Toward a CSP-based Approach for Energy Management in Rovers

Daniel Díaz and Maria D. R-Moreno
Universidad de Alcala
Alcala de Henares, Madrid, Spain
Email: {daniel.diaz, mdolores}@aut.uah.es

Amedeo Cesta, Angelo Oddi and Riccardo Rasconi
CNR – Italian National Research Council
ISTC, Rome, Italy
Email: ⟨name.surname⟩@istc.cnr.it

***Abstract*—This paper presents recent results on applying robust state-of-the-art AI Planning and Scheduling (P&S) techniques to mobile space robotic domains. We introduce an adaptation of an advanced constraint-based, resource driven reasoner for deciding feasible sequences of movements for a mobile robot in charge of executing a set of mission exploration-related jobs in a planetary terrain by reasoning upon complex temporal and resource constraints, in special energy demands. The major contribution of this paper is the inclusion of autonomous energy management capabilities within the general problem solving method.**

***Keywords*-Automation and Robotics; robust constraint-based action scheduling; precedence constraint posting; autonomous energy management**

I. Introduction

Traditionally, Artificial Intelligent (AI) Planning and Scheduling (P&S) techniques has been successfully applied to *mechanistic problems* that assume manufacturers-like conditions. Classical AI-based autonomous control systems are rooted on the conservative control theory basis which pursues an absolute and flawless governance, even in the face of non-nominal situations: contingency plans are designed in advance for every anomalous situation. But a new form of AI P&S systems that considers *uncertainty* as the main and partially controllable principle is increasingly demanded, since *real problems* are becoming scientifically interesting. With real problems we mean, close-to-human-intellect tasks such as *fluidly plan movements to avoid obstacles to achieve a specific goal*: a challenging AI P&S problem that claims more advanced capabilities than the actually provided by classical approaches, such as efficient mechanisms for managing complex and diverse temporal and resource constraints (e.g., energy demands, restrictive deadlines and resource availability, etc.).

Space exploration is actually self-proclaimed as a scientific discipline that is driving such paradigm shift, by disrupting old-fashioned deterministic rules that considered everything guessed in advance, in expenses of a more flexible and robust autonomous control systems.

Particularly, the European Space Agency (ESA) is currently bumping into different real problems that intrinsically hold uncertainty at its core definition: the increasing interest in evolving current telerobotics capabilities towards a complete mixed-initiative [1] strategy implementation that enables shifting control modes from purely manual to fully autonomous mission operations. Intelligent error handling mechanisms coupled with basic high level mission planning techniques currently represent the state-of-the-art in "Automation and Robotics (A&R)". Continuing to promote autonomy for future space missions certainly entails enormous benefits such as the reduction of operational costs, the enablement of opportunistic science, or the increase of mission quality in terms of safety, science return, reliability and flexibility. Very often, for some space missions such as deep-space probes, ground intervention may be slow (at the best cases) due to propagation signal delays and low bandwidth, or even impossible because of the occlusion of the line-of-sight communication when orbiting remote planets [2].

In order to formally cope with the infusion of autonomy in space mission operation, the ESA has established a reference decomposition scheme (ECSS Mission Execution Autonomy Levels[1]) (see table 1) that sort out the different flavours in automation applied to space mission operation in four well-founded levels (ranging from *dummy* or manually controlled to fully autonomous robots).

Level	Description	Functions
E1	Mission execution under ground control; limited on-board capability for safety issues	Real-time control for nominal operations. Execution of time-tagged commands for safety issues
E2	Execution of pre-planned, ground-defined, mission operations on-board	Capability to store time-based commands in an on-board scheduler
E3	Execution of adaptive mission operations on-board	Event-based autonomous operations. Execution of on-board operations control procedures
E4	Execution of goal-oriented mission operations on-board	Goal-oriented mission re-planning

Figure 1: ECSS Mission Execution Autonomy Levels.

[1]Contained within the ECSS-E-70-1 Space Segment Operability Standard document (available at www.ecss.nl)

It is basically a "functional authority" decomposition scheme ruled by a mixed-initiative principle and adjustable autonomy-based methods, that enables different degrees of cooperation between the human operators and the autonomous system (typically on-board). Since automation refers to the fully or partial replacement of a function previously carried by the human operators, it can be seen (the automation itself) as a continuum spectrum of which levels vary according to different criteria [3] such as human-intervention or shared responsibility degrees.

For instance, a close-to-manual operational mode (E1 level in the ECSS table) entrusts very few decision-making responsibilities on the autonomous system (typically restricted to detect critical system failures and set up system in standby), meanwhile in a highly-automated operational mode, the human is relegated as a mere supervisor within the mission execution loop (see figure 2).

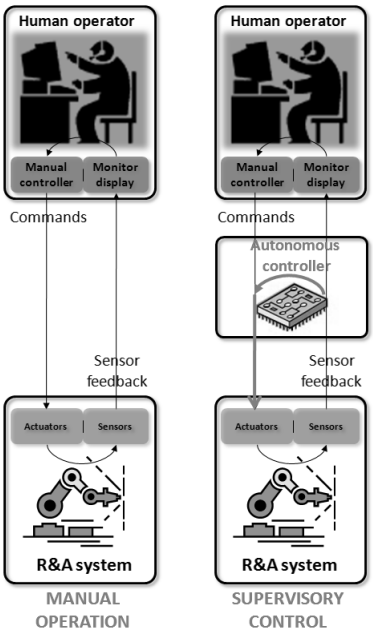

Figure 2: Adjustable autonomy-based execution control scheme: (left) manual operation; (right) mixed-initiative control.

In this context, the paper presents recent results obtained within a Ph.D. program on the topic of *Autonomy for Interplanetary Missions* funded and supported by ESA, which pursues to contribute on the development of a fully A&R system that succeed at executing E4-level (according to the ESA's ECSS Mission Execution Autonomy Levels) mission operational modes. More concretely, our current work is mainly aimed at developing a model-based autonomous control system for robust AI mission P&S and flexible reactive execution, applied to highly dynamic and unstructured mobile space robotics domains.

In this paper, we focus on a particular case scenario that involves synthesizing feasible sequences of movements for a mobile robot as it executes a set of mission exploration-related jobs, by reasoning upon complex temporal and resource constraints such as energy demands. This implies the ability of the robot of deciding smooth and efficient long-range routes by considering problem-specific parameters such as distances, energy costs for navigation and science activities, as well as energy gains due to solar charging. In spite of the complete problem description which also consider that the robot has to be capable of providing *rapid responses* in the face of contingent situations (such as an unexpected energy overconsumption), this work is mainly concerned with the scheduling dimension of the problem rather than on defining reactive strategies. We exploit state-of-the-art AI scheduling techniques both to provide baseline schedule solutions. More concretely, we use an advanced constraint-based, resource-driven solving procedure based on the last reported results in constraint-based P&S techniques, with particular attention to the "Precedence Constraint Posting" (PCP) approach as described in [4], [5], [6].

The remainder of the paper is structured as follows: Section II presents a detailed description of the previous problem. Section III describes the constraint-based, resource-driven solving procedure used to provide advanced reasoning capabilities, as well as a meta-heuristic optimization framework for solution optimization. In Section IV, important aspects of the energy model used, as well as its adaptation and use within our our problem solving method are outlined. Finally, a conclusions and future work section closes the paper.

II. Problem formulation

Synthesizing feasible sequences of movements for a mobile robot in charge of executing a set of mission exploration-related jobs in a planetary terrain, certainly involves enabling robot with highly autonomous and complex reasoning capabilities. In the current section, we study previous problem by considering the involved complexity sources as well as explaining the constraint-based representation that we use for problem solving.

A. Problem description.

Complexity nature of such a dynamic problem is mainly explained by its inherent uncertainty enclosed on the two following related reasoning premises: the highly unpredictability of the environment, and that the world can be only partially observed. Important consequences to be considered are inferred from previous assumptions, that in practice can be summarized as *the ephemeral validity of the plans during*

their execution. We actually propose a twofold strategy to tackle with uncertainty from scratch:

- *Statically:* both baseline and repaired plan solutions have to be robust enough in order to absorb (as much as possible) potential changes during the execution because of unexpected events (robust scheduling).
- *Dynamically:* a safe execution of the plans is performed by rapidly reacting to unexpected changes and providing new solutions consistent with the current situation (flexible reactive schedule execution).

Another important source of complexity arises from the combination of the different time and resource constrained parameters considered to reason upon during the generation of feasible plans. For instance, classical path planning involves reasoning (at a basic level) about the position of the robot. Hence, deciding feasible paths means mapping sequences of movements to reach a final location from a source point, by considering the Cartesian coordinates variables. But our problem is much more general since the robot's status is now given not only by its position but also by all the resources availability such as the battery level. For instance, in the case of a rover that uses solar power to charge its battery, the available battery energy may be necessary part of the robot's status since the available energy determines what movements or actions are feasible for the rover (if any) at each instant.

In order to better understand previous questions and other specific problem domain-related issues, next we present an hypothetical application example that illustrates how autonomy could be applied in deep space missions.

Mission Scenario. *The complete scenario describes a future ESA deep space mission where groups of autonomous rovers are in charge of collaborating for executing a set of linked experiments that take place at different laboratory facilities located along strategic Martian sites (see figure 3). Typical rover activities involve loading, transporting and unloading experimental products, as well as performing basic self-maintenance tasks. Rovers are able to autonomously synthesize efficient action plans by optimizing plan completion time meanwhile satisfying time and resource constraints such as energy availability. Furthermore, advanced onboard re-planning capabilities are necessary in order to hedge against environmental uncertainty during execution (i.e., rugged terrain, harsh weather conditions, etc.).*

We start our analysis from a significantly reduced version of the previous scenario that only considers a unique rover. We formally refer to the problem as "scheduling and executing a solar powered rover with limited battery capacity in an unstructured environment". It is a twofold problem

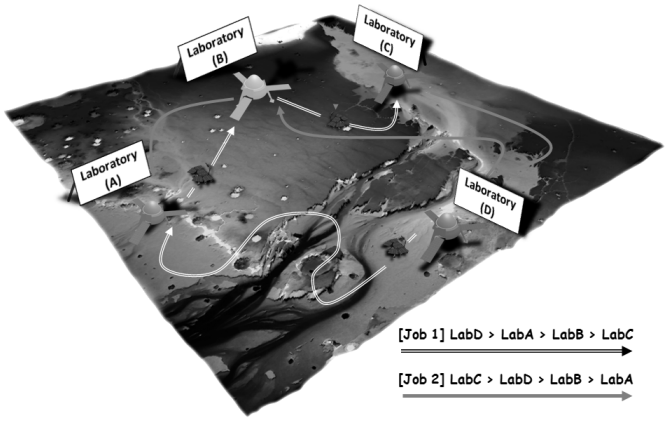

Figure 3: Futuristic mission scenario.

that basically aims at (1) synthesizing feasible sequences of movements for an arm-equipped rover *R* as it performs a set of jobs over time by synchronizing the use of a set of "laboratory facilities" $F = \{\mu_1, \ldots, \mu_n\}$; and (2) safety executing the jobs by taking into account environmental contingencies. Each job consists of a chain of activities related to the execution of a set of experiments $A_j = \{l_{0j}, u_{1j}, a_{1j}, l_{1j}, \ldots, u_{ij}, a_{ij}, l_{ij}, \ldots, u_{nj}, a_{nj}, l_{nj}, u_{(n+1)j}\}$ to be sequentially processed, where a_{ij} is the ith activity belonging to Job j performed by facility $\mu_{ij} \in F$, while u_{ij} and l_{ij} are, respectively, the *Unload* and *Load* activities that the rover must perform at laboratory μ_{ij} to release and hold items respectively.

l_{0j} is the Load activity performed by the rover at the initial storage location, while $u_{(n+1)j}$ is the Unload activity performed by the rover at the final storage location. Transportation activities τ_{ij} are considered as a different type of activities that correspond to the possible movements performed by the rover between two different facilities $\langle \mu_i, \mu_j \rangle$ within the same or distinct jobs, including initial and final storage locations.

The execution of all the activities is subject to the following temporal and resource constraints:

- *Resource availability*: each activity a_{ij} requires the exclusive use of μ_{ij} during its entire execution, i.e., no preemption is allowed. All the activities belonging to the same job use distinct laboratories. The activities u_{ij} and l_{ij} require the exclusive use of the rover *R*.
- *Processing time constraints*: all activities a_{ij}, u_{ij} and l_{ij} have a fixed processing time. Both the experimentation facilities and the rover can perform one operation at a time.

Previous constraints can be considered as "global" since they affect to all the activities and resources. On the contrary, the following constraints involve only a reduced group of

them and thus they are referred as "local":

- *Transportation time constraints*: for each pair $\langle \mu_{ix}, \mu_{jy} \rangle$ of facilities, the rover *R* is in charge of moving all items processed by the laboratory μ_{ix} to the laboratory μ_{jy}. This entails performing a Load activity l_{ix} at μ_{ix}, transporting the item at μ_{jy} (i.e., performing a τ_{ij} activity), and finally performing an Unload activity u_{jy} at μ_{jy}. The time necessary to travel from μ_{ix} to μ_{jy} is directly proportional to the travelling distance, and is modelled in the problem in terms of *sequence dependent setup times* st_{ij} which must be enforced between each $\langle l_{ix}, u_{jy} \rangle$ activity pair. All setup constraints satisfy the *triangle inequality property*, i.e., given three facilities μ_i, μ_j and μ_k, the condition $st_{ij} \leq st_{ik} + st_{kj}$ always holds.
- *rover maximum capacity constraint*: the rover is able to transport a maximum of *C* items at a time. This entails that the rover may chain different Load or Unload activities a maximum number of *C* times.
- *Battery capacity constraints*: the rover has a battery with a maximum Bat_{max} *(units of energy)* capacity of charge or *saturation level*. A specific usage threshold Bat_{thres} *(%)* delimits the total amount of energy that can be demanded as a safeguard for unexpected situations (i.e., if Bat_{max}*=100* and Bat_{thres}*=80(%)* it cannot be consumed less than 20 units of energy).
- *Energy production constraint*: the battery is continuously charged at a monotonic rate σ_{charge} *(unit energy/unit time)* through solar cells until the saturation level is reached (from this moment on, all collected energy is discarded).
- *Energy consumption constraints*: the rover activities (i.e., Load, Unload and Transporting) require a certain amount of energy that have to be available at the starting time of their execution. Meanwhile the Unload and Load activities always demand the same constant amount of energy, the consumption of the Transporting activities is given by an *energy cost map*: a table containing an estimation of the total energy consumed for travelling between each pair of locations. Such energy consumption estimation is computed on the basis of a combination of travelled distances and some specific terrain features such as slopes or rough soil surfaces.

The first part of the problem is referred to a purely scheduling problem whose goal is only about to synthesize a feasible schedule where both experiments and rover activities are synchronized towards the successful execution of all jobs. A feasible schedule solution is a schedule where, given an initial completion time or makespan, the start and end times of all activities are assigned while temporal and resource constraints are satisfied. Meanwhile the second part of the problem is involved with safety executing the resulting feasible schedule by considering the possible disturbances caused by the environmental threats. In this paper, we will be focused only on the scheduling side of the overall problem, i.e., provide feasible baseline schedule solutions.

B. Constraint-based problem representation.

Our solution considers the scheduling problem as a special type of a Constraint Satisfaction Problem (CSP) [7]. A general description of a scheduling problem as CSP involves a set of variables with a limited domain each, and a set of constraints that limits possible combinations. Hence, a feasible CSP solution is defined as an assignment of domain values to all variables which is consistent with all imposed constraints.

New problem modelling assumptions are considered in order to adapt the initial problem formulation to our constraint-based solution scheme. Hence, the problem is described now as follows (see figure 4): each laboratory μ_{ij} is considered as a binary resource that process all job operations a_{ij}. The Unload and Load activities are devised to be performed by a single robotic arm able to manage one item at a time (also modelled as a binary resource). We introduce an additional type of activity c_{ij} that requires the use of a cumulative resource (with a maximum capacity of *C*) to be processed by the rover, with the aim of modelling the rover capability of *simultaneously carrying multiple items*. The execution of c_{ij} activities starts when the corresponding Load activity l_{ij} starts, and finishes at the termination of the Unload activity u_{i+1j}.

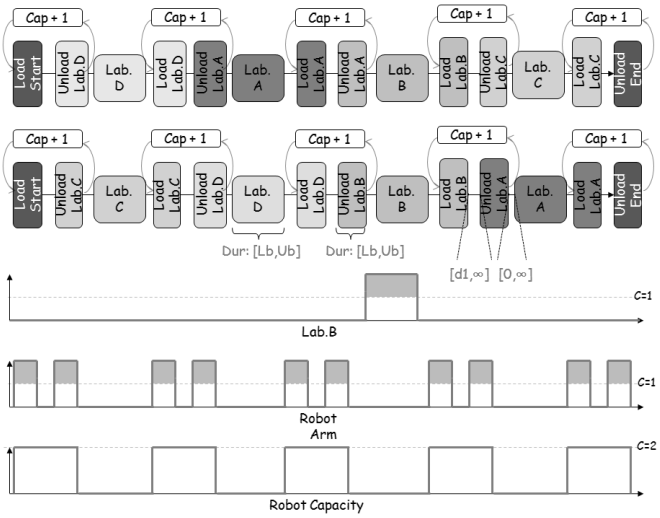

Figure 4: A cumulative resource is introduced to model the multi-capacity rover usage.

The battery is modelled as another cumulative resource according to the Simonis' model [8]. It has a maximum capacity of Bat_{max} and an usage threshold of Bat_{thres}), and processes two different sets of activities (see figure 5): the rover-related activities (i.e., Load, Unload and Transporting) that consume the battery, and the energy collecting activities that recharge the battery. The energy consuming activities are modelled so as to start when a specific amount of energy is demanded and finish at the "horizon" (the end of the schedule), i.e., this amount of energy is no longer available and thus the battery remains blocked until the end; meanwhile the energy producing activities are modelled as static activities that are anchored at the origin of the schedule and finish when energy is collected, i.e., the battery remains blocked from the beginning until a certain amount of energy becomes available.

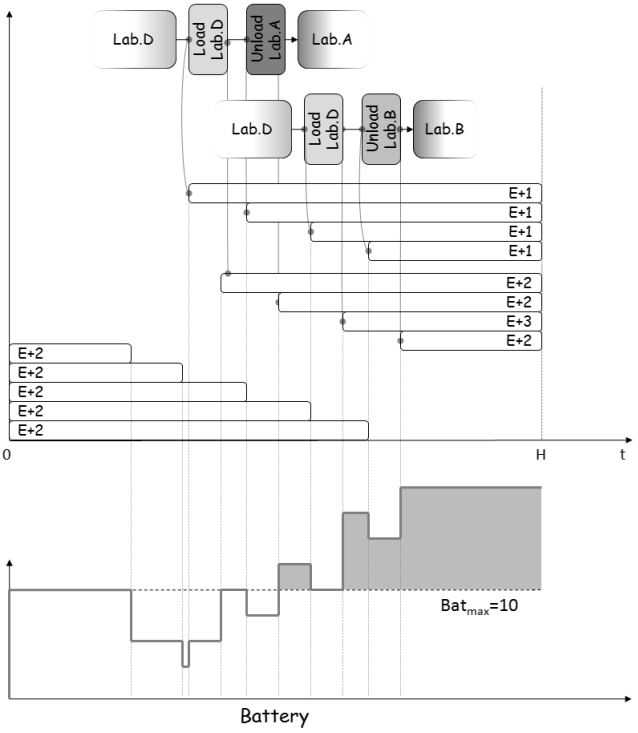

Figure 5: Example of an intersected energy collection/consumption profile

C. Related work.

We can mainly distinguish two different types of resources on literature attending to how a given activity demands and affects the availability of the resource: *reusable* and *consumable* resources. The former is also known as *renewable* or *cumulative*, and activities that use them may demand a certain quantity of resource during an interval of time. The sum of all activity demands cannot overpass the maximum level capacity. If the resource has unit capacity then it is known as *unary* or *discrete*, and the activities that use it have to be totally ordered since overlaps cannot occur. Examples of cumulative resources are the container of the rover used to transport items as a multi-capacity resource, and the robotic arm or the experimentation facilities as unary resource. The latter is also known as *reservoir* and it is a multi-capacity resource that can be consumed and/or produced by an activity. The battery of the rover is an example of consumable resource that is recharged by the solar cells and consumed by the robotic activities.

Research in CSP-based scheduling has been mainly focused on the development of effective heuristic-biased models that involve unary and cumulative resources, such as the precedence constraint posting algorithm proposed by [4] or the resource profile-driven algorithm ESTA [6] respectively. To the best of our knowledge, the majority of the work done in CSP-based scheduling with consumable resources is referred to "resource constraints representation and propagation problems" rather than proposing strategies for problem-solving. Hence, we can find expressive formalisms for expanding the basic Simple Temporal Network (STN) [9] foundations to explicitly deal with resource constraints in general, by defining positive and negative resource allocations to resource productions or consumptions events respectively. Some examples are: the work described in [10], that defines an augmented STN called "Activity Network" that allows a *piecewise constant* representation of the resource usage, on top of which an envelope-based resource consistency checking mechanism is built; or the "Linear Resource Temporal Network (LRTN)" proposed by [11] that enables bounding the resource availability for activity networks with linear resource impact, i.e., permits a *piecewise linear* representation of the resource usage along the duration of the activities.

A different direction to cope with consumable resource constraints proposes a simplified model that aims at reducing consumable constraints to a cumulative scheme by performing some simple transformations. An example is described in [8] and formally defines producer and consumer events with a "classical cumulative scheme" through the following conversion: since a producer event makes some amount of resource available at a specific time instant, such event is expressed as an activity that blocks the resource from the start until the resource becomes available; in the same way, a consumer event is modelled as an activity that uses the resource at a time point and remains blocked until the end, since the resource is no longer available.

III. The constraint-based, resource-driven reasoner (extended-ESTA)

To implement our solution for the first part of the problem (i.e., the purely scheduling part) we used a slight modification of the extended-ESTA [12] algorithm to enable *energy management capabilities*. Extended-ESTA is a constraint-based solving procedure based on the original ESTA [6] which also includes some basic principles of a recent extension of the SP-PCP (Shortest Path-based Precedence Constraint Posting) algorithm proposed in [13]. It is a constraint-based, resource-driven reasoner used in an previous work [12] aimed at providing a preliminary solution to the problem of "scheduling a single robot in a job-shop environment". In particular, extended-ESTA combines the ability of the original ESTA algorithm for reasoning upon cumulative resources, and the new set of *dominance conditions* introduced in [13] to consider the sequence-dependent setup times within the general resolution strategy. Both characteristics allows us, respectively, addressing conflicts on the multi-capacity robot usage and battery resources, meanwhile considering the distance-related metrics that constrain the duration of the transportation activities.

Generally, a CSP solution scheme can be seen as an iterative procedure that interleaves two main steps in each cycle:

- A *decision making step* where a "decision variable" is chosen to be assigned with a specific "domain value". The decision of the assignment to do next could be taken by either systematically following an exhaustive search technique (such as a simple depth-first search), or by using more efficient approaches that use variable and value ordering heuristics to guide the search process. Typical general purpose heuristics generate variable and value orderings by selecting the "most constrained variable (MCV)" and the "least constraining value (LCV)" respectively.
- A *propagation step* where a set of "inference rules" prune unfeasible solutions in advance, by removing elements from variable domains when a decision step is performed. Path consistency algorithms such as "all pairs shortest paths" are typically used.

More concretely, extended-ESTA basically implements the previous two steps in the following way: a (one-pass) greedy resource-driven method iterates by: (1) creating a meta-representation (meta-CSP) of the temporal CSP network to explicitly deal with resource aspects such as resource demands; and (2) solving the "resource contention peaks" (i.e., over-commitments) with new ordering constraints. The meta-CSP consist on the computation of the Minimal Critical Sets (MCSs) [14], i.e., minimal sets of activities that overlaps in time and demand the same resource causing over-commitments (see figure 6). The resolution of the resource contention peaks is biased by *slack-based search heuristics* [15] that selects the MCS with the smallest temporal flexibility (free temporal space), and post a precedence constraint between two activities that belong to the related MCS according to the following criteria: the greater the flexibility is retained after posting a precedence ordering constraint, the more desirable it is to post that constraint.

Algorithm 1: Conflict selection and resolution process.

```
Conflict ← SelectConflict (MetaCSP)
Precedence ← SelectPrecedence (Conflict)
GroundCSP ← PostConstraint (GroundCSP, Precedence)
```

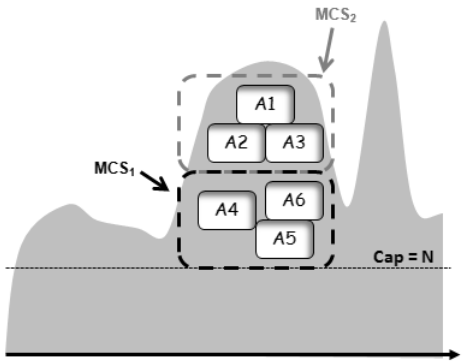

Figure 6: Meta-CSP generation example (MCSs computation).

Algorithm 1 depicts in detail the basic steps corresponding to the conflict resolution process previously explained: within the first step, a collection of candidate MCSs is computed; the second step selects the most constrained MCS and the ordering choice to solve it; last step imposes the new levelling constraint within the ground CSP.

Providing better solutions. Since the previous one-pass, greedy solution does not guarantee optimality, we used an efficient optimization framework (in contrast to the costly backtracking-based schemes) with the aim of providing better results. We adopted the advanced IFLAT [16] iterative sampling optimization schema that allows us to find lower solution's makespan and overcome situations where unsolved conflicts are encountered. The underlying idea is to iteratively run the extended-ESTA algorithm such that different paths are randomly explored within the search space, by using a "meta-heuristic strategy" to bias the process [17].

IV. Enabling autonomous energy management

The Simonis' model [8] in which we were inspired to represent the battery (a consumable resource by definition) as a

cumulative resource, has been modified and adapted in order to enable the extended-ESTA algorithm with the capability of efficiently reasoning upon energy-related issues. In the current section, we analyse some important limitations that arise from the adaptation of such a model to our cumulative-based scheme, and how we commit them.

A. The battery saturation problem.

Simonis' model assumes *stock-based* consumable resources with minimum/maximum values (such as a fuel tank or a storage warehouse), arising in flow shop or job shops environments. In other words, the model takes bounded consumable resource where the total amount of resource demanded/produced is know in advance, and globally will not exceed any level (when scheduled). For instance, within a scheduling system for crude oil transport in a pipeline network that aims at balancing the stock levels for the different sources and sinks. In this case, the whole crude oil may be enclosed within the total capacity of the whole pipeline network (if correctly distributed). In the contrary, our battery production constraints consider a virtually infinite source of energy (the sun) that will provide a continuum of input energy *despite of the capacity of the battery*.

We actually took a shortcut to this problem by assuming a limited amount of input energy and uniformly distribute it along a specific temporal horizon.

B. The energy consumption is path-dependent.

We recall that two different sets of energy consuming activities are considered: those referred to the robotic arm and locomotion systems respectively. The former always demand the same amount of energy since the duration of the arm-related activities is fixed and known in advance. Meanwhile the battery consumption in the latter case (i.e., the related to the transportation activities) will vary depending on the final route chosen.

Hence, the solution is to make some dependable estimation (during the solving process) for the transportation activities demand. More concretely, our approach iteratively estimates a *lower bound demand value for all the potential movements* (transportation activities that may take place after any Load/Unload activity) whenever a new solving constraint is posted. Algorithm 2 explains in detail the four basic steps of such recalculation process: within the first step, an Earliest Start Schedule (ESS)[2] is extracted for the partial CSP schedule solution (GroundCSP); the second step selects the (timely ordered) set of energy consuming transportation activities; third step resets all the demand values (to zero); and last step updates the demand values according to the energy cost map, if a potential movement may actually occur (i.e., when a pair of consecutive Load/Unload activities takes place at different locations).

[2]ESS: a consistent temporal assignment of all time points with their lower bound values.

Algorithm 2: Computation of the energy consumption profile (movements).

```
ESS ← extractESS (GroundCSP)
eTransportationActs ← extractOrdered (ESS)
ResetDemandValues (eTransportationActs)
UpdateDemandValues (eTransportationActs)
```

C. A preliminary empirical evaluation discussion.

Our current scheduling system, including the advancements described in this paper, has been implemented on top of the TRF software development environment [18]. First empirical results obtained by solving relatively-small problem instances (in the order of 2 jobs of 3 activities each) involved enormous computation times. We realized that one of the most critical bottlenecks was related to the large number of energy production activities managed. For instance, if we consider a small problem instance with an initial temporal horizon of 3000 seconds, and a monotonic energy production rate σ_{charge} = *0.5 (unit energy/unit time)*, we get 1500 production activities. In order to avoid the creation of such amount of activities we decided to consider the energy production profile as a piecewise constant function that, given a specific time instant, returns the amount of energy produced in such instant (see figure 7). This abstraction allows us to significantly improve the total computation time. An exhaustive experimentation with large problem instances is currently being performed, whose results are expected to be published shortly.

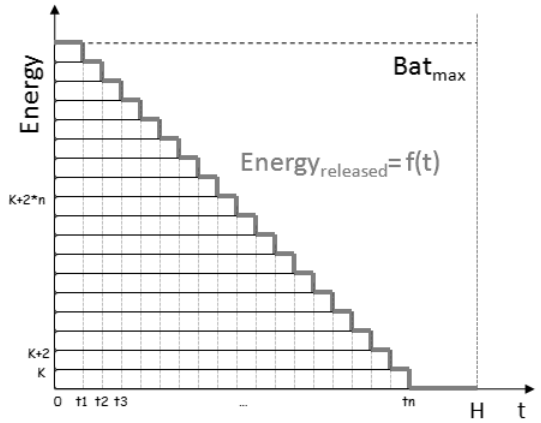

Figure 7: The energy production profile piecewise function.

V. Conclusions and future work

In this paper we have presented an enhanced version of the advanced constraint-based, heuristic-biased extended-ESTA solving method aimed at synthesizing feasible sequences of movements for a mobile robot as it executes a set of mission exploration-related jobs, by reasoning upon complex temporal and resource constraints, including energy demands. This problem implies the ability of the robot of deciding smooth and efficient long-range routes by considering problem-specific parameters such as distances, energy costs for navigation and science activities, as well as energy gains due to solar charging. The major contribution is referred to the adaptation of an existing resource model that allows us representing complex resources such as a battery, in the form of cumulative resources (i.e., those managed by the extended-ESTA solving method).

The work described in this paper represent part of a more general project aimed at developing a model-based autonomous control system for robust AI mission P&S and flexible reactive execution, applied to highly dynamic and unstructured mobile space robotics domains. Such envisioned control architecture will provide the rest of elements, such as reactive strategies and mechanisms, needed for coping with contingent situations arising during the solution schedule execution.

Acknowledgment

Daniel Diaz is supported by the European Space Agency (ESA) under the Networking and Partnering Initiative (NPI) *Autonomy for Interplanetary Missions* (ESTEC-No. 2169/08/NI/PA). CNR authors are partially supported by MIUR under the PRIN project 20089M932N (funds 2008). The second UAH author is supported by Castilla-La Mancha project PEII11-0079-8929. Special thanks to Michele Van Winnendael for his support throughout the whole course of the study.

References

[1] M. A. Goodrich, D. R. Olsen, J. W. Crandall, and T. J. Palmer, "Experiments in Adjustable Autonomy," in *Proceedings of IJCAI Workshop on Autonomy, Delegation and Control: Interacting with Intelligent Agents*, 2001, pp. 1624–1629.

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

[3] T. Inagaki and M. Itoh, "Trust, autonomy, and authority in human-machine systems: Situation-adaptive coordination for systems safety," in *Proceedings Cognitive Systems Engineering in Process Control International Symposium (CSEPC96)*, 1996, pp. 176–183.

[4] C. Cheng and S. Smith, "Generating Feasible Schedules under Complex Metric Constraints," in *Proceedings 12th National Conference on AI (AAAI-94)*, 1994.

[5] A. Oddi and S. Smith, "Stochastic Procedures for Generating Feasible Schedules," in *Proceedings 14th National Conference on AI (AAAI-97)*, 1997, pp. 308–314.

[6] A. Cesta, A. Oddi, and S. F. Smith, "A constraint-based method for project scheduling with time windows." *J. Heuristics*, vol. 8, no. 1, pp. 109–136, 2002.

[7] U. Montanari, "Networks of Constraints: Fundamental Properties and Applications to Picture Processing," *Information Sciences*, vol. 7, pp. 95–132, 1974.

[8] H. Simonis and T. Cornelissens, "Modelling producer/consumer constraints," in *Proceedings of the First International Conference on Principles and Practice of Constraint Programming*. London, UK: Springer-Verlag, 1995, pp. 449–462. [Online]. Available: http://portal.acm.org/citation.cfm?id=647484.726174

[9] R. Dechter, I. Meiri, and J. Pearl, "Temporal constraint networks," *Artificial Intelligence*, vol. 49, pp. 61–95, 1991.

[10] N. Muscettola, "Incremental Maximum Flows for Fast Envelope Computation," in *ICAPS-04. Proceedings of the* 14^{th} *International Conference on Automated Planning & Scheduling*, 2004, pp. 260–269.

[11] J. Frank and P. H. Morris, "Bounding the resource availability of activities with linear resource impact," in *In Proc. Int. Conf. on AI Planning and Scheduling, ICAPS*, 2007.

[12] D. Diaz, M. R-Moreno, A. Cesta, A. Oddi, and R. Rasconi, "Scheduling a Single Robot in a Job-Shop Environment throug Precedence Constraint Posting," in *Proceedings of the* 24^{th} *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2011)*, 2011.

[13] A. Oddi, R. Rasconi, A. Cesta, and S. Smith, "Iterative-Sampling Search for Job Shop Scheduling with Setup Times," in *COPLAS-09. Proc. of the Workshop on Constraint Satisfaction Techniques for Planning and Scheduling Problems at ICAPS*, 2009.

[14] P. Laborie and M. Ghallab, "Planning with Sharable Resource Constraints," in *Proceedings of the* 14^{th} *Int. Joint Conference on Artificial Intelligence (IJCAI-95)*, 1995.

[15] S. Smith and C. Cheng, "Slack-Based Heuristics for Constraint Satisfaction Scheduling," in *Proceedings 11th National Conference on AI (AAAI-93)*, 1993.

[16] A. Oddi, A. Cesta, N. Policella, and S. F. Smith, "Combining Variants of Iterative Flattening Search," *Journal of Engineering Applications of Artificial Intelligence*, vol. 21, pp. 683–690, 2008.

[17] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM Comput. Surv.*, vol. 35, no. 3, pp. 268–308, 2003.

[18] A. Cesta and S. Fratini, "The Timeline Representation Framework as a Planning and Scheduling Software Development Environment," in *PlanSIG-08.* 27^{th} *Workshop of the UK Planning and Scheduling Special Interest Group, Edinburgh, UK, December*, 2008, pp. 17–24.