N93-32130

Scheduling with Partial Orders and a Causal Model

Mark Boddy Jim Carciofini George D. Hadden
{boddy | carciofi | hadden}@src.honeywell.com
Honeywell Systems & Research Center, MN65-2100
3660 Technology Drive
Minneapolis, MN 55418

Abstract

In an ongoing project at Honeywell SRC, we are constructing a prototype scheduling system for a NASA domain using the "Time Map Manager" (TMM). TMM representations are flexible enough to permit the representation of precedence constraints, metric constraints between activities, and constraints relative to a variety of references (e.g., Mission Elapsed Time vs. Mission Day). The TMM also supports a simple form of causal reasoning (projection), dynamic database updates, and monitoring specified database properties as changes occur over time.

The greatest apparent advantage to using the TMM is the flexibility added to the scheduling process: schedules are constructed by a process of "iterative refinement," in which scheduling decisions correspond to constraining an activity either with respect to another activity or with respect to some timeline. The schedule becomes more detailed as activities and constraints are added. Undoing a scheduling decision means removing a constraint, not removing an activity from a specified place on the timeline. For example, we can move an activity around on the timeline by deleting constraints and adding new ones, and other activities constrained with respect to the one we move will move as well.

1 Introduction

We are interested in the solution of large, complex scheduling problems. Examples of the kinds of domains we are interested in include several NASA scheduling problems (e.g. Spacelab, Space Station operations, Shuttle ground processing), and the *Transportation Planning* problem being addressed by the joint DARPA/Air Force Planning Initiative.

A "solution" as we use the term is not simply an implementation of an algorithm for solving a particular constraint satisfaction or constrained optimization problem. For many domains, constructing schedules is an extended, iterated process that may involve negotiation among competing agents or organizations, scheduling choices made for reasons not easily implementable in an automatic scheduler, and last-minute changes when events do not go as expected. In such an enviroment, the process by which a schedule is constructed must be considered in any attempt to provide a useful scheduler for a given domain.

Even the more limited problem of generating a single schedule is becoming increasingly complex. Simple models solvable by straightforward application of standard operations research techniques such as linear programming are less and less relevant to current scheduling problems. For example, many NASA scheduling domains involve large problem instances (hundreds to thousands of activities and constraints), context-dependent activity effects (including context-dependent transitions or setup times as a special case), complex resource structures (e.g., a power bus that is divided into sub-busses), and user preferences on where activities appear in the final schedule (e.g., "as late as possible"). To provide an effective solution, a scheduling system must be expressive enough to represent or reflect these domain complexities as well as supporting the process by which a schedule is constructed.

In an ongoing project at Honeywell SRC, we are implementing a prototype scheduling system for a NASA domain using the "Time Map Manager" (TMM). TMM representations permit the expression of precedence constraints, metric constraints between activities, and constraints relative to a variety of references (e.g. Mission Elapsed Time vs. Mission Day). The TMM also supports causal reasoning (projection and persistence), dynamic database updates, and monitoring certain database properties as changes occur over time.

The greatest apparent advantage to using the TMM is the flexibility added to the scheduling process: schedules are constructed by a process of "iterative refinement," in which scheduling decisions correspond to constraining an activity either with respect to another activity or with respect to some timeline. The schedule becomes more detailed as activities and constraints are added. Undoing a scheduling decision means removing a constraint, not removing an activity from a specified place on the timeline. For example, we can move an activity around on the timeline by deleting constraints and adding new ones, and other activities constrained with respect to the one we move will move as well.

In the rest of this paper, we provide a brief introduction to the TMM, describe the application of the TMM to scheduling, and describe some related work.

2 TMM Overview

As part of the DARPA/RL Planning Initiative, Honeywell has developed a new implementation of Dean's Time Map Manager (TMM) [5], involving improvements in robustness, user interface, and documentation, in addition to a number of extensions in functionality. The TMM provides users and application programs (e.g., planners and schedulers) with the following functionality:

- Metric and ordering constraints between any two points.
- Causal reasoning.

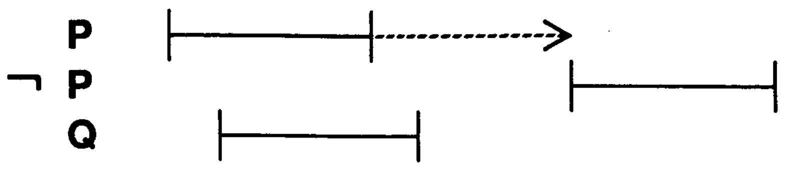

Figure 1: A simple temporal database

- Database monitors for temporal conditions and protections.
- Optimizations for large temporal databases.

The structure and capabilities of the TMM are described in more detail below.

2.1 Temporal Relations

The TMM lets users assert *constraints* between pairs of *time points*, resulting in a partial ordering among the points. TMM supports queries regarding necessary and possible temporal relations among the time points. The truth of facts over intervals of time is represented by *tokens*, which may include properties of *persistence* beyond their observed endpoints. In the current implementation, tokens may persist both forward and backward in time. The truth of a proposition over an interval is determined based on the ordering of token endpoints and the token's persistence properties. For example, Figure 1 is a simple temporal database, involving three tokens of three different types. In this example, P is true over the interval bounded by the vertical lines, and persists into the future. (not P) becomes true at a later time, and *clips* the forward persistence of P. The statement "P and Q" is true for an interval defined by the overlap of the tokens labelled P and Q.[1]

2.2 Causal Reasoning

The TMM supports reasoning about the changing state of the world as activities occur using three forms of inference:

- The persistence assumption. As described above, users of the TMM specify that certain facts are believed to be true over specific intervals of time. In addition, they can specify that those facts can be assumed to remain true until something occurs to make them false.
- Projection. This is inference of the form: given an event E and a set of preconditions $P_1, P_2, \ldots P_k$, and a result R, whenever the preconditions are believed to be true for the entire event E, R is believed to become true immediately following E.

[1]We are in the process of developing a formal semantics for the TMM. A draft version is available by request.

- Overlap chaining. Given a set of preconditions $P_1, P_2, \ldots P_k$, and a result R, R is believed to be true for any interval for which all of the preconditions are true.

All of these forms of inference are handled completely automatically: the user specifies which facts are persistent and asserts a set of projection and overlap rules, and the requisite inference is performed by the system.

2.3 Nonmonotonic Reasoning and Database Monitors

TMM supports two basic kinds of nonmonotonic reasoning:

- Possibly true temporal relations between time points (which may be invalidated by additional constraints), and
- Assumed truth of a temporal proposition over an interval based on a time token's persistence (which may be invalidated by the addition of a contradictory token, which *clips* the proposition during that interval).

In addition, the database itself is "nonmonotonic", in the sense that information can be deleted, and the inference performed by the system thus far will be checked to ensure that it continues to be supported by the current state of the database. [2]

The existence of specified database properties as changes are made over time can be tracked through the use of *monitors*. The existing types of TMM database monitors are *temporal conditions* and *protections*. Temporal conditions monitor whether specified relations among points can be derived from the current state of the database, maintaining this information as the database changes. Protections do the same thing for the truth of some fact over an interval. Between them, these two mechanisms provide support for monitoring the continued validity of previous inference, or triggering demons based on complex properties of the temporal database

2.4 Efficiency

Current and planned TMM optimizations for handling large databases include the use of a global reference point where appropriate (rather than forcing its use as some systems do), limiting search to that necessary to prove or disprove a query, caching search results for later use, graph decomposition, temporal indexing, lazy monitor evaluation, and algorithms that are designed to search only those parts of the database that may result in useful answers.

3 Scheduling Using the TMM

The assumptions underlying our scheduling work are as follows:

[2]This capability (*temporal reason maintenance*) is described in detail elsewhere [5].

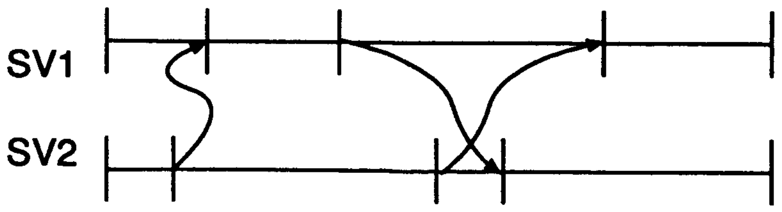

Figure 2: Time-line scheduling

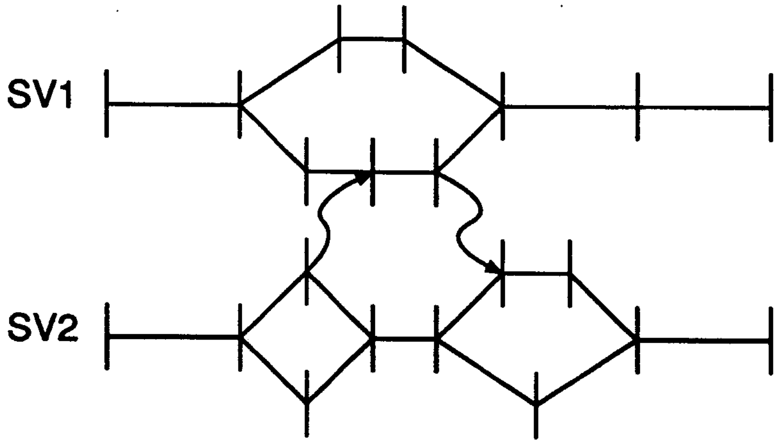

Figure 3: Constraint-posting scheduling and the resulting partial order

1. Explicitly modelling the constraints resulting from specific scheduling decisions makes the schedule easier to construct and modify.
2. Representing only those relationships required by the current set of constraints (the decisions made so far) provides a more useful picture of the current state of the scheduling effort.

The main consequence of this approach is that the scheduler does not manipulate totally-ordered timelines of activities and resource utilization. Instead, the evolving schedule consists of a partially ordered set of activities, becoming increasing ordered as additional constraints are added (or less so, as those decisions are rescinded).

Timeline schedules can be represented using linear sequences of tokens, one sequence for each resource. Figure 2 depicts a simple timeline schedule Arrows between the sequences represent constraints on parts of the two sequences that must obey the indicated ordering relationship. In contrast, schedules constructed by accumulating constraints have a structure like that in Figure 3. Here, the current set of constraints is insufficient to force a totally-ordered sequence of activities. Although providing increased flexibility (through delaying commitment), the explicit representation of partially-ordered activities in the time map makes reasoning about resource usage and other state changes more complicated. It is no longer possible to construct a single time-line representing (e.g.) changing resource availability over time. Instead, the system computes *bounds* on the system's behavior.

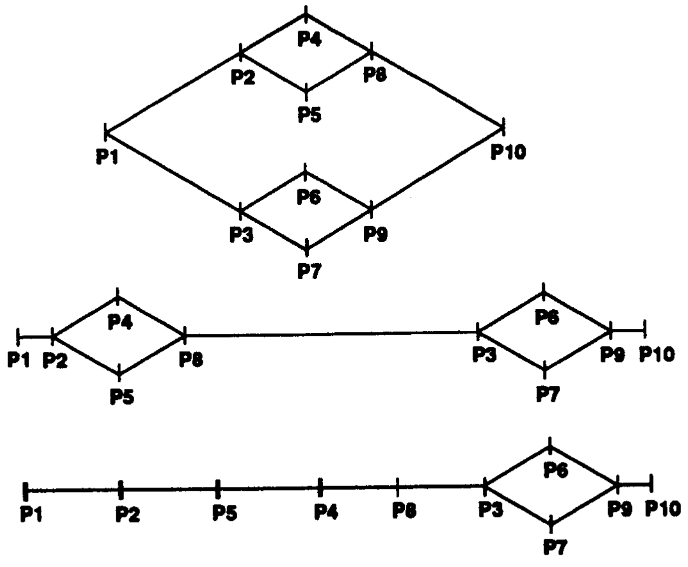

Figure 4: Gradual hardening of a partial order

Causal reasoning and resource profiles both depend on precise orderings of facts and activities in time, that is, on what propositions are true and what activities occur when. For a partial order, we can determine what facts might *possibly* or *necessarily* hold at a point, in some or all of the total orders consistent with the given partial order. With even a very simple causal model, this is an NP-complete problem [4]. The solution we have implemented (first presented in [3]) is to approximate the necessary quantification, implementing *strong* and *weak* reasoning as approximations for what is possibly or necessarily true, given the current partial order. [3]

Figure 4 depicts the process by which a partially ordered schedule is gradually refined into an executable, totally ordered schedule. In our approach to constructing the final schedule, this is one of the ways in which a partial or incomplete schedule is modified, the other being the addition of new activities.

The TMM's strong and weak reasoning provides a partial solution to the problem of reasoning about what will happen. For certain classes of inference, in particular problems involving resource capacity or the aggregate duration of mutually exclusive activities, strong and weak (even exact "necessary" and "possible") reasoning occasionally provides insufficient information. For these cases, there are two possible approaches: simulation (sampling) of totally-ordered sequences, or some kind of static graph analysis to determine better bounds on the system's behavior. The end result in either case is a measure of how likely it is that further constraints on the partial order will cause problems, requiring the scheduler to backtrack to earlier choices.

Despite the approximate nature of this reasoning, we are still ahead of the game: where the least-commitment approach to scheduling can at least provide

[3]See [5], or [13] for details.

approximate answers in support of scheduling decisions (e.g. what order activities should occur in), timeline schedulers make the same decisions arbitrarily—putting an activity on the timeline is a stronger commitment than constraining it to occur (say) between two other activities, or within a given time window.

4 Related Work

The idea that schedules should be constructed "from the side," looking at part or all of the schedule history rather that just sweeping forward or backward in time, has been implemented in several scheduling systems, e.g. [7, 18, 1, 14]. Typically, these systems also support an iterative process of schedule refinement or repair. Recent work on COMPASS provides a protocol for allowing different agents to modify the same schedule, wherein commitments made by one agent cannot be affected by the actions of any other. Research in constraint-based scheduling [8, 15, 12] has demonstrated the advantages of considering the structure of problem constraints over time and using this structure to dynamically focus decision-making on the most critical decisions. However, these systems have historically had a weak model of the interaction of activities and the evolving state of the domain.

Research in generative planning has focused on the construction of activity networks that bring about desired goal states, given basic representations of the effects of actions in the world. Classical domain modeling assumptions [6] [17] make it difficult to reason about the duration of activities, continuously varying quantities, and resource consumption. The consequence of these limitations is that automatic planners have not had any great success in applications to significant planning and scheduling problems [2, 16].

In addition to the TMM, several other temporal database systems have been implemented with an ability to reason about time "from the side" [11, 9, 10]. To date, the TMM's combination of expressive flexibility, precise semantics, support for database operations, and extensive optimization set it apart from other temporal reasoning systems.

5 Summary

Effective support for current scheduling domains requires a focus on the scheduling process as well as the scheduling problem. Using the TMM, we are in the process of constructing a novel form of scheduler that constructs schedules through the accumulation of constraints on the relations between activities, and between activities and various reference points (e.g. calendar time, mission elapsed time, etc.).

The TMM's support for flexible temporal relations, dynamic updates in large databases, and causal reasoning provide an effective base for building schedulers for complex problems. The TMM is freely available and written in Common Lisp. To obtain a copy of the software or a more detailed description of the system's design and capabilities, contact the authors at the address given on the first page.

References

[1] Biefeld, E. and Cooper, L., Scheduling with Chronology-Directed Search, *Proc. AIAA Computers in Aerospace VII, Monterey, California*, 1989, 1078–1087.

[2] Dean, T., Firby, R.J., and Miller, D., Hierarchical Planning Involving Deadlines, Travel Time, and Resources, *Computational Intelligence*, 4 (1988) 381–398.

[3] Dean, Thomas and Boddy, Mark, Incremental Causal Reasoning, *Proceedings AAAI-87 Sixth National Conference on Artificial Intelligence*, 1987, 196–201.

[4] Dean, Thomas and Boddy, Mark, Reasoning about Partially Ordered Events, *Artificial Intelligence*, 36(3) (1988) 375–399.

[5] Dean, Thomas and McDermott, Drew V., Temporal Data Base Management, *Artificial Intelligence*, 32 (1987) 1–55.

[6] Fikes, R.E., Hart, P.E., and Nilsson, N.J., Learning and Executing Generalised Robot Plans, *Artificial Intelligence*, 3 (1972) 251–288.

[7] Fox, B. R., Non-Chronological Scheduling, *Proc. AI, Simulation, and Planning in High Autonomy Systems, University of Arizona*, IEEE Computer Society Press, 1990.

[8] Fox, M.S. and Smith, S.F., ISIS: A Knowledge-Based System for Factory Scheduling, *Expert Systems*, 1(1) (1984) 25–49.

[9] Ghallab, M. and Alaoui, A.M., Managing Efficiently Temporal Relations through Indexed Spanning Trees, *Proceedings IJCAI 11, Detroit, Michigan*, IJCAI, 1989, 1297–1303.

[10] Koomen, J.A.G.M., *The Timelogic Temporal Reasoning System*, Technical Report 231 (revised), University of Rochester Computer Science Department, October 1988.

[11] Materne, S. and Hertzberg, J., *MTMM: Correcting and Extending Time Map Management*, Technical Report 511, GMD, February 1991.

[12] Sadeh, N. and Fox, M.S., Variable and Value Ordering Heuristics for Activity-based Job-shop Scheduling, *Proceedings of the Fourth International Conference on Expert Systems in Production and Operations Management, Hilton Head Island, S.C.*, 1990.

[13] Schrag, Robert, Boddy, Mark, and Carciofini, Jim, Handling Disjunction in Practical Temporal Reasoning, *KR-92*, 1992.

[14] Smith, S.F., Ow, P.S., LePape, C., McLaren, B. S., and Muscettola, N., Integrating Multiple Scheduling Perspectives to Generate Detailed Production Plans, *Proceedings of the SME Conference on AI in Manufacturing, Long Beach, CA*, 1986.

[15] Smith, S.F., Ow, P.S., Potvin, J.Y., Muscettola, N., , and Matthys, D., An Integrated Framework for Generating and Revising Factory Schedules, *Journal of the Operational Research Society*, 41(6) (1990) 539–552.

[16] Vere, S., Planning in Time: Windows and Durations for Activities and Goals, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5 (1983).

[17] Wilkins, D.E., *Practical Planning*, volume 4, (Morgan Kaufmann, 1988).

[18] Zweben, M., Deale, M., and Gargan, R., Anytime Rescheduling, *Proceedings of the DARPA Workshop on Innovative Approaches to Planning, Scheduling, and Control, San Diego*, DARPA, 1990.