Telescope Loading: A Problem Reduction Approach

John L. Bresina
Recom Technologies
AI Research Branch, Mail Stop: 269-2
NASA Ames Research Center, Moffett Field, CA 94035-1000
Phone: 415.604.3365, *Fax:* 415.604.3594, *E-mail:* bresina@ptolemy.arc.nasa.gov

KEY WORDS AND PHRASES

Automated telescope management, periodic observation campaign, problem reduction, scheduling, telescope loading.

ABSTRACT

This paper presents a problem reduction approach to telescope loading. To study time-varying celestial behavior, astronomers submit periodic observation campaigns which involve a sequence of observations at a given sampling frequency over months or years. The loader's task is to generate an assignment of observation tasks to each night in the time window such that resource demand does not exceed resource capacity and such that the observations usefully contribute to the campaigns' scientific purposes, in a manner that is fair to all participating astronomers.

INTRODUCTION

In order to carry out a scientific campaign involving the study of a time-varying celestial behavior, an astronomer submits requests for observation time on a telescope. Satisfying such a campaign requires periodic observations of the celestial object over an extended interval of time (months or years). The number of scientific campaigns that the community of astronomers would like to pursue overwhelms existing telescopes. Enabling astronomers to effectively pursue their campaigns is an important and difficult problem. This is the problem addressed by telescope loading. Telescope loading involves assigning each observation to a particular night for execution such that underutilization of telescope time is minimized, oversubscription is eliminated fairly, and the astronomers' scientific goals are served.

In our application domain, the telescopes are land-based and fully automatic; a telescope control computer opens the observatory at twilight and collects data through the night without human assistance [4]. We are implementing an overall automated management system [2; 3] to enable participating astronomers to submit observation requests and obtain results from a remotely located telescope, *via* electronic communication networks, without the necessity of human intervention. In addition to the telescope loader described in this paper, the system also includes a night scheduler [10]. Each night, the observations assigned by the loader are given to the scheduler to determine the time each one will be executed.

Simply demonstrating, by construction, the feasibility of a solution to a telescope loading problem is not sufficient — the quality of the solution is an important consideration. A "good" loading assignment uses all available telescope time on observations that usefully contribute to the submitted campaign goals, in a manner that is fair to all participating astronomers. To evaluate a particular loading assignment, the expected quality of the schedule for each of the nights must also be taken into account. The loader uses the night scheduler and its evaluation function to determine expected schedule quality for a night's candidate loading assignment.

Telescope time is typically oversubscribed and it is usually impossible to fully satisfy every submitted observation campaign. Hence, in this application domain, the problem-solving method must be able to relax the constraints of the initial problem until it is satisfiable. Since there are many alternative ways of relaxing the problem, when relaxation is necessary, the complexity of loading is increased. The ideal objective is to find an optimal solution to a minimal relaxation of the initial problem; however, this ideal is seldom realizable. Thus, problems in this domain cannot be effectively solved (in general) using existing optimization methods from the fields of operations research or artificial intelligence.

In this paper, we describe our proposed solution method for the telescope loading problem. In order to reduce the search complexity, our solution method employs a problem reduction approach. Problem reduction is a type of "divide-and-conquer" technique that recursively decomposes a problem into conjunctions of subproblems until the subproblems are simple enough to easily solve, then all the subproblem solutions are composed to form the solution to the original problem.

LOADING PROBLEM

A telescope loading problem consists of a time window, a set of campaign goals, and a specification of resource capacity for each night in the time window. Our campaign specification language is based on a proposal by Louis Boyd, Director of Fairborn Observatory [1]. In this paper, we only describe aspects of the specification language needed

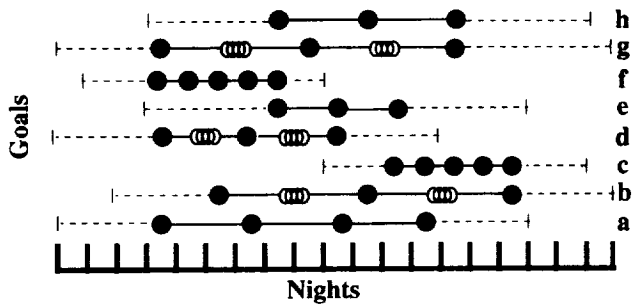

Figure 1: Schematic of an example loading problem.

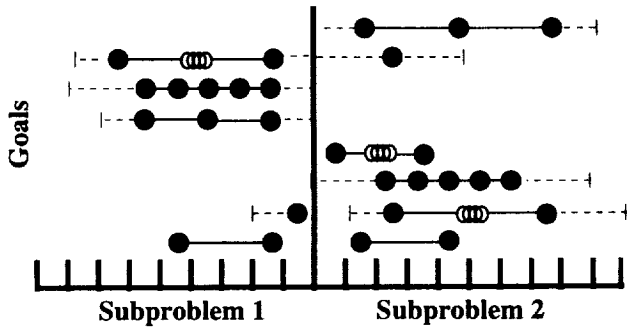

Figure 2: Schematic of an example problem reduction.

to explain the solution method.

Each campaign consists of one or more periodic observation goals. Briefly, a campaign goal specifies a number of repeated observations to execute within a given time window with a given time gap between executions. The number of repeated observations is specified with an ideal execution count and a minimum execution count; if it is not possible to obtain the minimum, then the campaign goal should not be attempted. An astronomer specifies the ideal gap (in days) between executions either with a fixed gap length or with a gap probability distribution (used, for instance, to reduce aliasing in the data or to determine the period of a recently discovered variable star). For example, a fixed gap length of one indicates that the observation should be executed every other night. An example of a gap distribution specification is the uniform probability distribution, $U(0, 2)$, which indicates that gaps should be randomly selected with a uniform probability from the set {0 days, 1 day, 2 days}.

Figure 1 illustrates a small example loading problem covering 19 nights and containing eight hypothetical campaign goals (**a**-**h**). Each goal is pictured as a sequence of observations which can slide within an window of nights, indicated by the dashed lines. A probabilistic (variable) gap between observations is indicated with a mechanical spring, and a fixed gap is indicated with a solid line. The loader's objective is to place each sequence within its window such that no night is overloaded. It may not be possible to achieve this objective with respect to the ideal gaps and the ideal number of observations specified by each astronomer. Some of the observations may not be done, and some of the gaps may be longer or shorter than ideal. The transformations that the loader can apply to the desired observation sequences are: (*i*) shrinking the goal's time window, (*ii*) clipping some observations from a sequence, and (*iii*) stretching or shrinking the gaps in a sequence. The first transformation restricts the possible nights to which the observations may be assigned, and the later two are problem relaxation transformations. If the time window is reduced too much, or if the execution count is reduced too much, or if the gap length is increased too much, then the scientific purpose of the campaign goal will not be served and the data collection activity will have wasted valuable telescope time. For each campaign goal, the astronomer specifies the limits of these transformations and specifies the relative desirability of the types of relaxations.

The resource capacity profile is a specification of the projected number of hours of observation time available for each night in the time window at a particular telescope. The amount of twilight time depends on the time of the year, and how much of that observation time can be expected to be available varies due to seasonal weather patterns. After each night, the loader tries to reassign the unexecuted tasks to future nights; if the initial loading assignment fills up all available observation time, then oversubscription could continue to grow worse over time. Therefore, in addition to accounting for changing observation availability, we need to leave some margin of capacity for future revision to the initial loading assignment.

LOADING PROBLEM REDUCTION

In this section, we describe our problem reduction approach to solving telescope loading problems. Our problem reduction approach for loading is a three stage process. First, a temporal decomposition process is applied to partition the problem's time window. Second, a campaign decomposition process is applied to the campaign goals one at a time; this process splits the campaign goals among the subproblems such that the average load is balanced. Third, a relaxation process is applied to each subproblem in an attempt to reduce oversubscription of resource capacity. Problem reduction is then recursively applied until all observation tasks are assigned to specific nights. Figure 2 illustrates a single application of problem reduction to the problem in Figure 1.

Our problem reduction technique implements a refinement process that incrementally restricts the set of possible nights that each campaign goal can be assigned for execution, incrementally balances the load, and incrementally relaxes the initial problem.

Each decomposition modifies campaign goals and shifts the load within the local context of a single subproblem. For problem solving efficiency, our reduction process ensures that the subproblems created are independently solvable; *i.e.*, that all subproblem solutions can be composed (*via* concatenation) into a valid solution to the original problem. We next discuss each reduction stage.

Temporal Decomposition

The objective of the temporal decomposition process is to partition the given problem's time window based on an analysis of resource contention. For each subinterval, a subproblem is created to cover that time window. For simplification, we restrict the temporal decomposition space by partitioning into only two subintervals. An heuristic evaluation function is used to select the best two-element partition of the problem's time window.

Resource contention is computed by subtracting resource capacity from resource demand; a contention greater than zero means the telescope is oversubscribed. The resource demand for a particular night is the observation time needed to satisfy the requests. The resource demand profile depends on how the requests will be assigned during the loading process. The expected demand for a given night is the summation, over all campaign goals, of the product of the goal's demand and the probability that the goal will be assigned to that night; it is assumed that all the possible start dates for a campaign goal are equiprobable (*cf.* Muscettola and Smith [7]).

Campaign Decomposition

The campaign decomposition stage involves placing each campaign goal of the parent problem into a subproblem or splitting it between the subproblems. The primary objectives are to restrict loading assignment alternatives and help balance the load.

If a campaign goal's time window is a subset of a subproblem's time window, then that goal can only be incorporated into that one subproblem's campaign set. All the goals of this type are processed first. In our example, only goals **c** and **f** are of this type. The remaining goals are processed one at a time based on which campaign goal can make the biggest impact on balancing the load between the two subproblems. The system selects the goal with the highest "load shift potential", which is the maximum amount of its demand that can be shifted from the high contention subproblem (*i.e.*, the one with a higher average contention) to the low contention subproblem.

Goals of this type can either be split across both subproblems, or their time windows can be shrunk to fit into one of the subproblems; the later was done to goal **h** in our example. When a goal is split, two campaign subgoals are created. In order for the subproblems to be independently solvable, the possible assignments for the two subgoals must be consistent with the original goal's constraints.

The system uses a "greedy" approach to load balancing — when decomposing the selected campaign goal, the system shifts as much demand from the high contention subproblem as can be accommodated in the low contention subproblem (without going past the balance point). After decomposing the selected goal, the average contention of the two subproblems is updated and the load shift potentials of the remaining goals are updated. Then the campaign goal with the current highest load shift potential is selected next. This process repeats until all goals have been decomposed.

Problem Relaxation

After the campaign goals have been split between the subproblems, further modification may be required in order to achieve a zero average contention in each subproblem. If a subproblem's average contention is greater than zero, then the resource demand of one or more of its goals must be reduced by decreasing the execution count. In our example, one observation task was removed from goal **d**.

The determination of how much to alter execution counts and gap lengths is impacted by both hard constraints and soft constraints (*i.e.*, preferences). When decreasing a goal's demand, there are limits (specified by the astronomer) to how much the observation sampling strategy can be modified. When the contention within a subproblem can not be sufficiently reduced without violating a goal's hard constraints, then one or more of the subproblem's goals must be entirely eliminated.

An heuristic evaluation function is employed to provide guidance to the relaxation process at a given subproblem. The evaluation of a candidate campaign set measures the desirability of the relaxation with respect to each astronomer's preferences and relative priorities of the campaign goals. The heuristic evaluation combines the scores of all these local evaluations and selects the campaign set that achieves the most effective and fairest relaxation.

Recursive Application & Termination

Upon completion of the problem relaxation stage, the expected demand is computed for each subproblem and then subtracted from the capacity profile to derive the contention profile. The decomposition process can then be recursively applied to each subproblem.

When a campaign goal's time window becomes too small, candidate night assignments are generated for the goal's observations according to the gap specification and ideal execution count. The problem reduction process terminates when decomposition has terminated for all goals and each observation task has been assigned to a particular night.

From problem to subproblem, each successive modification is a fine-tuning, or specialization, of the preceding modification. The average contention derived from the first temporal decomposition covers the entire time span considered by the loader and is a very abstract characterization of the contention

during that time window. Each successive subproblem produced (*via* recursive decomposition) has an average contention that covers a smaller time span and is a more accurate characterization of the contention. Hence, this combination of temporal decomposition and averaging automatically generates a hierarchy of abstraction levels.

CONCLUDING REMARKS

We have presented an approach to solving the telescope loading problem for periodic observation campaigns. The complexity of this loading problem is reduced by employing a problem reduction approach that reasons at different levels of abstraction and ensures that the subproblems created are independently solvable. The abstraction levels are automatically generated based on properties of the problem instance, namely, the contention profile. A given abstraction level not only depends on previous abstraction levels, but also on the decisions made in previous problem decompositions.

Our approach is a novel application of problem reduction to a domain in which reasoning about metric time is central, solution quality is important, and problem relaxation is necessary. The problem reduction technique implements an incremental refinement process. Each subproblem inherits the loading biases of all ancestor problems and imposes an additional bias on all descendant subproblems; the specific loading assignment produced is a result of the combination of all such biases.

The BAIT system [8] is a related automated telescope management system. The BAIT loader only considers the current night and uses a probabilistic selection technique to determine which of the active tasks to include. The assigned probability to a task is based on the desired (fixed) gap between observations and the time since the last observation.

Our loader addresses a similar problem to that addressed by the SPIKE system [5]. SPIKE solves the loading problem for the Hubble Space Telescope, at a grain size of about a week, employing a constraint satisfaction approach. Our loading approach is related to opportunistic scheduling approaches, especially those of Muscettola [6] and Sadeh [9], although neither of their systems has been applied to a problem domain with periodic requests. Though there are substantial differences between these two scheduling systems, both systems focus on bottlenecks and use variable ordering heuristics based on some type of contention analysis.

One of the primary differences from SPIKE and the opportunistic systems is that our approach incorporates problem relaxation. Another key difference from these three systems is that, in our approach, the reasoning is carried out at a much more abstract level, at least during the first few levels of problem reduction. The contention analyses, the problem solving decisions, and even the tasks assigned are initially quite abstract, in comparison to their approaches. As the problem is recursively decomposed, these aspects become more detailed. Though this research effort is in its early stages and system implementation is not yet completed, we conjecture that the combination of problem reduction and automatic, problem-specific abstraction should yield efficient problem solving and quality solutions.

Acknowledgements

Firstly, I would like to thank the rest of the telescope management project team: Mark Drummond, Keith Swanson, and Will Edgington. I would also like to thank Saul Amarel, William Borucki, Louis Boyd, David Genet, Russell Genet, Gregory Henry, Peter Friedland, Nicola Muscettola, Charles Schmidt, and Donald Smith.

REFERENCES

[1] Boyd, L., Epand, D., Bresina, J., Drummond, M., Swanson, K., Crawford, D., Genet, D., Genet, R., Henry, G., McCook, G., Neely, W., Schmidtke, P., Smith, D., and Trublood, M. 1993. Automatic Telescope Instruction Set 1993. *IAPPP Communications*, 52, T. Oswalt (ed.).

[2] Bresina, J., Drummond, M., Swanson, K., and Edgington, W. (1994). Automated Management and Scheduling of Remote Automatic Telescopes. In *Astronomy for the Earth and Moon*. D.P. Smith (ed.).

[3] Drummond, M., Bresina, J., Swanson, K., and Edgington, W. 1994. The Associate Principal Astronomer Telescope Operations Model. In *Proc. of i-SAIRAS 94*. JPL, Pasadena, CA.

[4] Genet, R.M., and Hayes, D.S. 1989. *Robotic Observatories: A Handbook of Remote-Access Personal-Computer Astronomy*. AutoScope Corp., Ft. Collins, CO.

[5] Johnston, M.D. 1990. SPIKE: AI Scheduling for NASA's Hubble Space Telescope. In *Proc. of the 6th Conf. on AI Applications*. IEEE Comp. Society Press.

[6] Muscettola, N. 1992. *Scheduling by Iterative Partition of Bottleneck Conflicts*. Tech. rpt. RI-TR-92-05, The Robotics Institute, CMU.

[7] Muscettola, N. and Smith, S.F. 1987. A Probabilistic Framework for Resource-Constrained Multi-Agent Planning. In *Proc. of the 10th IJCAI*. Milan, Italy.

[8] Richmond, M., Treffers, R., & Filippenko, A. 1993. The Berkeley Automatic Imaging Telescope. *Publications of the Astronomical Society of the Pacific*. No. 105 (October).

[9] Sadeh, N. 1991. *Look-Ahead Techniques for Micro-Opportunistic Job Shop Scheduling*. Tech. rpt. CS-91-102, School of Comp. Sci., CMU.

[10] Swanson, K., Bresina, J. and Drummond, M. 1994. Robust Telescope Scheduling. In *Proc. of i-SAIRAS 94*. JPL, Pasadena, CA.