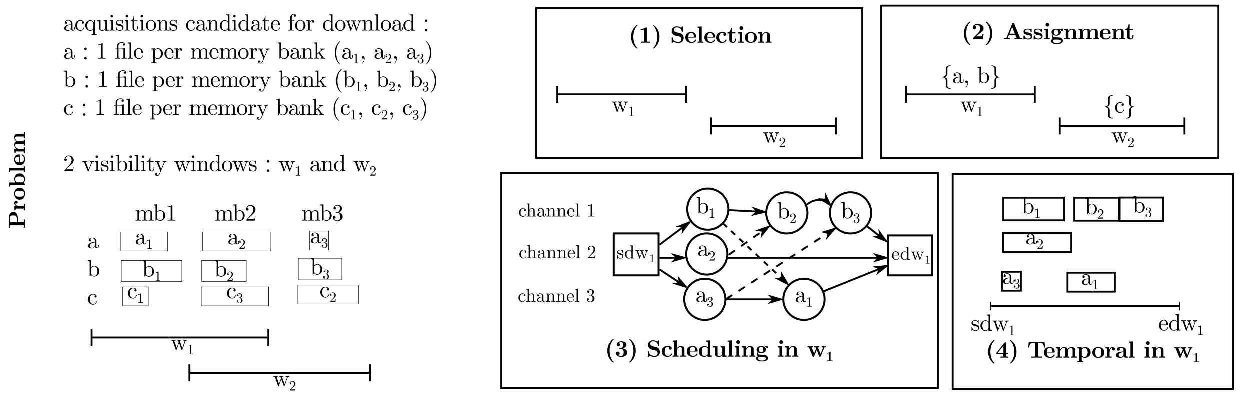
Figure 1. : How the data is acquired, stored, encrypted and downloaded

Download release and due dates Every acquisition download must start after acquisition and finish before a delivery deadline beyond which data is no longer valuable.

2.2 Decomposing the deterministic data download problem

The problem we face combines four connected problems :

1. a **sequencing** problem : the goal is to compute a sequence of non-overlapping download windows from the set of overlapping visibility windows.
2. an **assignment** problem : the goal is to allocate every observation to a download window. This part can be seen as a variant of the *Multiple-Knapsack Problem* where objects are the observations and sacks are the download windows. Because we have a sequence of floating download windows, sizes of the sacks are variable.
3. a **scheduling** problem : the goal is to compute a schedule of the file downloads on each download window (sequences on every memory bank and every channel). Given the constraints seen above, this subproblem can be seen as a *Flexible Open-Shop Scheduling* problem with two types of unsharable resoures : channels and memory banks. Each file download requires one resource of each type, but the choice of the channel is free, whereas the bank is pre-allocated.
4. a **temporal problem** : when the other subproblems are solved, the resulting problem has the form of a *Simple Temporal Network*. The only exceptions are the constraints of download duration and of transition between download windows which are time dependent (download and transition durations depend on the time at which they start). The result is a Time-dependent STN (TSTN) [7] for which STN techniques can be extended and polynomial algorithms



acquisitions candidate for download :
a : 1 file per memory bank (a1, a2, a3)
b : 1 file per memory bank (b1, b2, b3)
c : 1 file per memory bank (c1, c2, c3)
2 visibility windows : w1 and w2
Problem
mb1
mb2
mb3
(1) Selection
(2) Assignment
{a, b}
{c}
w1
w2
channel 1
channel 2
channel 3
sdw1
edw1
(3) Scheduling in w1
(4) Temporal in w1

Figure 2. : Illustration of the four subproblems on an example. There are 3 observations $\{a, b, c\}$ to be downloaded in 2 visibility windows $\{w_1, w_2\}$. The Scheduling and Temporal steps deal only with download window w_1.

can decide on consistency/inconsistency and compute the earliest/latest times for all the temporal variables.

An example illustrating all four subproblems can be seen Figure 2.

2.3 Criterion

Resource allocation is a major problem in economics and computer science in which several types of resources have to be distributed among several agents. The problem of allocating Earth-observing satellites resources has been studied in [5]. In our case, agents are the clients who have invested in the system. They may be the military or civil users. The criterion includes the user preference ("among my observations $(o_1, ..., o_n)$, o_i is prefered to o_j"). For that, each user assigns a weight $\mathbf{W}_a$ and a priority $\mathbf{P}_a$ to each of its requested observation a. Also, each user must have a right to use the satellite in regards to its participation.

The criterion presented below is a simplified criterion which does not include the objective of fair-sharing. It is defined as a vector of utilities, one per priority level. At each priority level p, the utility $\mathbf{U}_p$ is simply defined as the sum of the weights of the downloaded acquisitions of priority p, weighted by their freshness coefficient $\mathbf{Fr}$, in order to favour short delays between acquisitions and downloads :

$$\forall p \in [1; \mathbf{Np}] : \mathbf{U}_p = \sum_{a \in [1;\mathbf{Na}] | \mathbf{P}_a = p} \mathbf{W}_a \times \mathbf{Fr}_a(\mathbf{eda}_a) \quad (1)$$

where $\mathbf{Np}$ is the number of priority levels, $\mathbf{Na}$ the number of acquisitions, $\mathbf{Fr}_a$ the freshness level of a (between 0 and 1) is a monotonically decreasing function of the ending time $\mathbf{eda}_a$ of the download of a. Two download plans are compared by comparing the two associated utility vectors lexicographically from priority 1 to $\mathbf{Np}$: any improvement at any priority level is preferred to any improvement at lower priority levels.

3 Planning strategies in face of uncertainty

Data-download problems in space applications have already been considered [11] [2] but are very dependent on the considered system. More generally, ways of solving scheduling problems under uncertainty have been studied [10].

3.1 Sources of uncertainty

Satellites operate in a changing and dynamic environment. Several uncertainties impact the data download. Optical instruments and data emission antennas consume a significant amount of electrical power but power production and consumption on board is hard to predict exactly on the ground when planning observations and downloads.

The data link rate between ground and board depends on the distance between the satellite and the station but also of the atmospherical conditions (for some bandwiths) which are difficult to approximate on the ground. Thus, the data link rate may vary during a download window.

The uncertainty we will focus on concerns the data volumes. The onboard software uses sophisticated compression algorithms that compress the parts of the image containing clouds. Once again, the presence of clouds above a zone is difficult to anticipate on the ground. This makes the amount of data recorded on board very unpredictable and motivates flexible scheduling architecture. We will now see different ways to deal with such an uncertainty.

3.2 A ground approach

In this type of application, it is important to ensure that the satellite will always have an executable plan. One way of achieving that is by scheduling off-line on the ground with margins [4]. We remove uncertainties from the problem and tranform it into a deterministic problem. We set the volume to the maximum possible. In this case, any plan executable on the ground will be executable on board. The drawback is that these plans are far from optimal because volumes are frequently less than maximum.

We could also use a robust off-line algorithm on the ground that will produce one or several schedules maximizing an expected profit. In this case, we may have a model of our uncertainty or sample scenarios. But the schedule can become inconsistent when on board due to bigger volumes than expected. We will then need an onboard repairing procedure to restore consistency.

Another way of taking into account all possible outcomes is to synthesize a policy (with *Markov Decision Processes* for example). A policy specifies an action to perform for every possible state of the system. Unfortunately, in our case, it would be difficult to compute such a policy regarding the size of the state space.

3.3 A board approach

An onboard software can be in charge of producing schedules. In [3], acquisition requests may be generated either by ground users, or autonomously on board following the detection of ground phenomena such as volcanic eruptions, floods, or ice breakups by onboard data analysis algorithms. In such a context, acquisition and download plans are built on board using an iterative repair approach. In [9], authors have developed several greedy and local search algorithms for tackling the download scheduling problem described in the present paper and showed that an onboard scheduling increases the number of downloads and the information freshness compared to a full ground scheduling. Unfortunately, this approach, by taking all decisions on board, lacks predictability, decisions taken by the onboard software are known on the ground only after execution.

3.4 A flexible approach

In a critical application such as satellite operations, predictability of the system behaviour is a key aspect. While a pure ground-based approach lacks quality, a pure onboard approach lacks predictability. This is what motivates a flexible scheduling approach where the decision-making is shared between an off-line phase on the ground and an online phase on board. Such an approach allows to make use of the computing time and power on the ground and to maintain a certain level of predictability while making decisions about the problem when the uncertain data is revealed on board.

We define a flexible schedule as a entity composed of :

1. A **flexible** partially-instantiated plan that solves one or several of the subproblems.
2. A **contract** defined as a commitment coming with a *flexible* schedule. It renders the need for predictability and has to be fullfilled by the onboard algorithm when producing a consistent and then an *executable* schedule. A contract may cover all the observations in a plan or a subset of them. For example, a contract a_1 can cover only the top priority observations and another contract a_2 cover for the other observations. We define here three possible contracts :
   (a) *"Observation a must be downloaded during visibility window w"*. Ensuring this contract requires not only to assign the observation to a download window but also to provide the scheduling of all its files in the download window.
   (b) *"Observation a must be downloaded during visibility window w or earlier"*. This contract is more flexible than the previous one, it allows to move forward observations if possible.
   (c) *"The download of observation a must be finished by time t"*. This is the strongest contract and requires to send a fully instanciated schedule on board.
3. Optionnally, an **heuristic** guiding the onboard procedure in solving other subproblems to produce an executable schedule from the flexible schedule.

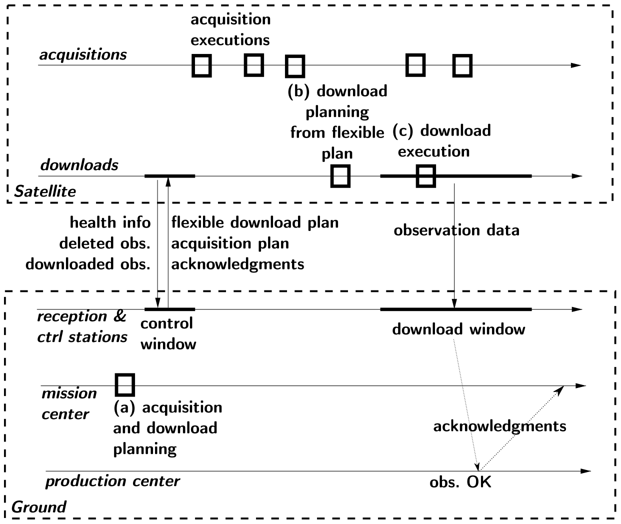


Figure 3. : Decision-making architecture

4 A flexible decision-making architecture

This part describes communications between the ground stations and the satellite. It also defines three scheduling phases which are different in terms of periodicity, uncertainty and computing power available. Fig. 3 shows the flexible decision-making architecture we assume.

4.1 Communication between ground and board

For the sake of simplicity, we omit some data exchanges. There are two types of ground stations, control stations and reception stations. A station can be of one or two types. During visibility windows the communication between the ground stations and the satellite is possible. Here are the data exchanges happening during these visibility windows :

— if it is a control window, the satellite downloads its current state (health check), the list of executed observations, the list of deleted observations and the list of downloaded observations. The station uploads a new observation plan and a new flexible download plan. Because these plans have been computed before the control window, data downloaded by the satellite from the previous control window has not been taken into account.

— if it is a reception window, observations are downloaded by the satellite.

When an observation is downloaded to a reception station, it is transfered to the production center where it is transformed into a user product. A signal called *acknowledgment* is sent to the mission center to confirm the reception of the observation. Then, it is sent onboard to allow the satellite to delete the observation from its internal memory. Acknowledgments also allow the ground scheduler to remove the observation from the pool of observations to be downloaded. The signal can take some time to arrive to the mission center because of the ground network. This leads to inconsistencies between the onboard state and the ground state. For example, assume that the satellite has downloaded observation o_1 on station s_1, but the signal is not yet arrived at the mission center when the next download planning happens. o_1 may then be scheduled in the next flexible download plan. More generally, there is an uncertainty about the state of observations (acquired, canceled, deleted and downloaded) on the ground. Because of these inconsistencies, we need to have a decision rule onboard. Here, the onboard software is optimistic and we consider that if a download has been performed, it has succeeded. This way, when the onboard software gets a new download request for an observation that has been already downloaded, this request is ignored.

4.2 Scheduling phases

We define three different scheduling phases (see Figure 3) to which we refer later :

(a) An off-line phase on the ground. This planning phase happens every 8 hours typically. During this phase, the acquisition plan and the *flexible* download schedule are produced. They are then sent onboard during the next control station visibility window. When received, the acquisition plan is instantly executed. At this moment, only the volumes of observations that have been acquired in the past and communicated to the ground during previous control station visibility windows are known. Thus, during this phase on the ground, we have computing time and power but a high uncertainty about data volumes.

(b) A *deliberative* off-line phase on board. During this phase happening before every group of overlapping visibility windows, a *consistent* schedule is first produced. This schedule ensures all constraints in a format allowing modifications to be made. Then, an *executable* schedule is produced from the consistent schedule. The scheduling horizon depends on the computing power available on board but it is typically small. Only the volumes of the observations finishing during the scheduling horizon are unknown.

(c) A *reactive* online phase on board in the course of which the download plan is executed. Reactivity is needed in this phase and changing a schedule during this phase is typically done by applying immediate decision rules.

5 Flexible approaches

In this part, we describe several levels of flexibility for the download problem (see Table 1). They vary depending on the decisions postponed on board. We highlight the advantages and drawbacks of each approach.

5.1 Time-Stamped Ground Scheduling (TSGS)

Time-stamped is the reference approach in which all subproblems are solved on the ground. Downloads have all a start date. There is no onboard procedure able to modify or repair the plan in case of inconsistency. To ensure that the schedule remains consistent whatever real volumes are, only maximum volumes are considered on the ground.

5.2 Time-Flexible Ground Scheduling (TFGS)

On the ground, a complete schedule with temporal flexibility is produced. On board, an algorithm instanciates a consistent plan by propagating earliest start dates in the resulting TSTN. An example of such instanciation is showed on Figure 2. In comparison of the **TSGS** approach, this approach allows repairing or modification of the plan by an onboard procedure like MoveForward (see section 5.7) and flexible execution (see section 6.2).

5.3 Ground Sequencing and Assignment + Board Scheduling (GSA+BS)

In this approach, the **Sequencing** and **Assignement** subproblems are solved on the ground is provided to the onboard software. The flexible plan contains the sequence of download windows and for each observation, a download window. Onboard, a greedy algorithm schedules files in the given download window for each observaton. The difficulty of scheduling files onto channels and memory

Decision	Approach					
	(ABS) Autonomous Board Scheduling	(BS) Board Scheduling w/ Priority Ordering	(GS + BAS) Ground Sequencing + Board Assignment and Scheduling	(GSA + BAS) Ground Sequencing and Assignement + Board Scheduling	(TFGS) Time-Flexible Ground Scheduling	(TSGS) Time-Stamped Ground Scheduling
Download windows sequencing (1)	•	•	◦	◦	◦	◦
Assignment (2)	•	•	•	◦	◦	◦
Scheduling (3)	•	•	•	•	◦	◦
File download start dates (4)	•	•	•	•	•	◦

Table 1. : Flexible approaches for the data download problem. ◦ decision made on the ground ; • decision made on board.

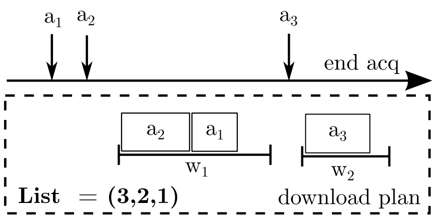


Figure 4. : Building a download plan from a priority list, we do not show individual files here but observations.

banks on the ground, without any information about their real volumes, motivates this approach.

5.4 Ground Sequencing + Board Assignment and Scheduling (GS+BAS)

On the ground, only the sequence of non-overlapping download windows is computed. **Assignment** and **Scheduling** are done on board and no guarantees is given. On board, there is a greedy algorithm, similar to those used in the Full **Autonomous Board Scheduling** approach. The large space of possible window sequences and thus the needed computing time to produce such a sequence motivates this approach.

5.5 Board Scheduling with priority ordering (BS)

On the ground, a priority list, ordering observations, is produced. On board, a non-chronological priority-based greedy algorithm produces a schedule from the list by removing the first element of the list at each step and allowing the earliest resources to it. Actually none of the subproblems are solved on the ground. The list is a heuristic for the onboard algorithm to allocate resources, it is an insertion ordering, not a temporal ordering. There are no guarantees with this approach. An example can be seen on Figure 4. With this approach, there is no need to take and compute the criterion on board, as the order between observations is provided, which leaves more computing time for scheduling.

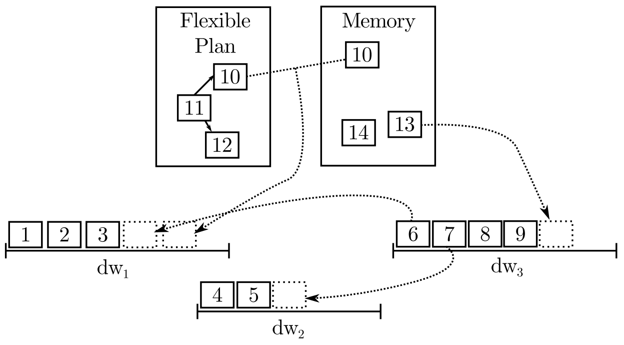


Figure 5. : MoveForward procedure example

5.6 Autonomous Board Scheduling (ABS)

In this approach, nothing is performed on the ground concerning the download schedule. Greedy and local search algorithms have been developed for this approach in [9]. In this approach, there are far less uncertainties about volumes than on the ground, only the volume of observations ending during the scheduling horizon are still unknown. Because of the onboard computing power, only greedy-derived algorithms are allowed and this impacts the quality of schedules. Also, there is no predictability at all on the ground.

5.7 MoveForward procedure (MF)

This onboard procedure takes any consistent plan produced onboard and can only improve its quality by moving forward or inserting downloads. It applies for approaches that compute the **Assignement** subproblem on the ground only because in any more flexible approach, the assignment would have been decided on board during the deliberative phase. There are three types of observations onboard :

— Those appearing in the consistent plan produced on board, these observations will be downloaded during the scheduling horizon (typically one or several download windows).

— Those not appearing in the consistent plan but present in the flexible download plan (which is produced on the ground and has a greater horizon than the consistent plan), these observations will be downloaded eventually.

— Those not appearing neither in the consistent plan nor in the flexible plan but in memory at the moment.

The algorithm can perform two different operations :

— Move forward observations from the first set to earlier download windows.

— Insert observations from the second and third set into the current consistent plan.

An example is displayed on Figure 5. Observations 1 – 9 are in the current schedules (channels are not represented), 10 – 12 are in the flexible plan and 13 – 14 are in memory but not in the flexible plan. The algorithm starts by moving forward obs. 6 and 7 to $dw1$ and $dw2$. Then it proceeds to insert the only observation both in the flexible plan and in memory at the time of the scheduling, obs. 10, in $dw1$. Finally, it inserts obs. 13 in $dw3$.

Sometimes, it is not a good idea to move forward observations in earlier download windows because of network bottlenecks on the ground (see section 4.1). Indeed, observations represent a large volume of data and ground stations receive data from several satellites. It can make the ground network between stations very busy and postpone delivery to the mission center. This issue can be adressed by, for example, analyzing the network load when solving the **Assignement** subproblem.

6 Experimental results

6.1 Constraint checking and propagation

The InCELL library (*Invariant-based Constraint EvaLuation Library* [8]) is used to model variables, constraints and criterion, to check non temporal constraints, to propagate temporal constraints and to evaluate the optimization criterion. InCELL draws its inspiration from *Constraint-based Local Search*. InCELL manages a Directed Acyclic Graph of *invariants* (one for each variable) that can be easily and incrementally re-evaluated (in a topological order) as the values of the variables change during local moves.

6.2 Execution

In the chosen decision-making architecture, schedules produced during the off-line phase on board assume real volumes for past observations and maximum volumes for observations to be completed in the scheduling horizon. This allows the schedule to always stay consistent when the real volumes are known. If actual volumes are smaller than maximum, it is however possible to start downloads earlier than expected in the plan and thus to improve on the plan quality without modifying scheduling decisions.

To implement such a flexible reactive way of executing schedules, we draw our inspiration from the Partial Order Schedule approach (POS [6]) and build from any download plan an execution precedence graph (a Directed Acyclic Graph) which represents all the precedences that must be met by execution. Once this precedence graph has been built, it can be executed in a topological order ; any node is executed as soon as all its predecessors in the graph have been executed. It can be easily shown that, because of the maximum volumes taken into account when scheduling, such an execution never leads to violation of the visibility window and download deadlines. Only the **Time-stamped** approach is not concerned by this flexible execution.

6.3 Scenarios

The flexible decision-making architecture has been experimented on two realistic scenarios provided by Airbus Defence and Space. Each scenario covers 24 hours of satellite activity. In these scenarios, the number of memory banks is equal to 5, the number of channels is equal to 3, the number of users is equal to 5, the number of priority level is equal to 2, the number of ground stations varies from 3 to 23 and the number of associated visibility windows vary from 20 to 115, the number of acquisitions is equal to 1364. If Vmax is the maximum volume of a file (without any compression), its actual volume is randomly generated between Vmax/4 and Vmax. Scenario 1 is a typical civil scenario in which there is a subsequent amount of reception stations located all around the globe, resulting in many download opportunities. Scenario 4 is a defence scenario and involves a small number of ground stations. This scenario is strongly oversubscribed. The volume of data that has to be downloaded is far more important than the one the satellite can manage.

6.4 Algorithms

To produce flexible plans on the ground, we use dedicated algorithms to solve each subproblem. For example, to get a sequence of download windows on the ground, there is a local search algorithm which evaluates a given sequence by computing the optimum flow on a relaxed version of the problem.

6.5 Comparison between timed and flexible planning

We focus our comparison on the number of downloaded observations (the higher the better) which is a quantity criterion and the mean age of information (the lower the better) which is a quality criterion. It is difficult to translate predictability in a criterion but it is important to keep this aspect in mind when comparing approaches. Experiments do not include the all the approaches as they are still under development.

Results can be seen in Table 2. Scenario 1 is not particularly oversubscribed, this is why we observe that the

	Scenario 1 : number of downloaded acquisitions				
	ABS	BS	TFGS + MF	TFGS	TSGS
prio. 1	267	269	269	267	267
prio. 2	947	949	942	908	909

	Scenario 4 : number of downloaded acquisitions				
	ABS	BS	TFGS + MF	TFGS	TSGS
prio. 1	244	244	244	244	244
prio. 2	636	527	592	141	132

	Scenario 1 : mean information age (seconds)				
	ABS	BS	TFGS + MF	TFGS	TSGS
prio. 1	5357	5907	5456	5469	5509
prio. 2	1482	2104	1776	2729	2815

	Scenario 4 : mean information age (seconds)				
	ABS	BS	TFGS + MF	TFGS	TSGS
prio. 1	6622	6788	7063	7129	7303
prio. 2	25889	23213	25582	34605	34343

Table 2. : Compararison between flexible approaches : number of downloaded acquisitions at priority levels 1 and 2 on Scenario 4

number of downloaded acquisitions does not vary a lot. On the contrary, scenario 4 is oversubscribed and we see dramatic changes in the number of downloaded acquisitions regarding the approach.The number of downloaded top priority observations does not vary much as they are always prefered to other observations.

A general comment is that flexibility increases the number of downloads and decreases the mean information age.

Time-Flexible Ground Scheduling, is not better than **Time-stamped Ground Scheduling** in its raw version, i.e. including only flexible execution. This mechanism allows to improve a bit mean information age (less than 100 seconds). However, the **MoveForward** mechanism dramatically improves performances of this approach in both number of downloads and information age.

In addition, we see that **Board Scheduling with priority ordering** is a bit better than the **Autonomous Board Scheduling** approach on information. It is because it has a greater scheduling horizon and because of the smaller number of observations it downloads (527/636).

7 Conclusion

In this paper, we showed how uncertainty about the amount of data generated by observation can be managed by using a mixed architecture where decision-making about downloads scheduling is shared between ground and board. The scheduling problem was analyzed and several levels of flexibility were proposed. Some work remains in developing efficient algorithms that produce *robust* flexible schedules on the ground. Validating these architectures on a real satellite would demonstrate its interest.

References

[1] Jean-Charles Billaut, Aziz Moukrim, and Eric Sanlaville. *Flexibility and robustness in scheduling*. Wiley, 2010.

[2] Amedeo Cesta, Gabriella Cortellessa, Simone Fratini, Angelo Oddi, and Nicola Policella. An innovative product for space mission planning : An a posteriori evaluation. In *Proc. of ICAPS*, pages 57–64, 2007.

[3] Steve Chien, Rob Sherwood, Daniel Tran, Benjamin Cichy, Gregg Rabideau, Rebecca Castano, Ashley Davies, Rachel Lee, Dan Mandl, Stuart Frye, et al. The EO-1 autonomous science agent. In *Proc. of AAMAS*, pages 420–427, 2004.

[4] R. Grasset-Bourdel, G. Verfaillie, and A. Flipo. Action and motion planning for agile earth-observing satellites. *Acta Futura*, 5 :121–131, 2012.

[5] Michel Lemaître, Gérard Verfaillie, Hélène Fargier, Jérome Lang, Nicolas Bataille, and Jean-Michel Lachiver. Equitable allocation of earth-observing satellites resources. In *Proc. of the 5th ONERA-DLR Aerospace Symposium*, 2003.

[6] Nicola Policella, Stephen F. Smith, Amedeo Cesta, and Angelo Oddi. Generating robust schedules through temporal flexibility. In *Proc. of ICAPS*, pages 209–218, 2004.

[7] C. Pralet and G. Verfaillie. Time-dependent simple temporal networks : Properties and algorithms. *RAIRO Operations Research*, 2013.

[8] Cédric Pralet and Gérard Verfaillie. Dynamic online planning and scheduling using a static invariant-based evaluation model. In *Proc. of ICAPS*, pages 171–179, 2013.

[9] Cédric Pralet, Gérard Verfaillie, Adrien Maillard, Emmanuel Hébrard, Nicolas Jozefowiez, Marie-Josée Huguet, Thierry Desmousceaux, Pierre Blanc-Paques, and Jean Jaubert. Satellite data download management with uncertainty about the generated volumes. In *Proc. of ICAPS, To appear*, 2014.

[10] Riccardo Rasconi, Amedeo Cesta, and Nicola Policella. Validating scheduling approaches against executional uncertainty. *Journal of Intelligent Manufacturing*, 21(1) :49–64, 2010.

[11] Gérard Verfaillie, Guillaume Infantes, Michel Lemaître, Nicolas Théret, and Thomas Natolot. On-board decision-making on data downloads. In *Proc. of IWPSS-11*, 2011.