Planning and replanning for a constellation of agile Earth observation 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

In this paper, we present the problem of planning offline 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.

Introduction

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).

In this paper, we present the physical system we have to manage, the physical constraints we must meet, the user requests we must satisfy as well as possible, and the organization of the management system we assume. Then, we describe the chronological forward search algorithm we developed to solve the planning problem. After that, we describe the replanning problem and how the planning algorithm can be adapted to a replanning setting. Experimental results on real-size scenarios show the right behavior of the chosen approach.

Satellite constellation

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 (see Figure 1).

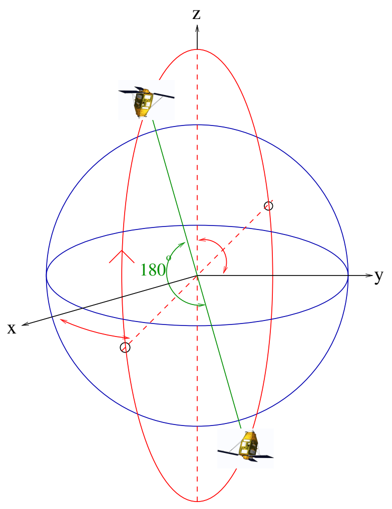

Figure 1: Two-satellite constellation.

Each satellite (see Figure 2) is equipped with thrusters which allow orbital manoeuvres to be performed in case of a too important drift with regard to the reference orbit and with gyroscopic actuators which allow very quick attitude movements (agility) useful to perform observations and transitions between observations.

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

Figure 2: Artist view of a satellite.

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.

Physical constraints

The physical constraints that must be met can be classified into six classes : attitude trajectory, observation, download, memory, instruments, and energy.

Attitude trajectory Thanks to gyroscopic actuators, the satellite is permanently moving around its gravity centre along the three axes (roll, pitch, and yaw). These attitude movements allow observations of areas on the ground to be performed by scanning them. They also allow transitions from the end of an observation to the beginning of the following to be performed relatively quickly. These movements are limited in terms of angular speed and acceleration, resulting in minimum times for moving from an attitude to another. However, the attitude that is required to observe a given area on the ground depends on the orbital position of the satellite and thus on the time at which the observation is performed. The result is a minimum time between the end of an observation and the beginning of the following that depends on the time at which the first ends (see Figure 3 for a schematic 2D illustration). Moreover, computing this minimum time requires solving a complex continuous optimization problem (see (Beaumet, Verfaillie, and Charmeau 2007)). For solving it efficiently inside planning algorithms, dedicated approximate algorithms were developed at ONERA, assuming three-phase movements (constant acceleration, constant speed, and constant deceleration) performed concurrently along each axis (roll, pitch, and yaw).

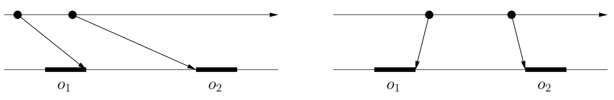

Figure 3: How the angular distance and thus the minimum transition time between observations depends on the time at which the first ends.

Observation Due to maximum observation angles, the observation of a given area on the ground must be performed within one of its visibility windows. Its duration is fixed, because it only depends on the required scanning speed on the ground. The satellite attitude trajectory to be followed during observation depends on the time at which it starts.

Download The same way, due to maximum communication angles, a data download must be performed within one of the visibility windows of one of the ground reception stations. However, this does not suffice because the satellite attitude must be compatible with download (the target station must remain within the satellite antenna communication cone). The result is a set of effective communication windows that depends on the satellite attitude trajectory. Observation and download can be performed concurrently. Two images (visible and infra-red) resulting from the same observation (within a day period) must be downloaded towards the same station and during the same station overflight.

Memory The amount of memory available on board for observation data recording must be never exceeded.

Instruments Concerning the three instruments (high-rate antenna, visible and infra-red focal planes) a minimum preheating time must be met before use, as well as a maximum total ON time and a maximum number of ON/OFF cycles over the planning horizon, for the sake of reliability. Temperature of the infra-red focal plane is automatically regulated by a cryothermic system, but temperatures of both the visible focal plane and the antenna must remain below a given level. Moreover, it must be checked that, at every point on the satellite attitude trajectory, the focal planes are not dazzled and thus not damaged by the sunlight (minimum angle between the satellite axis and the Sun direction).

Energy On-board energy cannot exceed a maximum level due to battery limitations. For the sake of safety, it must remain above a given level, particularly when the satellite is in eclipse and solar panels produce nothing. When the satellite is not in eclipse, the production of energy via the solar panels depends on the satellite attitude trajectory. On the other hand, energy consumption depends on instrument activations.

User requests

With each user request, are associated a polygon which is split into strips, a priority level, a weight, and a deadline.

Typically, three priority levels are available, from 3 (the highest) to 1 (the lowest). 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. In general, user requests exceed the constellation capacity and choices must be made using request priorities and weights.

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$\rangle$.

Management system

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 highest 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.

Differently from other studies that considered on-board planning and replanning (Chien et al. 2004; Beaumet, Verfaillie, and Charmeau 2011), planning and replanning are here performed on the ground. This choice is justified by the fact that the information used by planning and replanning (normal and urgent user requests) comes from the ground and not from board. In such a setting, there would be no advantage to plan on board. Limited computing resources on board would even make it disadvantageous.

Planning problem modeling

The planning problem can be modeled using for each satellite the following state variables:

- the current time and thus the 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:

1. orbital manoeuvres which are mandatory and characterized by starting and ending times and attitudes and by an energy production (function of the attitude trajectory during the manoeuvre);
2. observations which are characterized by a strip, a visibility window, and a starting time;
3. data downloads which are characterized by an image, a reception station, a communication window, and a starting time;
4. heliocentric pointings (solar panels directed towards the Sun in order to recharge batteries as fast as possible) which are characterized by starting and ending times;
5. geocentric pointings (satellite axis directed towards the Earth centre; default action when there is nothing else to do) which are characterized by starting and ending times too;
6. instrument switchings which are characterized by an instrument and a time.

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.

Any plan must satisfy all the constraints described above in Section *Physical constraints*.

We define the criterion to be optimized as a vector of utilities v_p, one for each priority level p. Two vectors resulting from two plans are lexicographically compared. 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 defined as the weight of r 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. At each priority level p, we assume that utility is additive: $v_p = \sum_{r \in R_p} w_r$. The result is a global hierarchical (lexicographic) criterion and a local additive criterion at each priority level.

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). At each priority level p, the starting point is the plan Pl produced at the previous level $p+1$, which includes orbital manoeuvres, observations (of priority $p+1$ or more), pointings (geo or heliocentric), data downloads, and instrument switchings. However, what is kept from Pl is only the sequence Seq of orbital manoeuvres and observations present in Pl, without their starting times. Other actions present in Pl, such as pointings, data downloads, or instrument switchings are disregarded. At level p, observations of priority p will be inserted into Seq by moving starting times when necessary. Other actions, such as pointings, data downloads, or instrument switchings will be added to build a consistent plan. At priority level 3, the starting point is the set of orbital manoeuvres which are imposed on the mission planning system by the satellite control system and can be classed as observations of priority 4.

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. This leads us to consider the sequence of observations present in the plan produced at level $p+1$ as being mandatory when building a plan at level 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. With any step of the algorithm, are associated the current time t, the next observation o of priority $p+1$ or more to be included in the plan because it belongs to Seq, and the set Os of observations of priority p that can be scheduled after t and before o. At the first step, $t = Ts$ and o is the first observation in Seq. The algorithm chooses an observation o' in Os as the next observation to be included in the plan and a starting time t' for o'. If $Os = \emptyset$, then $o' = o$ (observation o is chosen and then a starting time for it). At the next step of the algorithm, t is replaced by the ending time t'' of o' and, if $o' = o$, then o is replaced by the observation that follows it in Seq (empty when o is the last observation in Seq). Figure 4 illustrates two successive steps of the algorithm. The algorithm stops when o and Os are both empty (no other observation to be included in the plan).

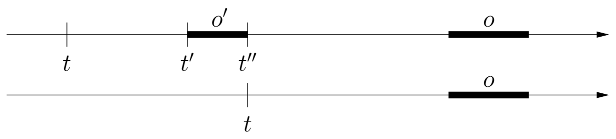

Figure 4: Two successive steps of the forward chronological algorithm.

Decision levels This is the first decision level (1) of the algorithm (choice of the next observation to be included). Once this choice is made, the algorithm makes other choices over the temporal horizon from t to t'' at other decision levels: (2) possible insertion of geo or heliocentric pointings, (3) possible data downloads, and (4) instrument activations.

At the second decision level, geo or heliocentric pointings are inserted between t and t' when possible. Once insertions are decided, the satellite attitude trajectory is completely fixed from t to t''. Hence, the production of energy and the effective communication windows can be computed and the absence of focal plane dazzle can be checked by simulating trajectories.

At the third decision level, data downloads are inserted within the effective communication windows from t to t'' and memory constraints can be checked. This means that observations (first decision level) have priority over downloads (third level). This choice is justified by mission and algorithm considerations: on the one hand, observation is the main system bottleneck and, on the other hand, it is necessary to know the effective communication windows and thus observations and pointings before planning downloads.

At the fourth decision level, instrument activations are inserted in order to satisfy the requirements in terms of observation (visible and infra-red focal planes) and download (high-rate antenna). Energy and instrument constraints can be checked.

Figure 5 shows an example of decisions at the four levels: at level 1, observation o', starting at t', is chosen; at level 2, a geocentric pointing followed by a heliocentric one are inserted before t'; at level 3, data downloads d_1 and d_2, followed by d_3 and d_4, are inserted between t and t''; at level 4, the following decisions are made concerning instrument activations: at time t, the visible focal plane was OFF and it is decided to switch it ON only before o'; on the contrary, the infra-red focal plane was ON and it is decided to maintain it ON between t and t''; at time t, the antenna was OFF, and it is decided to switch it ON before downloading d_1 and to maintain it ON until the end of d_4's download.

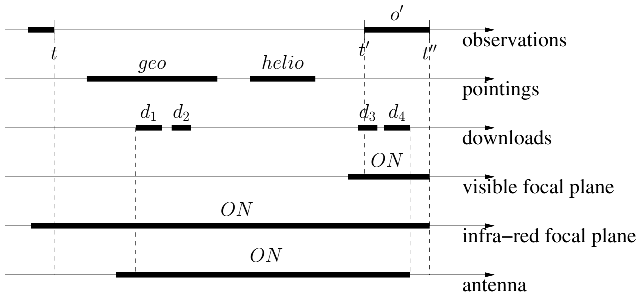

Figure 5: Example of decisions at the four levels: (1) observations, (2) pointings, (3) downloads, and (4) instruments.

Once decisions are 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 completely known satellite state.

This incremental process, which built 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 6).

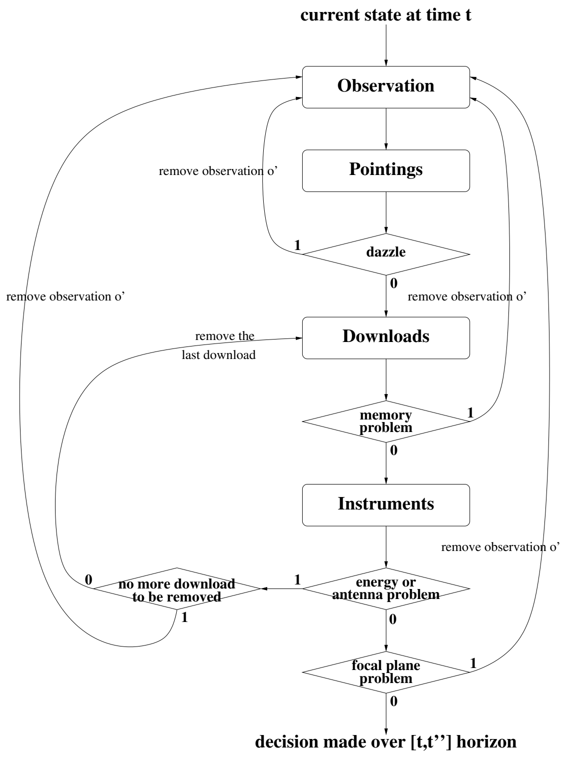

Figure 6: Hierarchical backtracks between decision levels.

At the first level, if $Os = \emptyset$ and thus o is chosen, but insertion of o 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 are propagated from the end to the beginning of Seq before planning.

Heuristics At all the decision levels, heuristics are necessary to make choices. 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. It may be important to stress the difference between the global optimization criterion defined in Section *Problem modeling* and the local heuristics described below which only aim at guiding the search towards good quality solutions.

The following heuristics were implemented at the various decision levels:

1. at the first level, as in the knapsack problem, one chooses an observation o' that maximizes the ratio between the increase in the criterion resulting from the insertion of o' (gain) and the time consumed by this insertion ($t'' - t$, considering the earliest starting time for o'; cost); once an observation o' is chosen, one chooses for it a starting time t' that maximizes the increase in the criterion resulting from the insertion of o' at time t' (function of the observation angle; gain) minus the sum of the decreases in the criterion resulting from this insertion (other observations that would become impossible and whose quality would degrade because of too large observation angles; cost);
2. at the second level, an expert rule aims at making easier energy production and data download; it systematically chooses a geocentric pointing when the satellite is in eclipse; when it is not in eclipse, it gives priority to a heliocentric pointing in order to recharge batteries, except in case of visibility of a ground reception station, because a geocentric pointing always allows data download, whereas a heliocentric one may prevent it;
3. at the third level, as in the knapsack problem, one chooses an image of maximum priority that maximizes the ratio between the increase in the criterion resulting from its download (gain) and the duration of this download (cost);
4. at the fourth level, the choice is, for each instrument, at the end of each mandatory activity period, between switching it OFF and maintaining it ON; these choices have an impact on four "resources": energy, temperature, total ON time, and number of ON/OFF cycles; the result is a kind of multi-criteria decision problem; for each resource and for each alternative a, it is possible to compute a ratio between remaining and maximum quantity, if a is chosen; finally, as usual in multi-criteria decision making, one chooses the alternative that maximizes the minimum ratio over the four resources.

The main difference between this algorithm and the one presented in (Beaumet, Verfaillie, and Charmeau 2011) is the use of backtrack mechanisms in case of constraint violation and of more sophisticated choice heuristics. In (Beaumet, Verfaillie, and Charmeau 2011), no backtrack was allowed and heuristics were limited to randomized decision rules.

Replanning problem modeling

When replanning, the main question in terms of modeling is how to manage the possibly contradictory two objectives: (1) the intrinsic quality of the new plan which can be measured using the same criterion as the one used when planning and (2) the plan stability which can be measured by the difference between the new and the previous plan.

The resulting two questions are: (1) how to define the difference between two plans? and (2) how to combine quality and stability objectives? These questions were discussed in planning (Fox et al. 2006; Cushing, Benton, and Kambhampati 2008), in scheduling (Sakkout, Richards, and Wallace 1998), and in constraint satisfaction (Verfaillie and Jussien 2005).

Our experience led us to consider that there is no generic answer to these questions. Answers depend on the problem at hand. 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: $vs_p = v_p - \alpha \cdot s_p$, with α a positive parameter to be set by system users according to the importance they attach to stability with regard to intrinsic quality.

To take an example, let us consider two requests A and B of the same priority and of weights w_A and w_B, both reduced to one strip and thus to one observation. Let us assume that A was previously planned, that B is a new urgent request, but that A and B are in conflict (it is impossible to satisfy both). The value of a new plan involving A (no change) is w_A, but the value of a new plan involving B (change) is $w_B - \alpha \cdot w_A$. The second one is preferred only if $w_B - \alpha \cdot w_A > w_A$, that is $w_B > w_A \cdot (1 + \alpha)$.

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;
- for each constellation satellite s, a replanning horizon from the first time t at which a new plan can be received for execution by s to the end of the current day: before t, the previous plan cannot be modified by replanning.

Replanning algorithm

When replanning, temporal pressure is generally higher than when planning. This pressure takes generally the form of a deadline for plan production. In our problem, this deadline is the beginning of the next visibility window between a ground control station and a constellation satellite.

Local search methods (Aarts and Lenstra 1997) are known to be able to produce quickly good quality solutions on hard combinatorial optimization problems. One of their strengths is that they can be used the same way, with the same local change operations, in a static setting (to solve a problem) and in a dynamic one (to solve a slightly modified problem, using a previously computed solution). This is why they are intensively used in a context of planning and replanning (Zweden et al. 1994; Chien, Knight, and Rabideau 2000).

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 develop a chronological forward search algorithm. On this basis, the idea is to use the same algorithm for replanning with slightly different data.

Let P be the set of observations that were considered when planning, let $S \subseteq P$ be the set of observations that were selected by planning (present in the previous plan), and let U be the set of observations associated with urgent requests.

We consider four possible modes of replanning.

1. in the first mode, the set of candidate observations is $S \cup U$; however, we favour stability and consider that all the observations in S are mandatory; for that, it suffices to consider them as observations of priority 4; in this mode, we try and insert the urgent observations in the previous plan without removing anything; however, starting times of observations in S can be moved; the same way, pointing, download, and instrument activation plans can be modified;
2. in the second mode, the set of candidate observations is the same: $S \cup U$; however, at each priority level, we consider that observations in S have priority over observations in U; for that, it suffices to add 0.5 to the priority level of each observation in S; as a result, the number of priority levels is multiplied by 2; in this mode, an observation in U of priority level 3 cannot remove an observation in S of the same priority level, but can remove an observation in S of lower priority level (2 or 1);
3. in the third mode, the set of candidate observations remains the same: $S \cup U$; at each priority level, there is no priority between observations in S and U; all of them are equally considered in terms of priority level;
4. finally, in the fourth mode, the set of candidate observations is $P \cup U$ (all observations); as in the previous mode, at each priority level, there is no priority between observations in P and U; all of them are equally considered in terms of priority level.

Roughly speaking, the search is less and less restrictive from the first to the fourth mode: less and less constraints imposing previously planned observations, 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.

Modes 1 and 2 naturally favour stability. Modes 3 and 4 do not so. To favour stability in the latter modes, it is sensible to modify the heuristics used at the first level (choice of the next observation to perform and of its starting date) by multiplying by $(1 + \alpha)$ the weight of a request r if one of its observations o was present in the previous plan ($o \in S$). The idea is to give these requests more importance when making observation choices (see the example in the previous section for an intuitive justification).

Scenarios and experimental results

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

first (easy) instance					
		mode 1	mode 2	mode 3	mode 4
CPU time (s)		130	275	225	232
# obs. removed	prio 3	0	0	1	0
	prio 2	0	0	0	0
	prio 1	0	1	5	4
# urgent obs. added		10	10	8	8
criterion	prio 3	103.7	102.4	101.5	101.7
	prio 2	115.2	114.8	114.8	114.8
	prio 1	96.9	95.4	93.7	94.3
second (medium) instance					
		mode 1	mode 2	mode 3	mode 4
CPU time (s)		133	270	221	231
# obs. removed	prio 3	0	0	1	0
	prio 2	0	4	1	1
	prio 1	0	4	4	7
# urgent obs. added		2	10	9	9
criterion	prio 3	101.9	104.6	103.1	103.3
	prio 2	115.3	111.6	113.6	115.2
	prio 1	96.9	93.4	93.0	92.7
third (hard) instance					
		mode 1	mode 2	mode 3	mode 4
CPU time (s)		129	263	217	225
# obs. removed	prio 3	0	0	3	2
	prio 2	0	2	2	2
	prio 1	0	5	6	13
# urgent obs. added		0	7	8	8
criterion	prio 3	100.8	103.0	102.7	103.2
	prio 2	115.3	112.3	113.2	114.0
	prio 1	97.0	93.4	93.1	93.2

Table 1: Results on the three instances using the four replanning modes

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 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. The fact that relatively more observations of priority 2 are performed than observations of priority 3 can be explained by the fact that, in this instance, observations of priority 3 are more geographically conflicting with each other.

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. For the combination of the quality and stability objectives, we set $\alpha = 0.5$. Following such a scenario, we built three replanning instances of increasing difficulty:

1. in the first one, strips associated with urgent requests are randomly generated on continents; the probability that these strips be in overloaded areas is low;
2. in the second one, strips associated with urgent requests are manually generated on areas where many requests of priority 1 or 2 are present, but few of priority 3;
3. in the third one, strips associated with urgent requests are manually generated on areas where many requests of priority 1, 2, or 3 are present.

Table 1 and Figure 7 show the results obtained by replanning in its four modes on these three instances: CPU time, number of observations removed from the previous plan at the three priority levels, number or urgent observations added in the new plan, value of the new plan at the three priority levels, taking into account quality and stability.

On the first (easy) instance, Mode 1 is clearly the most efficient: all the urgent requests can be added without removing anything; moreover this mode is the fastest in terms of CPU time.

On the second (medium) instance, Mode 2 produces the best results in terms of criterion value: all the urgent requests are added; no request of priority 3 is removed (it is anyway forbidden in Mode 2); only requests of priority 1 and 2 are removed (fewer of priority 2 than of priority 1); however, this mode is the most costly in terms of CPU time.

On the third (hard) instance, things are more complex. No urgent request can be added using Mode 1. 7 urgent requests can be added using Mode 2. One more (8) can be added using Modes 3 or 4. However, fewer requests of priority 3 are removed using Mode 4. Moreover, fewer requests of priority 3 and 2 are removed than of priority 1 using this mode. Mode 4 produces the best results in terms of criterion value, closely followed by Mode 2.

In terms of CPU time, replanning modes 2, 3, and 4 require nearly the same time as planning does. However, this time remains less than the maximum time specified in the mission requirements (5 minutes). Replanning mode 1 requires only half the time used for planning. Moreover, most of the time, replanning will be faster because it will be not performed over a one-day horizon, as in our scenario, but only on the remaining part of the day.

Conclusion

In this paper, we showed that is it possible to use the same chronological forward search algorithm for planning and replanning, only by modifying request priorities and weights,

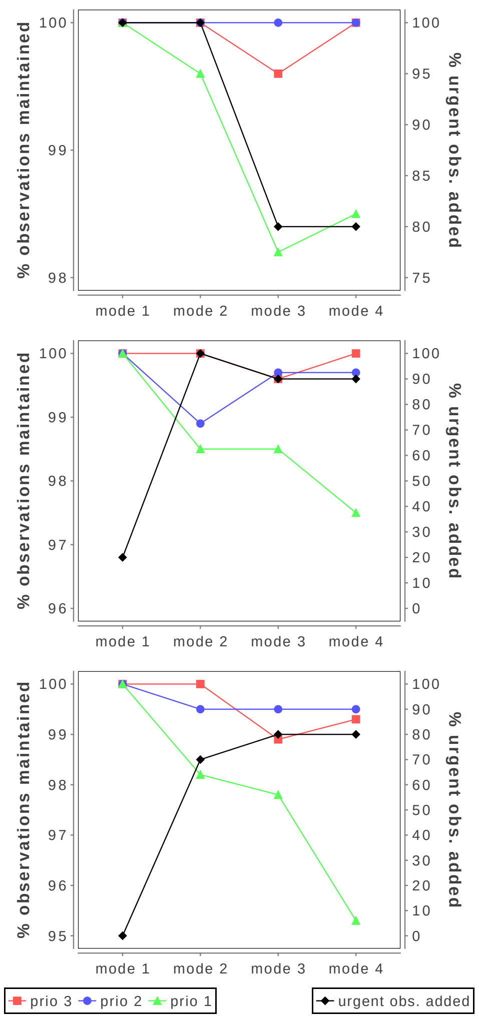

Figure 7: Graphical view of the replanning results; top: first (easy) instance; middle: second (medium) instance; bottom: third (hard) instance

as well as the set of candidate observations. We considered four more or less restrictive replanning modes. First experiments show that their efficiency 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: urgent requests either geographically spread, or concentrated on already overloaded areas.

References

[Aarts and Lenstra 1997] Aarts, E., and Lenstra, J., eds. 1997. *Local Search in Combinatorial Optimization*. John Wiley & Sons.

[Beaumet, Verfaillie, and Charmeau 2007] Beaumet, G.; Verfaillie, G.; and Charmeau, M. 2007. Estimation of the Minimal Duration of an Attitude Change for an Autonomous Agile Earth-observing Satellite. In *Proc. of the 13th International Conference on Principles and Practice of Constraint Programming (CP-07)*, 3–17.

[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. 2004] 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.; Van-Gaasbeck, 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.

[Chien, Knight, and Rabiddeau 2000] Chien, S.; Knight, R.; and Rabiddeau, G. 2000. An Empirical Evaluation of the Effectiveness of Local Search for Replanning. In *Proc. of the ECAI-00 Workshop on "Local Search for Planning and Scheduling"*.

[Cushing, Benton, and Kambhampati 2008] Cushing, W.; Benton, J.; and Kambhampati, S. 2008. Replanning as Deliberative Re-selection of Objectives. Technical report, Arizona State University.

[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)*.

[Sakkout, Richards, and Wallace 1998] Sakkout, H. E.; Richards, T.; and Wallace, M. 1998. Minimal Perturbation in Dynamic Scheduling. In *Proc. of the 13th European Conference on Artificial Intelligence (ECAI-98)*, 504–508.

[Verfaillie and Jussien 2005] Verfaillie, G., and Jussien, N. 2005. Constraint Solving in Uncertain and Dynamic Environments: A Survey. *Constraints* 10(3):253–281.

[Zweden et al. 1994] Zweden, M.; Daun, B.; Davis, E.; and Deale, M. 1994. Scheduling and Rescheduling with Iterative Repair. In Zweden, M., and Fox, M., eds., *Intelligent Scheduling*. Morgan Kaufmann. 241–256.