54-63
N93-18663

Decomposability and Scalability in Space-Based Observatory Scheduling

Nicola Muscettola and Stephen F. Smith
The Robotics Institute
Carnegie Mellon University
5000 Forbes Av.
Pittsburgh, PA 15213

Introduction

The generation of executable schedules for space-based observatories is a challenging class of problems for the planning and scheduling community. Existing and planned space-based observatories vary in structure and nature, from very complex and general purpose, like the Hubble Space Telescope (HST), to small and targeted to a specific scientific program, like the Submillimeter Wave Astronomy Satellite (SWAS). However the fact that they share several classes of operating constraints (periodic loss of target visibility, limited onboard resources, like battery charge and data storage, etc.) suggests the possibility of a common approach. The complexity of the problem stems from two sources. First, they display the difficulty of classical scheduling problems: optimization of objectives relating to overall system performance (e.g., maximizing return of science data), while satisfying all constraints imposed by the observation programs (e.g., precedence and temporal separation among observations) and by the limitations on the availability of capacity (e.g., observations requiring different targets cannot be executed simultaneously). Second, a safe mission operation requires the detailed description of all the transitions and intermediate states that support the achievement of observing goals and are consistent with an accurate description of the dynamics of the observatory; this constitutes a classical planning problem.

Another characteristic of the problem is its large scale. The size of the pool of observations to be performed on a yearly horizon can typically range from thousands to even tens of thousands, and, for large observatories, the dynamics of system operations involves several tens of interacting system components. To effectively deal with problems of this size, it is essential to employ problem and model decomposition techniques. In certain cases, this requires the ability to represent and exploit the available static structure of the problem (e.g., interacting system components); in other cases, where an explicit structure is not immediately evident (e.g., interaction among large numbers of temporal and capacity constraints), the problem solver should be able to dynamically focus on different parts of the problem, exploiting the structure that emerges during the problem solving process itself.

In this paper, we discuss issues of problem and model decomposition within the HSTS scheduling framework. HSTS was developed and originally applied in the context of the HST scheduling problem, motivated by the limitations of the current solution and, more generally, the insufficiency of classical planning and scheduling approaches in this problem context. We first summarize the salient architectural characteristics of HSTS and their relationship to previous scheduling and AI planning research. Then, we describe some key problem decomposition techniques supported by HSTS and underlying our integrated planning and scheduling approach, and discuss the leverage they provide in solving space-based observatory scheduling problems.

Planning and scheduling for space-based observatories

The management of the scientific operations of the Hubble Space Telescope is a formidable task; its solution is the unique concern of an entire organization, the Space Telescope Science Institute (STScI). The work of several hundred people is supported by several software tools, organized in the Science Operations Ground System (SOGS). At the heart of SOGS is a FORTRAN-based software scheduling system, SPSS, originally envisioned as a tool which would take astronomer viewing programs for a yearly period as input and produce executable spacecraft instructions as output. SPSS has had a somewhat checkered history [Wal89], due in part to the complexity of the scheduling problem and in part to the difficulty of developing a solution via traditional software engineering practices and conventional programming languages. To confront the computational problems of SPSS, STScI has developed a separate, knowledge-based tool for long term scheduling called SPIKE [Joh90]. SPIKE accepts programs approved for execution in the current year and partitions observations into weekly time buckets, each of which can then be treated as a smaller, more tractable, short term scheduling problem. Detailed weekly schedules are generated through the efforts of a sizable group of operations astronomers, who interactively utilize SPSS to place observations on the time line.

In the HSTS project we have addressed the short term problem in the HST domain, efficiently generating detailed schedules that account for the major telescope's operational constraints and domain optimization objectives. The basic assumption is to treat resource allocation (scheduling) and auxiliary task ex-

pansion (planning) as complementary aspects of a more general process of constructing behaviors of a dynamical system [Mus90].

Two basic mechanisms provide the basis of the HSTS approach:

1. a *domain description language* for modeling the structure and dynamics of the physical system at multiple levels of abstraction.
2. a *temporal data base* for representing possible evolutions of the state of the system over time (i.e. schedules).

The natural approach to problem solving in HSTS is an iterative posting of constraints extracted either from the external goals or from the description of the system dynamics; consistency is tested through constraint propagation. For more details, see [MSCD91].

Three key characteristics distinguish the HSTS framework from other approaches:

1. the explicit decomposition of the state of the modeled system into a finite set of "state variables" evolving over continuous time. This enables the development of scheduling algorithms that exploit problem decomposability and provides the necessary structure for optimizing resource utilization.
2. the flexibility along both temporal and state value dimensions that is permitted by the temporal data base (e.g., the time of occurrence of each event does not need to be fixed but can float according to the temporal constraints imposed on the event by the process of goal expansion). This flexibility contributes directly to scheduling efficiency, since overcommitment (and hence the greater possibility of the subsequent need to backtrack) can be avoided.
3. the flexibility of the constraint posting paradigm to accommodate a range of problem solving strategies (e.g., forward simulation, back chaining, etc.). This allows the incorporation of algorithms that opportunistically exploit problem structure to consistently direct problem solving toward the most critical tradeoffs that need to be made.

The importance of integrating these three features within a single framework can be appreciated by considering the limitations of other approaches that address them separately or partially.

Planning research has focused on the problem of "compiling" activity networks that bring about desired goal states from more basic representations of the effects of actions in the world. In contrast to HSTS, however, the modeling assumptions of most approaches [FHN72, Wil88] do not support explicit representation of temporal constraints depending on continuous time (e.g., task duration, temporal separation between events), and representation of the world state is not structured into state variables. More recent planning frameworks have only partially addressed these issues [Lan88, DFM88, Ver83]. Furthermore, in most cases, these frameworks have placed fairly rigid constraints on the manner in which solutions are developed (e.g., strict reliance on top down goal refinement with forward simulation[DFM88]), preventing an adequate consideration of efficient resource allocation over time, an issue of fundamental importance in the space-based observatory scheduling domain.

The monitoring of state variables over continuous time has always been at the core of scheduling research [Bak74]. Operations research has produced optimal solutions for very simple scheduling problems [Gra81, BS90] or has focused on the definition of dispatch priority rules [PI77] for more realistic problems. More recent research in constraint-based scheduling [SOM+90, Sad91], has demonstrated the advantages of dynamically focusing decision-making on the most critical decisions first. HSTS differs from other scheduling approaches in its temporal flexibility and in its ability to dynamically expand auxiliary goals and activities.

Issues in Integrating Planning and Scheduling

We now highlight some aspects of our approach that support the development of solutions for large scale scheduling problems in complex dynamical domains and, in particular, their relevance to space-based observatory domains.

Use of Abstraction

The use of abstract models has long been exploited as a device for managing the combinatorics of planning and scheduling. In HSTS, where models are expressed in terms of the interacting state variables of different components of the physical system and its operating environment, an abstract model is one which summarizes system dynamics in terms of more aggregate structural components or selectively simplifies the represented system dynamics through omission of one or more component state variables. Given the structure of space-based observatory scheduling problems, the use of an abstract model provides a natural basis for isolating overall optimization concerns, and thus providing global guidance in the development of detailed, executable schedules. In the case of the HST, a two-level model has proved sufficient. At the abstract level, telescope dynamics is summarized in terms of a single state variable, indicating, at any point in time, whether the telescope (as a whole) is taking a picture, undergoing reconfiguration, or sitting idle. The duration constraints associated with reconfiguration at this level are temporal estimates of the time required by the complex of actual reconfiguration activities implied by the detailed model (e.g., instrument warmup and cooldown, data communication, telescope repointing). Execution of an observation at the abstract level requires only satisfaction of this abstract reconfiguration constraint, target visibility (a non-controllable state variable accessible to both the abstract and detailed models), and any user specified temporal constraints. Thus, the description at the abstract level looks much like a classically formulated scheduling problem: a set of user requests that must be sequenced on a single resource subject to specified constraints and allocation objectives.

Planning relative to a full detailed level is necessary to ensure the viability of any sequencing decisions

made at the abstract level and to generate and coordinate required supporting system activities. The degree of coupling between reasoning at different levels depends in large part on the accuracy of the abstraction. In the case of HST, decision-making at abstract levels is tightly coupled; each time a new observation is inserted into the sequence at the abstract level, control passes to the detailed level and supporting detailed system behavior segments necessary to achieve this new goal are developed. Given the imprecision in the abstract model, goals posted for detailed planning cannot be rigidly constrained; instead preferences are specified (e.g., "execute as soon as possible after obs1"). The results of detailed planning at each step are propagated upward to provide more precise constraints for subsequent abstract level decision-making.

Model Decomposability and Incremental Scaling

Large problems are naturally approached by decomposing them into smaller sub-problems, solving the sub-problems separately and then assemble the sub-solutions. We can judge how the problem solving framework supports modularity and scalability by two criteria:

- the degree by which heuristics dealing with each sub-problem need to be modified when adding sub-problem assembly heuristics to the problem solver;
- the degree of increase of the computational effort needed to solve the problem versus the one needed to solve the component sub-problems

To test the scalability of the HSTS framework, we conducted experiments with three models of the HST operating environment of increasing complexity and realism, respectively denoted as SMALL, MEDIUM and LARGE model. All models share a representation of the telescope at the abstract level as a single state variable; they differ with respect to the number of components modeled at the detailed level. The SMALL model contains a state variable for the visibility of each of the celestial objects of interest with respect to the orbiting telescope, a state variable for the pointing state of the telescope, and three state variables for the state of an instrument, the Wide Field Planetary Camera (WFPC). The MEDIUM model adds two state variables for an additional instrument, the Faint Object Spectrograph (FOS), while the LARGE model includes eight additional state variables accounting for data communication. The LARGE model is representative of the major operating constraints of the domain. Figure 1 shows the relations among the various models.

The problem solver for the SMALL domain contains heuristics to deal with the interactions among the different components of the WFPC (e.g., when a WFPC detector is being turned on, make sure that the other WFPC detector is kept off), with the pointing of the HST (e.g., select a target visibility window to point the telescope), and with the interaction among WFPC state and target pointing (e.g., observe while the telescope is pointing at the proper target). The heuristics

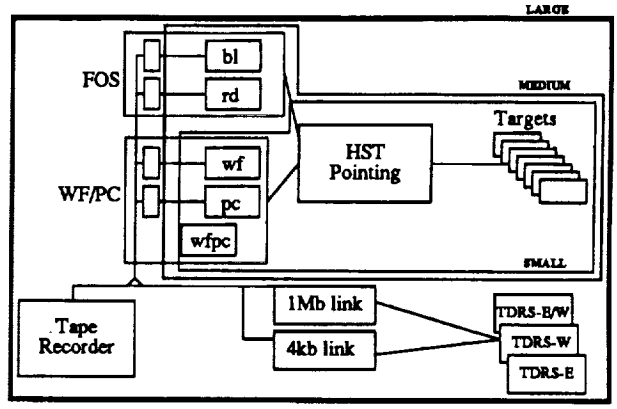

Figure 1: The SMALL, MEDIUM and LARGE HST models.

added for the MEDIUM domain deal with the interactions within the FOS, between FOS and HST pointing state, and between FOS and WFPC. Since the nature of the new interactions is very similar to those of the SMALL model, the additional heuristics are obtained by simply extending the domain of applicability of the SMALL's heuristics. Finally, for the LARGE model we have the heuristics used in the MEDIUM domain, with no change, plus heuristics that address data communication and interaction among instrument states and data communication (e.g., do not schedule an observation on an instrument if data from the previous observation has not yet been read out of its data buffer). The previous discussion supports the scalability with regard to the structure of the problem solvers.

To verify scalability with respect to the degree of computational effort, we run a test problem in the SMALL, MEDIUM and LARGE domain; the test consists of a set of 50 observation programs, each containing a single observation with no user-imposed time constraints. The experiments were run on a TI Explorer II+ with 16 Mbytes of RAM memory.

Table 1 supports the claim of scalability with respect to the required computational effort. The measure of the size of the model (number of state variables) excludes target and communication satellite visibilities since these can be considered as given data. The number of tokens indicates the total number of distinct state variable values that constitute the schedule. The temporal separation constraints are distance constraints that relate two time points on different state variables; their number gives an indication of the amount of synchronization needed to coordinate the evolution of the state variables in the schedule.

Notice that since the heuristics that guide the planning search exploit the modularity of the model and the locality of interactions, the average CPU time (excluding garbage collection) spent implementing each required compatibility constraint (corresponding to an atomic temporal relation among tokens) remains relatively stable. In particular, given the high similarity of the nature of the constraints between the SMALL and the MEDIUM models, this time is identical in the two

Model	SMALL	MEDIUM	LARGE
State Variables	4	6	13
Tokens	587	604	843
Time Points	588	605	716
Temporal Constraints	1296	1328	1474
CPU Time / Observation	11.62	12.25	21.74
CPU Time / Compatibility	0.29	0.29	0.33
Total CPU time	9:41.00	10:11.50	18:07.00
Total Elapsed Time	1:08:36.00	1:13:16.00	2:34:07.00
Schedule Horizon	41:37:20.00	54:25:46.00	52:44:41.00

Table 1: Performance results. The times are reported in hours, minutes, seconds and fraction of seconds

cases. The total elapsed time spent generating an executable schedule for the 50 observations is an acceptable fraction of the real time horizon covered by the schedules; this indicates the practicality of the framework in the actual HST operating environment.

Exploiting Opportunism to Generate Good Solutions

In the experiment just described, a simple dispatch-based strategy was used as a basis for overall sequence development: simulating forward in time at the abstract level, the candidate observation estimated to incur the minimum amount of wait time (due to HST reconfiguration and target visibility constraints) was repeatedly selected and added to the current sequence. This heuristic strategy, termed "nearest neighbor with look-ahead" (NNLA), attends directly to the global objective of maximizing the time spent collecting science data. However, maximization of science viewing time is not the only global allocation objective.

One critical tradeoff that must be made in space-based observatory scheduling is between maximizing the time spent collecting science data and satisfying absolute temporal constraints associated with specific user requests. The scheduling problem is typically over-subscribed; i.e., it will generally not be possible to accommodate all user requests in the current short term horizon and some must necessarily be rejected. Those requests whose user-imposed time windows fall inside the current scheduling horizon become lost opportunities if rejected. Those without such execution constraints may be reattempted in subsequent scheduling episodes.

As indicated above, the first objective (minimizing telescope dead time) is amenable to treatment within a forward simulation search framework. However, a forward simulation provides a fairly awkward framework for treating the second objective (minimizing rejection of absolutely constrained goals). A goal's execution window may be gone by the time it is judged to be the minimum dead time choice. Look-ahead search (i.e. evaluation of possible "next sequences" and potential rejections) can provide some protection against unnecessary goal rejection but the general effectiveness of this approach is limited by combinatorics. A second sequencing strategy of comparable computational complexity that directly attends to the objective of minimizing rejection of absolutely constrained goals

Sequencing Strategy	Pctg. Constrained Goals Scheduled	Pctg. Telescope Utilization
NNLA	72	21.59
MCF	93	17.20
MCF/NNLA	93	20.54

Table 2: Comparative Performance of NNLA, MCF and MCF/NNLA

is "most temporally constrained first" (MCF). Under this scheme, the sequence is built by repeatedly selecting and inserting the candidate goal that currently has the tightest execution bounds. This strategy requires movement away from simulation-based sequence building, since the temporal constraints associated with selected goals will lead to the creation of availability "holes" over the scheduling horizon. Adopting a sequence insertion heuristic that seeks to minimize dead time can provide some secondary attention to this objective, but effectiveness here depends coincidently on the specific characteristics and distribution over the horizon of the initially placed goals. As is the case with the simulation-based NNLA strategy, one objective is emphasized at the expense of the other. This second MCF sequencing strategy, incidentally, is quite close to the algorithm currently employed in the operational system at STScI.

Both NNLA and MCF manage combinatorics by making specific problem decomposition assumptions and localizing search according to these decomposition perspectives. NNLA assumes an event based decomposition (considering only the immediate future) while MCF assumes that the problem is decomposable by degree of temporal constrainedness. Previous research in constraint-based scheduling[SOM+90] has indicated the leverage of dynamic problem decomposition selective use of local scheduling perspectives. In the case of NNLA and MCF, one aspect of current problem structure that provides a basis for selection at any point during sequence development is the current variance in the number of feasible start times remaining for individual unscheduled goals. If the variance is high, indicating that some remaining goals are much more constrained than others, then MCF can be used to emphasize placement of tightly constrained goals. If the variance is low, indicating similar temporal flexibility for all remaining unscheduled goals, then emphasis can switch to minimizing dead time within current availability "holes" using NNLA.

To test this multi-perspective approach, a set of short-term (i.e. daily) scheduling problems where solved with each base sequencing strategy and the composite strategy just described (referred to as MCF/NNLA). The results are given in Table 2 and confirm our expectations as to the limitations of both NNLA and MCF. We can also see that use of the opportunistic MCF/NNLA strategy produces schedules that more effectively balance the two competing objectives. Further details of the experimental design and the strategies tested may be found in [SP92].

These results should be viewed as demonstrative and we are not advocating MCF/NNLA as a final solution. We can profitably exploit other aspects of the current problem structure and employ other decomposition perspectives. For example, the distribution of goals over the horizon implied by imposed temporal constraints has proved to be a crucial guideline in other scheduling contexts [SOM+90, Sad91], and we are currently investigating the use of previously developed techniques for estimating resource contention [MS87, Mus92]. There are also additional scheduling criteria and preferences (e.g., priorities) in space-based observatory domains that are currently not accounted for.

Conclusions

In this paper, we have considered the solution of a specific class of complex scheduling problems that require a synthesis of resource allocation and goal expansion processes. These problem characteristics motivated the design of the HSTS framework, which we briefly outlined and contrasted with other scheduling and AI planning approaches. To illustrate the adequacy of the framework, we then examined its use in solving the HST short-term scheduling problem. We identified three key ingredients to the development of an effective, practical solution: flexible integration of decision-making at different levels of abstraction, use of domain structure to decompose the planning problem and facilitate incremental solution development/scaling, and opportunistic use of emergent problem structure to effectively balance conflicting scheduling objectives. The HSTS representation, temporal data base, and constraint-posting framework provide direct support for these mechanisms.

References

[Bak74] K.R. Baker. *Introduction to Sequencing and Scheduling.* John Wiley and Sons, New York, 1974.

[BS90] K. R. Baker and G.D. Scudder. Sequencing with earliness and tardiness penalties: a review. *Operations Research*, 38(1):22–36, January-February 1990.

[DFM88] T. Dean, R.J. Firby, and D. Miller. Hierarchical planning involving deadlines, travel time, and resources. *Computational Intelligence*, 4:381–398, 1988.

[FHN72] R.E. Fikes, P.E. Hart, and N.J. Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence*, 3:251–288, 1972.

[Gra81] S.C. Graves. A review of production scheduling. *Operations Research*, 29(4):646–675, July-August 1981.

[Joh90] M.D. Johnston. Spike: Ai scheduling for nasa's hubble space telescope. In *Proceedings of the 6th IEEE Conference on Artificial Intelligence Applications*, pages 184–190, 1990.

[Lan88] A. Lansky. Localized event-based reasoning for multiagent domains. *Computational Intelligence*, 4:319–340, 1988.

[MS87] N. Muscettola and S.F. Smith. A probabilistic framework for resource-constrained multi-agent planning. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, pages 1063–1066. Morgan Kaufmann, 1987.

[MSCD92] N. Muscettola, S.F. Smith, A. Cesta, and D. D'Aloisi. Coordinating space telescope operations in an integrated planning and scheduling architecture. *IEEE Control Systems Magazine*, 12(1), February 1992.

[Mus90] N. Muscettola. Planning the behavior of dynamical systems. Technical Report CMU-RI-TR-90-10, The Robotics Institute, Carnegie Mellon University, 1990.

[Mus92] N. Muscettola. Scheduling by iterative partition of bottleneck conflicts. Technical report, The Robotics Institute, Carnegie Mellon University, 1992.

[PI77] S.S. Panwalker and W. Iskander. A survey of scheduling rules. *Operations Research*, 25:45–61, 1977.

[Sad91] N. Sadeh. *Look-ahead Techniques for Micro-opportunistic Job Shop Scheduling.* PhD thesis, School of Computer Science, Carnegie Mellon University, March 1991.

[SOM+90] S.F. Smith, J.Y. Ow, P.S. Potvin, N. Muscettola, , and D. Matthys. An integrated framework for generating and revising factory schedules. *Journal of the Operational Research Society*, 41(6):539–552, 1990.

[SP92] S.F Smith and D.K. Pathak. Balancing antagonistic time and resource utilization constraints in over-subscribed scheduling problems. In *Proceedings 8th IEEE Conference on AI Applications*, March 1992.

[Ver83] S. Vere. Planning in time: Windows and durations for activities and goals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5, 1983.

[Wal89] M. Waldrop. Will the hubble space telescope compute ? *Science*, 243:1437–1439, March 1989.

[Wil88] D.E. Wilkins. *Practical Planning*, volume 4. Morgan Kaufmann, 1988.