PLANET : a planning and replanning tool for a constellation of agile Earth-observing satellites

Romain Grasset-Bourdel[⋆†] and **Gérard Verfaillie**[⋆]
[⋆] Onera - The French Aerospace Lab, Toulouse, France
{Romain.Grasset, Gerard.Verfaillie}@onera.fr

Antoine Flipo[†]
[†] CNES, Toulouse, France
Antoine.Flipo@cnes.fr

Abstract

We present the problem of planning off-line on the ground all the activities of a constellation of next-generation agile Earth-observing satellites and the specific algorithm that was developed to solve it. Then, we present the replanning problem that arises when urgent observation requests are received during plan execution. We show how the planning algorithm can be used in this replanning setting, with some modifications that limit computing time and favour plan stability and optimality. We finally introduce PLANET as a tool based on these algorithms, and demonstrate algorithm efficiency.

Motivation

The context of the work we present in this paper is the European defence MUSIS project (Multinational Space-based Imaging System for Surveillance, Reconnaissance, and Observation) and more precisely the management of the MUSIS agile satellites that are equipped with high-resolution optical observation instruments.

As usual, such satellites are managed from the ground by a mission planning system which receives user observation requests, builds regularly satellite activity plans over a limited horizon ahead (typically one day), and receives plan execution reports. These plans must meet all the physical constraints and satisfy as well as possible the user requests.

However, such a management system is not very reactive. Any observation request, arriving at any time during the day, must wait for the next day to be taken into account. This led project managers to consider a more reactive management system that would take full advantage of the presence of several ground control stations and of the numerous associated satellite visibility windows that allow updated activity plans to be uploaded.

In such a setting, replanning may be called before any satellite visibility window. Replanning problem data is, on the one hand, a current activity plan involving hundreds of observations and, on the other hand, some urgent observation requests (at most some tens). The goal is to build quickly (efficiency) a new plan over the rest of the day that is of an as high as possible quality (optimality) and is as close as possible to the previous one (stability).

Planning problem

The constellation we consider is made up of two identical satellites[1] moving on the same orbit (circular, low altitude, quasi-polar, and heliosynchronous) with a phase shift of 180 degrees between the two satellites.

Each satellite is equipped with thrusters for potential orbital manoeuvres and gyroscopic actuators for quick attitude movements, useful to perform observations and transitions (Lemaître et al. 2002).

A telescope, with two focal planes, allows observations to be performed in the visible and infra-red spectra, with two images (visible and infra-red) within day periods (on the ground) and only one image (infra-red) within night periods. A mass memory allows observation data to be recorded and a high-rate large-aperture antenna allows it to be downloaded towards ground reception stations. Solar panels allow batteries to be recharged when the satellite is not in eclipse. For the sake of agility, all these equipments are body-mounted on the satellite.

We do not go into details here, but the numerous physical constraints that must be met can be classified into six classes : attitude trajectory, observation, download, memory, instruments, and energy. Some of them are similar to the thermal and pointing constraints considered in (Chien et al. 2010) for scheduling operations on board EO-1.

With each user request, are associated a polygon which has been split into strips, a priority level, a weight, and a deadline. Typically, three priority levels are available, from 3 to 1. It is assumed that any request of priority p is preferred to any set of requests of priority strictly less than p. Weights allow to express preferences between requests of the same priority level and are assumed to be additive.

It is assumed that any strip can be observed using only one strip overflight. With each strip, are associated a geographical definition, observation durations (day or night), image sizes (visible, day or night infra-red), a maximum observation angle, and a set of triples $\langle$satellite, visibility window, weather forecast$\rangle$.

[1] The planning algorithm we propose is able to manage any number of satellites, possibly not identical: not the same parameter values.

User requests may arrive at any time and, each day, at a given time, a plan is built for the next day from all the requests that are not out of date and not fully satisfied yet. This plan is built on the ground and then uploaded to the satellites for execution. Typically, up to ten minutes of computing are available for planning. After plan execution, observation data that has been downloaded to the ground is analyzed, taking into account the actual cloud cover, and satisfied requests are removed.

In addition to these normal user requests, urgent ones may arrive at any time too. The latter must be taken into account as soon as possible. To do that, before any visibility window between a ground control station and a constellation satellite, an updated plan is built for the rest of the day from all the requests, either normal or urgent. Replanning is guided by two objectives: on the one hand, to produce a new plan of highest quality, as in planning, and, on the other hand, to maintain in the new plan the greatest number of observations present in the previous one, because a plan is a kind of commitment facing users. In order to be able to take into account urgent requests until the last minutes, we consider that half of the computing time available for planning is available for replanning, that is up to five minutes.

The planning problem can be modeled using for each satellite the following state variables: the current time (orbital position); the attitude position and speed along the three axes; the available memory and energy; for each instrument, its status (ON or OFF), the remaining ON time, and the remaining number of ON/OFF cycles; for the antenna and the visible focal plane, its temperature.

Six types of action are available for each satellite: orbital manoeuvres which are mandatory, observations, data downloads, heliocentric pointings, geocentric pointings, and instrument switchings.

It must be observed that actions of all the types, but the third and sixth (data downloads and instrument switchings), constrain the satellite attitude and are thus mutually exclusive. They must be performed in sequence. Only data downloads and instrument switchings can be performed in parallel, at any time for instrument switchings, but only within effective communication windows for data downloads. As a consequence, a plan has the form of a sequence of actions of any type, except the third and sixth, with attitude movements between consecutive actions and with data downloads and instrument switchings in parallel.

The criterion to be optimized is a vector of numbers v_p, one for each priority level p. Two vectors resulting from two plans are lexicographically compared. For each priority level p, v_p is the sum of the weights of the requests r of priority p, weighted by four factors whose value is between 0 and 1 and which represent (1) the percentage of realization (observation and data download), (2) the mean percentage of cloud cover, (3) the mean observation angle, and (4) the mean data delivering delay, over all the strips of the polygon associated with r.

Planning algorithm

To solve this planning problem, we developed a specific chronological forward search algorithm with dedicated decision heuristics, constraint checking, limited lookahead, and backtrack in case of constraint violation, which guarantees the production of a plan that may be not optimal, but is really executable by the satellites.

Decreasing priorities First, the algorithm we developed works by decreasing priority levels from 3 (the highest) to 1 (the lowest). We consider the sequence of observations present in the plan produced at level $p + 1$ as being mandatory (without fixed starting times) when building a plan at level p. Such an approach is justified by the fact that any request of priority strictly greater than p is preferred to any set of requests of priority p.

A forward chronological algorithm At each priority level p, the algorithm builds a plan in a forward chronological way, from the beginning Ts of the planning horizon to the end Te. At each algorithm step (see Figure 1), if t is the current time and o is the next mandatory observation to be performed, the algorithm chooses the next observation o' of priority p to be performed before o (o if no such observation exists). The algorithm stops when there is no other observation to be included in the plan.

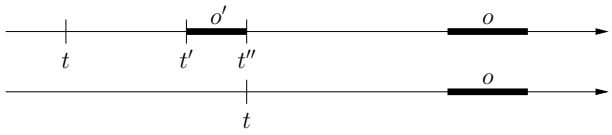

Figure 1: At each algorithm step, choice of the next observation to be performed: o is the next mandatory observation, o' is the chosen observation of priority p.

Decision levels This choice of the next observation to be performed is the first algorithm decision level. Once it has been made, the algorithm makes other choices over the temporal horizon from t to t'' (see Figure 2) at other decision levels: (2) possible insertion of geo or heliocentric pointings, (3) possible data downloads, and (4) instrument activations.

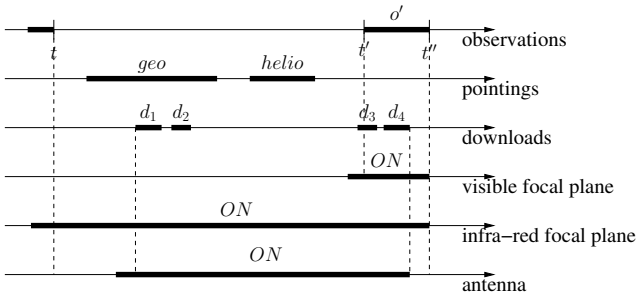

Figure 2: Example of decisions at the four levels: (1) observations, (2) pointings, (3) downloads, and (4) instruments.

Once decisions have been made at the four levels, a consistent plan is available from t to t'', extending the plan that already exists from Ts to t, and the planning process can continue from t'', starting from a known satellite state.

This incremental process, which builds incrementally a complex system trajectory, is the main justification for using a forward chronological search.

For the sake of simplicity, we present the algorithm by assuming only one satellite. However the planning process is in fact interleaved on the two satellites and the next planning step is the earliest one over the two satellites.

Backtracks At any decision level, in case of constraint violation, other choices are made. If no other choice is available, a hierarchical backtrack at the relevant decision level is triggered (see Figure 3).

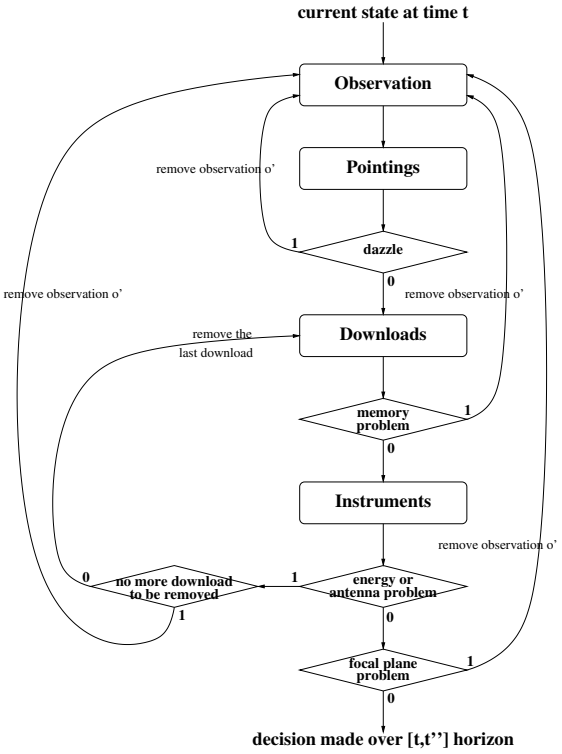

Figure 3: Hierarchical backtracks between decision levels.

At the first level, if the chosen observation is a mandatory one (higher priority), and the insertion is impossible, a chronological backtrack is triggered to the previous insertion of an observation of priority p. However, in order to avoid as much as possible such situations, the latest observation ending times of mandatory observations are propagated from the end to the beginning of the sequence before planning.

Heuristics Heuristics are necessary to make choices at all the decision levels. These heuristics are crucial to the production of good quality plans because, for the sake of efficiency, the algorithm backtracks only in case of constraint violation and never to try and improve on the current plan.

The implemented heuristics are not detailed here.

Dealing with replanning

The first question is how to define stability and how to combine quality and stability (Fox et al. 2006).

In our problem, the quality of a plan is measured by a vector of utilities v_p, one for each priority level p. We maintain this global hierarchical view when replanning. For each priority level p, let R_p be the set of requests of priority p. For each request r, let w_r be the utility associated with r. We have: $v_p = \sum_{r \in R_p} w_r$. Let $I_p \subseteq R_p$ be the set of requests r of priority p that are negatively impacted by replanning (at least one strip of the polygon associated with r was present in the previous plan, but does not appear in the new one). We define the stability as the sum over the impacted requests of the loss in utility: $s_p = \sum_{r \in I_p} (w'_r - w_r)$ with w'_r (resp. w_r) the previous (resp. new) utility associated with r. s_p is positive or null. The lower s_p, the more stable the plan. Then, we define the criterion to be optimized when replanning as a weighted combination of quality and stability: $vs_p = v_p - \alpha.s_p$, with α a positive parameter to be set by system users according to the importance they attach to stability with regard to intrinsic quality.

The data of a replanning problem is very similar to the one of a planning one: same requests, state variables, actions, and constraints. The main difference is in the definition of the criterion to be optimized. Specific data is however: the previous plan, a set of urgent requests to be taken into account and, for each constellation satellite s, a replanning horizon.

To solve our problem, we did not choose to use local search methods, mainly because of the high potential cost of a local change: adding or removing an action in the middle of a plan requires the complex system trajectory to be computed and checked again from the adding/removing point to the end of the planning horizon. We chose to use for replanning the same forward chronological search algorithm we used for planning, called with slightly different data.

We consider four possible modes of replanning. Roughly speaking, the search is less and less restrictive from the first to the fourth mode: less and less constraints imposing previously planned observations (by modifying priorities or weights in heuristics), more and more observations taken into account. It would be possible to run these modes sequentially or concurrently and to get the best result obtained by the deadline.

PLANET

Planning and replanning algorithms were implemented in a tool, called PLANET for PLanner for Agile observatioN satElliTes (see Figure 4), which was developed for this mission, on the basis of a previous tool (Beaumet, Verfaillie, and Charmeau 2011).

Algorithms were experimented on a real-size realistic instance, built by CNES (French Space Agency) and whose characteristics are the following ones: a one-day planning horizon; 8 ground reception stations; 3 priority levels; 1166 observation requests; all of them with polygons limited to one strip and all of them of the same weight (1); among

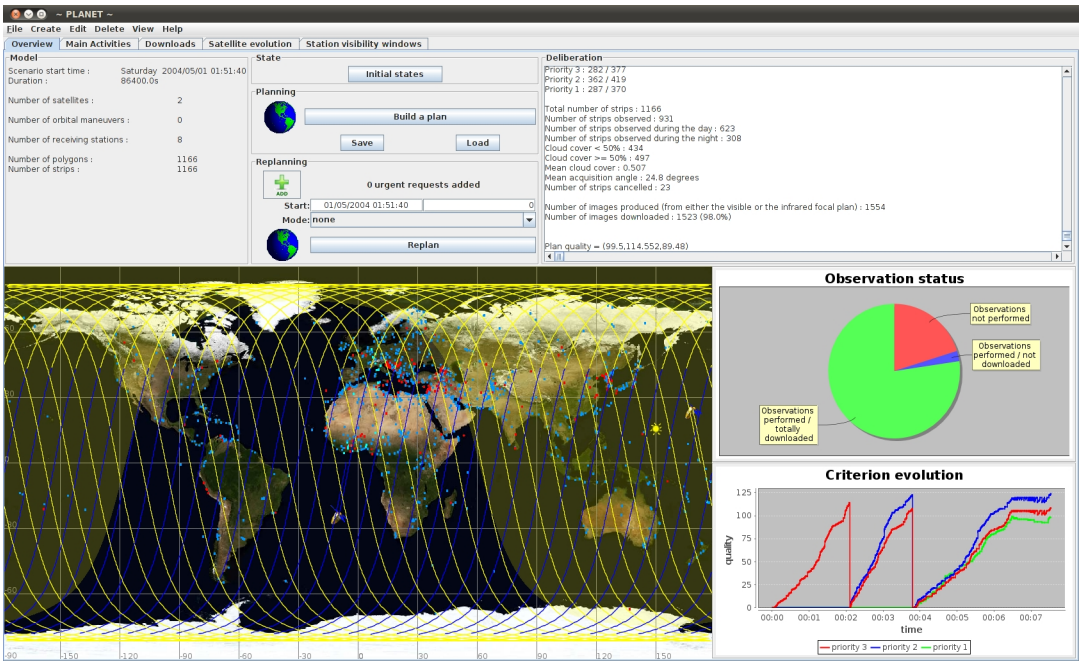

Figure 4: Top level interface of the PLANET tool when planning is complete.

them, 377 of priority 3 (the highest), 419 of priority 2, and 370 of priority 1 (the smallest); meteorological forecast built from climatological data. On this instance, planning takes 236 seconds (about 4 minutes), using a 3Ghz Intel processor with 2.5Go of RAM, running under Linux. In the resulting plan, 906 (78%) observations are performed and downloaded, 16 (1%) are performed, but not downloaded, and 244 (21%) not performed at all. Among the observations of priority 3, 280 (74%) are performed. Results are 367 (88%) for priority 2 and 275 (74%) for priority 1.

In order to evaluate the four replanning modes, we considered a scenario where 10 urgent requests of priority 3 (the highest) arrive some minutes before uploading the daily plan. Such a scenario is one of the most stressing for replanning because planning must be performed again over the whole one-day planning horizon. Following such a scenario, we built three replanning instances of increasing difficulty (urgent requests either geographically spread, concentrated on already overloaded areas . . .). Relative efficiency of each mode in terms of quality, stability, and computing time depends on the instance type. Running these four replanning modes in parallel would be an option. Another option would be to run them in sequence. For that, the order according to which modes are called could be determined for each replanning instance by performing a quick analysis of the setting.

Conclusion

We built a planning algorithm which (i) is able to handle all the complex physical constraints (in particular those related to attitude trajectory), (ii) guarantees the production of a plan that may be not optimal, but is really executable thanks to constraint checking, and (iii) is able to produce in some minutes, over a one-day planning horizon, a plan with hundreds of observations and downloads, which covers satellite attitude trajectory as well as observation, data download, satellite pointing, and instrument activations.

We adapted the algorithm to run in a repair mode, taking into account urgent observation requests: modification of, first, the optimization criterion and, then, request priorities and weights in heuristics (in order to favour plan stability).

Algorithms were implemented in the PLANET tool which allows planning and replanning to be performed and produced plans to be visualized in the form of timelines.

References

[Beaumet, Verfaillie, and Charmeau 2011] Beaumet, G.; Verfaillie, G.; and Charmeau, M. 2011. Feasibility of Autonomous Decision Making on board an Agile Earth-observing Satellite. *Computational Intelligence* 27(1):123–139.

[Chien et al. 2010] Chien, S.; Tran, D.; Rabideau, G.; Schaffer, S.; Mandl, D.; and Frye, S. 2010. Timeline-based Space Operations Scheduling with External Constraints. In *Proc. of the 20th International Conference on Automated Planning and Scheduling (ICAPS-10)*.

[Fox et al. 2006] Fox, M.; Gereveni, A.; Long, D.; and Serina, I. 2006. Plan Stability: Replanning versus Plan Repair. In *Proc. of the 16th International Conference on Automated Planning and Scheduling (ICAPS-06)*.

[Lemaître et al. 2002] Lemaître, M.; Verfaillie, G.; Jouhaud, F.; Lachiver, J.-M.; and Bataille, N. 2002. Selecting and scheduling observations of agile satellites. *Aerospace Science and Technology* 6:367–381.