An Anytime Planning Approach for the Management of an Earth Watching Satellite

Sylvain Damiani
ONERA, Toulouse, France
Sylvain.Damiani@onera.fr

Gérard Verfaillie
LAAS-CNRS, Toulouse, France
Gerard.Verfaillie@laas.fr

Marie-Claire Charmeau
CNES, Toulouse, France
Marie-Claire.Charmeau@cnes.fr

Abstract.

This paper presents the challenge of the autonomous on-board management of the mission of an Earth watching satellite within a constellation of identical satellites, then the anytime planning approach that we developed to deal with it. After a presentation of the reference mission we started from, we describe the operational constraints on the management of this mission, the resulting architectural choices we made, the need for an anytime planning module on-board each satellite, and the dynamic programming approach we chose. Then, we present the results of the simulations that have been carried out on an *a priori* difficult scenario to assess algorithm parameter settings and tradeoffs between computing power and decision quality. We conclude with the work that remains to be done to design a satisfying management system for each satellite and for the whole constellation.

1 Introduction

This paper presents the results of an one year work on the management of the mission of a constellation of Earth watching satellites, dedicated to the detection and observation of forest fires and volcanic eruptions. For this work, we have been provided with an artificial, but realistic, reference mission by the French Space Agency (CNES, (Charmeau 2002)) This mission, as well as the associated technical constraints and choices, is freely inspired from the *Bird* (http://spacesensors.dlr.de/SE/bird/) and *Fuego* (Escorial *et al.* 2001) projects.

After a presentation of this reference mission, we will focus on the operational constraints the mission management system must satisfy and on the resulting architectural choices we made. We will emphasise especially the need for an anytime planning module on-board each satellite. Then, we will describe and justify the dynamic programming approach we chose to implement this planning module. We will show and analyse the results of the simulations we carried out on a difficult scenario to assess algorithm parameter settings and possible tradeoffs between computing power and decision quality. We will conclude with the work that remains to be done if we want to build the prototype of a completely satisfying system.

2 An Earth watching mission

2.1 Fire and eruption detection and observation

The reference mission we consider is dedicated to the detection and observation of forest fires and volcanic eruptions. Starting fires and eruptions must be automatically detected, localised and roughly identified. In case of detection, an alarm must be immediately sent to the concerned ground mission centres. Observations of the fire or eruption areas must be triggered either automatically after a detection, or at the request of a ground mission centre. Associated data must be then delivered to the interested ground mission centres.

2.2 A constellation of Earth watching satellites

The physical components that can fulfil this mission are of four types:

1. a constellation of 12 identical low-orbit (LEO) satellites, arranged according to a *Walker* schema: 3 orbital planes, each with an inclination angle of $47,5°$ with regard to the polar axis, 4 satellites per orbital plan, evenly distributed on a circular orbit at an altitude of 700 km;
2. a set of 3 geostationary (GEO) satellites which together cover the whole Earth surface;
3. a set of ground mission centres, possibly dedicated to a specific area and to a specific kind of event (either forest fire or volcanic eruption).
4. a ground system control centre.

Given their altitude, the LEO satellites have a revolution period round the Earth of about 100 mn. Thus, 25 mn go on between the flight of a satellite over a given area and the flight of the following satellite on the same orbital plane over the same area. Figure 1 shows the track on the ground of the 12 satellites of the constellation within a 25 mn period.

The GEO satellites are used as low rate communication relays between the LEO ones and the ground mission centres.

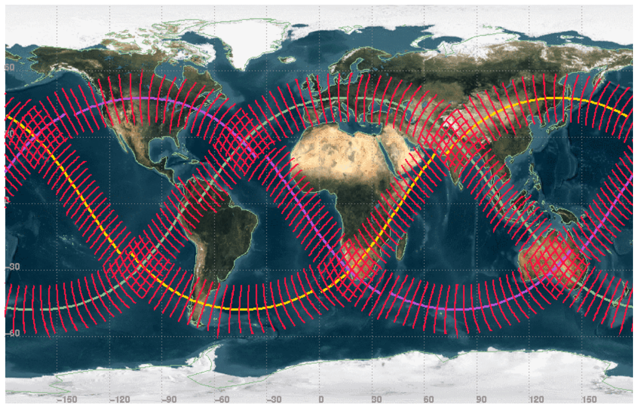


Figure 1: Track on the ground of the 12 satellites of the constellation within a 25 mn period.

2.3 Detection and observation instruments

Each LEO satellite is equipped with two instruments (see Figure 2):

1. an infrared detection instrument, the swath of which is 2500 km wide. This instrument is permanently active and pointed 30°, that is 400 km, in front of the satellite. Data analysis is instantly performed on board. In case of fire or eruption detection, an alarm is sent to the concerned ground mission centres via the currently visible GEO satellite and an observation request is sent to the observation instrument;

2. an observation instrument, the swath of which is only 176 km wide. Four observation modes, in the visible, near infrared, and thermic infrared spectrums, are available, according to the kind of phenomenon to observe. This instrument is not permanently active. It is permanently pointed under the satellite, but a mobile mirror in front of it allows it to observe laterally any ground area in the strip that is swept by the detection instrument. Data that result from an observation are not analysed on-board. They are downloaded to the concerned ground mission centres during visibility windows.

Note that, because the detection instrument is systematically pointed 30° in front of the satellite, there is an one minute delay between the detection of an unexpected phenomenon by a satellite and its possible observation by the same satellite.

Because the satellite can observe an area only when it arrives roughly at the same latitude, the start and end times of the observation of a given area from a given revolution of a given satellite are fixed and two areas the latitudes of which are too close may be incompatible: they cannot be observed by the same satellite from the same revolution because either a temporal overlapping, or an insufficient time to modify the mirror orientation (see Figure 3).

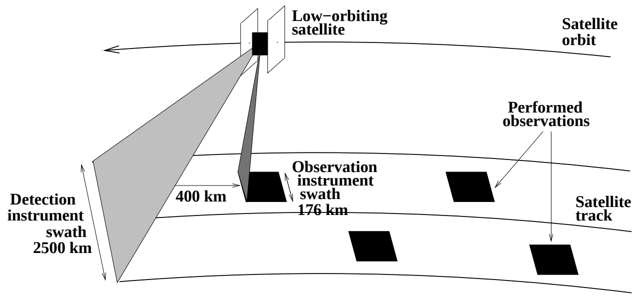
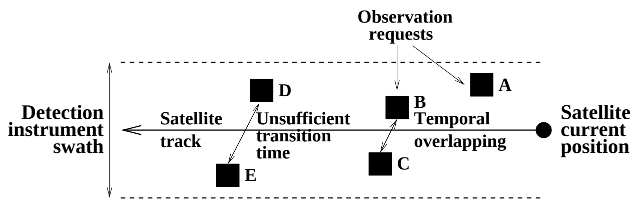


Figure 2: Detection and observation on-board each LEO satellite.



Observation requests
Detection instrument swath
Satellite track
D
Unsufficient transition time
E
B
Temporal overlapping
C
A
Satellite current position

Figure 3: Incompatibilities between observations from the same satellite and the same revolution.

2.4 On-board energy and memory

Each LEO satellite is limited in terms of energy and memory available on-board. Figure 4 shows the permanent and temporary productions and consumptions of energy and memory that must be taken into account.

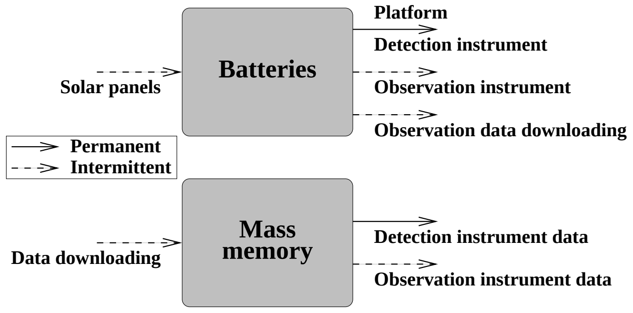


Figure 4: Productions and consumptions of energy and memory.

It is assumed that solar panels are powerful enough to cover the maximum energy consumption during day windows, but enough energy must be stocked into batteries to cover night windows. Energy and memory are not independent because observations consume energy and memory and because data downloading produces memory (by releasing memory space), but consumes energy.

2.5 Communications

Communications between the various components of the system are not permanent (see Figure 5):

- communications between the LEO satellites and the ground, via the GEO satellites, are limited to unidirectional small size alarm messages in case of fire or eruption detection;
- only direct visibility windows can be used for larger size messages, to upload observation requests or to download observation data;
- there is no direct communication between LEO satellites.

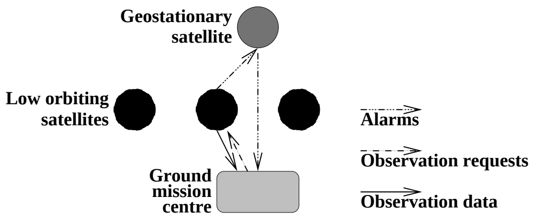


Figure 5: Communications between system components.

3 Operational constraints and architectural choices

Two kinds of operational constraint have a determining effect on the type of management we can consider for this system:

1. apart the possible alarm messages from the LEO satellites to the ground mission centres, there is no permanent communication link, neither between LEO satellites, nor between them and any ground mission centre;
2. after an unexpected event detection, a LEO satellite must be able to perform an observation of the associated area and to download the resulting data to the concerned ground mission centres. It must do that as quickly as possible, because the main interest of such a system is the quick delivery of these observation data to the mission centres, which can then trigger any appropriate action on the ground. Late observations and data deliveries are of low utility.

To sum up, a LEO satellite cannot wait for explicit observation and data downloading requests, coming from any ground mission centre or any other satellite, to perform what it has to do. It must be able to decide autonomously at any moment upon observations and data downloading.

These operational constraints turn down any choice for a mission management system which would be either centralised in the unique ground system control centre, or decentralised with information exchanges and negotiations between the ground mission centres or between the LEO satellites themselves.

The only choice that remains sensible, given the strongly limited communication means, is an autonomous mission management system on-board each LEO satellite. This is the choice we made and we assume for the sequel of this paper.

At this point, it may be interesting to stress the difference between Earth watching and Earth observation missions:

- in Earth observation missions, like *Spot*, *Pléiades*, or others including missions that involve constellations of observation satellites (Gabrel *et al.* 1997; Sherwood *et al.* 1998; Muraoka *et al.* 1998; Potin 1998; Potter and Gasch 1998; Bensana *et al.* 1999; Wolfe and Sorensen 2000; Pemberton 2000; Niezette 2000; Vasquez and Hao 2001; Frank *et al.* 2001; Lemaître *et al.* 2002; Globus *et al.* 2002)), the requests come from the ground and the deadlines for the achievement of the observations and the delivery of the associated images are not very short (typically, some days or some weeks). In these conditions, the choice of a centralised mission management system on the ground, able to optimise the activities of all the components of the system, seems sensible, although onboard management systems have been explored too (Bornschlegl *et al.* 1998; Gout *et al.* 1999; Verfaillie and Bornschlegl 2000; Honvault *et al.* 2001; Creasey and Teston 2001);
- in Earth watching missions, like *Fuego* or others (Escorial *et al.* 2001; Chien *et al.* 2001; Chien *et al.* 2002; Khatib *et al.* 2003), on the contrary, the requests come mainly from the satellite itself and the deadlines for the achievement of the observations and the delivery of the associated images are shorter (typically some minutes or some hours). In these new conditions, the choice of a autonomous mission management system on-board each satellite seems more sensible.

4 An anytime planning module

In these conditions, each LEO satellite must be equipped with a decisional system which must be able:

1. to take into account the current state of the satellite (orbital position and trajectory, energy, memory, mirror position . . .) and the current set of observation requests, coming either from the detection instrument, or from ground mission centres;
2. to select among them a subset of observations that can be performed all together over a given temporal horizon and the global utility of which is as high as possible;
3. to build an associated feasible activity plan;
4. to trigger and control the execution of this plan;
5. to adapt observation selection and activity plan in case of any unexpected change in the state of the satellite or in the set of observation requests.

Note however that the situation to face (satellite state and current set of observation requests) is potentially highly dynamic, mainly because of observation requests coming from the detection instrument. Even if such events are globally rare, it is precisely in such situations that an appropriate quick reaction of the decisional system is waited for: it has been built for that.

In such a context of high uncertainty, one can consider that building and maintaining an activity plan over a large temporal horizon may not be of great utility: why to spend time to build a plan which has every chance not to be executed as it stands?

Moreover, because the decisional system is implemented on-board the satellite, decision to trigger an observation or a data downloading can be made at any time: rather than an activity plan, we need a mechanism which allows the satellite to decide at any time upon the next observation to perform and the next data to download.

There are basically three ways of building such a decision mechanism, which is often referred to as a strategy or a policy:

1. one can compute off-line the optimal decision associated with any possible situation. Because of the astronomical number of situations the satellite may have to face (combinations of all the possible satellite states and all the possible sets of observation requests), this technical solution can be immediately turned down;
2. one can define off-line decision rules which allow the satellite to face on-line any possible situation. An example of such a decision rule is to trigger any observation that can be executed, given the temporal and resource constraints. In the example of Figure 3, this policy would lead the satellite to perform observations A, B, and D. But, if the utilities of observations A, B, C, D, and E are for example 5, 1, 10, 2, and 20, this activity plan is clearly sub-optimal (global utility of 8), whereas there is an obvious optimal plan (A, C, and E, with a global utility of 35);
3. one can design off-line an optimisation mechanism which allows the same way the satellite to face on-line any possible situation, the main difference with the previous approach being that we will try to build a mechanism which is as less myopic as possible. This is the approach we chose and we assume for the sequel of this paper.

The on-line optimisation mechanism we built is based on two principles:

1. firstly, reasoning shall not preclude or delay actions. For example, the fact that the decisional system is still reasoning and computing when time is up to trigger an observation of high utility shall not lead the satellite to leave it. In other words, the decisional system must always know what to do when time is up to decide upon triggering an observation or not;
2. secondly, the decisional system must use all the time it has at its disposal to reason as thoroughly as possible, in order to decide upon the best observation to perform next, taking into account its knowledge about the current satellite state and the current set of observation requests. In other words, the fact that the reasoning time is limited and that the situation to face may change at any time is not an excuse for a short-term arbitrary limited reasoning. Long-term reasoning must be triggered and carried on each time information and reasoning time are available for that.

To implement these principles, we designed an on-line optimisation mechanism which is based on three temporal horizons (see Figure 6):

1. a commitment horizon, which lies between the current time and the end of the currently performed observation, if it exists, and is empty otherwise. Over this horizon, nothing can be decided because activities have been already committed by the executive;
2. a decision horizon, which is limited to the next observation to perform. This is over this horizon that the planning module will decide upon activities and send its decisions to the executive;
3. a reasoning horizon, the length of which depends on the time that is available for reasoning. Even if the planning module reasons over this horizon, in order to assess the consequences of the decisions it envisages, it does not make any definitive decisions over it. At the beginning of the reasoning, this horizon is limited to the first observation request along time. This observation, we note i, is selected if it can be executed, given the temporal and resource constraints. Each time more time is available for reasoning, this horizon is incremented by adding to the set of considered observation requests the next one along time. The best next observation to perform, possibly different from i, is then selected among them, taking into account all the possible observation sequences. This process stops as soon as time is up to trigger i. If i is the currently selected one, it is triggered. If not, nothing happens. In both cases, reasoning continues without i, still to decide upon the next observation to perform.

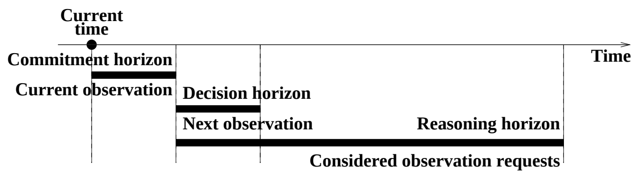


Figure 6: The various horizons considered by the decisional system.

This mechanism can be seen as a form of resource-bounded and anytime reasoning system (Boddy and Dean

1994; Zilberstein 1996), which adapts itself to the available time. If the temporal pressure is too high, its behaviour is the one of the simple decision rule we presented above: as soon as a candidate observation can be executed, given the temporal and resource constraints, it is triggered. If the temporal pressure is lower, an optimisation is carried out among the currently considered observation requests. This set of considered observation requests grows with time, resulting in a less myopic decision mechanism which allows generally better decisions to be done. In contrast to decision mechanisms the duration of which is strictly limited at the design time in order to be sure to provide the supervisor with a decision whatever the temporal pressure is, this mechanism exploits all the reasoning opportunities. For example, the highly variable time between the starting time of the currently performed observation and the starting time of the first observation that can be performed next is completely exploited to decide upon the next observation to perform. Moreover, one can guarantee that the supervisor is provided with a decision each time it is required, that is each time a candidate observation can be executed, in order to trigger it or not.

5 A dynamic programming approach

At least concerning the observation decisions, which are the most important, such a mechanism can be implemented using a dynamic programming approach (Bellman 1957).

This approach uses the fact that the global utility of a plan is defined as the sum of the utilities of the selected observations and the fact that all the observation requests can be ordered according to an increasing starting time. It is based on the recursive computing of two things, for each observation request i, from the first to the last one, for each possible energy level e and each possible free memory level m:

1. the maximum global utility $U^*(i,e,m)$ that can be obtained from the current time to the ending time of i, by going out of i with an energy level e and a free memory level m;
2. an associated observation plan (see Figure 7).

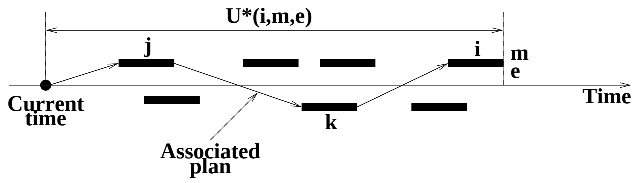


Figure 7: Maximum utility $U^*(i,e,m)$ and associated observation plan.

Equations 1 are a simplified version of the equations that are actually used for this recursive computing. In these equations, $C(i)$ is the set of observations that can start before i and are compatible with i, $u(i)$ is the utility of i, and $m(i)$ and $e(i)$ are the amount of energy and memory that are consumed by i. Actual equations take into account energy production during day or night windows, permanent energy consumption, data downloading windows, and data downloading energy consumption and memory production. See Figure 8 for an example of observation plan, taking into account all these features.

$$\begin{aligned} U^*(i,m,e) \quad = \quad & max_{j \in C(i)} \\ & [u(i) + U^*(j, e+e(i), m+m(i)] \end{aligned} \tag{1}$$

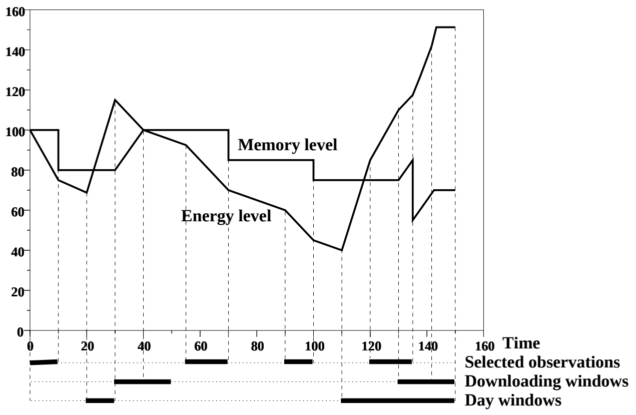


Figure 8: Example of plan with energy and free memory levels as functions of time.

Note that the complexity of this recursive computing is only a quadratic function of the number of considered observation requests, that is of the length of the current reasoning horizon, and a linear function of the number of considered levels of energy and memory, which depends of the number of discretisation steps used for energy and memory.

Note also an interesting feature of this computing. When $U^*(i,m,e)$ has been computed for an observation request i, for all the possible levels of energy and memory, an optimal plan over a reasoning horizon of length i is available via equations 2. The first observation of this plan can be considered as the optimal decision by reasoning over this horizon. In these equations, $P(i)$ is the set of observations that can start before i, compatible with i or not, and E_{max} and M_{max} are the maximum levels of energy and memory.

$$\begin{aligned} U^*(i) \quad = \quad & max_{j \in P(i), e \leq E_{max}, m \leq M_{max}} \\ & U^*(j,e,m) \end{aligned} \tag{2}$$

This way, a decision j is available at each step i of the algorithm. Its quality, that is the utility of a plan of maximum utility over the horizon of maximum length, starting with j, is a globally increasing function of i, although a systematic increase is not guaranteed. In contradiction with a conventional wisdom, increasing the length of the reasoning horizon may indeed decrease the decision quality (Pearl 1983; Bulitko *et al.* 2003; Khatib *et al.* 2003) (see also Figure 12). Figure 9 shows two optimal plans, computed at two successive steps of the algorithm (i and $i + 1$). It shows that the optimal decision (j') at step $i + 1$ may be different from the one (j) at step i.

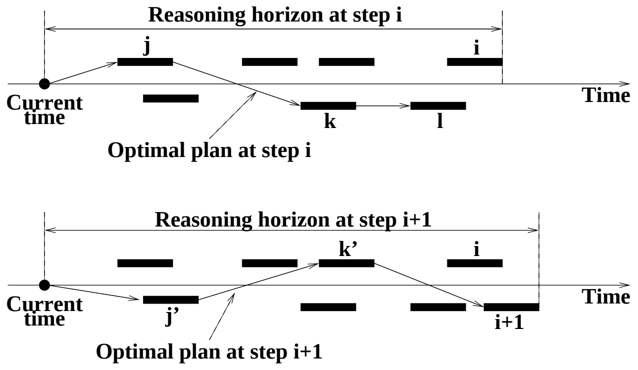


Figure 9: Optimal decisions at step i and $i + 1$.

At this point, it must be emphasised that, although dynamic programming fits very well the specific problem we face, it may not be always the case, for example when observation starting times are not fixed and scheduling decisions between observations must be made. But the global anytime approach does not depend on the use of dynamic programming. Any search algorithm, as, for example, tree or local search, which is able to build a good quality plan over a given horizon, can be used instead of dynamic programming. But the quicker algorithm, the higher plan quality, and the larger amount of information reused when searching for a plan over a larger horizon, the better global result.

6 A decisional system

6.1 A global architecture

As usually (Alami *et al.* 1998; Muscettola *et al.* 1998), the whole decisional system involves three main components (see Figure 10):

1. a supervisor, which is the core of the decisional system in charge of receiving observation requests either from the detection instrument, or from ground mission centres, execution reports from the executive, and decisions from the planning module, in charge also of activating the planning module, as well as observations and data downloadings via the executive;
2. a planning module, which may be activated and interrupted at any moment by the supervisor and provides it with observation decisions each time an observation may be executed;
3. an executive which serves as an interface between the supervisor and the physical system.

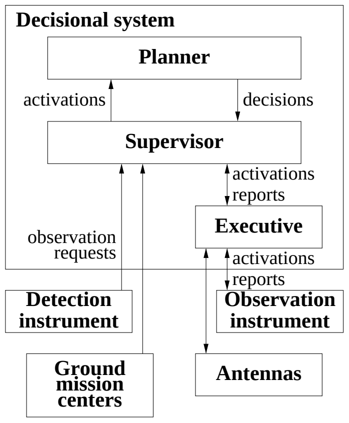


Figure 10: Global architecture of the decisional system.

6.2 A reactive supervision loop

The supervisor takes the form of an infinite loop, the length of which has been arbitrarily set to one second. During each loop, that is in less than one second, the supervisor:

- handles all the events that occurred during the previous loop: information about unexpected states of the satellite, execution reports, new observation requests;
- updates accordingly its internal data structures;
- extracts from the current set of observation requests the subset T of the ones that can be triggered during this loop;
- if $T \neq \emptyset$,
  - then extracts the first observation j of the plan currently provided by the planning module; if $j \neq \emptyset$
    - then if $j \in T$, then triggers j, else does nothing (case of a non empty plan);
    - else uses a basic decision rule to decide upon the observation $i \in T$ to trigger (case of an empty plan);
  - else does nothing (nothing to do);
- restarts the planning module in case of unexpected change in the current state of the satellite or in the set of current observation requests.

7 Simulations

7.1 Experimental objectives

The experiments we built and carried out aimed at:

- analysing the behaviour of the dynamic programming planning mechanism in terms of computing time, used memory, and decision quality;
- studying the influence of the energy and memory discretisation step number, and of the reasoning horizon length on this behaviour;

- comparing the results that can be obtained by such an anytime optimisation mechanism with the ones that are obtained by using simple decision rules, like the two following ones:
  - each time an observation can be triggered, trigger it (rule 1);
  - each time an observation can be triggered, trigger it if it is not in conflict with a future observation of higher utility (rule 2).

7.2 Experimental scenarios

For the moment, experiments have been carried out only on one randomly generated, but highly loaded, scenario (many observation requests, not uniformly distributed along time) in order to assess the optimisation mechanism in a stressing setting. The main features of this scenario are the following ones:

- it covers one satellite revolution, that is 6000 seconds;
- at time 0, the satellite is provided with 200 randomly generated observation requests: 100 of them with a uniform probability distribution of the observation starting times over the whole revolution, and the other 100 with a Gaussian distribution of these observation starting times around time 100;
- moreover, the satellite is progressively provided with 50 randomly generated observation requests, with a Gaussian distribution of the request arrival times around time 4000 and a uniform probability distribution of the observation starting times between the request arrival time and time 6000;
- observation utilities are generated with a uniform probability distribution among the multiples of 10 between 10 and 100;
- all the observation requests (250) are generated with a uniform probability distribution of the lateral positions with regard to the satellite track over the whole detection instrument swath, leading to a possible mirror orientation between $-55°$ and $+55°$; the mirror takes 11.5 seconds to go between these two extreme positions ($9.6°/sec$);
- each observation is 176 km large and long, and thus takes 26 seconds; each observation mode (1 or 2 per observation) consumes $1/600$ of the maximum energy level and $1/1000$ of the maximum memory level; its downloading takes 11 seconds and consumes $1/2000$ of the maximum energy level;
- at time 0, memory is free, but only 17% of the maximum energy level is available;
- a 4000 second day window begins at time 1000 and two 360 second data downloading windows begin at time 119 and 3473;
- during day windows, the solar panels are powerful enough to cover the maximum satellite energy consumptions.

7.3 Experimental results and analysis

Artificial offline setting We first considered an artificial offline setting where planning should be achieved only once before the beginning of the satellite revolution and for the whole revolution. Our objective was mainly to assess how the CPU-time and the used memory evolve as functions of the length i of the reasoning horizon, that is the number of considered observation requests.

Figure 11 shows the results that have been obtained with 100 discretisation steps for energy and memory. It shows that CPU-time goes from 0 second for $i = 1$ to more than 400 seconds for $i = 200$ and that used memory lies between 9 and 24 Mbytes. An interesting piece of information is that 1 second of reasoning allows us to look until 25 requests ahead, 10 seconds until 80 requests, and 1 minute until more than 130 requests.

With regard to the decision quality which can be measured as the difference between the maximum utility that can be obtained over the whole 6000 second horizon and the maximum utility that can be obtained over the same horizon by starting with the first observation of the optimal plan associated with a reasoning horizon of length i, it is, at least on this scenario and in such offline setting, always equal to 0: whatever i is, the same observation is selected to be triggered first and it is optimal. But we will see that that this is not always the case, particularly in an online setting (see next subsection).

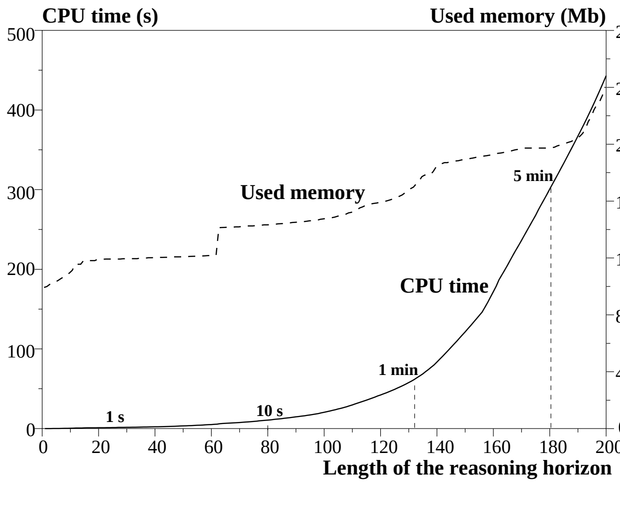


Figure 11: CPU-time and memory as functions of the length of the reasoning horizon.

Artificial online setting Then, we considered an artificial online setting where the arrival of new observation requests forces planning to be done again often during the satellite revolution, but where there is no temporal pressure on the planning reasoning. Reasoning takes its time to go as far as it wants.

Figure 12 shows four graphs, one for each of the following number of discretisation steps for energy and memory: 10, 50, 100, and 1000. Each graph shows how the global utility over the whole revolution evolves as a function of the length of the reasoning horizon.

This figure shows first that this utility is not mandatorily an increasing function of the length of the reasoning horizon, although reasoning on large horizons is generally better than reasoning on very short ones. For example, for 10 discretisation steps, it is more profitable to reason on an horizon of length 4 than on an horizon of length 40. This phenomenon may be the result of the arrival of new observation requests for which no prior knowledge is available and which may lower abruptly the utility of previous decisions that resulted from long-term optimisations. Note that this undesirable phenomenon tends to decrease by using a sufficiently large number of discretisation steps (100 or 1000).

This figure shows also that increasing the number of discretisation steps behind 1000 may not be very profitable in terms of global utility, above all if we remember that this increases unavoidably CPU-time and memory consumptions.

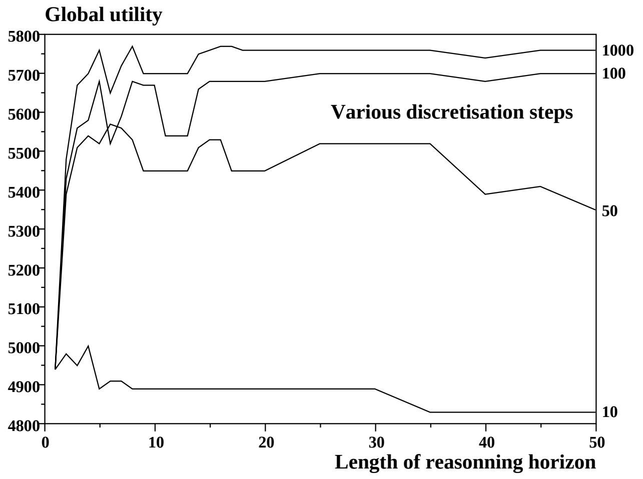


Figure 12: Global utility as a function of the length of the reasoning horizon and of the number of discretisation steps used for energy and memory.

Realistic online setting Finally, we considered a more realistic online setting where, as previously, the arrival of new observation requests forces planning to be done again often during the satellite revolution and where reasoning is temporally limited by observation triggering deadlines.

Table 13 shows the global utility that results from two classes of decision methods:

1. the ones that use *decision rules* such as Rules 1 and 2 described above: DR_1 and DR_2;
2. the ones that use the *anytime dynamic programming* approach described above: $DP_{nds/cp}$, where nds is the number of discretisation steps used for energy and memory, and cp is the ratio between the power of the computer used for simulations (a SUN station SPARC 5, with 333 Mhz and 256 Mbytes of RAM) and the power of the computer that could be used onboard each satellite. For nds, we used the same values as previously: 10, 50, 100, and 1000. With nds set to 1000, we used the following values for cp: 1, 5, 10, and 100 (for example, $cp = 100$ means that the onboard computer is 100 times less efficient than the one we used for simulations). Note that, for all these variants of DP, the decision rule DR_2 has been chosen to be the default behaviour of the supervisor when a decision must be made to trigger or not an observation and no plan is available.

Method	DR_1	DR_2		
Utility	4860	5510		
Method	$DP_{10/1}$	$DP_{50/1}$	$DP_{100/1}$	$DP_{1000/1}$
Utility	4870	5340	5530	5620
Method	$DP_{1000/100}$	$DP_{1000/10}$	$DP_{1000/5}$	$DP_{1000/1}$
Utility	5540	5520	5560	5620

Figure 13: Global utility obtained in a realistic online setting by methods based on decision rules or anytime dynamic programming.

Note that such global utilities result from the achievement of about 75 observation requests over the whole revolution.

These results show first that, although the performance of DR_1 is somewhat weak, the one of DR_2 is surprisingly good in spite of the simplicity of this rule. Then, they confirm that the number of discretisation steps used for energy and memory has a significant impact on the resulting utility: the best results are obtained with 1000 discretisation steps. Finally, they show that the computing power has a surprisingly weak impact on the resulting utility: whatever the computing power is, results remain better than the ones of DR_2, the decision rule used by the supervisor as a default behaviour in case of absence of plan.

Although further experiments with other kinds of scenarios are necessary, these results suggest that:

- studying decision rules that would be a bit more sophisticated than DR_2 might be profitable;
- working on more accurate physical models of the satellite and of the observation and data downloading activities might be more immediately profitable than working on more efficient algorithmic versions of the anytime dynamic programming approach.

8 Present and future work

But important pieces of work remain to be done if we want to design and to evaluate a management system for each satellite and for the whole constellation.

8.1 Data downloading management

First, for the moment, the planning module optimises only the observation decisions. Data downloading decisions are managed using a simple decision rule: a static order decides upon the ground mission centre to which data are downloaded in case of conflict between mission centres, and data are downloaded to a given mission centre in the order of the associated observations (FIFO decision rule).

It would interesting to extend optimisation to the data downloading decisions. For that, the utility of a given observation can no longer be associated with the observation achievement, but with the data downloading achievement. It is more precisely a decreasing function of the delivery time. Such an unusual optimisation criterion and the free decision upon the order two observation data are downloaded both result in a data downloading management which is more complicated that the observation management.

In such a setting, the way data downloading will be optimised and observation and data downloading optimisations will be combined within a global anytime approach remains to be decided.

Note also the ability to replace data associated with a low utility observation by data associated with a higher utility one in case of a full mass memory, as suggested in (Khatib *et al.* 2003).

8.2 Uncertainty management

For the moment, we consider that all the decided activities have a deterministic behaviour: each of them succeeds, takes a known time, and consumes a known amount of energy and memory. Execution results that differ from what was expected are only managed in a reactive way, the same way as new observation requests are.

But, for example, the amount of memory that is consumed by an observation after data compression is not known precisely in advance. Lower and upper bounds can be provided, but the precise amount depends on the observation.

In such a setting, there is certainly a tradeoff to tune between optimality (optimistic decisions) and robustness (pessimistic decisions).

8.3 Change management

Always for the moment, all the results of the optimisation of the planning module are lost in case of any change in the current state of the satellite or in the current set of observation requests. The whole reasoning must restart from scratch with a reasoning horizon of length 1.

Such a situation could be improved by designing incremental mechanisms which would be able to reuse as many as possible of the previous results.

8.4 Inter-satellite cooperation

Letting the most important point for the end, we only considered for the moment the autonomous management of each isolated satellite.

But, Earth watching needs the cooperative work of the whole constellation. It is thus necessary to study how the satellites will be able to cooperate with each other in order to satisfy as best as possible the global mission. It may be for example useless that many satellites perform the same observation each one after the other, especially if this leads all of them to leave out other important observations. One way or another, observations must be as evenly as possible distributed among satellites.

The difficulty is to get this result with distributed autonomous management systems and without permanent communications between satellites and ground mission centres. We plan to study how each satellite could use any piece of knowledge about the behaviour of the other satellites to make autonomously the best decisions: for example static knowledge about the track of the other satellites, or dynamic knowledge they could be provided with by ground mission centres during the visibility windows about the current set of observation requests and the current observation plans of the other satellites, and about the observations they already performed and delivered.

9 Acknowledgements

Thanks to Dominique Séguéla from CNES and Michel Lemaître from ONERA who helped us in this study.

References

[Alami *et al.* 1998] R. Alami, R. Chatila, S. Fleury, M. Ghallab, and F. Ingrand. An Architecture for Autonomy. *The International Journal of Robotics Research*, 17(4):315–337, 1998.

[Bellman 1957] R. Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[Bensana *et al.* 1999] E. Bensana, M. Lemaître, and G. Verfaillie. Earth Observation Satellite Management. *Constraints : An International Journal*, 4(3):293–299, 1999.

[Boddy and Dean 1994] M. Boddy and T. Dean. Deliberation Scheduling for Problem Solving in Time-Constrained Environments. *Artificial Intelligence*, 67(2):245–285, 1994.

[Bornschlegl *et al.* 1998] E. Bornschlegl, P. David, and H. Schindler. On-Board Software and Operation for PROBA Small Satellite Autonomy. In *Proc. of DASIA-98*, Athens, Greece, 1998.

[Bulitko *et al.* 2003] V. Bulitko, L. Li, R. Greiner, and I. levner. Lookahead Pathologies for Single Agent Search. In *Proc. of IJCAI-03*, Acapulco, Mexico, 2003.

[Charmeau 2002] M.C. Charmeau. Mission de Référence pour le DE Autonomie. Note technique DTS/AE/SEA/IL/02-112, CNES, 2002.

[Chien *et al.* 2001] S. Chien, R.Sherwood, M. Burl, R. Knight, G. Rabideau, B. Engelhardt, A. Davies, P. Zatocha, R. Wainwright, P. Kuplar, P. Cappelaere, D. Surka, B. Williams, R. Greeley, V. Baker, and J. Dohm. The Techsat-21 Autonomous Sciencecraft Constellation. In *Proc. of the ESA Workshop on On-Board Autonomy*, pages 167–174, Noordwijk, The Netherlands, 2001.

[Chien *et al.* 2002] S. Chien, B. Engelhardt, R. Knight, G. Rabideau, and R.Sherwood. Onboard Autonomy of the Three Corner Sat Misssion. In *Proc. of the Third IWPSS NASA Workshop*, Houston, TX, USA, 2002.

[Creasey and Teston 2001] R. Creasey and F. Teston. Project for Onboard Autonomy: PROBA. In *Proc. of the ESA Workshop on On-Board Autonomy*, Noordwijk, The Netherlands, 2001.

[Escorial *et al.* 2001] D. Escorial, I. Tourne, and F. Reina. FUEGO: A Dedicated Constellation of Small Satellites to Detect and Monitor Forest Fires. In *Proc. of the Third IAA Symposium on Small Satellites for Earth Observation*, Berlin, Germany, 2001.

[Frank *et al.* 2001] J. Frank, A. Jónsson, R. Morris, and D. Smith. Planning and Scheduling for Fleets of Earth Observing Satellites. In *Proc. of i-SAIRAS-01*, Montreal, Canada, 2001.

[Gabrel *et al.* 1997] V. Gabrel, A. Moulet, C. Murat, and V. Paschos. A New Single Model and Derived Algorithms for the Satellite Shot Planning Problem Using Graph Theory Concepts. *Annals of Operations Research*, 69:115–134, 1997.

[Globus *et al.* 2002] A. Globus, J. Crawford, J. Lohn, and R. Morris. Scheduling Earth Observing Fleets Using Evolutionary Algorithms: Problem Description and Approach. In *Proc. of the Third IWPSS NASA Workshop*, Houston, TX, USA, 2002.

[Gout *et al.* 1999] J. Gout, S. Fleury, and H. Schindler. A New Design Approach of Software Architecture for an Autonomous Observation Satellite. In *Proc. of i-SAIRAS-99*, pages 491–497, Noordwijk, The Netherlands, 1999.

[Honvault *et al.* 2001] C. Honvault, C. Simon, P. David, and E. Bornschlegl. An Autonomous On-board Mission Manager for LEO Satellite Powered by the ERC32SC. In *Proc. of the ESA Workshop on On-Board Autonomy*, Noordwijk, The Netherlands, 2001.

[Khatib *et al.* 2003] L. Khatib, J. Frank, D. Smith, R. Morris, and J. Dungan. Interleaved Observation Execution and Rescheduling on Earth Observing Systems. In *Proc. of the AIPS-03 Workshop on "Plan Execution"*, Trento, Italy, 2003.

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

[Muraoka *et al.* 1998] H. Muraoka, R. Cohen, T. Ohno, and N. Doi. ASTER Observation Scheduling Algorithm. In *Proc. of SpaceOps-98*, Tokyo, Japan, 1998.

[Muscettola *et al.* 1998] N. Muscettola, P. Nayak, B. Pell, and B. Williams. Remote Agent: To Boldly Go Where No AI System Has Gone Before. *Artificial Intelligence*, 103(1-2):5–48, 1998.

[Niezette 2000] M. Niezette. Mission Planning Systems for Earth Observation Missions. In *Proc. of the Second IWPSS NASA Workshop*, pages 19–24, San Francisco, CA, USA, 2000.

[Pearl 1983] J. Pearl. On the Nature of Pathology in Game Searching. *Artificial Intelligence*, 20:427–453, 1983.

[Pemberton 2000] J. Pemberton. Towards Scheduling Over-constrained Remote-sensing Satellites. In *Proc. of the Second IWPSS NASA Workshop*, pages 84–89, San Francisco, CA, USA, 2000.

[Potin 1998] P. Potin. End-to-End Planning Approach for Earth Observation Mission Exploitation. In *Proc. of SpaceOps-98*, Tokyo, Japan, 1998.

[Potter and Gasch 1998] W. Potter and J. Gasch. A Photo Album of Earth: Scheduling LANDSAT 7 Mission Daily Activities. In *Proc. of SpaceOps-98*, Tokyo, Japan, 1998.

[Sherwood *et al.* 1998] R. Sherwood, A. Govindjee, D. Yan, G. Rabideau, S. Chien, and A. Fukunaga. Using ASPEN to Automate EO-1 Activity Planning. In *Proc. of the IEEE Aerospace Conference*, 1998.

[Vasquez and Hao 2001] M. Vasquez and J.K. Hao. A Logic-constrained Knapsack Formulation and a Tabu Algorithm for the Daily Photograph Scheduling of an Earth Observation Satellite. *Journal of Computational Optimization and Applications*, 20(2):137–157, 2001.

[Verfaillie and Bornschlegl 2000] G. Verfaillie and E. Bornschlegl. Designing and Evaluating an On-line On-board Autonomous Earth Observation Satellite Scheduling System. In *Proc. of the Second IWPSS NASA Workshop*, pages 122–127, San Francisco, CA, USA, 2000.

[Wolfe and Sorensen 2000] W. Wolfe and S. Sorensen. Three Scheduling Algorithms applied to the Earth Observing Systems Domain. *Management Science*, 46(1):148–168, 2000.

[Zilberstein 1996] S. Zilberstein. Using Anytime Algorithms in Intelligent Systems. *AI Magazine*, 17(3):73–83, 1996.