Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling

Ground and Onboard Decision-Making on Satellite Data Downloads

Adrien Maillard and **Cédric Pralet**
ONERA – The French Aerospace Lab
F-31055, Toulouse, France
FirstName.LastName@onera.fr

Jean Jaubert and **Isabelle Sebbag**
Centre National d'Etudes Spatiales
18 avenue Edouard Belin, 31400 Toulouse, France
FirstName.LastName@cnes.fr

Frédéric Fontanari and **Julien L'Hermitte**
Airbus Defence and Space
31 Rue des Cosmonautes, 31400 Toulouse, France
FirstName.LastName@astrium.eads.net

Abstract

In this paper, we show how decision-making on data download plans can be shared between ground and onboard planning to face uncertainty about the volumes of data generated by observations: flexible plans are produced on the ground when volumes are uncertain, together with commitments to high-priority data downloads; these plans are then adapted on board when exact volumes are known, while guaranteeing that ground commitments be satisfied.

This setting arises in a space application where Earth observation satellites perform observations, which produce data to be recorded on board and then downloaded to ground reception stations, but the volume of data generated by observations is uncertain, due to the execution of sophisticated compression algorithms between acquisition and data recording.

1 Introduction

Earth observation satellites are space sensors which acquire data, compress and record it on board, and then download it to the ground using reception station visibility windows. Because of the use of more and more sophisticated compression algorithms, the amount of data that results from an acquisition and thus is recorded on board and must be downloaded to the ground is more and more unpredictable. It depends on the data that has been acquired. For example, in the case of optical instruments, the presence of clouds over the observed area allows high compression rates and results in a low amount of data to be recorded and downloaded.

In such conditions, it is still possible to build data download plans on the ground with the assumption of maximum volumes (minimum default compression rate). The resulting plans are always executable on board, but suboptimal due to an under-use of the station visibility windows.

It is also possible to build data download plans on board just before any set of overlapping station visibility windows with the exact knowledge of the volumes of the already recorded data. This approach has been explored in (Pralet et al. 2014). However, the resulting plans are unpredictable and users cannot know when their data will be delivered. This is problematic especially for high-priority acquisitions.

This is why this paper explores a third option, intermediate between the previous two: flexible data download plans are built on the ground, by considering maximum volumes for the high-priority acquisitions and expected volumes for the others; these plans have the form of a sequence of downloads, each download being a pair made of an acquisition a and of a visibility window w assigned to a; moreover, for any high-priority acquisition a, if w is the visibility window chosen for a in the ground plan, two kinds of commitment are possible: either a will be downloaded in w, or a will be downloaded in w or in an earlier window; on board, these plans are followed as much as possible in a chronological greedy way by removing low-priority downloads when they are unfeasible, by adding or moving forward downloads when this is possible, and by always guaranteeing that commitments to high-priority downloads be satisfied. Another flexible approach exploiting the problem structure was studied in (Maillard et al. 2014). Several decision-making sharing mechanisms were designed, by varying the number of subproblems solved on the ground or on board.

This study is part of a technological program started by the French space agency aiming at developing next-generation agile Earth-observing satellites. At the moment, our work is used for early system design. For instance, it allows system designers to assess whether the number of download channels available on board is sufficient, to have an idea of the number of acquisitions which can be downloaded each day, to assess whether memory capacities proposed in first design phases are sufficient, to determine the impact of the number and repartition of ground stations on the system efficiency, and to determine how much flexible planning can decrease system needs.

Section 2 describes the data download planning problem (physical system, decisions, constraints, and criterion). Section 3 describes how acquisitions are planned and executed. Sections 4 and 5 describe how flexible download plans are

---

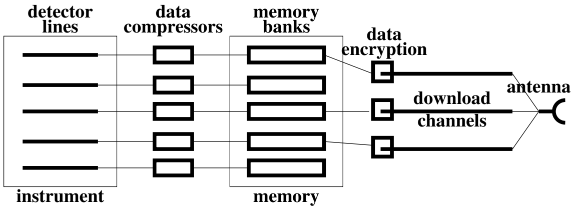


Figure 1: How data is acquired, recorded, and downloaded.

built on the ground, adapted on board and executed. Section 6 presents the results of the experiments that have been performed on real-world scenarios. Finally, Section 7 points out some related works.

2 The Data Download Planning Problem

2.1 Physical System

The space system we consider is made of an agile Earth observation satellite (possibly more than one), several ground satellite control stations, several ground data reception stations, and several data processing centers, each one associated with a satellite user.

Acquisition An agile Earth observation satellite is able to move quickly around its gravity center, along the roll, pitch, and yaw axes, thanks to gyroscopic actuators, while moving along its orbit. It is equipped with an optical observation instrument which is body-mounted on the satellite. Hence, to observe a given ground area, the whole satellite must be trained on it. Acquisition takes some time, as well as moving from an acquisition to the following one in the sequence. Agility provides users with a great flexibility in the choice of the sequences of acquisitions performed by the satellite.

Data Recording Any acquisition activates several instrument detector lines. For each activated detector line, a file is generated, compressed, and recorded in one of the mass memory banks (see Fig. 1). The compression rate depends on the data and cannot be predicted before acquisition. It is only known after recording.

Data Downloading The satellite is also equipped with an orientable emission antenna which allows recorded data to be downloaded to the ground. To download data towards a ground reception station, the satellite must be within one of the station visibility windows and the antenna must be trained on the station. Moving the antenna from one station to another takes some time. This duration depends on the time at which the movement is triggered, due to the movement of the satellite along its orbit and on itself and to the movement of the Earth on itself. Data acquisition and download can be performed concurrently.

Several concurrent channels are available for data emission (see Fig. 1). Any file can be downloaded using any channel. A file download cannot be preempted. The files generated by an acquisition can be downloaded in any order on any channels. If two files are downloaded on the same channel, their downloads cannot overlap: channels are unsharable resources. The same way, if two files are recorded in the same memory bank, their download cannot overlap: memory banks are unsharable resources. The duration of a file download depends on the file size and on the download rate. This rate depends itself on the satellite-station distance and thus on the download time, because of the movement of the satellite along its orbit and of the Earth on itself.

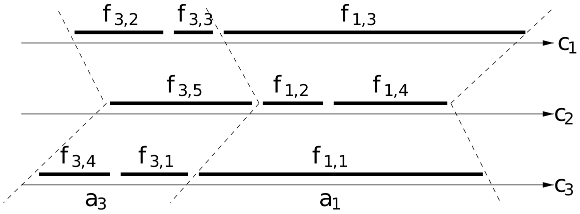


Figure 2: Sequence of downloads of two acquisitions a_3 and a_1 on channels c_1, c_2, and c_3. $f_{i,j}$ is the jth file generated by acquisition a_i.

Data Encryption The system has several users and each acquisition is required by one of the users. An encryption key is associated with each user and data is encrypted before download, independently on each channel (see Fig. 1). A key change table which contains key changes and their precise times is associated with each channel. The number of changes it is possible to record in this table is limited. Resetting this table takes some time and requires that download on all the channels be interrupted.

User Requirements In addition to these physical constraints, some user requirements must be met. First, we require that all the files generated by an acquisition a be downloaded within the same station visibility window and that data download towards this station be allowed by the user who required a. Second, we forbid that file downloads associated with two different acquisitions be interleaved on the same channel. As shown in Fig. 2, this does not forbid that acquisition downloads start and end at different times on different channels.

Data Processing Centers Once received by a station, the data generated by an acquisition a is inserted in a queue of files to be sent via a ground communication network to the processing center of the user requiring a. When data is correctly received by the processing center, an acknowledgement is sent back to the ground satellite control stations and from them to the satellite.

Ground Satellite Control Stations Ground control stations differ from data reception stations. They are used to send to the satellite acquisition and download plans, as well as data acknowledgements, using visibility windows.

2.2 Decisions To Be Made

To build a download plan, decisions must be made at three levels:

1. acquisition download selection, assignment, and scheduling: this consists in deciding which acquisitions will be downloaded and in building a sequence of acquisition downloads, each download being made of an acquisition a and of a reception station visibility window w_a assigned to a;
2. file download assignment and scheduling: this consists in assigning each file to be downloaded a channel and in building on each channel and on each memory bank a sequence of file downloads;
3. time management: this consists either in computing minimum and maximum dates, or in setting precise dates for all the events: start and end of file downloads, of antenna movements, and of key change table resettings.

2.3 Constraints To Be Satisfied

The set of constraints to be satisfied are the following :

- an acquisition is downloaded at most once;
- an acquisition, required by a user u, is downloaded towards a station allowed by u;
- all the files of an acquisition a are downloaded exactly once within the visibility window assigned to a;
- for each channel, there is no interleaving between file downloads associated with different acquisitions;
- for each channel c, there is no overlapping between file downloads assigned to c;
- for each memory bank b, there is no overlapping between downloads of files recorded on b;
- an acquisition download cannot start before acquisition end;
- the download of an acquisition a (download of all its files) must start after the beginning (and finish before the end) of the visibility window assigned to a;
- for each file, there must be a sufficient time to download it, taking into account the file size and the current download rate;
- if two successive acquisition downloads are assigned to two different stations, there must be a sufficient time to move the antenna between the end of the first and the beginning of the second;
- if resetting the key change table is necessary (when the maximum number of changes that can be recorded in the table is reached on one of the channels), there must be a sufficient time to reset it, without any download on all the channels.

2.4 Criterion To Be Optimized

With each acquisition, are associated a user, a priority level, and a weight.

Information age becomes more and more important in Earth observation systems. Hence, we assume that a note can be associated with any acquisition a whose download is planned in a plan P. This note takes into account the age of information of a in P, that is the distance between the end of acquisition of a and the time at which data generated by a will be delivered to the processing center of the user who required a, taking into account both data download and ground data transfer.

Moreover, we assume that the note of a plan P at a priority level p for a user u is the sum of the products weight by note for all the acquisitions that are downloaded and required by u (utilitarian approach for each priority level and each user). Then, the note of a plan P at a priority level p is a function of the notes of P at level p for all the users. Such a function favours the fair sharing of the space system between users (more or less egalitarian approach between users at each priority level (Bouveret et al. 2005)).

Finally, the note of a plan P is simply the vector of its notes at all its priority levels, from the most to the less important. Two plans are compared by comparing lexicographically their associated vectors of notes. This implies that any improvement at a priority level p is always preferred to any improvement at a priority level less than p (hierarchical approach between priority levels).

3 Acquisition Planning and Execution

3.1 Acquisition Planning on the Ground

Acquisition plans are built offline on the ground, as it is usually done until now, some time before any ground satellite control visibility window, over a planning horizon that runs at least from this window to the next. This is justified by the fact that building them is computationally expensive and that all the information about acquisition requests is available on the ground. These plans consider neither memory and download limitations, nor download activities. From the acquisition plan, it is possible to deduce, for any ground reception station s, the windows over which downloading data to s is effectively possible (pointing the mobile antenna towards s is possible, taking into account the satellite position and orientation, and the maximum antenna orientation angle). The acquisition plan and the resulting visibility windows are then uploaded to the satellite.

3.2 Acquisition Execution on Board

These acquisition plans are then executed on board without any modification except when an acquisition a would lead to a memory overflow: in this case, a is performed if and only if it is possible to free enough memory by removing from memory lower priority acquisition data.

4 Building Flexible Download Plans on the Ground

Now, we can present the approach we propose to share decision-making on data downloads between ground and on-board planning. See Fig. 3 for an illustration. For the sake of simplification, we assume only two priority levels: high and low-priority acquisitions.

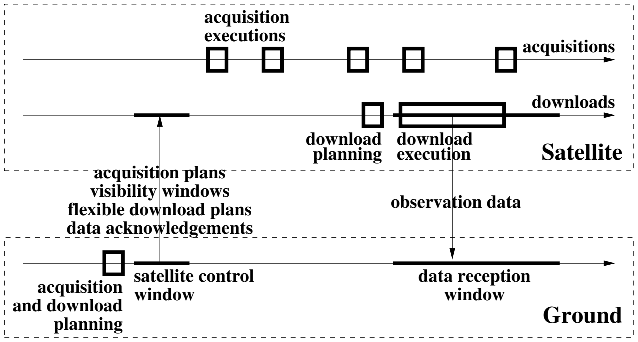


Figure 3: How decision-making is shared between ground and onboard planning.

Flexible download plans are built offline on the ground together with acquisition plans, over the same planning horizon. When planning, maximum volumes are taken into account for high-priority acquisitions, but only expected volumes for low-priority acquisitions.

Due to the huge number of acquisitions and files to be managed (> 1000 acquisitions, > 5000 files), the usage of exact optimal methods for planning downloads must be left out, even with the computational resources available on the ground. Following several experiments, we chose to use an SWO algorithm (Squeaky Wheel Optimization (Joslin and Clements 1999)). The algorithm we use performs a sequence of non-chronological greedy searches. Each greedy search consists in inserting acquisitions one by one into the plan, following an acquisition insertion order O. For inserting each acquisition $a \in O$, a visibility window is selected, as well as a position in the current sequence of downloads, with a preference for pairs window/position that allow either the data delivery date or the download duration to be minimized. Then, all the files associated witg a are scheduled from the largest to the smallest. For each file, a download channel c is selected, with a preference for channels that allow idle times to be minimized. At any step, all the non-temporal constraints are checked and all the temporal constraints are propagated in the current TSTN (Time-dependent Simple Temporal Network (Pralet and Verfaillie 2013b)). TSTN is an extension of the STN framework (Simple Temporal Network (Dechter, Meiri, and Pearl 1991)) which allows not only constant minimum distances between events, but also minimum distances that depend on event times (for example, time-dependent download durations), to be taken into account. If any constraint is violated, another pair window/position is selected for a until all the possible pairs are exhausted. If all of them have been exhausted without any success, a is rejected. All constraint checks and constraint propagations are handled using the InCELL local search library (Pralet and Verfaillie 2013a).

For the first greedy search, acquisitions are ordered in O from the highest to the lowest priority level and, at the same priority level, from the earliest to the latest acquisition end date, in order to favour high-priority acquisitions and to minimize information age. After each greedy search, producing a plan P, insertion order O is updated by moving acquisitions forward in O proportionally to the gap between their age and the mean age of acquisitions of the same priority in P.

The best plan over all the runs according to the optimization criterion (see Sect 2.4) is selected.

Flexible Plan Format To limit the unpredictability of the decisions that will be finally made on board, we introduce some form of commitment to high-priority acquisitions. A commitment c_a is associated on the ground with each high-priority acquisition a, with two possible kinds of commitments:

- if c_a is of the first type, we enforce that a will be downloaded within w_a: the download window may not be changed on board and the user knows the window w_a within which its acquisition will be downloaded;
- if c_a is of the second type, we enforce that a will be downloaded within w_a or within an earlier window: the download window may be changed on board for an earlier one and the user knows the latest window w_a within which its acquisition will be downloaded.

As this will be shown in Sect. 5, considering maximum volumes for high priority acquisitions allows these commitments to be guaranteed. It is the user who chooses which type of commitments he needs for each acquisition.

For a low-priority acquisition a, there is no commitment: the download window may be changed on board for any other window; it is even possible not to download a. Because of the absence of commitment to these acquisitions, considering expected volumes is possible. One can also look at (Bonfietti, Lombardi, and Milano 2014) for an empirical and theoretical justification of such a choice.

From the best plan computed on the ground, the latest starting dates of all the high priority downloads are computed by ignoring low-priority acquisitions. They are recorded to be used later on board (see the next section).

Finally, the flexible download plans that are sent to the satellite are made of two lists:

- a flexible plan $\mathcal{FP}$ which is a list of quadruples $\langle a, w, c, l \rangle$, where a is an acquisition, w a visibility window, c a commitment (with three possible values 0, 1, and 2: 0 for no commitment, 1 for a commitment of first type, and 2 for a commitment of second type), and l a latest starting date in case of commitment ($c \neq 0$); $\mathcal{FP}$ represents the sequence of downloads that has been planned on the ground, as well as associated commitments;
- a complementary list $\mathcal{CL}$ which is a list of acquisitions whose download has not be planned on the ground, but might be planned on board due to volumes being lower than expected; this list is ordered from the most to the less preferred download.

In this paper, two priority levels are considered. In practice, there can be more than two priority levels (for instance, seven priority levels are used for currently active satellites). Although, for commitments, there is always a binary scheme between on one hand high-priority acquisitions concerned by commitments and handled using maximum volume assumptions, and on the other hand low-priority ones (binary distinction between hard constraints and soft constraints). However, it makes sense to have several priority levels among non-prioritary acquisitions, in order to determine the candidate low-priority acquisitions to be considered first for insertion into the plan (acquisitions with the highest priority are selected first).

5 Adapting Download Plans on Board

Executable download plans are built on board some time before any set of overlapping visibility windows (two visibility windows are considered to overlap if they effectively overlap or if they do not, but there is not enough time to plan between them) with this set as a planning horizon. When planning, the exact volumes of the already recorded acquisitions are known and taken into account. Only, the volumes of the acquisitions that will end between the planning start time and the end of the planning horizon remain uncertain. To guarantee that the download plans will always be executable, we consider maximum volumes for all of the latter, whatever their priority levels are.

Standard onboard processors are between 100 and 1000 times less powerful than classical processors available on the ground. Because the non chronological greedy algorithm built on top of InCELL may take several seconds (for one run) on a ground processor and requires approximately 600 MBytes of memory (to store and manage the constraint network), it cannot be used on board. This is why we chose to implement a chronological greedy algorithm without InCELL for onboard download planning, which takes several milliseconds on a ground processor and requires far less memory (no constraint network to be managed).

The onboard algorithm (see Alg. 1) follows in a chronological greedy way the flexible plan $\mathcal{FP}$ that has been built on the ground. The main difficulty is to design a chronological greedy algorithm which guarantees that commitments to high-priority acquisitions be satisfied. This kind of algorithm is blind and some form of look-ahead is necessary to guarantee the satisfaction of hard requirements. Hence, because look-ahead takes time, the challenge is to limit it as much as possible. As we will see, this will be done by using latest starting dates computed on the ground for high-priority downloads.

At each step, the current state of the greedy algorithm is a quadruple $\langle t, s, nd, nc\rangle$ where t is the current time, s the current (simulated) state of the physical system, nd the next download in $\mathcal{FP}$, and nc the next download with commitment in $\mathcal{FP}$.

At each step, the algorithm checks first whether or not inserting the next download nd would lead to an idle period (see the examples of Fig. 4). If there is an idle period, the algorithm tries to take advantage of it and to insert an acquisition download before nd. This download is chosen first in the complementary list $\mathcal{CL}$ (addition to the download plan) and second in the sequel of the flexible plan (forward movement in the download plan). If there is no idle period or if no insertion before nd is possible, it tries to insert nd in the download plan.

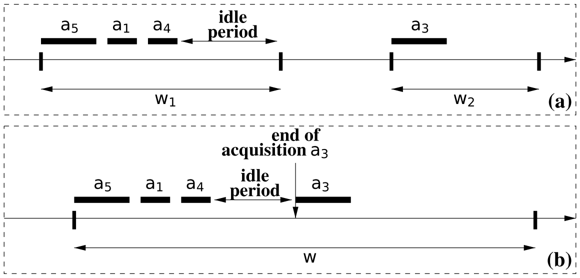


Figure 4: Examples of idle periods in the sequence of downloads: a_3 is the next download in the flexible plan, but one must wait in (a) for the start of visibility window w_2 where the download of a_3 is planned and in (b) for the end of acquisition of a_3.

Algorithm 1 Repair algorithm (main loop)

```
Data : a flexible plan made of $\mathcal{FP}$ and $\mathcal{CL}$; $nd$ : $\langle a, w, c, l\rangle$ the next download in $\mathcal{FP}$; $\mathcal{PB}$ the current onboard plan;
1. if inserting $nd$ immediately would lead to an idle period in the plan:
   (a) try to insert an alternative download from $\mathcal{CL}$ or $\mathcal{FP}$ before $nd$;
   (b) if there is nothing to download, jump to the next event in the horizon.
2. otherwise :
   (a) if $a$ is high-priority :
       i. insert $nd$ in $\mathcal{PB}$;
   (b) else if $nd$ is physically possible and does not violate the next commitment : insert $nd$ in $\mathcal{PB}$;
   (c) otherwise :
       i. if lookahead($nd$) = false then : add $nd$ to $\mathcal{CL}$;
       ii. otherwise insert $nd$ in $\mathcal{PB}$.
```

When trying to insert a download d, the algorithm checks whether or not this is physically possible (see Fig. 5a) and does not impact the future high-priority downloads (see Fig. 5b). The latter check is performed by looking at the latest starting dates of the high-priority downloads that have been computed on the ground based on maximum volume assumptions. Because they took into account maximum volumes, these dates are pessimistic. They are lower bounds on the actual latest starting dates. Fig. 2 shows that, for an acquisition, the end of its last file download on each channel materializes some kind of *boundary* before another acquisition download. Because they exist also for each channel, latest starting dates also materialize a boundary. Checking whether or not a new download can be inserted between an already inserted download and the next high-priority download is checking whether its file downloads *fits* on the channels between these two boundaries. It can be quickly done because a scheduling of these file downloads is given in the flexible plan[1]. If the latest starting date of the next high-priority download nc is not violated, insertion of d is possible. If it is violated, one checks whether or not the sequence

---

[1] For all high-priority acquisitions, the flexible plan actually also contains a way to schedule file downloads on channels and memory banks, since it can be difficult to find on board a schedule meeting the commitments. The flexible plan also contains, for all high-priority acquisitions, latest starting dates on each channel. Such elements are not detailed in the following for the sake of clarity.

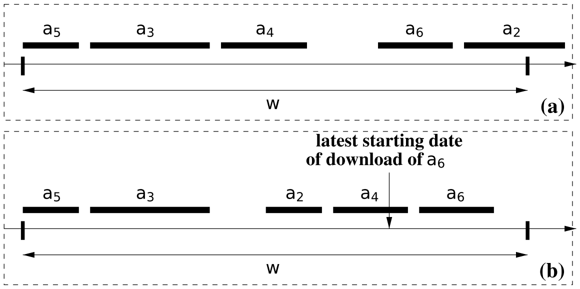


Figure 5: Examples of negative tests when trying to insert download a_2 in window w: violation in (a) of the end of visibility window w and in (b) of the latest starting date of the next high-priority download a_6.

of future high-priority downloads is actually possible, taking into account exact volumes (when they are known) and ignoring low-priority downloads. This look-ahead check is stopped as soon as the latest starting date of the next high-priority download is not violated. In this case, insertion of d is possible. The download d is then inserted and the latest start dates of the acquisitions used during the look-ahead are updated. If the whole sequence is exhausted with violation at every step, insertion of d is impossible. If this look-ahead procedure is too time consuming, it is possible to limit it, for example by considering only the sub-sequence of high-priority downloads in the current visibility window (or even only the next high-priority download): in this case, if the sub-sequence is exhausted with violation at every step, insertion of d is rejected, even if it might be possible.

If insertion of d is possible, the download plan and the current state of the greedy algorithm are updated. If it is impossible, d is added to the complementary list $\mathcal{CL}$ (when it did not already belong to $\mathcal{CL}$).

To take into account the uncertainty that remains about the volumes of the acquisitions that are not already recorded, the plans produced on board are temporally flexible. Hence, a download plan has the form of a directed acyclic graph which represents all the precedences between events (Policella et al. 2004): start and end of acquisition and file downloads, start and end of visibility windows, start and end of antenna movements, and start and end of key change table resettings, end of acquisitions, and key change times.

The precedence graph is then executed with the following policy: any event is activated as soon as all its parents in the graph are executed, until all the events be executed.

This policy may allow acquisitions to be downloaded earlier than expected in the plan. Moreover, due to the exact or maximum volumes taken into account, it is guaranteed that it will never lead to a violation of any temporal constraint.

It can also be easily proven that, because maximum volumes are considered on the ground for high-priority acquisitions, commitments to high-priority downloads will always be satisfied by onboard planning.

6 Experiments

6.1 Compared Approaches

To assess the positive impact of sharing decision-making between ground and onboard planning, we implemented and compared the following four approaches:

1. planning on the ground (`Ground`) : download planning is performed on the ground with maximum volumes for all the acquisitions, using an SWO approach built on top of a non-chronological greedy algorithm; this plan is executed on board without any change;
2. onboard planning (`Onboard`) : download planning is performed on board with exact volumes for the acquisitions that are already finished and maximum volumes for the others, using a chronological greedy algorithm;
3. planning shared between ground and onboard planning with very simple onboard adaptation mechanisms (`SimpleRepair`): download planning is performed on the ground with maximum volumes for the high-priority acquisitions and expected volumes for the others, using the same SWO approach on top of a non chronological greedy algorithm; this plan is then adapted on board, using a chronological greedy algorithm which simply removes downloads when they are physically unfeasible (see Fig. 5a), whatever their priority is;
4. planning shared between ground and onboard planning with intelligent onboard adaptation mechanisms (`SmartRepair`); this is the approach presented in this paper: download planning is performed on the ground with maximum volumes for the high-priority acquisitions and expected volumes for the others, still using the same SWO approach; this plan is then adapted on board, using a chronological greedy algorithm which uses some kind of look-ahead; this algorithm removes low-priority downloads either when they are physically unfeasible or when they lead to a violation of the commitments; it also adds and moves forward downloads when possible; it always guarantees that commitments to high-priority downloads be satisfied.

6.2 Scenarios

We use two realistic scenarios produced by our partner space company. These scenarios cover one day. They involve 5 memory banks, 3 channels, 5 users, 2 priority levels, 3 to 23 ground reception stations, 20 to 115 associated visibility windows, 1364 acquisitions and 6820 files to be downloaded, and actual file volumes randomly generated between $Vmax/4$ and $Vmax$, with $Vmax$ the maximum file volume.

The two scenarios differ from each other according to the number of available ground reception stations : Scenario 1 involves a large number of ground reception stations (23) resulting in many download opportunities, whereas Scenario 2 involves a small number of ground reception stations (3) resulting in only few download opportunities. The two scenarios also differ in the way acquisitions are distributed between users and priority levels: 275 (resp. 247) acquisitions of high priority and 1089 (resp. 1117) acquisitions of low priority for Scenario 1 (resp 2). 15 instances of actual volumes have been randomly generated and all the results are mean values over these 15 instances.

6.3 Evaluation Criteria

Execution results are compared according to several criteria:

1. the number of downloads, at each priority level;
2. the mean information age (see Sect. 2.4) over all the downloaded acquisitions, at each priority level;
3. the criterion described in Sect. 2.4, at each priority level;
4. only for the third and fourth approaches, the differences between ground and onboard plans: number of downloads that are added, removed, moved forward, and moved backward, at each priority level;
5. the onboard computing time consumed by the planning algorithm.

The first three criteria measure the quality of the plans executed on board. The fourth one measures their stability with regard to the plans produced on the ground. The fifth one measures the efficiency of the onboard planning algorithm.

6.4 Comparison between Planning Approaches

Number of Downloads With regard to the number of downloads (see Fig. 6a), all high-priority acquisitions (whichever the scenario) are downloaded except with the `SimpleRepair` approach that does not check whether or not future high-priority acquisitions are endangered when inserting low-priority ones. For low-priority downloads, we can see that the `SmartRepair` approach performs as well as the `Onboard` approach does, downloading all available acquisitions, and much better than the `Ground` and `SimpleRepair` approaches do.

Mean Information Age Significant differences appear only for low-priority acquisitions in Scenario 2 (see Fig. 6b). Mean information age decreases from the `Ground` approach to the `Onboard` one, suggesting that the higher flexibility in onboard decision-making, the lower information age.

Criterion The criterion aggregates user preferences and fair sharing of the system. Fig. 6d shows that for high-priority acquisitions, the `Ground` and `SmartRepair` approaches are better than the `Onboard` approach (neglecting the `SimpleRepair` because it sometimes removes high-priority downloads thus degrading greatly the criterion). We can observe that producing complete or flexible plans on the ground with larger horizons allows to be more accurate on sharing and less myopic than with short horizons. With low-priority acquisitions of Scenario 2, we can see that `SmartRepair` is superior. It shows that combining large horizon ground-based planning with onboard improving operations such as inserting new acquisitions or moving forward acquisitions can greatly improve the performance of the system.

Computing Time Computing resources onboard a satellite are scarce. Unsurprisingly, results in Fig. 6e (the mean computing time on board for the `Ground` approach is too small to be showed) show that the higher flexibility in onboard decision-making, the higher computing time. We must stress that, by using the computing power available on the ground, `SmartRepair` is able to achieve similar or superior performances than `Onboard`, with 1.5× to 3× less computing time on board.

Stability Tables 6c and 6f show the mean number of movements (additions, removals, forward and backward movements of downloads) performed over the one-day scenarios by the two flexible approaches : `SimpleRepair` and `SmartRepair`. Concerning `SimpleRepair`, it must be first stressed that it performs only removals when physical constraints are violated. Second, it can be observed that, because it performs no lookahead, it may remove several high-priority downloads: on average 11 in Scenario 1, and 2 in Scenario 2. On the contrary, because it performs lookahead, the `SmartRepair` approach removes no high-priority download and satisfies commitments.

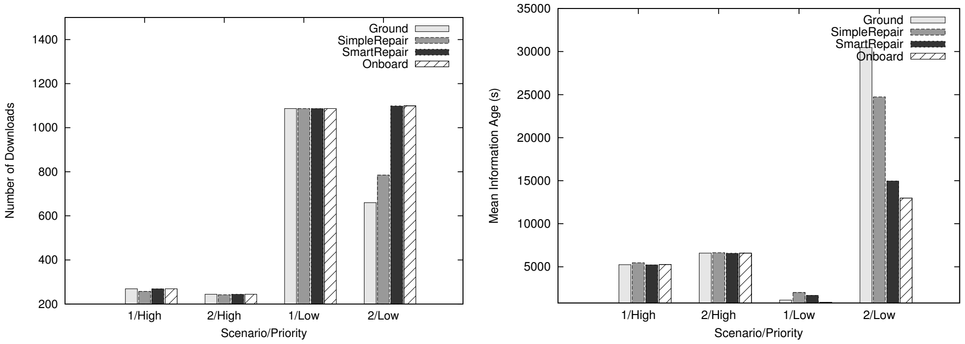
We can see that `SmartRepair` is adding a lot more acquisitions in Scenario 2 than in Scenario 1. It is not surprising because Scenario 1 is a lot less oversubscribed than Scenario 2. In other words, in Scenario 1, the onboard software does not need to insert new downloads into the plan because all possible downloads are already in it. This approach is also moving forward a lot of low-priority acquisitions (162) which can explain why the information age is somewhat close to the age obtained with the `Onboard` approach.

Results confirm the important role of the onboard insertion procedure in the increase of the number of downloads with relation to the `Ground` and `SimpleRepair` approaches.

7 Related Work

The data download planning problem from a space engine (Earth satellite or not) to Earth data reception stations has been addressed in several publications:

- in (Oddi et al. 2003; Cesta et al. 2007), the problem of downloading data from a Mars orbiter (ESA Mars Express mission) to Earth is described; in (Oddi and Policella 2004; Righini and Tresoldi 2010), it is shown that this specific problem can be modeled as a Max-Flow problem and thus solved using either Max-Flow or Linear Programming polynomial algorithms; unfortunately, because of differences in terms of constraints and criterion, a Max-Flow formulation does not fit our problem; in (Oddi and Policella 2004; Righini and Tresoldi 2010), uncertainty about the generated volumes of data is also addressed; methods able to produce plans that are as robust as possible are proposed: however, planning is only performed on the ground and no sharing of the decisions between ground and onboard planning is considered;
- (Chien et al. 2004) describes how acquisition and download plans are built and repaired on board the EO-1

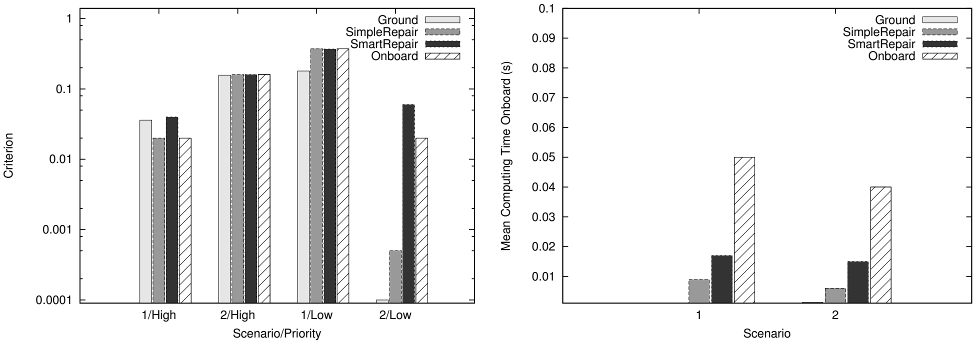


(a) Number of downloads. The higher the better.

(b) Mean information age (in seconds). The lower the better.

	Priority	Simple Repair	Smart Repair
added	High	/	0
	Low	/	3
removed	High	11*	0
	Low	141	124
moved forward	High	/	5
	Low	/	20
moved backward	High	/	0
	Low	/	54

(c) Onboard movements for Scenario 1. Violated commitments are highlighted with *.



(d) Criterion (on a logarithmic scale). The higher the better.

(e) Mean computing time on board. The lower the better.

	Priority	Simple Repair	Smart Repair
added	High	/	0
	Low	/	309
removed	High	2*	0
	Low	72	64
moved forward	High	/	0
	Low	/	162
moved backward	High	/	0
	Low	/	0

(f) Onboard movements for Scenario 2. Violated commitments are highlighted with *.

Figure 6: Evaluation on the four main criteria and stability measures. For each criterion, the four approaches `Ground`, `SimpleRepair`, `SmartRepair` and `Onboard` are compared. $i \in \{1, 2\}$ is the scenario number and $p \in \{\text{High}, \text{Low}\}$ the priority level.

Earth observation satellite, using an iterative repair approach (Chien et al. 2000): local search algorithms implemented in the generic CASPER planning tool; however, the number of acquisitions and files we have to manage (see Sect. 6) is far higher than in the setting of the experimental EO-1 mission;

- the problem of downloading data from electromagnetic Earth surveillance satellites to ground data reception stations, with uncertainty about the volumes of data to be downloaded, is also addressed in (Verfaillie et al. 2011); several algorithms for onboard download planning are proposed and compared; however, our problem is far more complex in terms of constraints to be managed: memory banks, emission channels, encryption keys . . .

More globally, our problem is related to robust offline planning and scheduling and to reactive online planning and scheduling approaches.

In robust offline planning and scheduling, the objective is generally to produce offline plans/schedules that resist as much as possible to possible events, that is plans/schedules that for example maximize the expected gain or the expected value of any optimization criterion, taking into account probability distributions on events. See for example all the works around Stochastic Programming (Birge and Louveaux 1997) and the specific works on Resource-Constrained Project Scheduling Problems (RCPSP) under uncertainty about activity durations (Fu et al. 2012; Artigues, Leus, and Nobibon 2013).

In reactive online planning and scheduling, the objective is to build after any event occurrence a new plan that takes into account this event ; generally, one seeks for a compromise between three competitive objectives : optimality of the new plan/schedule with regard to the chosen optimization criterion, stability with regard to the previous plan/schedule in order to limit as much as possible perturbations, and efficiency of the planning and scheduling algorithm to meet strong or soft decision-making deadlines. Many greedy or local search algorithms, which try and repair the previous plan/schedule, have been proposed for that (see for example (Chien et al. 2000)). One of the distinctive features of our setting is the presence of commitments to high-priority downloads, that is of strong constraints that must be met by the final plan/schedule.

8 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 is shared between ground and onboard planning, and by using onboard fast and efficient repairing procedures and execution algorithms. We also showed the operational advantages that can be provided by such a mixed approach, when compared to usual approaches where all decisions are made only on the ground or only on board.

References

Artigues, C.; Leus, R.; and Nobibon, F. 2013. Robust Optimization for Resource-Constrained Project Scheduling with Uncertain Activity Durations. *Flexible Services and Manufacturing Journal* 25:175–205.

Birge, J., and Louveaux, F. 1997. *Introduction to Stochastic Programming*. Springer Series in Operations Research.

Bonfietti, A.; Lombardi, M.; and Milano, M. 2014. Disregarding Duration Uncertainty in Partial Order Schedules? Yes, We Can! In *Proc. of the International Conference on Integration of Artificial Intelligence and Operations Research Techniques in Constraint Programming for Combinatorial Optimization Problems (CP-AI-OR-14)*, 210–225.

Bouveret, S.; Fargier, H.; Lang, J.; and Lemaître, M. 2005. Allocation of Indivisible Goods: A General Model and some Complexity Results. In *Proc. of the 4th Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-05).*

Cesta, A.; Cortellessa, G.; Fratini, S.; Oddi, A.; and Policella, N. 2007. An Innovative Product for Space Mission Planning: An A Posteriori Evaluation. In *Proc. of the 17th International Conference on Automated Planning and Scheduling (ICAPS-07).*

Chien, S.; Knight, R.; Stechert, A.; R.Sherwood; and Rabideau, G. 2000. Using Iterative Repair to Improve the Responsiveness of Planning and Scheduling. In *Proc. of the 5th International Conference on Artificial Intelligence Planning and Scheduling (AIPS-00)*, 300–307.

Chien, S.; Sherwood, R.; Tran, D.; Cichy, B.; Rabideau, G.; Castano, R.; Davies, A.; Lee, R.; Mandl, D.; Frye, S.; Trout, B.; Hengemihle, J.; D'Agostino, J.; Shulman, S.; Ungar, S.; Brakke, T.; Boyer, D.; VanGaasbeck, J.; Greeley, R.; Doggett, T.; Baker, V.; Dohm, J.; and Ip, F. 2004. The EO-1 Autonomous Science Agent. In *Proc. of the 3rd Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-04)*, 420–427.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal Constraint Networks. *Artificial Intelligence* 49:61–95.

Fu, N.; Lau, H.; Varakantham, P.; and Xiao, F. 2012. Robust Local Search for Solving RCPSP/max with Durational Uncertainty. *Journal of Artificial Intelligence Research* 43:43–86.

Joslin, D., and Clements, D. 1999. Squeaky Wheel Optimization. *Journal of Artificial Intelligence Research* 10:353–373.

Maillard, A.; Pralet, C.; Verfaillie, G.; Jaubert, J.; and Desmousceaux, T. 2014. Building Flexible Data Download Schedules for Agile Earth-Observing Satellites. In *Proc. of International Symposium on Artificial Intelligence, Robotics, and Automation in Space (i-SAIRAS-14).*

Oddi, A., and Policella, N. 2004. A Max-Flow Approach for Improving Robustness in a Spacecraft Downlink Schedule. In *Proc. of the 4th International Workshop on Planning and Scheduling for Space (IWPSS-04).*

Oddi, A.; Policella, N.; Cesta, A.; and Cortellesa, G. 2003. Generating High Quality Schedules for a Spacecraft Memory Downlink Problem. In *Proc. of the 9th International Conference on Principles and Practice of Constraint Programming (CP-03)*, 570–584.

Policella, N.; Smith, S.; Cesta, A.; and Oddi, A. 2004. Generating Robust Schedules through Temporal Flexibility. In *Proc. of the 14th International Conference on Automated Planning and Scheduling (ICAPS-04)*, 209–218.

Pralet, C., and Verfaillie, G. 2013a. Dynamic Online Planning and Scheduling using a Static Invariant-based Evaluation Model. In *Proc. of the 23rd International Conference on Automated Planning and Scheduling (ICAPS-13).*

Pralet, C., and Verfaillie, G. 2013b. Time-dependent Simple Temporal Networks: Properties and Algorithms. *RAIRO Operations Research* 47(2):173–198.

Pralet, C.; Verfaillie, G.; Maillard, A.; Hebrard, E.; Jozefowiez, N.; Huguet, M.-J.; Desmousceaux, T.; Blanc-Paques, P.; and Jaubert, J. 2014. Satellite Data Download Management with Uncertainty about the Generated Volumes. In *Proc. of the 24th International Conference on Automated Planning and Scheduling (ICAPS-14).*

Righini, G., and Tresoldi, E. 2010. A Mathematical Programming Solution to the Mars Express Memory Dumping Problem. *IEEE Transactions on Systems, Man, and Cybernetics, Part C* 40(3):268–277.

Verfaillie, G.; Infantes, G.; Lemaître, M.; Théret, N.; and Natolot, T. 2011. On-board Decision-making on Data Downloads. In *Proc. of the 7th International Workshop on Planning and Scheduling for Space (IWPSS-11).*