Toward the Development of Robust Scheduling Tools

Stephen F. Smith
The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213 sfs@cs.cmu.edu

Abstract

In this note, I summarize ongoing research in the Intelligent Coordination and Logistics Laboratory (ICLL) at Carnegie Mellon University toward the development of robust scheduling tools. Three inter-related but complementary threads of research are considered: (1) the development of delayed-commitment scheduling algorithms which enable generation of schedules that are less sensitive to executional uncertainty, (2) the development of architectures that support incremental, mixed-initiative scheduling and rescheduling, and (3) the development of (re)configurable scheduling tools which allow rapid construction of high-performance application systems.[1]

Introduction

In analyzing the requirements of practical scheduling systems, the issue of robustness comes up at several levels. First, one can consider the robustness of the solutions that that the system generates. Almost invariably there is some amount of uncertainty in the executing environment (and often quite a lot). To hedge against this uncertainty, it is desirable to produce solutions that retain executional flexibility where possible and leave decision-making options open. A second requirement for robustness concerns the system's ability to respond to changed constraints and assumptions. Despite any attempt to pro-actively account for uncertainty, unexpected events will occur and circumstances generally demand an ability to incrementally revise the current solution. This may be due to the associated cost of implementing solution changes, or to support controlled convergence to acceptable solutions in collaborative problem solving contexts. In either case, the ability to efficiently and flexibly resolve conflicts and exploit improvement opportunities is a crucial requirement. Finally, one can consider the robustness of a scheduling system design across different applications. Though there is commonality in requirements across domains, different scheduling problems invariably present different challenges. There may be different dominating constraints and objectives, different types of uncertainty, peculiar problem structure and so on. High performance in a particular domain depends in large part on an ability to capitalize on domain idiosyncrasies, and the ease/cost of system development thus becomes another important concern.

In the following sections, I describe three inter-related but complementary threads of ongoing research within ICLL that aimed at addressing these issues of robustness in practical scheduling system design. Work on so-called constraint-posting scheduling algorithms, which defer commitment with respect to the exact timing of activities, is discussed first. This is followed by an overview of current work on incremental, mixed-initiative scheduling (and planning) frameworks. Finally, the development of a scheduling ontology for use as a device for configuring domain models (and ultimately full application systems) is mentioned.

Constraint-Posting Scheduling Algorithms

Most typically, a scheduling problem is formulated as one of finding a consistent assignment of start times for each input (or goal) activity. However one can alternatively formulate the problem strictly as a sequencing problem. In this case, the objective is to determine and post precedence constraints between pairs of activities contending for the same resources so as to ensure that all time and capacity constraints are satisfied. Solutions generated in this way generally represent a set of feasible schedules (i.e., the sets of activity start times that remain consistent with posted sequencing constraints), as opposed to a single assignment of start times. As such, they provide a measure of robustness against executional variances.

Our work has developed a family of constraint-posting scheduling procedures, each derived from a basic constraint satisfaction search model called PCP (Precedence Constraint Posting) (Smith and Cheng 1993). Within this model, the "variables" to be assigned are ordering decisions $O(i,j,r)$ (for activities i and j contending for resource r), each with two possible "values": *i before j* or *j before i*.. The PCP search procedure utilizes estimates of sequencing flexibility as a heuristic basis for directing the search. Estimates of sequencing flexibility are used (1) to detect unconditional sequencing decisions and perform early

[1] The research reported in this paper has been sponsored in part by the National Aeronautics and Space Administration under contract NCC 2-976, by the Department of Defense Advanced Research Projects Agency under contracts F30602-95-10018 and F30602-97-20227 and the CMU Robotics Institute.

pruning of the search space and (2) as a basis for variable and value ordering at each step of the search. The effectiveness of PCP as a constraint satisfaction scheduling procedure has been demonstrated under very general representational assumptions (Cheng and Smith 1994).

Use of the basic PCP model as a basis for schedule optimization has also been explored, leading to development of the Multi-PCP procedure for minimizing schedule makespan (Cheng and Smith 1997). Experimental results with Multi-PCP have been quite impressive. Running on classical benchmark job shop scheduling problems from the Operations Research (OR) community, Multi-PCP produced solutions comparable to state-of-the-art OR approximation algorithms. Multi-PCP has also generated the best known solutions to a more idiosyncratic "hoist" scheduling problem, which requires treatment of temporal separation constraints and sequence-dependent resource setups. The reader is referred to (Cheng and Smith 1997) for further details.

A similar constraint-posting approach has also been successfully used as the basis for an experiment scheduler for a robotic chemistry work station (Aarts and Smith 1994). In this case, the ability to accommodate so-called "flexible experiment protocols" (i.e., specification of execution intervals and flexible experiment step durations as opposed to rigid temporal constraints) was shown to yield almost a 100% improvement in overall work station throughput.

One limitation of the basic PCP model, is its fall-back reliance on a back-tracking search model in circumstances where its heuristics fail to produce a feasible solution. In practice, this approach has proved to be computationally prohibitive and most work has instead relied on backtrack-free versions of PCP, which generate relaxed solutions if necessary instead of expanding the search. To provide an alternative to the basic backtrack search model (and reduce strict reliance on the power of PCP's search heuristics), recent work has investigated the development of randomized variants of basic PCP search heuristics and their use within an iterative sampling search model (Oddi and Smith 1997). The key idea here is to vary the level of randomness according to how well informed the heuristic is in a given choice context. Experimental results on complex constraint satisfaction scheduling problems have demonstrated the efficacy of this approach.

Current work focuses on extending PCP-based search procedures and heuristics to operate with more complex resource capacity models, as well as integration with complementary heuristics for resource assignment.

Incremental, Mixed-Initiative Scheduling

In practice, planning and scheduling is rarely (if ever) a one-shot generative task aimed at satisfying pre-specified constraints and objectives. It is an iterative (and ongoing) process, concerned concurrently with (1) building understanding of the evolving problem and execution state, (2) determining what the constraints and objectives are (or should be), and (3) generating and maintaining an acceptable solution. Despite this fact, most current planning and scheduling systems are based rigidly on a "specify and solve" model of user-system interaction, where the user specifies problem inputs and constraints up-front, and the back-end problem solver is then invoked to generate a solution. This model of interaction does not match the requirements of the larger planning and scheduling process. There is no persistence of decisions over time, no ability to control how solutions change in response to changed inputs or constraints, and no ability to converge to and maintain an acceptable solution in an incremental, controlled manner.

A better framework for user-system interaction follows from a view of planning and scheduling as an incremental change process. This is the approach taken in the design of Ozone, a configurable framework for mixed-initiative scheduling and planning under continuing development at CMU (Smith et. al. 1996). Within Ozone, incremental constraint-based problem solving techniques are combined with graphical solution visualization and manipulation techniques to provide a flexible, "spreadsheet-like" planning and scheduling model. The user analyzes solution elements and formulates change directives from aggregate task-oriented perspectives; the system manages the details of implementing change directives in accordance with user goals and expectations.

The need for change arises both in advance planning contexts, when solutions are found to be less than satisfactory, and to reactively respond to problems and opportunities that arise as execution proceeds. In either case, decision-making responsibility can be variably apportioned from user to system. Various graphical metaphors are used to convey essential attributes of a desired user change, including focus (which decisions must change), scope (which decisions should not), relaxable constraints (if necessary) and search biases (for use within constraint satisfying subspaces). An agenda-based controller, descended from the earlier-developed OPIS scheduling system (Smith 1994), is responsible for mapping change directives to appropriate system change procedures. The set of change procedures available are designed to provide complementary (re)optimization, conflict resolution and change localization capabilities.

The Ozone framework has been applied to develop application systems in a range of complex planning and scheduling domains. Most notable is the DITOPS transportation scheduler (Smith and Lassila 1994b, Smith et. al. 1996), which was demonstrated originally in the domain of large-scale strategic deployment and is now being adapted for transition into operation at the US Air Mobility Command as a day-to-day airlift and tanker scheduling tool. Current research focuses on elaboration of the mixed-initiative scheduling framework to support collaboration with other planning agents and on integration with advanced data visualization and exploration tools.

Configurability and Reuse in Scheduling System Design

A second major thrust of research within the Ozone project has been the development of a configurable scheduling system: a generic framework that promotes rapid construction of domain-specific tools for specific scheduling applications through reuse and extension of previously developed components. Like other recent work toward the design of more flexible scheduling systems (Smith and Lassila 1994a, LePape 1994, Pinedo & Weh 1997), Ozone has been developed with strong emphasis on object-oriented design principles, and is organized as an extensible class library. However, while reliance on object programming techniques and practices provides a foundation for achieving configurability in scheduling system design, a class library by itself does not necessarily promote configurability. Without a higher-level, conceptual model for configuring scheduling applications, reuse of system components and class libraries tends to be quite difficult for anyone other than the original system designers.

Within Ozone, this conceptual model for scheduling system design is achieved by first committing to basic architectural constraints, which provides a structure for identifying and decomposing functional components, and then super-imposing a mechanism for mapping application characteristics and requirements to implied functional components. The first step can be seen as a domain specific counterpart to the use of architectural patterns in software engineering (Gamma et. al. 1994). As already mentioned in the previous section, the Ozone system architecture derives from commitment to a constraint-based scheduling model.

The second step integrates these ideas with recent trends in artificial intelligence toward (1) treating knowledge engineering and knowledge acquisition as a modeling activity (Wielinga et. al. 1992), and (2) development and use of ontologies as a basis for knowledge sharing and reuse (Gruber 1993). The Ozone scheduling ontology (Smith and Becker 1997) defines a meta-model of the scheduling domain. It provides a vocabulary for describing those aspects of the scheduling domain that are relevant to construction of an application system, and a set of constraints on how concepts in the ontology fit together to form consistent domain models. Consistency, in this context, relates to the information and knowledge required to insure executability of the model. Generally speaking, the Ozone ontology serves to map user-interpretable descriptions of an application domain to application system functionality.

This linkage is established through the inclusion of *capabilities* as an integral part of the definition of concepts in the ontology. Capabilities designate encapsulated behaviors that are intrinsic to various domain concepts. These behaviors refer to specific software components in the underlying class library. As a result, the concepts in the ontology can be seen as the actual building blocks for composing and customizing constraint-based solution methods. For example, the concept of a "resource" contributes capabilities for querying and managing its available capacity over time, and different resource types (e.g., reusable, consumable) provide specific "implementations". Given a solution method that incorporates these capabilities, the ontology provides a direct basis for its customization to match the resources in any target domain. It promotes rapid configuration of executable systems and allows concentration of modeling effort on those idiosyncratic aspects of the target domain.

Illustrating the potential of this approach, Ozone was recently applied to develop a system for reactive re-planning of areo-medical evacuation missions in a period of just two person months (Lassila et. al. 1996). Current research focuses on extension of the Ozone ontology to encompass scheduling methods, and on development of interactive, model building tools.

Acknowledgements

The work reported in this paper is the result of the collective efforts of many individuals, including Robert Aarts, Marcel Becker, Eric Biefeld, Stephen Chen, Casper Cheng, Ora Lassila, Dirk Lemmermann, Angelo Oddi, Gary Pelton, Mark Shieh and Ben Werle.

References

Aarts, R. and Smith, S.F. 1994. A High Performance Scheduler for an Automated Chemistry Workstation. In *Proceedings 1994 European Conference on Artificial Intelligence*, Amsterdam

Cheng, C., and Smith, S.F. 1997. Applying Constraint Satisfaction Techniques to Job Shop Scheduling. *Annals of Operations Research* 70:327-357.

Cheng, C. and Smith, S.F. 1994. Generating Feasible Schedules under Complex Metric Constraints. In *Proceedings 12th National Conference on Artificial Intelligence*, Seattle, WA.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. 1994. *Design Patterns: Elements of Reusable Object-Oriented Design*. Addison-Wesley.

Gruber, T.R. 1993. Towards Principles for the design of Ontologies used for Knowledge Sharing, Technical Report KSL-93-04, Stanford University, Palo Alto, CA, Aug.

Lassila, O., Becker, M. and Smith, S.F. 1996. An Exploratory Prototype for Aero-Medical Evacuation Planning, CMU Robotics Institute Technical Report CMU-RI-TR-96-02, January.

Le Pape, C. 1994. Implementation of Resource Constraints in ILOG schedule: A Library for the Development of Constraint-based Scheduling *Intelligent Systems Engineering*.

Oddi, A. and Smith, S.F. 1997. Stochastic Procedures for Generating Feasible Schedules. In *Proceedings 14th National Conference on Artificial Intelligence*, Providence RI, AAAI Press.

Pinedo, M. and Weh, B. 1997. On the Design of Object-Oriented Scheduling Systems. *Annals of Operations Research*, 70.

Smith, S.F. 1994. OPIS: A Methodology and Architecture for Reactive Scheduling. In *Intelligent Scheduling* (eds. Zweben and Fox), Morgan Kaufmann.

Smith, S.F. and Cheng, C. 1993. Slack-Based Heuristics for Constraint Satisfaction Scheduling. In *Proceedings 11th National Conference on Artificial Intelligence*, Washington DC.

Smith, S.F., and Lassila, O. 1994a. Configurable Systems for Reactive Production Management. *In Knowledge-Based Reactive Scheduling*, IFIP Transactions B-15, North-Holland, Amsterdam (The Netherlands).

Smith, S.F. and Lassila, O. 1994b. Towards the Development of Flexible Mixed-Initiative Scheduling Tools. In *Proceedings ARPA/Rome Laboratory Planning Initiative Workshop*. Tucson AZ.

Smith, S.F., Lassila, O. and Becker, M. 1996. Configurable Systems for Mixed-Initiative Planning and Scheduling. In *Advanced Planning Technology*. (ed. A. Tate), AAAI Press.

Smith, S.F. and Becker, M. 1997. An Ontology for Constructing Scheduling Systems. In *Proceedings 1997 AAAI Spring Symposium on Ontological Engineering*. AAAI Press Technical Report, forthcoming.

Wielinga, B., Velde, W.V., Schreiber, G., and Akkernamans, H. 1992. The KADS Knowledge Modeling Approach. In *Proceedings 2nd Japanese Knowledge Acquisition for Knowledge-Based Systems Workshop*. Hitachi Advanced Research Laboratory