Temporal Reasoning for Planning and Scheduling

Mark Boddy
Honeywell Systems and Research Center, MN65-2100
3660 Technology Drive
Minneapolis, MN 55418
boddy@src.honeywell.com

Abstract

We briefly describe the current implementation of the *Time Map Manager* (TMM), followed by a description of the system's suitability for and application to planning and scheduling tasks.

1 Introduction

As part of the DARPA/Rome Lab Planning Initiative, Honeywell's Systems and Research Center has developed a new implementation of Dean's Time Map Manager (TMM), involving improvements in robustness, efficiency, user interface, and documentation, in addition to a number of extensions in functionality. The TMM development contract is a 3 year, $1M effort, on which work commenced in August 1990. Honeywell's TMM software and User Manual were initially released in August 1992, with several incremental releases since.

The TMM is a unique tool for representing and reasoning about temporal information. The TMM supports the representation of both ordering and metric constraints, reasoning about state changes, and dynamic database updates, all in a tool that has been engineered specifically to support large-scale temporal reasoning problems. Using the TMM, we are now developing tools that add a new dimension of flexibility and power to planning and scheduling applications.

In this paper, we provide a brief overview of TMM capabilities, a description of the system's employment as a basis for building planning and scheduling tools, and a description of selected problems to which these tools have been applied.

2 TMM Overview

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.
- 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]

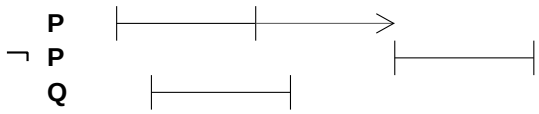

Figure 1: A simple temporal database

2.2 Causal Reasoning

The TMM currently supports reasoning about the changing state of the world as activities occur using two 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.

These forms of inference are handled completely automatically: the user specifies which facts are persistent and asserts a set of projection rules, and the requisite inference is performed by the system. An additional form of causal reasoning under investigation is *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. Providing an efficient implementation of this kind of inference is proving difficult, for reasons that have been discussed elsewhere [5].

2.3 Nonmonotonic Reasoning and Database Monitors

TMM supports two basic kinds of nonmonotonic reasoning:

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

- 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. The representation of temporal information as a graph makes it easy to construct application-specific algorithms, for example providing a graph-based algorithm for efficiently determining interval durations.

3 Scheduling Using the TMM

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. 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.

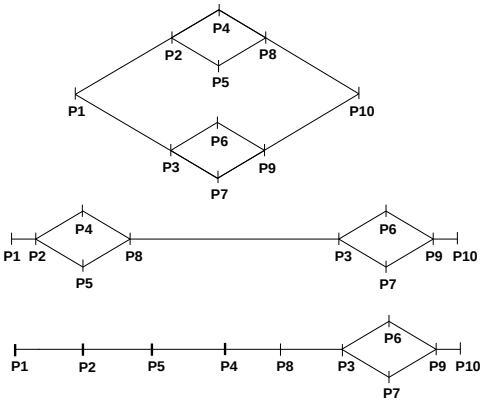

Figure 2: Gradual hardening of a partial order

The assumptions underlying our scheduling work are as follows:

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).

The greatest advantage we have found 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. Figure 2 depicts the process by which a partially ordered schedule is gradually refined into an executable, totally ordered schedule.

Timeline schedules can be represented using linear sequences of tokens, one sequence for each resource. Figure 3 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 4. Here, the current set of constraints is insufficient to force a totally-ordered sequence of activities.

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

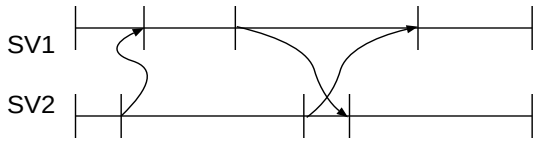

Figure 3: Time-line scheduling

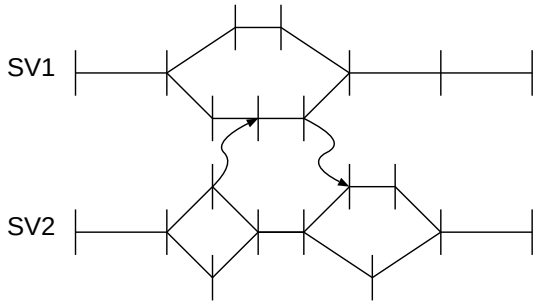

Figure 4: Constraint-posting scheduling and the resulting partial order

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. 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 given 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]

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 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.

[3] See [5], or [14] for details.

4 Planning Using the TMM

To date, we have expended less effort on the application of the TMM to planning tasks. Our preliminary investigations of the application of the TMM to planning problems have had mixed results. It is increasingly clear that the metric temporal reasoning and incremental updates supported by the system are going to be useful. TMM support for efficient reasoning about partial orders was initially intended to support nonlinear planning.

We have been able to show that for many problems, schedulers can be built as a relatively thin "shell" around the TMM. The correct level of integration between a planner and a temporal reasoning system such as the TMM is an open question. For example, in doing projection, the TMM builds data dependencies that look much like causal links in an SNLP-style planner, but without the bookkeeping required to explore the set of possible links in a systematic manner [9]. Thus, an SNLP-style planner could make use of the lower-level temporal reasoning capabilities of the TMM without making effective employment of the causal reasoning machinery. Backstrom and Nebel provide additional arguments that persistence, projection, and overlap chaining may not be useful forms of inference for current planning techniques [11].

5 Applications

Within the context of the Planning Initiative, the TMM has made its presence felt in several ways. In a series of benchmark studies conducted at the University of Rochester, the TMM was shown to provide a great deal of functionality and expressive power in a tool that scaled realistically to application to problems involving tens of thousands of activities. The TMM has been integrated with a generative planning system (SOCAP) in a Technology Integration Experiment (TIE), and more such integrations are planned. At the recent PI meeting in San Antonio, we demonstrated a successful application of the TMM within the current transportation planning software environment, as a system for checking and applying doctrinal constraints that were previously checked by hand, if at all.

Planning and scheduling tools using the TMM have been developed for a wide variety of domains, including operations planning for a Space Shuttle science module, satellite data analysis and retrieval, and processor and communication scheduling for the Boeing 777 Flight Management System, in addition to the Planning Initiative applications described above. An additional application is under development for printed circuit board manufacturing scheduling. Other application areas under consideration include scheduling for the Shuttle Mission Simulator, the Space Station Engineering Master Schedule, satellite ground processing for the army, real-time database transaction scheduling, generating conditional plans for an automated telescope, and an additional manufacturing domain.

6 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, 1, 15]. 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, 16, 13] 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] 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 much success in applications to significant planning and scheduling problems [2, 18]. Recent work addressing the integration of constraints and classical planning may result in a resolution of some of these issues [10, 12, 17].

7 Summary and Conclusions

We have described the current implementation of the TMM, a system for temporal reasoning originally developed by Tom Dean. The current version includes improvements in semantics, robustness, and expressive power. At Honeywell SRC, we have been using the TMM as a basis for constructing planning and scheduling tools. One question currently under investigation in this effort is what set of functionality is required from the temporal reasoner for building such tools. For scheduling problems, the answer is somewhat surprising: we have constructed a variety of flexible and powerful scheduling tools, where the necessary extensions to the TMM were small in comparison to the size of the TMM itself, consisting entirely of interface support and activity and resource models. Schedulers using the TMM have been applied to problems between 9000 and 10000 constraints constraints, with response times in the 1 to 5 second range.

Our preliminary investigations of the application of the TMM to planning problems have had mixed results. On the one hand, it is increasingly clear that the metric temporal reasoning and incremental updates supported by the system are going to be useful. On the other, it is not immediately apparent that projection and overlap chaining are suitable forms of inference for current planning techniques (see, for example [9, 11].

Both planning and scheduling capabilities are required for complex domains (e.g., Space Station operations scheduling). Our current approach to integrating these capabilities is incrementally extend the capabilities of our existing planning tools to handle state changes, task reduction, conditional effects, and other functionality as needed for a particular domain or class of problems.

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 Generalized 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] McAllester, D. and Rosenblitt, D., Systematic Nonlinear Planning, *Proceedings of the Ninth National Conference on Artificial Intelligence, Anaheim, California*, 1991, 634–639.

[10] Muscettola, N., *HSTS: Integrating Planning and Scheduling*, Technical Report CMU-RI-TR-93-05, The Robotics Institute, Carnegie Mellon University, 1993.

[11] Nebel, Bernhard and Backstrom, Christer, On the Complexity of Temporal Projection and Plan Validation, *Proceedings AAAI-92 Tenth National Conference on Artificial Intelligence*, 1992, 748–753.

[12] Penberthy, J. Scott and Weld, Daniel S., Temporal Planning with Constraints (Preliminary Report), *Working Notes, AAAI Spring Symposium on Classical Planning*, Also available as a AAAI tech rpt, forthcoming, 1993, 112–116.

[13] 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.

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

[15] 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.

[16] 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.

[17] Tate, Austin, Drabble, Brian, and Kirby, Richard, *Spacecraft Command and Control Using AI Planning Techniques*, Technical Report AIAI-TR-109, University of Edinburgh, 1992.

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