Challenges in Explainable Planning for Space Operations

Simone Fratini and **Nicola Policella**
Solenix Deutschland GmbH
Darmstadt, Germany
name.surname@solenix.de

Abstract

This paper discuss the role of explanation and the solutions adopted in the development of planning and scheduling tools for the support of space mission operations at the European Space Agency (ESA). A key point to strengthen the effectiveness and success of fielding planning and scheduling applications in space is the capability of providing an explanation of the solutions as well as of the solving process to the end users. The approach has been consolidated over the last decade while developing several tools and functionality to support explanation, but the challenge is still open and far away for being properly resolved.

Even though the approach has been utilized only in the space operations domain, we think that the concepts and problems highlighted are general enough to be applied to other domains and/or planning algorithms.

Introduction

The success of ”deep learning” methods in the 2010s has raised the attention on explaining the decisions taken by an AI based system. In fact these methods are naturally opaque, that is, their behavior cannot in general easily be understood by humans. In order to foster research in the area of Explainable AI (XAI), agencies like DARPA have recently launched ad-hoc programs (DARPA 2016).

AI Planning seems instead to be in a better position (with respect to Machine Learning for instance) to provide a correct and complete explanation of its behavior and to result more transparent to a not-expert end-user (Fox *et al.* 2017).

Even though the nature of AI planning is more adequate to achieve transparent systems, explaining the behavior and the solutions of planning algorithms is still far to be trivial. In this paper we discuss the general approaches we use to facilitate the explanation of the automated solving process. We also make use of two recent applications where explanation is playing a crucial role.

Motivation

A key point to strengthen the effectiveness and success of fielding planning and scheduling applications in space is the capability of providing an exact explanation of the solutions as well as of the solving process to the end users.

This is very important for at least three reasons. First of all the planning tool is usually introduced in an already running, operational environment in order to optimize and/or automate some steps of the process. In this scenario, the users already have been generating plans for some time, and when the system provides solutions that they do not ”recognize” it becomes of primary importance to be able to explain how they have been generated and justify differences they might find in the solutions generated by the ”new tool”.

Second, the planning tool will rarely work as ”black-box” or pure plan generator. Usually a ”man-in-the loop” approach is the preferred, and the planning technologies shall provide assistance to the human operators in generating or updating plans. In this interactive scenarios, it is necessary to provide explanations and to justify what is being done to put the users in the position to properly interact with the system.

Finally, since in operation usually the planning activity is shared among different groups, each one responsible for a sub-set of the whole plan, a proper explanation is also needed to have effective iterations between the different teams involved.

The remainder of the section provides a few examples of the different types of explanation we encountered in our experience in developing Planning & Scheduling solutions to support space operations.

Mexar2 experience

Our first project done in ESA-ESOC to introduce automated planning was the MEXAR2 system which is responsible for the generation of the activity plan used to download the satellite data from the on board memory to ground (Cesta *et al.* 2007).

The tool enables MARS-EXPRESS mission planners to rapidly produce a data dumping plan as well as to explore alternative solutions, providing the means to choose the more robust plan for execution. Additionally, the user is currently able to analyze the dumping problem over multiple days and identify payload overcommitments that cause resource bottlenecks and high risk of data losses. As a consequence MEXAR2 has allowed a significant increase of data return over the whole mission duration. These features have effectively made the system a fundamental work companion for the mission planners.

From this first experience was already clear the crucial role of being able to explain automatically the steps taken by the planning algorithm. For instance, at first was not clear why the solver was creating plans with many small activities. In fact such a fragmentation was the resulting of the optimization process which was trying to use at best all the available resources. Optimization was necessary to take into account different priorities for different packet stores and to provide robust solutions[1].

A fundamental aspect for users acceptance of the tool was to keep them into the loop. Mission planners in charge of spacecraft operations must be capable of understanding in detail any step of the solution and maintaining authority on any decision. This important requirement encouraged us to develop an interaction tool to foster the collaboration between the mission planners and the automated algorithms. In parallel we also modified the original algorithm in a way to make possible to the user to better control through specific tuning parameters the solving process.

Whereas at the time the approach followed was very specific to tool and the users, several lessons learned were collected and then drove the design of the generic planning and scheduling platform. This also applied to the explanation features.

Alphasat Payload Planner

Since 2013 the TECO planning system (just TECO below) is managing in an completely autonomous way the different payload experiments for the Alphasat satellite (Policella *et al.* 2013). The satellite carries four technology demonstration payloads. While each payload has a dedicated operations center responsible for defining and requesting the different experiments, the spacecraft resources (e.g., power, thermal constraints), downlink data, and telemetry budget are shared among the four payloads. This required a coordination and selection of the different experiments.

By exploiting advanced planning and scheduling technologies, the TECO planning system has been designed to have a high degree of autonomy where the final activity plans are generated automatically (no user intervention). Moreover the generation of the plan is done as an iterative process between the different payload centers on one side and the TECO planner on the other side. The goal of this iterative approach is to maximize the scientific return of the technology demonstration by helping to highlight the possible conflicts among the different experiment requests.

A relevant aspect of the output generation process is the need of providing sufficient explanation about the solving process. This is crucial in our case as the system has been designed to be completely autonomous. Our objective is to provide the system users with the necessary information to understand the decisions taken during the solving process and the final activity plan. A proper explanation is also needed to have effective iterations between the different entities involved. Considering that the time available to provide a new set of task requests is limited, it becomes fundamental to provide the right explanation on, for instance, why a task was not allocated.

Cluster-II Pass Planner

ESAs four Cluster-II satellites conduct three dimensional insitu measurements of the Earths magnetosphere. The mission is operated from the European Space Operations Centre in Darmstadt, where the FCT (Flight Control Team) prepares the routine operations for the spacecraft. A major aspect of the mission operation is the planning of ground station passes. Aside from the input of the science operation planning, the ground station plan also depends on the input of the Flight Dynamics team and on ground station availabilities. Ground station passes enable the FCT to download the scientific data from the spacecraft and check the overall health of the satellites. Given the large number of constraints involved in pass scheduling and the complexity of the problem, a tool based on AI technologies, TIAGO, Tool for Intelligent Allocation of Ground Operations, has been developed to ease such a process (Fratini *et al.* 2017).

A key operational aspect in this scenario is the capability of providing explanation on the choice of the ground station, as well as pass duration and temporal allocation. Users inspect visually the solutions and want to understand why a ground station has been chosen, for instance in place of a different one that might looks better or more suitable. Or why dumps are delayed (according to user's expectation), causing a dangerous filling of the on board memory, or, on the contrary, anticipated, resulting in an expensive over use of the ground stations that looks not justified by the filling level of the on board memory.

Explanation Approach

In pursuing a ”process driven” approach to support operations at ESA[2] we aim at general purpose tools for facilitating the design and synthesis of new products. The general idea is the one of improving the ”process” of tool development, taking advantage of the state of the art planning and scheduling technology.

In this view, the APSI Framework (APSI 2017) is a Java architecture for rapid prototyping of planning and scheduling applications currently in use at ESA for deploying new tool to support mission operations. The platform is designed for constraint-based temporal planning and scheduling. Constraint-based temporal planning, often referred to as “timeline-based planning”, is an approach to temporal planning which has been applied to the solution of several space planning problems – e.g., (Muscettola 1994; Jonsson *et al.* 2000; Smith *et al.* 2000; Frank and Jonsson 2003; Chien *et al.* 2010; Cesta *et al.* 2011). This approach pursues the general idea that planning and scheduling for controlling complex physical systems consist in the synthesis of a set of desired temporal behaviors, named *timelines*, for system

[1]In particular, the idea for robustness in this scenario is to maintain a non-critical level of memory bank usage so as to allow run time variations to be absorbed with a low risk of overwriting.

[2]As an attempt of innovation with respect to the traditional ”product driven” approach aimed at deploying solutions for specific mission problems.

features that vary over time. In this approach, problem solving consists of controlling components by means of external inputs in order to achieve a desired behavior. Hence different types of problems (e.g., planning, scheduling, execution or more specific tasks) can be modeled by identifying a set of inputs and relations among them that, together with the model of the components and a given initial set of possible temporal evolutions, will lead to a set of final behaviors which satisfy the requested properties; for instance, feasible sequences of states or feasible resource consumption[3].

Explanation in APSI

Being APSI the core of planning and scheduling applications we deploy for ESA, the explanation approach for TECO and TIAGO (presented later in this Section)is strongly grounded on modeling primitives and services provided by the framework.

APSI provides two general classes of modeling primitives: state variables and resources. Planning statements are specified as temporal assertions on timelines subject to temporal and transition constraints. Cause-effect relationships among planning statements and/or resource allocations are specified by means of temporal and logical synchronizations. Applications built using the framework can exploit the domain independent framework explanation primitives (for the solving process and for the solution) to build up a user oriented, application specific explanation significant for the final user.

A relevant aspect to support the generation of explanation is the fact that the APSI solving approach is a so-called flaw-based approach. This consists in an iterative, constructive paradigm which generates the final solution step-by-step driven by the identification and resolution (using the "right" solver) of flaws in the current plan. The causal relations between flaws and planning decisions are here exploited to drive the explanation process. In fact the flaws in a plan can be seen as the difference between this plan and an actual solution of the problem.

In general in APSI four elements are playing a role in generating the final explanation to the users:

1. The tree of flaws and decisions that are behind the generation of a solution. Here we are talking about trees and not simple chains as also failures in the solving process should be considered to provide a complete explanation. It is also worth remarking as the decisions can be resulting from the application of one of the solvers as well as from the propagation of the planning model.
2. The explanation of each decisions provided by the specific solver. In fact, all solving decisions are labeled with information about the solver who has taken the decision and the motivation of the decision.
3. A communication protocol. This is used to exchange the relevant information with the users of the system. This protocol has been introduced to address differences in backgrounds among the different partners. In fact, while the users are experts in their specific domain, they are not required to be experts in AI planning and scheduling.
4. An Explanation Generator module. The goal of this module is to interpret the information provided together with the solving decisions and of generating information for the system users by applying the given protocol.

[3]A detailed description of the planning approach, state of the art and basic concepts is out of the scope of this paper. More information can be found, for example, in (Muscettola 1994; Frank and Jonsson 2003; Fratini and Cesta 2012).

It is worth remarking as the first two points are generic aspects already part of APSI. On the contrary the last two are specific of the final application and can have a different implementation from case to case.

In the remainder of the section we describe the generic explanation elements in the framework. In particular the different explanation generated by the model propagation functions and the solving process.

State Variables. State variables represent components that can take sequences of symbolic states subject to various (possibly temporal) transition constraints. This primitive allows the definition of *timed automata*.

The values taken by a timeline instantiated for a state variable component in the APSI framework (and temporal relations among them) can support explanation considering the timeline as a valid sequence of symbols accepted by the timed automata. A value for instance can be justified with the need of instantiating valid transitions, or with the need of enforcing temporal constraints on the durations of the states. If the timeline is not complete (it contains unspecified intervals of time or *gaps*), temporal constraints on unspecified intervals can be justified by the need of valid sequences of states to fill the gaps. If the timeline does not represent a valid sequence for the automata, information can be provided on violations: invalid transitions between specified values, invalid durations of the values or of the unspecified intervals. This low level explanation is used by solvers to build higher level explanations.

Resources. Resources are used to model any physical or virtual entity of limited availability, such that its timeline (or profile) represents its availability over time whereas a decision on the resource models a quantitative use/production/consumption of the resource over a time interval or in a time instant.

The timeline of a resource supports the explanation process by providing information about intervals of time that violate the resource capacity constraint. Given that, the explanation process can justify temporal relations on resource allocations, like deadline or release times, precedence or duration constraints generated by the solving process to remove peaks of consumption.

Synchronizations. The physical and technical constraints that influence the interaction between subsystems (modeled either as state variables or resources) are represented by means of temporal and logical synchronizations among the values taken by the automata and/or resource allocations on the timelines. Languages for timeline-based planning have constructs, called *synchronization* in APSI, to represent the interaction among the different timelines that model the domain. Conceptually these constructs define valid schema of values allowed on timelines and link the values of the timelines with resource allocations. Despite the syntactic differences, they allow the definition of Allen's relations (Allen 1983) like quantitative temporal relations among time points and time intervals as well as constraints on the parameters of the related values.

The synchronization construct on which the timeline based planning is grounded provides a significant support to the explanation process. The synchronization, in fact, gives teleological explanation for the values in the timeline. A value (or a resource allocation) can be traced back, trough a chain of synchronizations, to the goals that justify its allocation in the plan. These goals can originate in the planning problem or can be generated during the planning process to complete a timeline or to produce a resource. On the other way, following the synchronizations applied to justify a value or a resource allocation, it is possible to build a list of supports for that statement. In other words, the synchronization gives explanation about 'why' a given value is there, and to 'what' this value is a support for.

Problem Solving. An application in the APSI framework, being a generic planner and/or scheduler or a domain specific deployed application is designed as a collection of *solvers*. Based on the constraint-based paradigm, the search space of an APSI solver is made of planning and scheduling statements on timelines and temporal and data relations among them. An APSI solver provides explanation based on the actions taken to update the plan/schedule, grounded on the information collected on state variable and resource timelines.

Available solvers and applications include state-of-the-art binary and multi-capacity schedulers as well as integrated planners and schedulers (Fratini *et al.* 2015). Solvers are chosen and activated on a flaw detection base. When a flaw is detected on a timeline the planner activates the corresponding solver to fix the problem. A flaw is any type of violation in a plan. It can be a logical flaw, when unsupported actions are added to the plan, or a resource violation flaw, when a resource is over or under used, or any other impairment of temporal allocation on the values over a timeline. The flaw-based solving approach provides a natural way for building explanation by connecting issues detected in the partial plan with the explanation provided by the solver chosen for fixing it. The explanation of the plan process is then given as a sequence of steps taken to update the plan, with each step motivated by the issue detected and explained with the actions taken to fix it. Some of the solvers currently available in APSI are:

Domain Theory Unfolder The DTU detects unsupported statements in the plan and justify them by applying synchronizations. The DTU can apply the synchronization by expansion (adding new statements) or by unification (constraining existing statements). The DTU explains the process by listing the effects and the supports of a statement, i.e. expansion and unification applied to justify it.

Partial Order Scheduler The POS is supporting the scheduling process resulting from planning to guarantee temporal flexibility.[4] The explanation in this case refers to the consumer-producer relations created by the solver. Moreover in case of no solution the solver provides as explanation a set of activities which are associated to the unsolvable conflict.

Resource Activity Generator It makes sure that linear resources can be adequately managed by avoiding any overconsumption. The generator add production or consumption statements to the plan, grounding the choice on the domain theory (which states of a state variable can produce/consume resource, how much can be produced/consumed and what kind of opportunity are needed for the production/consumption). The explanation includes the resource violation that fostered the process and which production/consumption status has been chosen on the basis of the temporal position and entity of the violation and available opportunities.

Resource Profile Bounder It uses a MaxFlow algorithm to bounds position and duration of activities to assure that all the resource constraints and/or requirements are consistent. The explanation provided by a profile bounder links duration and position constraints generated with the resource violation that fired the solving process, for instance stating that a consumption activity has been shortened to avoid a resource overconsumpion in a specific intareval of a given resource timeline.

Timeline Completer It fills gaps in state variable timelines. The explanation provides information on which sequences of states have been generated considering the transition and temporal constraints induced by the automata.

As pointed out, the solving process is driven by the different flaws identified and solved - therefore by considering the flaws collected it is possible to have a first analysis of the planner behavior (and explain it).

However this is not sufficient. As shown in the TECO example below, one important factor to consider is also the order in which the different flaws are collected and moreover when they first appear – were they part of the problem or introduced during the solving process (by one of the solvers)?

Explanation in TECO

In the TECO operations, a proper explanation is crucial to have effective iterations between the different entities involved – the goal is make the users to better understand why a task has not been allocated and how the requests can be updated.

There might be different explanations (and combinations of these) for which a task is not allocated: wrong input files, ill-defined task requests, tasks conflicting with spacecraft status, tasks conflicting with other task requests, tasks not allocated because linked to conflicting tasks, etc. Once the

[4]The POS (Policella *et al.* 2007) is a set of activities partially ordered such that any possible complete order that is consistent with the initial partial order is a resource and time feasible schedule.

explanation is identified by the system, before distributing it, it is necessary to put it in a form that can be understood by the receivers.

A first aspect worthy of attention is that with this approach the explanation provided to the user is not generated directly by one of the different solvers.[5] This is required as 1) the single solvers has a limited view of the general solving process and 2) it is necessary to decouple the solvers from the final explanation generation process (and the associated protocol).

Figure 1 shows two simple examples where a task cannot be allocated. In both cases the explanation cannot be provided directly by a single solver. In both examples both the payload (TDP) and the ground activity timeline are represented.

Figure 1(a) contains two tasks: t_a which is associated to the payload TDP_a and triggers the mode of the payload from the mode X to W and back to X (see payload timeline). t_a requires a fixed start time (and duration). Task request t_b is instead flexible in time (in the figure this is represented by the double arrow on the ground timeline). The task is associated to the TDP_b payload and changes the mode of the payload from F to A. Also one of the domain constraints requires that mode W of payload TDP_a shall be synchronized with the mode A of TDP_b. Finally, the ground is not always available to execute the tasks (see not "not available" activity in the ground timeline).

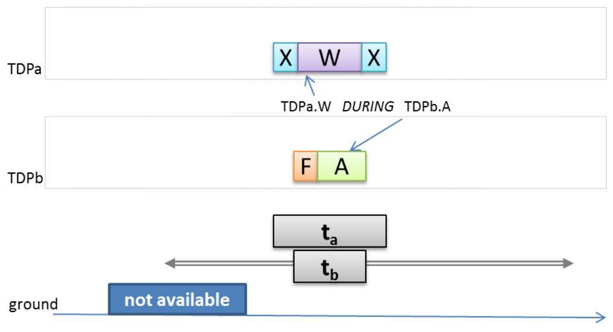
The solution of this problem contains only t_a. In fact to satisfy the domain constraint the planner adds the solving decision $et(t_b) \leq st(t_a)$ and pass it to the scheduler. The scheduler will then fail as t_b cannot be scheduled before the start of t_a due to the not availability of the ground. In generating the explanation for the missing allocation of t_a it will then be necessary to consider not only 1) the conflict identified by the scheduler but also 2) the decision taken by the planner, and 3) the domain theory (which contains the constraint $TDP_a.W$ during $TDP_b.A$).

In the second example, Fig. 1(b), only one task request, t_a, is present. Additionally it is required to switch off all the payloads (during the time interval represented on the ground timeline). In this case, while the task could be allocated in a time window, the planner will find a conflict between the final status of the task (X) and the value requested by the platform activity "TDPs OFF." In this case the explanation process will need to consider both 1) the planner conflict and 2) the domain theory.

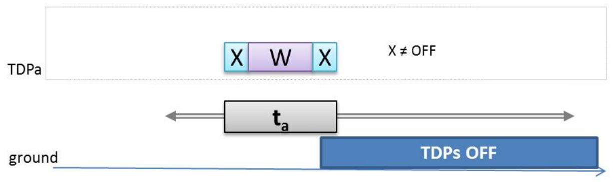
The objective of the previous examples is to show that for providing the correct explanation to the users it is necessary to consider all the decisions coming from the different solvers. Even if they represent very basic cases, the examples are also useful to give an idea of the effort needed to retrieve proper information especially when considering more complex and real cases, with more activities, resources, constraints, etc.

The approach implemented to generate explanations consists in "tracing back" all the different solving decisions that are associated to a task request. This is done by exploiting the different annotations added during the solving process. Once all this information is collected, the explanation generator module identifies the specific case based both on the solvers which generate the annotations (for instance the scheduler) and on the content of the annotations (e.g., conflict with spacecraft activity).

[5]In TECO a planner and a scheduler are actually used(Policella *et al.* 2013).



(a) Planning decision and scheduling conflict



(b) Planning conflict

Figure 1: Example of explanations

Explanation in TIAGO

A key operational aspect in Cluster-II pass scheduling problem is the possibility for providing explanation on the choice of the passes, duration and temporal allocation. Users inspects visually the solution and need to understand why a ground station has been chosen, for instance in place of a different one that might looks better or more suitable. The explanation provided by the planner trough the APSI framework primitives described above is then used to synthesize an higher level explanation semantically significant for the final user.

The decision of allocating a pass in this domain is driven by the flaw in the on-board memory allocation. The memory is continuously being consumed by science operation. The position and the entity of the flaw give the justification for allocating a pass in the plan. The explanation provided by the Resource Activity Generator (that decided to allocate the pass) and the Resource Profile Bounder (that bounded the pass duration) includes information for the time window where the pass has been allocated and its duration. The earliest and latest start time for the possible pass allocation depends on the resource profile. The earliest start time depends on the minimal pass duration and on the amount of data in the memory: since a pass dumps at a rate higher than all the science data rate production, a pass cannot be allocated if

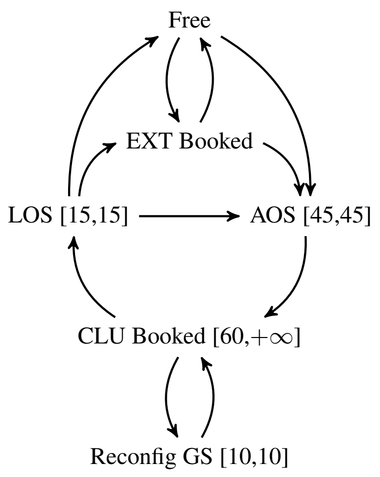


Figure 2: Ground station bookings state varible.

there are not enough data to download, which gives a lower temporal bound for the allocation. Conversely the latest start time for allocating a pass depends on the position of the flaw: the pass must start not too late to avoid the memory overwriting. The duration bounds depend on the ratio between the production rate in the interval and available dump rate.

Moreover, in the Cluster-II domain some of the constraints on ground stations booking are modeled in a state variables definition. Before the actual begin of tracking, a ground station configuration activity of 45 min has to be considered. During this time, the ground station must be available and should not be booked by any other activity. After the end of tracking, a ground station post-pass activity of 15 min has to be considered. Similar to the prepass activity, no other activity should be booked during this time. When two consecutive passes of two different Cluster-II spacecrafts are scheduled on the same ground station, the time accounting for the post and prepass activities can be reduced to exactly 10 min (in order to change between the different carrier frequencies of the spacecraft). If this 10 min interval cannot be satisfied, a configuration time of at least 1 h (15 min post-pass and 45 min pre-pass activities) has to be considered. Moreover, for cost efficiency reasons, scheduled passes should have a duration of at least 1 h.

To represent the configuration time, intermediate states are introduced on the booking timeline of the ground station and transition rules are specified, hence the state variable in Figure 2 models the ground station booking logic. In this state variable, a cluster booking has a minimal duration of 60 minutes and it is possible to pass between two bookings either trough a reconfiguration status of 10 minutes or passing by an LOS (Loss Of Signal) status of 15 minutes and a successive AOS (Acquisition Of Signal) of 45 minutes. Non-cluster bookings are modeled as states of external booking and for them it is always required an AOS phase before that a Cluster pass can be booked.

In addition to that there are constraints among the values taken by the different state variables and the resource allocations: a ground station must be visible and should not be booked by any other satellite in order to be able to book a Cluster ground station pass. The pass is also synchronized with an activity that dumps the memory, according with the link budget available and it has to be scheduled against science modes (a pass start or end should stay 6 to 15 minutes away from a science mode switch because of the unaffordable computational effort on board to start/finish a pass and contemporary switch a science mode).

Hence the position and duration of a pass (in within the bounds decided by the resource allocator described above) is also constrained by other planning decisions: (a) the pass must be synchronized with ground station's visibility windows; (b) must be properly scheduled with respect to other events on the same ground station, to acquire and release the station; (c) must be synchronized with the science modes, to avoid start or end points to be too close to science modes switch; (d) must be properly synchronized with available downklink rates for the selected station and (e) must guarantee a proper de-overlapping with other passes scheduled for the same satellite on other ground stations. (a), (c) and (d) are enforced by the Domain Theory Unfolder, (b) is enforced bu the Timeline Completer and (e) is enforced by the schedulers. Combining the explanations provided by these solvers, to the user is provided a trace of pass allocation in the plan that includes: (a) a window of opportunity for allocating the pass, explained in terms of memory filling; (b) a choice of a ground station, explained in terms of station availabilities and other concurrent use of the same station; (c) a position and duration of a pass, explained in terms of station usage logic, other passes allocation for the same satellite and memory consumption.

Discussion

Our planning/solving approach is based on a constructive/iterative schema driven by flaw resolution. A flaw entails a resolution step which provides explanation on the updates performed on the plan to fix the flaw. This approach integrates different types of information that concur in explaining the plan process and result: temporal information, cause-effect relationships and resource allocations.

Compared to explanations that can be provided on action based planning (STRIPS and PDDL), our approach has the advantage of exploiting explicit temporal information and scheduling constraints to justify plan statements not only causally but also temporally (and in space domain often the most important issue is not to explain why something appears in the plan but mostly when things happens and in which order). On the other side, the timeline-based approach to planning does not make explicit the preconditions for an action. As a consequence, it is more difficult, in case of failures not induced by resource or temporal conflicts, to identify a closed set of violated logical properties to justify the issue.

Besides that, in our experience there are various aspects to be considered in order to make the explanation an added value in the process.

First of all the planner is part of a tool which can be seen as a partner of the end-user in solving the problems ('man in the loop' principle). Therefore, in order to have an efficient collaboration, it is fundamental not only to explain the final solution but also the intermediate states and, more in general, the decisions taken during the solving process (Smith 2012). In this respect, the flaw based approach provides a step-by-step explanation not only on which actions are taken but also on why and when they are taken.

A second important aspect is the explanation of failures. In real applications it is not only important to be able to justify a plan, but we need often also to explain why a plan could not be found. This helps in tailoring system configurations and in analyzing problems. Again the flaw based approach gives a great support on this: starting from the unresolvable flaw that led to the failure of the planning process, it is often possible to frame the issue and decide for instance if it is a logical problem or a resource allocation problem.

Last but not least, the hierarchical approach that bases the explanation on increasing level of abstraction allows a good trade-off between generality and significance of the information. Temporal e logical allocation of values on the plan are explained, bottom up, as effect of propagation of temporal statements generated as consequence of solving steps (plan expansion or scheduling for instance), which in turns gives an explanation of higher level statements in the plan that play a role in that allocation. This analysis resulted useful in explaining solutions and failures of the planning process.

Pursuing a top down approach, the explanation starts from the high level goals and following the trace of the solving steps applied, frames the effects of the process on the plan (what provides support for the plan, which effect the goal has on the resource profiles and so on). This analysis proved to be a valid support for re-planning and mixed-initiative planning, where is useful to understand which effects a goal has on the whole plan and how it affects the planning process.

Notwithstanding this initial achievements, providing explanation remains in general an hard issue to face. First of all the solving steps and the set of constraints that influence a plan statement not always give a significant or usable information.

Regarding failure explanation for instance, a flaw can be the result of large set of combined effects, to the point that is not identifiable where and how to react. Also the flaw resolution process can sometimes make hard the explanation of the result. Some flaws can be resolved too early for instance, influencing the following steps of the process to the point that the final plan is not explainable. Optimization also makes the explanation more difficult: some optimization algorithms are not explainable (as genetic ones for instance).

Probably the most important issue is related with the explanation generation vs presentation: to be useful the explanation must be presented semantically close to the user, while is currently syntactically generated close to the data model and/or solving process. The synthesis of a semantically significant explanation, as shown in the examples presented in this paper, is still a cognitive process that requires a deep understating of the solving process first and of the planning domain also.

References

James Allen. Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 26(11):832–843, 1983.

APSI. APSI Software Distribution Web Site. https://essr.esa.int/project/apsi-advanced-planning-and-scheduling-initiative, 2017.

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

Amedeo Cesta, Gabriella Cortellessa, Simone Fratini, Angelo Oddi, and Giulio Bernardi. Deploying Interactive Mission Planning Tools - Experiences and Lessons Learned. *JACIII*, 15(8):1149–1158, 2011.

S. Chien, D. Tran, G. Rabideau, S. Schaffer, D. Mandl, and S. Frye. Timeline-Based Space Operations Scheduling with External Constraints. In *ICAPS-10. Proc. of the 20th International Conference on Automated Planning and Scheduling*, 2010.

DARPA. Explanable Artificial Intelligence (XAI), August 2016. Defense Advanced Research Projects Agency, Program n. DARPA-BAA-16-53.

M. Fox, D. Long, and D. Magazzeni. Explainable Planning. In *IJCAI Workshop on XAI*, 2017.

J. Frank and A. Jonsson. Constraint Based Attribute and Interval Planning. *Journal of Constraints*, 8(4):339–364, 2003.

S. Fratini and A. Cesta. The APSI Framework: A Platform for Timeline Synthesis. In *Proceedings of the 1st Workshops on Planning and Scheduling with Timelines at ICAPS-12 (PSTL-12), Atibaia, Brazil*, 2012.

S. Fratini, N. Policella, N. Faerber, A. De Maio, A. Donati, and B. Sousa. Resource Driven Timeline-Based Planning for Space Applications. In *Proceedings of the 9^{th} International Workshop on Planning and Scheduling for Space, IWPSS15*, 2015.

S. Fratini, N. Faerber, N. Policella, and B. Sousa. TIAGO Tool for Intelligent Allocation of Ground Operations on Cluster-II. In *SPARK-17. Scheduling and Planning Applications woRKshop at ICAPS*, Pittsburgh, USA, 2017.

A.K. Jonsson, P.H. Morris, N. Muscettola, K. Rajan, and B. Smith. Planning in Interplanetary Space: Theory and Practice. In *AIPS-00. Proc. of the Fifth Int. Conf. on Artificial Intelligence Planning and Scheduling*, pages 177–186, 2000.

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

N. Policella, A. Cesta, A. Oddi, and S. F. Smith. From Precedence Constraint Posting to Partial Order Schedules. *AI Communications*, 20(3):163–180, 2007.

N. Policella, H. Oliveira, and E. Benzi. Planning Spacecraft Activities: An automated approach. In *Proceedings of the 23rd International Conference on Automated Planning and Scheduling, ICAPS 2013*, 2013.

D.E. Smith, J. Frank, and A.K. Jonsson. Bridging the Gap Between Planning and Scheduling. *Knowledge Engineering Review*, 15(1):47–83, 2000.

D.E. Smith. Planning as an iterative process. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence, AAAI 2012*, pages 2180 – 2185, 2012.