MrSPOCK: a Long-term Planning Tool for MARS EXPRESS

Amedeo Cesta and **Gabriella Cortellessa** and **Simone Fratini** and **Angelo Oddi**
ISTC-CNR, Institute of Cognitive Science and Technology - National Research Council of Italy
Via S. Martino della Battaglia, 44, 00185 Rome, Italy.
name.surname@istc.cnr.it

Abstract

This paper describes MrSPOCK (MARS EXPRESS Science Plan Opportunities Coordination Kit), a decision support system for long-term mission planning. The work was carried out within the *Advanced Planning and Scheduling Initiative* (APSI), a project funded by ESA which aims at creating the basis for a general, flexible and reusable software *framework* (named here APSI-TRF – APSI Timeline-based Representation Framework) to facilitate injection of AI Planning and Scheduling in space missions to enhance ESA mission operation management performance. The paper first overviews the framework features then shows them at work in supporting the realization of MrSPOCK. The paper shows how to solve an interesting multi-objective optimization problem requiring the satisfaction of various temporal and causal constraints. It then shows how the modeling capabilities of the APSI-TRF supports both the synthesis of an end-to-end approach to the problem, and open the possibility to flexible extensions of the application with added value in terms of modularity, extensibility and reusability. To this purpose, the main steps for the development of an extended application, which includes a model of the satellite power management subsystem are also described.

Introduction

Over the past three decades automation of complex procedures in space missions has always represented a challenge for AI planning and scheduling (P&S) techniques. Planning systems research has been deeply influenced by challenges offered by space applications. Innovations have concerned initial works on temporal planning (Vere 1983), real time control of the space shuttle (Ingrand, Georgeff, and Rao 1992), the broad concept of autonomy (Muscettola et al. 1998), the planning and execution loop, e.g., (Knight et al. 2001), the allocation of Earth Observations on a satellite (Bensana, Lemaitre, and Verfaillie 1999), negotiation tools for on-ground decision making of Mars missions (Ai-Chang et al. 2004), etc.

The European Space Agency (ESA) through the Advanced Planning and Scheduling Initiative (APSI) (Steel et al. 2009) is currently supporting cutting-edge research on AI planning and scheduling. The study focuses on the possible application of Artificial Intelligence techniques to enhance the ESA missions operations management. To this end APSI was explicitly requesting the synthesis of a software infrastructure to support different applications and the verification of such an infrastructure through the design and implementation of systems for three different planning scenarios, coming from current ESA missions (MARS EXPRESS, Integral, and XMM were selected as case studies).

As described in (Steel et al. 2009) the consortium in charge of the study comprehends VEGA Deutschland GmbH as industrial partner, ISTC-CNR (Rome, Italy), ONERA (Toulouse, France), and Politecnico di Milano (Milan, Italy) as research centers. The development of a general-purpose architecture was a fundamental requirement in the study and has been in charge of ISTC-CNR that developed the APSI-TRF (APSI Timeline-based Representation Framework). Then, the three research centers have been responsible for the development of the test cases as respectively described in this paper for MARS EXPRESS, in (Pralet and Verfaillie 2009) for Integral and in (Castellini and Lavagna 2009) for XMM.

The APSI framework follows the *timeline-based approach* initially proposed in (Muscettola et al. 1992; Muscettola 1994), since then used in a number of space related tools (e.g., (Jonsson et al. 2000; Chien et al. 2000)) and studied in several works (e.g., (Frank and Jónsson 2003)). In particular the APSI framework uses, as proposed in (Fratini, Pecora, and Cesta 2008), the generic term of "component" to identify a modeling primitive that refer to features endowed with a temporal behavior. Specific example of components in the framework are the multi-valued state variables, *a-la* (Muscettola et al. 1992) and the constraint-based representation of resources *a-la* (Cheng and Smith 1994).

This paper aims at giving a general view of the approach followed by ISTC-CNR[1] for solving the MARS EXPRESS problem using the APSI-TRF. It first presents the definition of the addressed problem, then describes the main capabilities of the framework, then presents MrSPOCK and its evaluation. Finally, it addresses an important issue for the whole approach, the support to extensibility that is underlying the APSI-TRF and that here supports seamless extension of MrSPOCK.

[1] http://pst.istc.cnr.it/

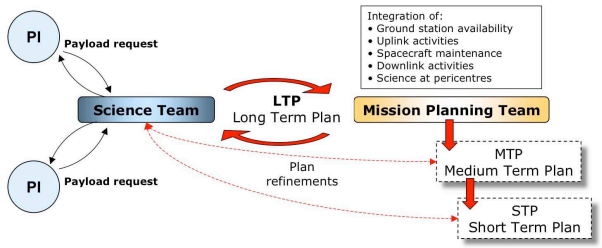

Figure 1: The problem scenario

The MEX-LTP Problem

The open problem we address is called the MARS EXPRESS Long Term Planning problem (MEX-LTP). The broad ESA request was for support in the collaborative problem solving process between the SCIENCE TEAM and the MISSION PLANNING TEAM of the space mission. These two groups of human planners (see Figure 1) iteratively refine a plan which eventually contains all activities for the mission. The process starts at the long term plan (LTP) level – three months of planning horizon – and is gradually refined to obtain fully instantiated activities at short term plan (STP) level – one week of planning horizon. This process continuously leads to weekly STPs, which are then further refined every two days to produce final executable plans. The lack of an accurate model of the spacecraft on the SCIENCE TEAM side is one of the main cause for performing many expensive iterations between the two groups. In addition, on the other side, the MISSION PLANNING TEAM has only partial information about the requested science operations for MARS EXPRESS, thus adding further sources of uncertainty to the decision process. The objective is to generate a *pre-optimized skeleton LTP* subject to subsequent cooperative SCIENCE TEAM/MISSION PLANNING TEAM refinement, which can guarantee both a reduction in the time spent in the iterative refinements and the minimization of a set of objective functions.

An LTP skeletal plan is composed by three types of activities corresponding to the three phases of each orbit around MARS: (1) time interval around the *pericentre* (the closest orbital point to the planet); (2) time interval around the *apocentre* (the farthest orbital point from the planet); (3) time interval *between* the pericentre and apocentre passages. During the pericentre period the spacecraft is preferably requested to point to the planet thus allowing observations of the planet surface with its payloads – this is generically referred to as *Science Operation*. Between pericentre and apocentre passages, the spacecraft can transmit data to Earth (*Communication*), thus pointing to Earth. This activity should occur within ground station availability windows. Finally, *Maintenance* operations should occur around the apocentre passages.

Given a problem instance, a *feasible* solution $S = \{op_1, op_2, \ldots, op_{no}\}$ is a set of *total ordered* operations op_i with three different operative modes: maintenance, science, and communication. The set of operations $op_i \in S$ hold the following set of constraints, which are only briefly described in this section. Apocentre slots for spacecraft maintenance windows must be allocated between o_{min} and o_{max} orbits apart (2 and 5 are respectively the usual values) and has a maximal duration (90 minutes is the usual value). Communication operations require the exclusive use of a single ground station and are source of several temporal constraints. It is required a non-preemptable four-hours uplink time each 24 hours (there is also the possibility to split a four-hour uplink window into two-hour uplink windows allocated each 12 hours). In general, since downlink and uplink operations can be always indifferently executed on the available ground stations, we require to have non-preemptable communication operations with duration δ (e.g., four hours), such that the maximal distance between two consecutive operations is T_{ud} (e.g., 24 hours). We consider this communication constraints as the only *soft-constraint* of the problem, so we can accept a small degree of violation for the constraint and we cast its satisfaction as a minimization problem (see below the definition of an optimal solution). Further temporal constraints on communication operations impose a minimum/maximal durations for the X-band transmitter in both the *on* and the *off* state.

An *optimal* solution S_{opt} is a feasible solution which *minimizes* the following objective function that has been defined according to the mission planners needs:

$$f(S) = \alpha f_{sc}(S) + \beta f_{dw}(S) + \gamma f_{up}(S) + \epsilon f_{ta}(S) \quad (1)$$

where α, β, γ and ϵ are non negative constant real values.

1. $f_{sc}(S) = 1 - NP(S)/NP_{max}$ where $NP(S)$ and NP_{max} are respectively the number of pericentre events associated to a science operation and the total number of pericentre events within the problem planning horizon $\mathcal{H}$. $f_{sc}(S)$ measures the fitness with respect to the science opportunity.

2. $f_{dw}(S) = 1 - DV(S)/DV_{max}$ where $DV(S)$ and DV_{max} are respectively the volume of data which can be down-linked by the set of communication operations included in the solution and the maximum volume of data which can be downlinked within the given planning horizon $\mathcal{H}$. $f_{dw}(S)$ measures the fitness with respect to the downlink opportunity.

3. $f_{up}(S) = LA_{ud}(S)/T_{ud}$ represents a measure of the uplink *smoothness*, that is a measure of the uniform distribution of the uplink operations over $\mathcal{H}$. Where $LA_{ud}(S)$ is the standard deviation of the set of distance values computed for each pair of subsequent pair of uplink operations.

4. $f_{ta}(S) = TA_{ud}(S)/T_{ud}$ represents a measure of the "uplink tardiness", i.e., the violation degree of the maximum constraint T_{ud}. Where TA_{ud} is the average *tardiness* value, that is the average violation value of the constraint T_{ud} imposed on each subsequent pair of communication operations with duration greater or equal to δ (e.g., four hours).

The objective function f reduces the original multi-objective optimization problem to a single-criterion optimization problem. This approach is quite common in the optimization literature and has pros and cons. As pros, we can directly use a single-criterion optimization algorithm for solving our problem (see the section on the solver). As cons, we have to consider that in many multi-objective problems, the semantics of the desired solutions is context dependent and can be dictated by individual preferences (in our case the preferences of the mission planners). Therefore the task of constructing the combined evaluation function in a way that preserves the semantics of the desired solutions may require some experience in solving the problem and the need to involve the mission planner in the solving loop (see the section on the user interaction services).

Developing Problem Solvers with the APSI Framework

As said before the basic step in APSI has been the definition of an open software architecture that acts as a software development environment for planning and scheduling applications. Such an architecture is called Timeline Representation Framework (or APSI-TRF) to underscore the basic representation choice it contains: a timeline-based approach to problem solving.

The APSI-TRF architecture has a layered design (see Figure 2). Each layer is responsible for dealing with a particular aspect of the problem and uses the services provided by the underlying layers to implement its functionalities. The constraint-based nature of the approach is visible in the way the different layers exchange information: constraints are posted on the underlying levels as a consequence of decisions taken on higher levels, and decisions are taken on higher levels by analyzing the domains of the variables in the underlying levels.

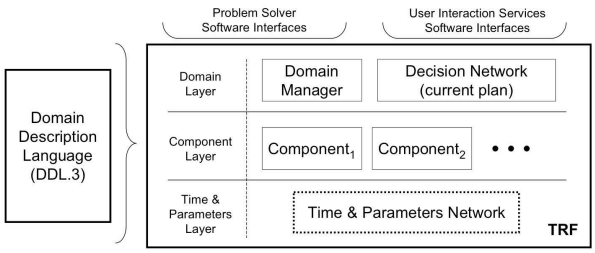

Figure 2: The layered implementation of the APSI-TRF

APSI-TRF's software layers are: a *Time/Parameters* layer, a *Component* layer and a *Domain* layer. A planning domain is modeled as a set of concurrent threads (the *timelines*) which are instantiation of components. A problem is solved by synthesizing a set of decisions to obtain a desired timeline behavior and, at the same time, by synchronizing the timeline threads to respect the component interactions.

Time and Parameters Layer. This is the lowest layer in the APSI-TRF 's architecture. Temporal and parameters' information is managed at this level. The interface provided by this level is simple and straightforward. Higher levels create temporal elements and parameters, impose constraints on them and query the database to access the information on events temporal positions and parameters values. The temporal information is managed in shape of *Temporal Constraint Networks* (TCNs) (Dechter, Meiri, and Pearl 1991). Parameters are managed through the external CSP solver CHOCO[2].

Component Layer. The component layer plays a key role for the extension of the APSI-TRF architecture. A *component* is a software module that encapsulates the logic for (1) computing a timeline resulting from decisions, temporally tagged functions of parameters; (2) evaluating the consistency of the computed timeline with respect to a set of constraint rules and (3) computing a set of temporal and/or parameter constraints and further decisions to solve (if possible) any threat to the consistency of the computed timeline.

The current version of the APSI-TRF provides two types of components: *state variables* and *reusable resources*. *State variables* (as introduced in (Muscettola et al. 1992)) have behaviors that are piecewise constant functions over a finite, discrete set of symbols which represent the *values* assumed by the state variable. Each behavior represents a different sequence of values over a finite interval of time. A *decision* $d = ([s, e], v)$ imposes a value v over the finite interval $[s, e]$. The consistency notion is stated as a set of sequence constraints, i.e., a set of rules that specify which transitions between allowed values are legal, represented as a *timed automaton*. In addition, temporally intersecting decisions must require the same values, otherwise the resulting timeline will be inconsistent. If two decisions that require, for instance, $P(x)$ and $P(y)$ overlap, the state variable component must be able to deduce $x = y$ to ensure the consistency. A *resource* (represented as in (Cheng and Smith 1994)) is any physical or virtual entity of limited availability, such that its profile (or behavior) represents its availability over time, a *decision* represents a quantitative use/production of the resource over a time interval. A *reusable resource* abstracts any real subsystem with a limited *capacity* c_{max}, a decision $d = ([s, e], q)$ uses a quantity q of resource during the limited interval $[s, e]$. For example, an electric generator has a maximal available power P_{max} (its capacity). A decision d uses power during $[s, e]$ and as soon as the activity d ends, the amount of resource q can be *reused* by other activities. A set of decisions are *feasible* when for each time t the aggregate demand $p(t)$ (or profile) is below or equal to the resource capacity c_{max}.

A component provides to the higher level basic timeline-management primitives, like timeline extraction and inconsistencies detection. As said before is a key to extension of the APSI-TRF architecture, because components make the architecture independent from the actual implementation of the functionalities they provide, encapsulating specific algorithms and hiding differences about behaviors, inconsistency

[2]http://choco.sourceforge.net/

detection and resolution behind a common interface. An example of how the modeling capabilities of the basic APSI-TRF can be extended by adding components is described in the last section of the paper (see the definition of a new component named *piecewise linear* resource).

Domain Layer. The Domain layer plays two important roles in APSI: (a) it allows for the definition of an application domain through the *Domain Manager* of Figure 2; (b) it defines and maintains the *Decision Network* the basic data structure that stores the solving decisions that modify the temporal behavior of the components to synthesize a solution to the open problem.

The Domain Manager glues together the concurrent threads represented by components of the underlying level. This module is responsible for providing domain theory management functions (e.g., sub-goaling and/or unification possibilities) and to maintain knowledge of needed synchronizations among components. At present, an extension of the DDL.3 language introduced in (Fratini, Pecora, and Cesta 2008) is used for specifying the domain theory. It is worth pointing that the proposed domain theory can represent a wide range of problems.

The Decision Network manages *relations* among solving decisions maintaining the current solution updated. The decision network provides a unified vision of the current solution, while the synchronizations that constitute the domain theory provide a unified means for expressing the constraints that decisions must satisfy. The Decision Network is the basic interface with respect to external solvers. The pursued scenario is the following: a solver implements search procedure and heuristics, while the APSI-TRF maintains the search space, additionally providing services for helping maintaining such a search space. It is also worth noting that the APSI-TRF provides the primitives to capture the specificity of a planning and scheduling application domain. To deploy a fully operational application it is necessary to complete the representation aspect by adding (a) a *solver engine* and (b) *user interaction services*. We will described this instantiation in MrSPOCK.

MrSPOCK: the APSI-TRF Model

We use the APSI-TRF features and represent the MEX-LTP domain with two different types of timelines: (1) *Controllable State Variables*, which define the search space of the problem, and whose timelines ultimately represent the solution to the problem; (2) *Uncontrollable State Variables*, representing values imposed over time which can be only observed. In particular, a single controllable state variable is used to model the spacecraft operative mode, which specifies the temporal occurrence of science and maintenance operations as well as the spacecraft's ability to communicate. The values that can be taken by this state variable, their durations (represented as a pair $[min, max]$) and the allowed transitions among them, are synthesized by the automaton in Figure 3, and represented in DDL.3 as showed in Figure 5(a).

In addition, we instantiate two uncontrollable state variables to represent contingent events such as orbit events and

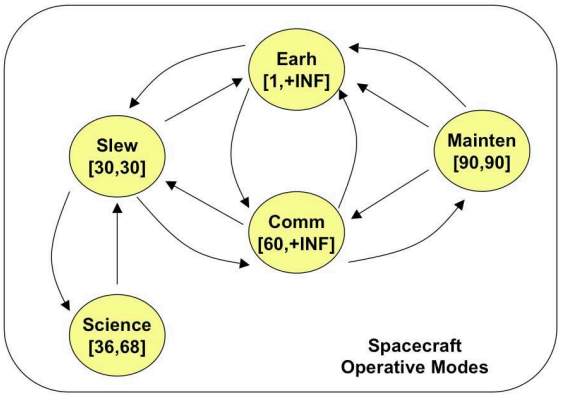

Figure 3: Legal value transitions on the state variable describing the operational mode of the spacecraft

communication opportunity windows. One state variable type component maintains the temporal occurrences of pericentres and apocentres ("PERI" and "APO" values on the timeline in Figure 4, top) of the spacecraft's orbit (they are fixed in time according to the information found in an orbit events file), while the other state variables maintains the visibility of three ground stations ("MAD","CEB" and "NNO" timelines in Figure 4, bottom). These state variables have as allowed values {*Available(?rate,?ul_dl,?station), Unavailable()*}, where the *?rate* parameter indicates the bitrate at which communication can occur, *?ul_dl* indicates whether the station is available for upload, download or both, and the *?station* parameter indicates which dish is available for transmission.

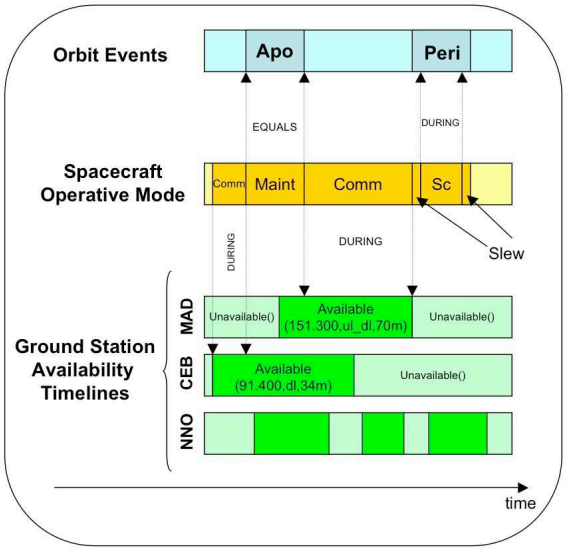

Figure 4: Timeline synchronizations.

Any valid plan needs synchronizations among the spacecraft Operative Mode timeline (Figure 4, middle) and the un-

```
COMP_TYPE StateVariable OPERATIVE (Earth(),
    Comm(RATE,UL_DL,STATION), Science(), Maintenance(), Slew()) {
 VALUE Earth() [1,+INF] MEETS
    {Slew(), Maintenance(), Comm(?rate,?availability,?station)}
 %Duration 60 minutes at least
 VALUE Comm(?rate,?availability,?station) [3600000,+INF] MEETS
    {Earth(),Slew(),Maintenance() }
 %Duration [36,68] minutes
 VALUE Science() [2160000,4080000] MEETS {Slew()}
 %Duration 90 minutes
 VALUE Maintenance() [5400000,5400000] MEETS {Earth()}
 %Duration 30 minutes
 VALUE Slew() [1800000,1800000] MEETS {Earth(),Comm(?x0,?t0,?a0)}}
```

(a) Operative Mode state variable

```
COMPONENT OPERATIVE_MODE:OPERATIVE {
  %A ground station must be visible
  VALUE Comm(?rate,?avail,?station) {DURING [0,+INF][0,+INF]
     DSS_STATIONS Available (?rate,?avail,?station), ?avail= ul }
  %Maintenance During Apocentres
  VALUE Maintenance() {EQUALS ORBIT_EVENTS Apocentre()}
  %Science During Pericentres
  VALUE Science() {DURING [0,+INF] [0,+INF]
       ORBIT_EVENTS Pericentre()}}
```

(b) Synchronization constraints for the Operative Mode state variable

Figure 5: MrSPOCK's DDL.3 specifications

controllable timelines (represented as dotted arrows in Figure 4 and in the domain description language DDL.3 in Figure 5(b)): science operations must occur during Pericentres (meaning that a *Science* value must start and end during a *Peri* value), maintenance operations must occur in the same time interval as Apocentres (meaning that a *Maint* value must start and end exactly when the *Apo* value starts and ends) and communications must occur during ground station visibility windows (meaning that a *Comm* value must start and end during an *Available* value). In addition to those synchronization constraints, the pointing mode timeline must respect the transition among values specified by the automaton and the minimal and maximal duration specified for each value (in the automaton as well). It is worth pointing that the synchronizations and the allowed transitions among the values taken by the state variables model the planning causal relationships among the states that the modeled sub-systems can take.

On top of this representation, MrSPOCK's solver (described in the next section) builds the spacecraft's operative mode timeline allocating science, maintenance and communication activities. A solution is obtained when a consistent timelines for the controllable component is defined and all the operational constraints represented by synchronizations are satisfied.

MrSPOCK Solver

Given the multi-objective nature of the proposed optimization problem we have considered the possibility to build an optimization procedure based on Genetic Algorithms (GA), a population-based optimization procedure inspired from the study of population genetics. A GA uses a *population* of possible solutions, which are subject to modifications aimed at the determination of the optimal solution. Every possible solution is encoded into a *chromosome* – a summarized representation of an individual or a solution – and positions in the chromosome are called genes. The value a gene takes is called an *allele* (or *allelic value*). Given an initial set of feasible solutions (the initial population), individuals are selected according to their fitness. The fitness of the N_p individuals is made explicit by means of a *fitness function* which is related to the *objective function* of the problem. After selection, individuals are randomly crossbred allowing the *recombination* of genetic material with probability p_c. The resulting individuals can then be *mutated* with a specific mutation probability p_m. The new population so obtained undergoes again a process of natural selection which favors the survival of the fittest individuals (the best solutions), and provides the basis for a new evolutionary cycle (this is iterated for N_g generations). The key idea of the integration of GA with the APSI-TRF representation relies on a simplified and *encoded* solution representation, the chromosome, which can be manipulated by a classical GA. The GA environment uses classical operators for Selection, Recombination and Mutation, then the chromosome are decoded by a *greedy algorithm* which works on the APSI-TRF representation. In this way the GA leads the way for the optimization but the greedy part of the solver still maintains responsibility to create a ground complete solution that satisfies the full set of problem constraints. Such an encoding/decoding phase is another interesting original contribution of MrSPOCK.

Given a solution S for a pre-planning problem instance, it is *encoded* into a chromosome ch by just reporting the position over time of the science and maintenance operations. In particular, a solution S is encoded by a vector of integer values *0-1* of size $|E|$, where $|E|$ is the size of the set of reference events E. The sequence of allelic values (0 or 1) respectively represents the position of science or maintenance operations according to the position of the corresponding reference events $e_i \in E$. In particular, for each event representing an apocentre (pericentre) the value 1 indicates the allocation of a maintenance (science) operation around the event, the value 0 indicates a free event.

A chromosome ch is decoded into a solution S by a constraint-based procedure that exploits the APSI-TRF features. A procedure *DecodeChromosome* scans the problem horizon from left to right and generates the total ordered sequence of operations (the solution) $S = \{op_1, op_2, \ldots, op_{no}\}$ that are translated into a temporal occurrence of proper values for the pointing state variable. It takes as input an instance of the problem, a chromosome

ch and a reference operation op_0. According to a reference events E – we include only apocentre and pericentre events – over the horizon $\mathcal{H} = [0, H]$ three different type of decisions de_i are considered:

1. Around apocentre events – in this case it possible to select between two type of operations: maintenance (`mn`) or communication `cm(j)` with a ground station j. Maintenance operations are decided according to the chromosome and they are considered as mandatory decisions. When no maintenance is decided a communication operation is selected if a ground station j is available on the basis of the so-called ground stations *de-overlapping* strategy sketched below;
2. Between an apocentre and a pericentre event – in this case only communication operations are possible, which can be joined to the last operation op_{k-1} inserted in the solution and are decided on the basis of the *de-overlapping* strategy;
3. Around pericentre events – in this case it possible to select between two type of operations: science (`sc`) or communication `cm(j)` with a ground station j. Science operations are decided according to chromosome, however this kind of decisions are not mandatory. In fact, a communication operation is selected when the decision of a science operation cannot generate a tardiness value between two consecutive communication operations with duration greater or equal to a given threshold.

The core of the procedure *DecodeChromosome* is a loop which iteratively select an operation op_k and post it into the current solution S, translating this decision into appropriate temporal occurrence of timeline values. The loop continues until there are no more operations possible.

Furthermore, a solution is completed by specifying the decisions concerning a *de-overlapping* strategy that assigns the communication operation with mode `cm(j)` and the selection of the related ground station j. Basically, the *de-overlapping* strategy fills the gaps between maintenance and science operations with communication operations. It takes into account both a set of temporal constraints and the objective function $f(S)$.

It is worth noting as our idea to use a chromosome with references to only two kind of operations – science and maintenance – is based on the observation that communication represent a kind of *default* operation for the satellite. In fact, when no maintenance or science operation is executed, communication is the only possible option. So, the chromosome implicitly influences when to perform communication operations.

User Interaction Services

The basic layout for MrSPOCK is centered on the concept of timelines. It describes both the uncontrollables (Ground Station Availability and Orbit Events) and the controllable (Spacecraft Operative Mode) components. The choice of centering the interaction on the concept of components which evolve over time allowed us taking advantage

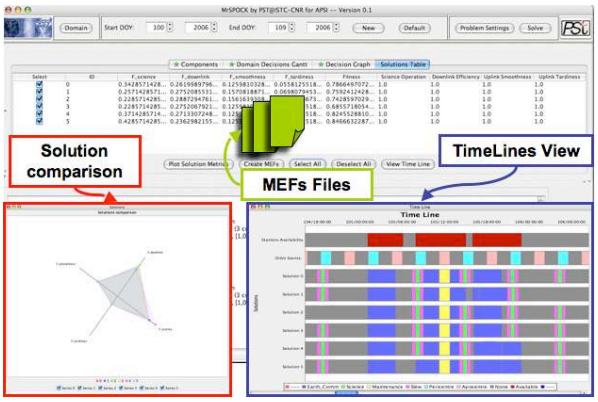

Figure 6: Examples of interaction for end users

of the users' ability on reasoning over timelines to be completed and refined. Showing timelines, even in a preliminary version of the interface, resulted very useful to set up a context for the users and to facilitate our dialog with them since the early stages of the project.

The interaction is based on few focal concepts to meet users' expectations on the open problem, in particular we focused on: (a) the need to explore alternative solutions, (b) the ability to control some parameters to favor an optimization criteria or another, (c) the easy visualization of the solution.

Figure 6 presents a sketch of aspects that directly cope with these requirements. The main outcome of the genetic algorithm run is gathered in a solution table that gives an immediate view of the fitness values specified according to the different metrics like Science and Downlink efficiency and Uplink Tardiness. We have given the user the possibility to act on the parameters that influence the different fitness and to inspect the effects of this manipulation on the single fitness component (same table). Additionally a graphical version of the optimization values offer an alternative and cumulative view (left bottom of the figure) that allows to easily see the comparisons of alternative solutions. The connection with the existing legacy of the mission planning at ESA has been preserved by providing the users with the possibility to generate the files containing all the activities for the spacecraft in the format required (MEFs file in figure) directly from the MrSPOCK environment.

Exploiting the central concept of the timeline shared between users and system developers, an additional graphic service has been built for the users which consists in the comparison of the pointing mode timelines corresponding to alternative solutions (see bottom right of the figure).

This additional graphical view guaranteed a twofold beneficial effect. On the system developer side we were able to quickly check the validity of our solving approach since the overall view highlights features of the different solutions and consequently the solving choices. On the users' side they were able to compare and reason on their choices using this environment as a means to perform "what -if" analysis.

Also in this case the choice of centering the interaction on the timeline comparison, appeared particularly successful. It is possible to speculate that in space domain the idea of taking decision over time is a quite "natural concept" which facilitate the choice of the main shared concept in term of what to show to the user at first glance.

A further aspect in the user interface is dedicated to the work done to show to the users the underlying domain model. This effort is motivated by the goal of showing the user aspects connected to the reusability of this technology within different contexts and space missions. Examples of this interaction are the high level textual form domain description, an inspection of the single state variables, a graphical view of the automaton regulating the internal state transitions of the pointing mode component. This is somehow both irrelevant for the core application and also very simple but, together with other representations not shown here for lack of space, have obtained the effect of making explicit the generality of the underlying APSI-TRF representation module.

Experimental analysis

This section offers a quantitative measure of the effectiveness of the current version of the optimization algorithm. The aim of this analysis is to give an evaluation targeted to the mission planner needs. In particular, it proposes two different analysis, first, a comparison with the performance obtained at operative level without the use of the tool; second, a test on the flexibility offered by MrSPOCK in generating different solutions.

Experimental Setup. The evaluation concerns the synthesis of pre-optimized skeleton plans within an horizon of four weeks. The problem instances were generated on the basis of the real data obtained from ESA on the ongoing interplanetary MARS EXPRESS mission. In particular, we have chosen the reference period, *[06-100, 06-128]*[3] and used the corresponding Master Event File (MEF) file run by the MISSION PLANNING TEAM at ESA-ESOC during the same period in order to compare our results. The MEF file can be seen as a set of *timelines* representing the main events connected to the mission. In particular, it represents the changing of visibility of the ground stations, the evolution of the operative mode of the spacecraft, and its communication activities.

The objective function's weights, the problem constraints' parameters and the genetic algorithm parameters are set to the following values. α, β, γ and ϵ to the value 1.0. The maintenance constraints $o_{min} = 3$ and $o_{max} = 5$ (i.e., the minimal and maximal allowed orbit distance between two maintenance operations), the parameters for the communication operations $T_{ud} = 24$ hours (distance between two uplink operations) and $\delta = 4$ hours (duration of non-preemptable uplink communication). Finally, we set genetic algorithm's parameters, each generation is composed by $N_p = 50$ individuals, the number of generation $N_g = 400$, the probability of recombination is set to $p_c = 0.8$, and finally the mutation probabilities is $p_m = 0.05$. Given the reference setting, we then compare the results obtained with the ones generated by imposing different values for the weights (α, β, γ, ϵ) of the objective function $f(S)$. MrSPOCK is coded in Java and, as such, can run on multiple HW/SW platforms. All the experiments run on a AMD Athlon 64 processor at 3.5 GHz, 2 GB Ram, under Windows XP.

Results. Table 1 compares the values of the objective function components ($f_{sc}(S)$, $f_{dw}(S)$, $f_{up}(S)$, and $f_{ta}(S)$)) obtained by our system for a subset of *non-dominated*[4] solutions with the corresponding component values calculated for the used MEF file. Among the best non-dominated solutions we report one solution for each component of the objective function (f_{sc}, f_{dw}, f_{up}, and f_{ta}) which gives the minimum of the corresponding value (*Best*). We also report one solution which minimizes the overall value $f(S)$ (first row in the table). Best objective values are reported in boldface characters and *Cpu* times are reported in seconds.

As expected (see Table 1), one solution which minimizes the value $f(S)$ does not contain the best values for all the single components of the objective function. Nevertheless, we observe an improvement over the MEF performance (last row in the tables) such that three solutions out of five dominate the MEF performance. That is, the values of the MrSPOCK objective function components are either as good as or better than the MEF ones in at least one component. It is worth noting that MrSPOCK's performance is based on a relaxed version of the MARS EXPRESS Long Term Planning problem, whereas the MEF performance are calculated on a final operative file. The generation of solutions which dominate the current MEF performance is therefore an encouraging result. Our next step will be the integration of the tool in the process of plan generation for MARS EXPRESS. Nevertheless, according to some preliminary comments from our end-users (the mission planners), the pre-optimized skeleton plans generated by MrSPOCK have a good chance to improve considerably the performance of the final operative plans for MARS EXPRESS.

Table 2 shows the performance of our system with respect to two different vectors of weights $(\alpha, \beta, \gamma, \epsilon)$ for the objective function, one targeted to maximize the number of pericentre opportunities for science operations (*Science*$(1.0, 0.5, 0.25, 0.25)$) and a second one targeted to maximize the total communication time *Comm*$(0.5, 1.0, 0.25, 0.25)$. As we can see in Table 2, the two vectors of weights generate two different set of solutions, which find different trade-offs among the values of the objective function's components. In particular, we obtain quite different values for the components f_{sc} and f_{dw}. Such

[3]Dates are in the format *YY-DDD*, *YY* represent the last two digits of the year and *DDD* is the day of the year.

[4]Given two solutions S_1 and S_2, S_1 *dominates* S_2, or the vector $(f_{sc}(S_1), f_{dw}(S_1), f_{up}(S_1), f_{ta}(S_1))$ dominates $(f_{sc}(S_2), f_{dw}(S_2), f_{up}(S_2), f_{ta}(S_2))$, when $f_i(S_1) \leq f_i(S_2)$ for each $i \in \{sc, dw, up, ta\}$. That is, when a solution S_1 is as good as S_2 in all the objective function's components, or better in at least one component.

Table 1: Performance in the period *[06-100, 06-128]*

Source	Best	f	f_{sc}	f_{dw}	f_{up}	f_{ta}	Cpu
MrS	f	**0.54**	0.12	0.26	0.12	0.04	210
	f_{sc}	0.54	**0.12**	0.26	0.12	0.04	
	f_{dw}	0.59	0.21	**0.22**	0.12	0.04	
	f_{up}	0.56	0.18	0.23	**0.11**	0.03	
	f_{ta}	0.56	0.17	0.24	0.11	**0.03**	
MEF	-	-	0.36	0.24	0.59	0.19	-

flexibility can be used for finding different pre-optimized skeleton plans, which can bias the generation of the final operative plan towards different objectives.

Table 2: Using different objective function's weights

Source	Best	f	f_{sc}	f_{dw}	f_{up}	f_{ta}	Cpu
Science	f	**0.27**	0.1	0.27	0.12	0.04	205
	f_{sc}	0.27	**0.1**	0.27	0.12	0.04	
	f_{dw}	0.30	0.13	**0.25**	0.12	0.03	
	f_{up}	0.29	0.12	0.26	**0.11**	0.03	
	f_{ta}	0.28	0.11	0.26	0.11	**0.03**	
Comm	f	**0.39**	0.42	0.14	0.12	0.04	215
	f_{sc}	0.40	**0.41**	0.15	0.12	0.04	
	f_{dw}	0.41	0.47	**0.13**	0.12	0.04	
	f_{up}	0.40	0.43	0.14	**0.12**	0.04	
	f_{ta}	0.40	0.43	0.14	0.12	**0.04**	
MEF	-	-	0.36	0.24	0.59	0.19	-

The issue of *flexibility* is a key in MrSPOCK, through the available user interaction services it is possible to use the solving algorithm and to generate different set of solutions for accommodating different and constrasting needs arising in the mission planning environment. Such flexibility exists at level of modeling as well as we will try to show in the next session.

MrSPOCK the ... *Next Generation*

It is worth saying that MrSPOCK is a tool in advanced testing in the MARS EXPRESS operational environment and shows an interactive help to the long term planning negotiation process. For sure we can claim that although the role of MrSPOCK in APSI has been the one of validating the flexibility of the software development environment called APSI-TRF the tool described and evaluated in the previous sections shorten the development cycle and offer support to the user that previously relied on a set of distributed tools and their own personal intervention (e.g., the phone calls between MISSION PLANNING TEAM and SCIENCE TEAM described in the problem section). Indeed our goal in the introduction of the APSI-TRF in the development cycle is slightly more ambitious and aims at creating a general software environment that can evolve with the missions and takes advantage of the modularity of the components to create problem models that can address different levels of detail. To show this general direction we end the paper presenting an example of MrSPOCK extension supported by a APSI-TRF extension through the definition of new components.

In fact, the modeling capabilities of the basic APSI-TRF can be extended by plugging in new components. Let us suppose we need to reason about something that cannot be described using basic components like state variables or reusable resources. Let us suppose we need a resource with a piecewise linear behavior instead of a piecewise constant one like reusable resources. In this case the software engineer can design and implement a new component, plug it in the APSI-TRF and use it by specifying the appropriate synchronizations in the DDL.3 among the new component's behavior and other component behaviors. In this way it is also easy to extend already existing models by adding new features, as in the example we show here, where we define an extended MEX-LTP problem for the MrSPOCK application (see the section on the problem definition). In particular, each operation requires power for its execution, energy is supplied by the solar arrays and accumulated in the on board battery. A mandatory constraint is imposed on the discharge level of the battery, which must be maintained under a given maximal value. Hence, the extended model considers the Power Management System of the satellite. In particular, for this extension we have plugged a new type of component to the APSI-TRF: the *Piecewise Linear* consumable resource.

The *Piecewice Linear* consumable resource is characterized by a minimal and maximal capacity c_{min} and c_{max}. The main difference with basic resources is the type of imposed decisions, which are pairs $d = ([s, e], r)$, where $[s, e]$ is a limited time interval and r is a modification *rate*. A set of decisions are feasible when the aggregate demand $p(t)$ holds the constraint $c_{min} \leq p(t) \leq c_{max}$. It is worth noting $p(t)$ is a piecewise linear function. $p(t)$ can be represented as a discrete-time function on the set $T = \{t_0, t_1, t_2, \ldots, t_{n_r}\}$, such that, t_k are the time instants where the aggregate modification rate $R(t)$ - the sum of the rates associated to the active decisions at the instant t - changes. Given the initial value $p(t_0)$, there are two options for the calculus of $p(t_k)$, $k = 1, \ldots, n_r$.

1. $p(t_k) = p(t_{k-1}) + R(t_{k-1})(t_k - t_{k-1})$
2. $p(t_k) = min\{c_{max}, p(t_{k-1}) + R(t_{k-1})(t_k - t_{k-1})\}$

A fuel tank within a complex system is an example related to the former option. Fuel is consumed and filled up at different constant rates from a set of connected subsystems. In order to avoid fuel shortage or fuel loss (a filling up over the tank's capacity c_{max}) It is mandatory to hold the constraints $c_{min} \leq p(t) \leq c_{max}$. About the latter option, there is no violation of the maximal constraints c_{max}, because the profile $p(t)$ is *chopped* by definition to the maximal value c_{max}. It is only possible to violate the minimal constraint c_{min}. An example of modeled subsystem is a satellite's battery charged by solar arrays. The battery is discharged by the supplied subsystems and charged by the solar arrays at different constant rates[5] (the resulting battery power flow). It is

[5]The rates are not constant at all in reality. For simplification they are represented to be constant.

mandatory to maintain a minimum level of charge in the battery (e.g., 90 %). However, when the battery is full charged, the loss of solar power is not represented as a constraint violation, because solar energy is free and always available. On the contrary, in the tank case, a violation of the constraint c_{max} might represent a costly fuel loss.

The spacecraft's Power Management System can be integrated into the overall infrastructure as three additional components and a new set of DDL.3 forms, which represent the additional needed constraints. The new components model three basic aspects of the Power Management System: the source of the power, that is the *solar flux*; a representation of the power requirements (uses); a model of the *battery*, which is charged when the overall power requirement is less than the power produced (when the so-called *battery power flow* is positive) and discharged in the opposite situation. Under our model all single power requirements/productions are constant over a limited period, hence the aggregate power demand is a piecewice constant function. In the following a more detailed description of the added APSI-TRF components and constraints.

Solar Flux – it is an *uncontrollable* component which models the maximal input power to the satellite system, in particular to the so-called Array Power Regulator (APR). Within our model the the solar flux is represented as a sequence of decisions $d_f(i) = ([s_i, e_i], f_i)$, where f_i represent the average solar flux over the sampling interval $[s_i, e_i]$.

Battery – it is a *piecewise linear resource* component (using the chopping option), with minimum charge level constraint c_{min}[6] and two type of decisions. A power *production* $d_p = ([s, e], p)$, representing the average power supplied to the Battery Charge Regulator over the interval $[s, e]$. A power *requirement* (consumption) $d_c = ([s, e], p)$, representing the average power required to the Battery Discharge Regulator over the interval $[s, e]$.

Power – this *reusable* resource represents an additional constraint imposed on the set of power requests $d_c = ([s, e], p)$, such that limits the aggregate power demand to the maximal value $P_{max} = (1 + ovd)P_{SA}$, where P_{SA} is the average power produced by the solar arrays over the reference planning horizon and ovd is an overloading coefficient. The idea behind this choice is that on the long-term we cannot require an average power greater than P_{SA}, however, it is possible to require an higher power for a limited interval of time.

The model of the Power Management System is completed by the description of the constraints imposed among the set of components.

– Any valid plan needs a temporal synchronization among the *Pointing* timeline and the *Battery* timeline, such that at each pointing status corresponds a power production $d_p = ([s, e], \alpha p)$. Where p is the average power produced over the interval $[s, e]$ and α is a constant coefficient depending on the pointing status (`slew`, 0.5; `nadir`, 0.75; `along track/across track`, 0.75; `inertial`, 1.0; `eclipse`, 0).

– There is a further synchronization between the Earth Pointing value of the Pointing timeline and power production on the Power timeline, which takes into account the power needed by the Telecommunication subsystem.

– A constant power requirement on the Power timeline generated on MTP basis, which represent an estimation of the power required by the Platform subsystems, that is, Thermal Control, Attitude Control, Power Control, Command and Data Systems.

Hence, in our model the battery is charged at different rates according to the solar flux and the pointing status, and discharged by the activities of the Telecommunication and the other Platform subsystems. Our extended model considers two additional *virtual* power requirements, the so-called *Operative Margin* and the Payload Power Budgets. They are constant power requirements, such that the former is generated on MTP basis and the latter on orbit basis. Operative Margins are set on the basis of the Mission Planning System Configuration List and represent a further safety margin on the availability of power for the satellite. Finally, the assignments of the Payload Power Budgets can be estimated as residual values (as the difference between the average power generated by the solar arrays and an estimation of the power required by all the other satellite subsystems). In the current version of the system Power Budgets values are set for checking the DOD value of the battery. However, it is also possible to consider the Payload Power Budgets (generated on orbit basis) as additional decision variables of the MEX-LTP problem.

[6]The charge level c is a percentage value of the full battery capacity (the so called energy nameplate), so the Depth of Discharge parameter (DOD) can be calculated as $DOD = 100 - c$

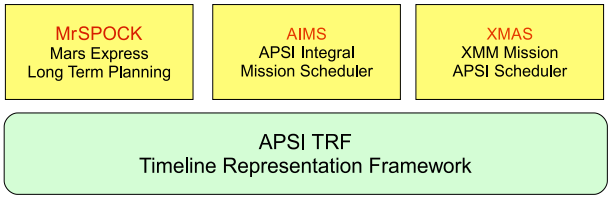

Figure 7: Different tools developed with the APSI-TRF

Conclusions

The main outcomes of APSI are sketched in Figure 7. The figure shows how the APSI-TRF has been the layer that allowed development of three different planning applications in real ESA missions: MARS EXPRESS (this paper), Integral (Pralet and Verfaillie 2009), and XMM (Castellini and Lavagna 2009).

This paper has described our approach to the development of MrSPOCK strongly based on the features of the APSI-TRF. We have described MrSPOCK in its different parts

but also described the application support that is underlying the APSI-TRF. Additionally we have described steps in extending both APSI-TRF and MrSPOCK to cope with the spacecraft power management subsystem and shown how this can be done taking advantage of the modularity offered by the timeline-based concept of component. It is worth underscoring how in developing MrSPOCK we have followed the general architectural approach of previous experiences, e.g., MEXAR2, (Cesta et al. 2007), this time using the support of an software platform. It is also worth saying that the APSI-TRF is not a domain independent tool like RAX-PS (Jonsson et al. 2000) or ASPEN (Chien et al. 2000) but rather a software development environment which guides application development through the domain ontology based on timelines and constraints. In some respect this approach shares some features with the SPIKE environment (Zimmerman Foor and Asson 2002) that directly allows for the incremental modification of the underlying software environment. Indeed our aim is at creating libraries of components and a robust general methodology for create new applications. We have done an important step in this direction within the APSI project.

Acknowledgments. The authors thank Alessandro Donati and Nicola Policella from ESA-ESOC for their support in the APSI study and all the members of the APSI team for fruitful exchanges. A special thank to Robin Steel and to the Mars-Express team from ESOC who supported the authors in understanding and modeling the MARS EXPRESS long-term planning problem.

References

Ai-Chang, M.; Bresina, J.; Charest, L.; Chase, A.; Hsu, J. C.; Jonsson, A.; Kanefsky, B.; Morris, P.; Rajan, K.; Yglesias, J.; Chafin, B. G.; Dias, W. C.; and Maldague, P. F. 2004. MAPGEN: Mixed-Initiative Planning and Scheduling for the Mars Exploration Rover Mission. *IEEE Intelligent Systems* 19(1):8–12.

Bensana, E.; Lemaitre, M.; and Verfaillie, G. 1999. Benchmark problems: Earth observation satellite management. *Constraints* 4(3):293–299.

Castellini, F., and Lavagna, M. 2009. Advanced Planning and Scheduling Initiative XMAS Tool: AI for Automatic Scheduling of XMM-Newton Long Term Plans. The 6th International Workshop on Planning and Scheduling for Space, IWPSS-09, July 19th-July 21st 2009, Pasadena, CA.

Cesta, A.; Cortellessa, G.; Denis, M.; Donati, A.; Fratini, S.; Oddi, A.; Policella, N.; Rabenau, E.; and Schulster, J. 2007. MEXAR2: AI Solves Mission Planner Problems. *IEEE Intelligent Systems* 22(4):12–19.

Cheng, C.-C., and Smith, S. F. 1994. Generating feasible schedules under complex metric constraints. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, volume 2, 1086–1091. Seattle, Washington, USA: AAAI Press/MIT Press.

Chien, S.; Rabideau, G.; Knight, R.; Sherwood, R.; Engelhardt, B.; Mutz, D.; Estlin, T.; Smith, B.; Fisher, F.; Barrett, T.; Stebbins, G.; and Tran, D. 2000. ASPEN - Automated Planning and Scheduling for Space Mission Operations. In *Proceedings of SpaceOps 2000*.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal constraint networks. *Artificial Intelligence* 49(1-3):61–95.

Frank, J., and Jónsson, A. 2003. Constraint-based attribute and interval planning. *Constraints* 8(4):339–364.

Fratini, S.; Pecora, F.; and Cesta, A. 2008. Unifying Planning and Scheduling as Timelines in a Component-Based Perspective. *Archives of Control Sciences* 18(2):231–271.

Ingrand, F. F.; Georgeff, M. P.; and Rao, A. S. 1992. An architecture for real-time reasoning and system control. *IEEE Expert* 7(6):33–44.

Jonsson, A.; Morris, P.; Muscettola, N.; Rajan, K.; and Smith, B. 2000. Planning in Interplanetary Space: Theory and Practice. In *Proceedings of the Fifth Int. Conf. on Artificial Intelligence Planning and Scheduling (AIPS-00)*.

Knight, R.; Rabideau, G.; Chien, S.; Engelhardt, B.; and Sherwood, R. 2001. Casper: Space Exploration through Continuous Planning. *IEEE Intelligent Systems* 16(5):70–75.

Muscettola, N.; Smith, S.; Cesta, A.; and D'Aloisi, D. 1992. Coordinating Space Telescope Operations in an Integrated Planning and Scheduling Architecture. *IEEE Control Systems* 12(1):28–37.

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

Muscettola, N. 1994. HSTS: Integrating Planning and Scheduling. In Zweben, M. and Fox, M.S., ed., *Intelligent Scheduling*. Morgan Kauffmann.

Pralet, C., and Verfaillie, G. 2009. AIMS: A Tool for Long-term Planning of the ESA INTEGRAL Mission. The 6th International Workshop on Planning and Scheduling for Space, IWPSS-09, July 19th-July 21st 2009, Pasadena, CA.

Steel, R.; Niezette, M.; Cesta, A.; Fratini, S.; Oddi, A.; Cortellessa, G.; Rasconi, R.; Verfaille, G.; Pralet, C.; Lavagna, M.; Brambilla, A.; Castellini, F.; Donati, A.; and Policella, N. 2009. Advanced Planning and Scheduling Initiative: MrSPOCK AIMS for XMAS in the Space Domain. The 6th International Workshop on Planning and Scheduling for Space, IWPSS-09, July 19th-July 21st 2009, Pasadena, CA.

Vere, S. A. 1983. Planning in Time: Windows and Durations for Activities and Goals. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 5(3):246–276.

Zimmerman Foor, L., and Asson, D. J. 2002. Spike: A Dynamic Interactive Component in a Human-Computer Long-range Planning System. In *Proceedings of the 3rd International NASA Workshop on Planning and Scheduling for Space*.