Decision upon observations and data downloads by an autonomous Earth surveillance satellite

Cédric Pralet and **Gérard Verfaillie**
ONERA, 2 av. Édouard Belin, BP 74025, F-31055 Toulouse Cédex 4, France
{Cedric.Pralet,Gerard.Verfaillie}@onera.fr

Abstract

In the context of the French CNES-ONERA-LAAS AGATA project (Charmeau & Bensana 2005) which aims at developing technical means of improving spacecraft autonomy, we are working on the scenario of an autonomous Earth low-orbiting satellite dedicated to the detection and observation of ground phenomena such as forest fires or volcanic eruptions (Damiani, Verfaillie, & Charmeau 2005).

Following detection of ground phenomena in front of the satellite, alarms are automatically sent to the ground using geo-orbiting relay communication satellites and observation requests are generated on-board. Because the satellite is not permanently in communication with a ground control center and because first observations can start only one minute after detection, decisions upon observations must be made on-board. The same way, decisions upon data downloads following observations must be made on-board. Although observations and data downloads can be performed concurrently, decisions upon observations and data downloads cannot be made independently as previously proposed in (Damiani, Verfaillie, & Charmeau 2005), because both consume energy and because observations consume free memory while data downloads produce it.

In this paper, after describing the mission and the satellite, we show how the task of making decisions upon observations and data downloads can be tackled using an architecture which combines a reactive task and a deliberative one, how the reactive task can be implemented using the synchronous language *Esterel* (Berry & Gonthier 1992) and how the deliberative one can use a constraint-based model inspired from (Verfaillie, Pralet, & Lemaître 2007) and an anytime iterated stochastic greedy search algorithm inspired from (Bresina 1996).

The AGATA project

The AGATA project (Charmeau & Bensana 2005) (http://www.agata.fr) is a joint project between CNES (French Space Agency), ONERA (French Aerospace Lab), and LAAS-CNRS (French Research Center), in Toulouse, France, which started in 2004 and whose objective is to develop the technical background necessary to develop spacecraft on-board autonomy, in order to increase spacecraft availability, reactivity, and dependability. To do so, the main research topics of the project are (i) the design of a control architecture for autonomy, (ii) methods for autonomous failure detection, identification, and recovery, (iii) methods for autonomous decision-making, (iv) approaches for control validation, and (v) means of a correct interaction between an autonomous spacecraft and human ground control operators.

The Hotspot mission and the Frankie satellite

The AGATA project uses several space missions as examples and benchmarks for concepts, approaches, methods and algorithms. The first of these missions, referred to as *Hotspot* and already considered in (Damiani, Verfaillie, & Charmeau 2005), assumes an autonomous Earth low-orbiting satellite, referred to as *Frankie*, dedicated to the detection and observation of hot ground phenomena such as forest fires or volcanic eruptions.

The *Frankie* satellite is moving around the Earth using a circular, low altitude orbit (800 km), with a 60° angle with regard to the equatorial plane. It is permanently maintained in a geocentric attitude. Its equipments include:

1. a wide swath detection instrument, referred to as the *hotspotter*, equipped with a line of infrared detectors, pointing 30° ahead the satellite, permanently active, analyzing data instantly, and able to detect hot areas at the Earth surface and to distinguish between forest fires and volcanic eruptions;
2. a low rate antenna, able to send alarms to geo-orbiting relay satellites in case of hot area detection; once received by a geo-orbiting satellite and then by a ground station, this alarm is sent to the dedicated mission center (at least one for forest fires and one for volcanic eruptions).
3. a narrow swath observation instrument, referred to as the *imager*, equipped with a line of optical or near-infrared detectors, pointing to the nadir, using a mobile sight mirror to point laterally anywhere in the swath of the hotspotter, active on demand, but unable to analyze data;
4. a mass memory able to record observation data;
5. a high rate antenna able to download observation data to ground stations; once received, this data is forwarded to the dedicated mission center.

In order to avoid useless alarms on known hot areas, the satellite is provided with a black list of areas on which detection must be ignored.

Each time a hot area at the Earth surface which does not belong to the black list is detected by the hotspotter, an observation request is generated with the highest priority level (3). When this area has been observed by the imager, this request becomes a regular observation request of priority 2 (observation required typically four times each day). But, when this area has not been detected for a given time (typically one day), it becomes a regular observation request of priority 1 (observation required typically only each day). Finally, when this area has not been detected for a given time (typically seven days), it is removed from the set of observation requests.

In addition to the observation requests that are generated on-board by the hotspotter, observation requests can be sent to the satellite from ground mission centers in order to observe regularly some specific areas (for example, inactive volcanoes). Once received, these regular observation requests have the lowest priority level (0). But if a detection occurs on an area whose associated observation request has priority 0 or 1, its priority is immediately increased to 3. Whatever its priority and its origin (ground or board) are, any observation request can be removed at any time by a ground mission center.

With each regular observation request and each period of time (for example, one day for a request of priority 1) are associated one or several temporal windows which would allow the request to be satisfied in this period of time, each one with fixed start and end times: because the satellite is not agile, there is no temporal freedom for observation, except the choice of one window among a set of possible ones.

After an observation has been performed, the associated data recorded in the mass memory must be downloaded as soon as possible, using the visibility window of any ground station. Data associated with several requests can be downloaded sequentially within a ground station visibility window, one immediately after the other. But, unlike observation, data downloading can be freely set within this window and one may take advantage of this temporal freedom.

Moreover, nothing prevents from removing from the mass memory the data associated with an observation request of low priority to record data associated with an observation request of higher priority.

From the user point of view, what is really important is the data that is downloaded to the ground and, more precisely, (i) the priorities of the associated requests, (ii) the regularity of observations, and (iii) the freshness of delivered data, that is the difference between delivery and observation times.

Decision-making upon observations and data downloads

Fully satisfying all the observation requests is unfortunately impossible because of temporal conflicts (it is impossible to point the imager mirror to two different ground areas at the same time; moreover, moving the mirror between two successive observations takes a significant time) and of resource conflicts (observations and data downloads both consume energy; observations consume mass memory, whereas downloads release it; both resources are limited on-board).

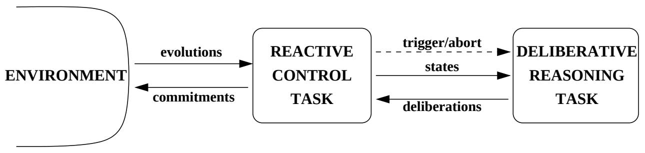

Figure 1: **Global schema of the interactions between the environment, a reactive task and a deliberative one**

Choices must be consequently made, aiming at the highest possible user satisfaction.

More precisely, for each known elementary observation request, one must decide upon (i) the window used for observation (null when no observation is decided) and (ii) the window used for data downloading (null when no downloading is decided). Moreover, for each visibility window of a ground station, one must decide upon (iii) the time at which starting downloading. Finally, for each data already recorded, one must decide upon (iv) keeping it in the mass memory or removing it.

In previous works on a similar scenario (Damiani, Verfaillie, & Charmeau 2005), we tried to decompose the problem and to manage separately observation decisions and data downloading decisions. This solution was not completely satisfactory, mainly because of a systematic priority given to downloading. So, the model and the algorithms presented in this paper aim at dealing in one go with the whole problem: observation and downloading decisions.

A reactive/deliberative architecture for decision-making

Because the first observation of a detected hot area can start around one minute after detection and because the satellite is rarely in communication with its control center, decisions about observations and downloads must be made on-line and on-board. Moreover, they must be made before strict deadlines: it is useless to decide upon the observation of a given area if it is too late to trigger this observation because of the movement of the satellite on its orbit.

Because the problem is dynamic (observation requests may arrive at any time either from the hotspotter, or from the ground; actual energy and memory consumptions may differ from what is forecast by the spacecraft model), it may be counterproductive to build and to commit to precise activity plans far ahead. In fact, what is really necessary is to decide upon the next actions to be performed just before triggering them.

This is why we adopted a decisional architecture built around two interacting tasks, one reactive and one deliberative, inspired from (Lemaître & Verfaillie 2007). Figure 1 presents the global schema of such an architecture:

1. the reactive task is in charge of the interaction between the environment (physical system, users) and the decisional architecture; it receives from the environment information about the way the latter evolves; from that, it is at any time able to compute a deadline for action commitment and thus for decision-making (for example, the start

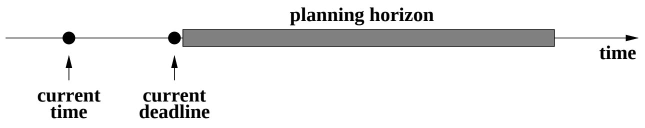

Figure 2: **Deadline and planning horizon**

time of the next data downloading window); when this deadline is immediate, it makes a reactive decision and commits immediately to the associated action; but, when it is not, it can trigger the deliberative task by providing it with a temporal reasoning horizon (starting at the deadline) and the state of the environment as it can be forecast just before the beginning of this horizon (that is, just before the deadline); see Figure 2;

2. when triggered by the reactive task, the deliberative one is in charge of building activity plans over the specified temporal horizon, from which it is possible to extract the first action or set of actions, to be sent to the reactive task as a decision proposal; it is assumed to have an anytime behavior, that is to be able to build quickly a first activity plan and to improve on it as long as computing time is available; each time a better quality activity plan is produced, its first action or set of actions is extracted and sent to the reactive task, when it differs from the previous one;
3. when the deadline occurs, the reactive task must make a decision and commit to the associated action; when no proposal is available because the deliberative task did not have enough time to build a plan, the reactive task makes a reactive decision; but, when a proposal is available, the reactive task checks consistency between this proposal and the current state of the environment before committing to the associated action (to guarantee, for example, that this action will not endanger the satellite).

Abstractly, the reactive task reacts instantly to any event from the environment or from the deliberative task. In fact, it is assumed to be able, in case of any event arrival, to compute its reaction and to update its internal state before the next event arrival. We use the synchronous language *Esterel* (Berry & Gonthier 1992) to implement it. As for the deliberative task, it can be implemented in any basic language, such as *Java*. Finally, for the interaction between the reactive and deliberative tasks, we use the so-called *Task Esterel* mechanism which allows any external task to be triggered, synchronized, and aborted from an *Esterel* program.

What we present now in the sequel of this paper is the model and the algorithm used by the deliberative task to build an activity plan over the specified temporal horizon ahead the deadline.

A constraint-based planning model

Problem data

We assume that, when triggered, the deliberative task is provided with the data summarized in Table 1.

The first important piece of information is related to the planning temporal horizon. Its start time STA is the current deadline. Its end time END is the end time of the last eclipse window we want to be considered by planning. Choosing systematically the end time of an eclipse window as the end of the planning horizon is justified by the need to check that the plan will allow the satellite to end an eclipse window with a sufficient level of energy.

The second piece of information is related to the observations OBP that can be performed over the planning horizon and to the observations OBM that are already recorded at time STA and can be downloaded over the planning horizon. When an observation is currently performed at time STA, it is considered as already recorded and thus included in OBM (the set made of the observation initially in progress, if any, is denoted OBC). But, when observations are currently downloaded at time STA, they are not included in OBM. With each observation in $OB = OBP \cup OBM$, are associated its value (function of its priority), its size in memory, and the duration of its downloading. With each observation in OBP, are associated the duration of its observation and its observation windows over the planning horizon. An observation window is itself defined by its starting time and the required mirror orientation. Finally, with each observation in OBM, is associated the end time of the observation which has already been performed.

Other pieces of information are related to the downloading and eclipse windows over the planning horizon.

Finally the deliberative task is provided with information about the satellite state just before time STA.

Reasoning assumptions

The planning reasoning task is based on some assumptions:

1. there is no uncertainty about the way the satellite state evolves and on the effects of actions; for example, when triggered, an action always succeeds; for example too, energy and memory productions and consumptions are perfectly known; in fact, these uncertainties are taken into account by the whole decisional architecture via reaction and replanning;
2. observations and data downloading can be performed in parallel without any restriction;
3. there is a conflict between two observation windows associated with two different observations if there is a temporal overlapping between both windows or not enough time to move the mirror from the first required orientation to the second one;
4. when the satellite is not in eclipse, solar panels provide it with enough energy to execute any set of actions in parallel and, in addition, to recharge batteries;
5. ground stations are not dedicated to a kind of observation; observations can thus be downloaded to any ground station; there is no overlapping between ground station visibility windows;
6. a data downloading involves a set of recorded observations; selected observations are downloaded one immediately after the other within a ground station visibility window; too old observations are not downloaded;

Observations	
OBP	set of observations to be performed
OBM	set of observations initially memorized and to be downloaded
$OB = OBP \cup OBM$ $(OBP \cap OBM = \emptyset)$	set of observations
$OBC \subseteq OBM$	set made of the observation initially in progress, if any ($\emptyset$ otherwise)
Observation features	
$\forall o \in OB, VL(o)$	value of observation o
$\forall o \in OB, SZ(o)$	memory used by observation o
$\forall o \in OB, DDO(o)$	downloading duration of observation o
$\forall o \in OBP : DO(o)$	duration of observation o
$\forall o \in OBM, EO(o)$	end time of observation o
Observation windows	
$\forall o \in OBP, NO(o)$	number of possible windows for observation o
$\forall o \in OBP, \forall k, 1 \leq k \leq NO(o), SO(o,k)$	start time of the kth window for observation o
$\forall o \in OBP, \forall k, 1 \leq k \leq NO(o), OR(o,k)$	mirror orientation required for observation o in the kth window
Downloading windows	
ND	number of possible windows for downloading
$\forall k, 1 \leq k \leq ND, SD(k)$	start time of the kth downloading window
$\forall k, 1 \leq k \leq ND, DD(k)$	duration of the kth downloading window
Eclipse windows	
NE	number of eclipse windows
$\forall k, 1 \leq k \leq NE, SE(k)$	start time of the kth eclipse window
$\forall k, 1 \leq k \leq NE, DE(k)$	duration of the kth eclipse window
Initial state	
EO_0	end time of the observation initially in progress, if any (STA otherwise)
ED_0	end time of the downloading initially in progress, if any (STA otherwise)
EE_0	end time of the eclipse initially in progress, if any (STA otherwise)
SZ_0	memory which will be released by the downloading initially in progress
EN_0	initial level of energy
MM_0	initial level of free memory
OR_0	initial mirror orientation
Satellite features	
$ENmin$	minimum level of energy
$ENmax$	maximum level of energy
$MMmax$	maximum level of free memory
$Psun$	power provided by the solar panels
Pob	power consumed by observation
Pdl	power consumed by data downloading
$Psat$	power consumed by the platform
MS	mirror orientation speed
DV	validity duration of any recorded observation
Reasoning horizon	
STA	start time of the planning horizon
$END = \max_{1 \leq k \leq NE} (SE(k) + DE(k))$	end time of the planning horizon

Table 1: **Problem data**

7. when an observation is triggered, all the memory needed for data recording is immediately consumed; but, when a data downloading is triggered, the memory is freed only at the end of the downloading; at any time, it is possible to remove a recorded observation in order to record another one;

8. the free memory does not change and the energy evolves linearly between two successive changes in the satellite state in terms of observation, data downloading, or eclipse mode.

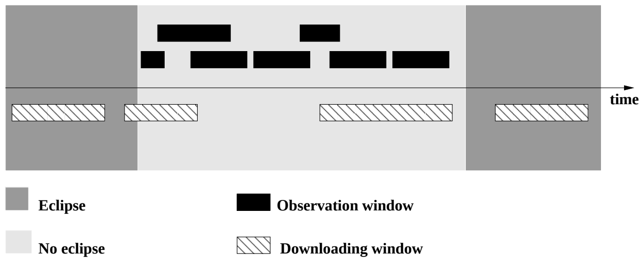

Figure 3: **Observations, data downloads, and eclipses**

Modeling overview

We adopt a constraint-based modeling approach, that is based on variables and constraints, inspired from (Verfaillie, Pralet, & Lemaître 2007). Variables are used to represent the decisions and the satellite state. Constraints are used to restrict possible decisions, mainly due to physical limitations, and to express the impact of decisions on the satellite state.

We observe that, to represent the way the satellite state evolves and the impact of decisions, it suffices to consider the satellite state at a finite set of steps, which are the start and end times of observations, downloads, and eclipses, that is the times at which the satellite state changes in terms of observation, data downloading, or eclipse mode (see Figure 3). This results from the assumption that energy evolves linearly between two successive steps and that the other satellite state features, including free memory, do not change between two successive steps.

Problem variables

The problem variables and their domains are summarized in Table 2. They are all discrete, including time.

Following the modeling overview, we distinguish two kinds of variables: (i) the static decision variables which represent the possible decisions in terms of observation, memory management, and data downloading, and (ii) the dynamic state variables which represent the satellite state evolution.

The main decision variables are, (i) for each $o \in OBP$, the chosen observation window $no(o)$ (null in case of no observation), (ii) for each $o \in OBM$, the choice $fg(o)$ of forgetting it or not, (iii) for each $o \in OB$, the chosen downloading window $nd(o)$ (null in case of no downloading), and (iv) for each downloading window, the downloading start time $sd(k)$ within this window. All the other decision variables are consequences of the previous ones.

The dynamic state variables are duplicated at each step. At each step i, they allow to represent the current time t_i, the current levels of energy and free memory (en_i and mm_i), and the current observation, downloading, and eclipse modes (ob_i, dl_i, and ec_i).

Problem constraints

Constraints on static decision variables The following constraints restrict the possible decisions to take into account physical limitations and hard user requirements.

$$\begin{array}{l} \forall o \in OBP: \\ (no(o) \neq 0) \rightarrow (so(o) = SO(o, no(o))) \\ (no(o) \neq 0) \rightarrow (eo(o) = so(o) + DO(o)) \\ (no(o) = 0) \rightarrow (so(o) = END + 1) \\ (no(o) = 0) \rightarrow (eo(o) = END + 1) \end{array}$$

The start time of an observation is the start time of the chosen observation window. The end time is the start time plus the observation duration. If no observation window is chosen, then start and end times are arbitrarily set to $END+1$, beyond the temporal horizon.

$$\begin{array}{l} \forall k \in \{1, \ldots, ND\}: \\ ed(k) = sd(k) + \sum_{o \in OB} (nd(o) = k) \cdot DDO(o) \\ (sd(k) = ed(k)) \rightarrow (sd(k) = ed(k) = SD(k)) \end{array}$$

The end time of a downloading is its start time plus the sum of the durations of all the downloads that are performed within the downloading window. If no downloading is performed, then start and end times are arbitrarily set to the start time of the window.

$$\begin{array}{l} \forall o \in OBP: \\ (no(o) = 0) \rightarrow (nd(o) = 0) \\ (nd(o) \neq 0) \rightarrow (eo(o) \leq sd(nd(o)) \leq eo(o) + DV) \end{array}$$

If an observation is not performed, then it is not downloaded. Else, the associated downloading starts after the observation end time and before the observation validity end time.

$$\begin{array}{l} \forall o \in OBM: \\ (fg(o) = 1) \rightarrow (nd(o) = 0) \\ (nd(o) \neq 0) \rightarrow (sd(nd(o)) \leq EO(o) + DV) \end{array}$$

If a recorded observation is removed, then it is not downloaded. Else, the downloading starts before the observation validity end time.

$$\begin{array}{l} \forall o \in OBC: \\ (nd(o) \neq 0) \rightarrow (EO(o) \leq sd(nd(o))) \end{array}$$

If an observation is in progress at time STA, then the associated downloading starts after the observation end time.

$$\begin{array}{l} \forall o \in OBP, \forall k \in \{1, \ldots, NO(o)\} \\ \forall o' \in OBP, \forall k' \in \{1, \ldots, NO(o')\} \mid \\ ((o \neq o') \ \wedge \ (SO(o', k') \leq SO(o, k)) \ \wedge \\ (SO(o, k) < SO(o', k') + DO(o') + \frac{|OR(o,k) - OR(o',k')|}{MS})): \\ (no(o') = k') \rightarrow (no(o) \neq k) \end{array}$$

If two observation windows associated with two different observations are temporally in conflict and if the first one is chosen, then the second one cannot be chosen.

$$\begin{array}{l} \forall o \in OBP, \forall k \in \{1, \ldots, NO(o)\} \mid \\ (SO(o, k) < EO_0 + \frac{|OR(o,k) - OR_0|}{MS}): \\ (no(o) \neq k) \end{array}$$

If an observation window is in conflict with the observation initially in progress or with the initial mirror orientation, then it cannot be chosen.

Static decision variables	
$\forall o \in OBP, no(o) \in \{0, \ldots, NO(o)\}$	window used for observation o
$\forall o \in OBP, so(o) \in \{STA, \ldots, END+1\}$	start time of observation o
$\forall o \in OBP, eo(o) \in \{STA, \ldots, END+DO(o)\}$	end time of observation o
$\forall o \in OBM, fg(o) \in \{0, 1\}$	to forget or not observation o
$\forall o \in OB, nd(o) \in \{0, \ldots, ND\}$	window used for downloading observation o
$\forall k, 1 \leq k \leq ND, sd(k) \in \{SD(k), \ldots, SD(k)+DD(k)\}$	start time of the kth downloading
$\forall k, 1 \leq k \leq ND, ed(k) \in \{SD(k), \ldots, SD(k)+DD(k)\}$	end time of the kth downloading

Dynamic state variables	
$\forall i, t_i \in \{STA, \ldots, END\}$	current time
$\forall i, ob_i \in \{0, 1\}$	current observation mode
$\forall i, dl_i \in \{0, 1\}$	current downloading mode
$\forall i, ec_i \in \{0, 1\}$	current eclipse mode
$\forall i, en_i \in \{ENmin, \ldots, ENmax\}$	current level of energy
$\forall i, mm_i \in \{0, \ldots, MMmax\}$	current level of free memory

Table 2: **Problem variables**

Constraints on the initial satellite state

$$
\begin{aligned}
&t_0 = STA\\
&ob_0 = (EO_0 > STA)\\
&dl_0 = (ED_0 > STA)\\
&ec_0 = (EE_0 > STA)\\
&en_0 = EN_0\\
&mm_0 = MM_0 + \textstyle\sum_{o \in OBM-OBC} fg(o) \cdot SZ(o)
\end{aligned}
$$

The initial level of free memory takes into account the observations one decides to forget.

Constraints on the satellite state transitions The following constraints apply at each step i.

$$
\begin{aligned}
&t_1 = \min\{t \in TO \cup TD \cup TE\}\\
&((i \geq 1) \wedge (t_i < END)) \rightarrow\\
&(t_{i+1} = \min\{t \in TO \cup TD \cup TE \mid t > t_i\})
\end{aligned}
$$

with:

$$
\begin{aligned}
&TO = \left(\cup_{o \in OBP \mid no(o) \neq 0}\{so(o), eo(o)\}\right) \cup\\
&(\{EO_0\} \text{ if } ob_0 = 1,\ \emptyset \text{ otherwise})\\
&TD = \left(\cup_{k \in \{1,\ldots,ND\} \mid sd(k) \neq ed(k)}\{sd(k), ed(k)\}\right) \cup\\
&(\{ED_0\} \text{ if } dl_0 = 1,\ \emptyset \text{ otherwise})\\
&TE = \left(\cup_{k \in \{1,\ldots,NE\}}\{SE(k), SE(k)+DE(k)\}\right) \cup\\
&(\{EE_0\} \text{ if } ec_0 = 1,\ \emptyset \text{ otherwise})
\end{aligned}
$$

These constraints express how the successive step times to consider depend on decision variables. Note the difference between a step i and its associated time t_i.

$$
\begin{aligned}
&(t_i = END) \rightarrow (t_{i+1} = END)\\
&(t_i = END) \rightarrow (ob_{i+1} = ob_i)\\
&(t_i = END) \rightarrow (dl_{i+1} = dl_i)\\
&(t_i = END) \rightarrow (ec_{i+1} = ec_i)\\
&(t_i = END) \rightarrow (en_{i+1} = en_i)\\
&(t_i = END) \rightarrow (mm_{i+1} = mm_i)
\end{aligned}
$$

These constraints express that, from the planning point of view, the satellite state is frozen at the end of the temporal horizon. Nothing is considered beyond.

$$
\begin{aligned}
&(ob_i = 1) \leftrightarrow ((t_i < EO_0) \vee\\
&(\exists o \in OBP,\ (so(o) \leq t_i < eo(o))))
\end{aligned}
$$

$$
\begin{aligned}
&(dl_i = 1) \leftrightarrow ((t_i < ED_0) \vee\\
&(\exists k \in \{1, \ldots, ND\},\ (sd(k) \leq t_i < ed(k))))
\end{aligned}
$$

$$
\begin{aligned}
&(ec_i = 1) \leftrightarrow ((t_i < EE_0) \vee\\
&(\exists k \in \{1, \ldots, NE\},\ (SE(k) \leq t_i < SE(k) + DE(k))))
\end{aligned}
$$

These constraints express how the observation, downloading, and eclipse modes depend on decision variables.

$$
\begin{aligned}
&(t_i < END) \rightarrow\\
&(en_{i+1} = \min(ENmax, en_i + (t_{i+1} - t_i) \cdot\\
&((1 - ec_i) \cdot P_{sun} - ob_i \cdot P_{ob} - dl_i \cdot P_{dl} - P_{sat})))
\end{aligned}
$$

$$
\begin{aligned}
&(t_i < END) \rightarrow (mm_{i+1} = mm_i -\\
&(\textstyle\sum_{o \in OBP}(t_{i+1} = so(o)) \cdot SZ(o)) +\\
&(\textstyle\sum_{o \in OB}(t_{i+1} = ed(nd(o))) \cdot SZ(o)) +\\
&((t_{i+1} = ED_0) \cdot SZ_0) +\\
&((t_{i+1} = EO_0) \cdot (\textstyle\sum_{o \in OBC} fg(o) \cdot SZ(o))))
\end{aligned}
$$

Finally, these constraints express how the energy and free memory levels evolve step after step. Changes in the level of energy take into account the current observation, downloading, and eclipse modes. Memory is consumed at the beginning of an observation. It is freed at the end of a downloading. Note that the constraints on the minimum and maximum levels of energy and memory are stated by the domain definitions of the variables en_i and mm_i.

Optimization criterion We choose to optimize the following criterion which is a weighted sum of two criteria: (i) the sum of the values of the downloaded observations and (ii) the sum of the values of the performed observations that have not been downloaded yet, with the weighting parameter α set between 0 and 1:

$$(\sum_{o \in OB} (nd(o) \neq 0) \cdot VL(o)) + \\ \alpha \cdot (\sum_{o \in OBP} (no(o) \neq 0) \cdot (nd(o) = 0) \cdot VL(o))$$

Any other optimization criterion could be considered.

An iterated stochastic greedy search algorithm

The deliberative task must be able to quickly provide the reactive task with "good" decisions, and to progressively improve their quality during the available time. To get such an anytime behavior for the deliberative task, we use an *iterated stochastic greedy search algorithm* (Bresina 1996; Cicirello & Smith 2005) to reason about the decision model previously introduced. The principles of this algorithm are given below.

First, this algorithm is *greedy* in the sense that it explores the search space without backtracking on the selected choices. More precisely, we consider the start and end times of the observation, download, and eclipse windows in a chronological order, starting from STA, without reversing any anterior choice:

1. when the time currently considered is the start time of the kth window for observation o, the satellite follows some heuristics for deciding to trigger or not the observation (decision $no(o) = k$); if memory is needed, the satellite can decide to forget observations o' initially recorded (decision $fg(o') = 1$); if decision $no(o) = k$ is made, state variable values are updated, and the start and end times of all posterior observation windows for o are ignored in the sequel;
2. when the time currently considered is the start time of the kth download window, the satellite can decide to download observations by following some heuristics; for each downloaded observation o, decision $nd(o) = k$ is recorded; in our current implementation, the start time of the kth download is always chosen as $sd(k) = SD(k)$;
3. when the time currently considered is the start or end time of an eclipse window, the end time of a performed observation window, or the end time of a download in progress, the current state of the satellite is updated.

The greedy search terminates when the END time is reached. All unassigned decision variables $no(o)$, $nd(o)$, $fg(o)$ are set to 0 and the quality of the plan obtained is evaluated. It is important to note that before advising any decision, the heuristics we use perform some simple checks by considering the decision model and the current state of the satellite. These fast checks prune unfeasible decisions and notably ensure that all hard constraints of the decision model are satisfied when the search terminates.

Second, the algorithm is *stochastic* since choice heuristics are stochastic:

1. when facing the start time of an observation window, the probability of performing this observation depends on its value and of the amount of memory and energy currently available; if several observation windows have a common start time, a highest value observation is selected (ties broken randomly); in case of memory needs, observations initially recorded are forgotten with some probability function of their value;
2. when facing the start time of a download window, the satellite chooses to download highest value observations first (ties broken randomly) and the number of observations downloaded is a random quantity function of the current level of energy.

These stochastic heuristics are chosen quite arbitrarily. They could be learned for a more efficient approach.

Last, the search is *iterated*: each time a stochastic greedy search terminates, the algorithm restarts from an empty plan and performs a new stochastic greedy chronological search starting from STA. When the quality of the best known plan is improved, the deliberative task can send the first action (or set of actions) of this plan to the reactive task, when it differs from the previously sent one. The restarts combined with the stochastic choices enable the search to be diversified, by opposition to a deterministic greedy search algorithm which would always make the same choices.

Experiments

Experiments were performed for several input parameters. We give the results obtained for one parameter setting. We consider a temporal horizon $[STA, END]$ covering 3 revolutions. During these 3 revolutions, 100 observations must be performed. For each observation, from 1 to 3 observation windows are available. The 3 revolutions contain 7 download windows. Owing to the download duration of each observation, at most 8 observations can be downloaded during each download window. The mass memory enables 30 observations to be recorded on-board (in our experiments, all observations have the same size). Initially, half of the memory is used, i.e. 15 observations are recorded. Apart from a few cases, the initial level of energy does not constrain the feasible decisions. Experiments are performed on a Pentium 4, 2.4Ghz, 512Mo RAM. The iterated stochastic greedy search is implemented via the COMET constraint-based local search solver (Hentenryck & Michel 2005; Hentenryck & Michel).

Figure 4 gives the mean, worst and best evolutions of the plan quality, obtained from 20 iterated stochastic greedy searches. Each iterated stochastic greedy search uses 100 restarts and each restart takes about 150ms. The results obtained show that the algorithm, which produces good solutions in less than 2s, is anytime. The worst and best cases evolutions show the robustness of this stochastic algorithm.

The iterated greedy stochastic search was compared with an exact tree search using constraint propagation to prune branches in the search tree. This tree search first assigns static decision variables $no(o)$, $nd(o)$ and $fg(o)$, by following some stochastic heuristics, functions of the values of observations. As we consider that $sd(k) = SD(k)$ in our experiments, the assignment of these variables completely determines the assignment of all other variables. The search is complete since choices not selected initially are systematically explored when backtracks occur. This tree search

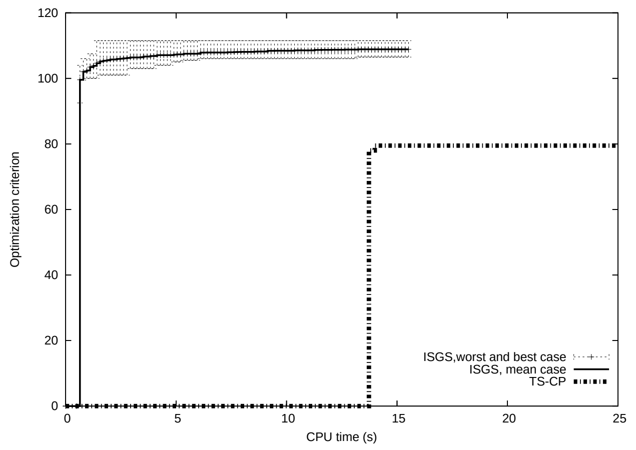

Figure 4: **Evolution of the plan optimization criterion as a function of CPU time; ISGS: Iterated Stochastic Greedy Search; TS-CP: Tree-Search with Constraint Propagation.**

algorithm was tested via OPL Studio 3.71 and Ilog Solver 6.1 (Hentenryck 1999; Ilog).

Figure 4 shows that the complete tree search using constraint propagation is not anytime. Not only no solution is produced in less than 13s, but also the solution produced is of bad quality. After 5 minutes of computing, the quality of the best plan found is still far from that found with the iterated stochastic greedy search. On cases for which memory and energy are more constrained, the tree search finds no solution in 5 minutes, whereas the iterated stochastic greedy search finds good solutions in a few seconds. These results can be explained by multiple reasons: first, the tested tree search algorithm loses a lot of time to propagate constraints on the dynamic variables on the whole reasoning horizon; second, tree search assigns the static decision variables first, without considering directly their chronological effects on the satellite state; third, tree search takes a long time before backtracking on initial choices, which may be inappropriate.

Considering space, the iterated greedy stochastic search uses just 24Mo to run. This size could be decreased by using a dedicated implementation, since parts of this memory usage come from data structures used by COMET. In fact, the implemented iterated stochastic greedy search uses constraints on the satellite state as a transition function of a dynamic system: it maintains only the current state of the satellite and updates this current state each time a decision is made. This differs from the tree search method, which duplicates all dynamic variables and transition constraints over a horizon of H time steps, chosen high enough so that $t_H = END$ necessarily holds. Such a duplication leads to a higher memory consumption (290Mo).

Conclusion

In this article, we presented a constraint-based model for on-board decision upon observations and data downloads, used in the context of the *Hotspot* mission. This model enables both temporal and resource features to be taken into account. From an algorithmic point of view, we showed the anytime behavior of an iterated stochastic greedy search considering the decisions chronologically. The interest of this approach is that it allows the search to be quickly diversified, by opposition to a tree search scheme. A possible future algorithmic direction could be to try and mix these two schemes, by first diversifying search to produce good quality plans, and then by intensifying search starting from these plans.

In the context of AGATA, the current objective is to implement the deliberative task of the mission planning module of the Frankie satellite, based on the proposed decision model, including its interaction with the reactive task, in order to simulate the whole mission control system.

References

[Berry & Gonthier 1992] Berry, G., and Gonthier, G. 1992. The Esterel Synchronous Programming Language: Design, Semantics, Implementation. *Science of Computer Programming* 19(2):87–152.

[Bresina 1996] Bresina, J. 1996. Heuristic-Biased Stochastic Sampling. In *Proc. of the 13th National Conference on Artificial Intelligence (AAAI-96)*, 271–278.

[Charmeau & Bensana 2005] Charmeau, M.-C., and Bensana, E. 2005. AGATA: A Lab Bench Project for Spacecraft Autonomy. In *Proc. of the 8th International Symposium on Artificial Intelligence, Robotics, and Automation for Space (i-SAIRAS-05).*

[Cicirello & Smith 2005] Cicirello, V., and Smith, S. 2005. Enhancing Stochastic Search Performance by Value-Biased Randomization of Heuristics. *Journal of Heuristics* 11(1):5–34.

[Damiani, Verfaillie, & Charmeau 2005] Damiani, S.; Verfaillie, G.; and Charmeau, M.-C. 2005. Cooperating On-board and On the ground Decision Modules for the Management of an Earth Watching Constellation. In *Proc. of the 8th International Symposium on Artificial Intelligence, Robotics, and Automation for Space (i-SAIRAS-05).*

[Hentenryck & Michel] Hentenryck, P. V., and Michel, L. Comet. http://www.comet-online.org/.

[Hentenryck & Michel 2005] Hentenryck, P. V., and Michel, L. 2005. *Constraint-based Local Search.* MIT Press.

[Hentenryck 1999] Hentenryck, P. V. 1999. *The OPL Optimization Programming Language.* MIT Press.

[Ilog] Ilog OPL Studio http://www.ilog.com/products/oplstudio/.

[Lemaître & Verfaillie 2007] Lemaître, M., and Verfaillie, G. 2007. Interaction between reactive and deliberative tasks for on-line decision-making. In *Proc. of the ICAPS-07 Worshop on "Planning and Plan Execution for Real-world Systems".*

[Verfaillie, Pralet, & Lemaître 2007] Verfaillie, G.; Pralet, C.; and Lemaître, M. 2007. Constraint-based modelling of discrete event dynamic systems. In *Proc. of the CP/ICAPS-07 Workshop on "Constraint Satisfaction Techniques for Planning and Scheduling Problems".*