AI Magazine Volume 11 Number 5 (1990) (© AAAI)

Issues in the Design of AI-Based Schedulers: A Workshop Report

Karl Kempf, Claude Le Pape, Stephen F. Smith, & Barry R. Fox

Abstract

Based on the experience in manufacturing production scheduling problems which the AI community has amassed over the last ten years, a workshop was held to provide a forum for discussion of the issues encountered in the design of AI-based scheduling systems. Several topics were addressed including : the relative virtues of expert system, deep method, and interactive approaches, the balance between predictive and reactive components in a scheduling system, the maintenance of convenient scheduling descriptions, the application of the ideas of chaos theory to scheduling, the state of the art in schedulers which learn, and the practicality and desirability of a set of benchmark scheduling problems. This article expands on these issues, abstracts the papers which were presented, and summarizes the lengthy discussions that took place.

Introduction

Since its first formal business meeting in August of 1988, the American Association for Artificial Intelligence Special Interest Group in Manufacturing (SIGMAN) has held a number of workshops, three of which have been concerned with the application of AI techniques to the problem of manufacturing scheduling. Here we report the results obtained at the workshop held during the International Joint Conference on Artificial Intelligence in Detroit.

From its inception, this workshop was aimed at identifying and discussing the issues involved in the design of AI-based manufacturing production schedulers. This issues-centered perspective was grounded in the large number of reports which have appeared in the literature over the past ten years concerning AI approaches to scheduling. It was felt that enough experience had been gained in this problem domain to warrant a forum to develop a shared overview among those currently active in the area.

Prior to the event, a number of steps were taken to identify the issues. Papers were solicited which focused on issues encountered during the application of "identifiable theory under realistic conditions". The sessions were organized in response to the interest of the participants as reflected in the papers submitted. Papers were selected for presentation which represented the spectrum of positions on the issue of the session. Each of the session chairmen supplied a one page description of his issue which was sent to all participants along with instructions to those selected to be prepared to give a concise five minute statement of their position. Finally, the bound compilation of the submissions was made available to each participant prior to the start of the workshop.

During the event, each of the four sessions opened with the session chairman reiterating the issue. Those selected to present representative positions on the issues were each given 5 minutes to address the workshop. The session chairmen were instructed to be sure to leave at least an hour for discussion of each issue, and to moderate in such a way that everyone in attendance was brought into the discussion. Notes were taken by each of the committee members during the discussions.

After the event, the committee members made their discussion notes available for the preparation of the report on each session. Each report, to be written by the session chairman, was to start with the statement of the issue, include a brief synopsis of the selected papers, and end with the edited highlights of the discussion. The concatenation of these reports forms the body of this article.

Expert versus Deep versus Interactive Schedulers

Barry Fox, Moderator

The Issues

There are those who feel that expert system technology can be used to build manufacturing production schedulers. There are certainly manufacturing engineers available in every factory to serve as domain experts. Others believe that "deeper" methods must be applied. There is an existing Operations Research (OR) literature on mathematically-based "deeper" methods, and a growing AI literature concerning "deeper" methods based on, for example, constraint satisfaction. Still others argue that fully automated schedulers are not as desirable as interactive schedulers. The point here is that the man and the machine bring complementary skills to the scheduling task, and that both are necessary to produce high quality schedules. Prototype systems have been built based on each of these three approaches. While all of the papers in the first session addressed these issues, they were very diverse in subject matter. There was great variety in the problems that they addressed and in the methods that they used.

The Contributions

The first paper, "Application of Simulation and AI Techniques for Daily Production Planning", was authored by Fahid Bayat-Sarmadi, Bernd Hellingrath, Andreas Marx, and Gabriele Schroder. It describes a job shop scheduling system under development at the Fraunhofer Institute in Dortmund. The system operates in two phases. It builds a pool of orders that can be produced in one day of production, then employs a discrete event simulator to develop a detailed schedule for that one day. Heuristic methods are used in both phases to maximize productivity while respecting deadlines. This work is representative of many knowledge-based scheduling systems. In spite of its

limitations, discrete event simulation can be conveniently combined with a rule base which controls the detailed sequence of events. The rules can be easily acquired from the human expert and then synthetically modified when common-sense or analysis reveals that more sophisticated decisions are required. Unfortunately, because the system is simulation-based, these rules tend to be myopic, considering only the state or configuration of the shop at the temporal frontier of the simulation.

The second paper, "Toward the Knowledge-Based Optimization and Knowledge-Based Relaxation of Complexity", was submitted by Jae Kyu Lee, Min Soo Suh, Yong Uk Song, and Yoan Sung Yi. It describes a crude oil delivery scheduling system under development at the Korean Advanced Institute of Science and Technology. The problem is formulated as a 0-1 integer programming model which is then simplified by knowledge-based relaxation. The relaxed problem is then solved by simulating the operation of the refinery and the delivery of crude oil. Heuristic methods are used in both the relaxation and the simulation. This is a hybrid system combining both natural and synthetic knowledge. The heuristics for problem simplification are synthetic, derived from theoretical knowledge of integer programming. The heuristics for routing and production are natural, based upon knowledge acquired from human experts.

Norman Sadeh and Mark Fox of the Computer Science Department at Carnegie Mellon University collaborated on the third paper titled "CORTES: An Exploration into Micro-Opportunistic Job-Shop Scheduling". The method described in the paper is to search the space of partially constructed schedules beginning with an initial schedule in which none of the activities have been scheduled and all resources are available. The research is concerned with identifying and scheduling activities that have serious resource conflicts with other activities. The underlying method is to create a number of schedules by essentially random methods in order to identify bottleneck resources and regions of contention. This is a synthetic method based upon the "deep" knowledge that hard-to-schedule activities should be scheduled first coupled with an apparently brute-force method based upon the "deep" knowledge that there is no simple analytic method that can reveal regions of contention.

The fourth paper was entitled "A User Interface Architecture for Interactive Scheduling of Software Manufacturing Projects" and was presented by Ali Safavi. This paper describes a (software) job shop scheduling system also under development at the Computer Science Department at Carnegie Mellon University. The research is concerned with creating an interactive system for traversing and exploring a search tree of schedules beginning with an initial schedule in which none of the activities have been scheduled and all resources are available. The assumption is that the human can exercise intuition and creativity to speed the search and improve the quality of the final schedule. This is an attempt to apply natural expertise interactively in an essentially synthetic method. In fact, it may be possible to include some of the more useful natural expertise to improve automated search methods.

Perry A. Zalevsky presented the fifth paper which had the title "Knowledge-Based Finite Capacity Production Scheduling". This paper describes a job shop scheduling system under development at Alcoa Laboratories. It discusses the kinds of knowledge that are used to construct a schedule and identifies two problems that diminish the value of emulating human expertise: separate production centers may have contradictory goals that reduce overall productivity, and established operating procedures may negatively affect productivity. The author concluded by stating a goal to improve current scheduling methods rather than simply emulate the human expert and current methods.

Sanjiv Sidhu of Intellection authored the sixth paper which addressed "Avoiding Typical Mistakes While Building Intelligent Scheduling Systems". Unlike the others, this paper describes a number of common design mistakes that lead to the failure of automated scheduling systems. These include inadequate understanding of dominant domain characteristics, improper problem segmentation, unwarranted reliance on shallow expert knowledge or locally greedy strategies, and inappropriate handling of fluctuations and trivialities. The examples given illustrate that regardless of "natural" or "synthetic" methods, "shallow" or "deep" knowledge, "autonomous" or "interactive" methods, there is no substitute for insight and foresight during system design.

The Discussion

Much of the discussion in this session revolved around three issues: the role of domain experts and expert knowledge; the role of the scheduling specialist and "deeper" synthetic methods; and the role of interactive systems which can be used to compensate for the inadequacies of totally automated systems.

Some participants openly questioned the role of the domain expert and expert knowledge in automated scheduling systems. Recent attempts at building knowledge-based schedulers have met with mixed success. In some cases, the knowledge-engineering process has failed because of the tremendous amount of knowledge that must be acquired from the domain expert and mapped into a suitable representation. In other cases, the implementation has failed because the rules acquired from the domain expert did not define a clear model of the scheduling process. In still other cases, it is questionable whether the humans responsible for scheduling were really expert within their domain. The discussion raised many questions about the role of domain experts and expert knowledge in the implementation of automated scheduling systems including:

1) Concerning scheduling constraints: Can we rely upon the domain expert to accurately describe the production environment and the constraints that it imposes? Can we rely upon the domain expert to understand the constraint representation that we build? Will the completed scheduler fail because it fails to represent and resolve constraints that the domain expert failed to enumerate? Will the completed scheduler fail because the constraints enumerated by the domain expert are really context specific?

2) Concerning scheduling methods: Can we rely upon the domain expert to accurately describe the methods that he or she uses to produce their own schedules? Can these methods be translated into data structures and algorithms or rules

and inference engines? Will the completed scheduler fail because it fails to adapt to the great variety of circumstances that are encountered on the shop floor? Will the completed scheduler fail because the methods described by the domain expert are really context specific?

3) Concerning system validation: Can we rely upon the domain expert to critique the schedules produced by the completed system? Can the domain expert critique large scale schedules as well as small? Are the flaws identified by the domain expert superficial or significant? Will the domain expert accept local anomalies in schedules that are globally very good?

4) Concerning system acceptance: Will the completed scheduler fail to be used because it fails to exactly emulate the behavior and performance of the domain expert? Will the domain expert accept representations and methods that are foreign and hard to comprehend? At the other extreme, can we develop enough general scheduling expertise to build our systems without the close cooperation of the domain expert?

The basic conclusion was that it is clearly necessary to involve human experts in the process of designing an AI-based scheduling system, but their knowledge and methods may not be sufficient. Systems that do not emulate human experts adequately may not be readily accepted, but systems that model domain experts too closely may exhibit inadequate performance. Striking a balance is difficult.

Some participants expressed confidence in the knowledge and methods that scheduling specialists bring to the problem. Regardless of the constraints and heuristics that apply, scheduling systems will be based upon one of a small number of possible methods: simulation-based chronological scheduling, incremental non-chronological job-machine assignment, search-based constraint satisfaction, and so on.

Some argued that it is not necessary to simply emulate the methods of the human expert. Human schedulers must adopt methods that are tractable with pencil and paper, while computerized systems can adopt methods that involve detailed time maps of resource availability and extensive search. Scheduling specialists can use their general knowledge of scheduling problems to select an appropriate computerized method and to embed the knowledge acquired from the human expert into that method. However, the methods adopted by scheduling specialists may be so foreign that it can become extremely difficult to acquire appropriate advice from the domain experts.

Since it requires great effort for human schedulers to produce even one schedule, it is hard for the human to develop a feeling for the global effect of their decisions. Instead, their attention is myopic, focusing on individual decisions and their immediate effects. Scheduling specialists can use their general knowledge of scheduling problems to identify and focus on global metrics and develop methods that improve performance measured by such metrics.

The papers and discussion covered a full range of methods including traditional OR optimization, constraint satisfaction, constraint directed search, and simulation. It was primarily concluded from the presentations that scheduling specialists are developing methods and metrics far more sophisticated than the pencil and paper methods employed by domain experts, and the performance of these systems is expected to be far superior to that exhibited by humans.

A secondary conclusion was also considered in the session. Many deployed scheduling systems contain only a small amount of AI. Successful systems can be dominated by other issues such as the user interface, database connections, real-time data collection, and so on. Good data processing and software engineering can be as important to success as AI and scheduling technology. (See AI-Based Schedulers in Manufacturing Practice: Report of a Panel Discussion by Kempf, Russell, Sidhu, and Barrett in this issue.)

Finally, there was open discussion about the role of interactive systems. Given questions about the role of human experts and conclusions about the power of synthetic methods, proposals for interactive methods received mixed reactions. Clearly, interactive systems allow the human manager to pursue goals and to enforce constraints that cannot be (or have not been) given an accurate computational representation or that change rapidly over time. Interactive systems allow the human to build schedules by methods that they naturally use but are hard to represent as algorithms. Interactive systems allow the human to guide search in the directions that they might naturally follow based upon intuition and knowledge. Interactive systems can be used to educate the domain experts about schedules and scheduling methods, and can be used to negotiate with production managers and clients about release dates and due dates. In the extreme, if we cannot build automated systems that produce satisfactory schedules, then there is good reason to involve humans in the scheduling process.

However, there are obvious risks in using interactive schedulers. The human manager may simply enforce local metrics, the scheduling problems may be too large for the human scheduler to comprehend, or the pace of human interaction may waste valuable computing time that could be better spent in intensive search. If we can build automated systems that do produce satisfactory schedules, then there is no reason to involve humans.

It was concluded that interactive scheduling systems can be used to enforce constraints or implement algorithms that may be difficult to incorporate into an automated scheduler. However, there are risks associated with their use and they should only be used when there is clear benefit from human interaction.

Integrating Predictive and Reactive Decision-Making

Stephen F. Smith, Moderator

The Issues

In most manufacturing domains, coordination of factory production implies some combination of predictive, dynamic, and reactive scheduling. Predictive scheduling serves to provide guidance in achieving global coherence in local decision-making. Dynamic scheduling serves as a basis for execution-time decision-making within the confines of the current predictive schedule. Reactive scheduling is concerned with revising predictive schedules as unexpected events force changes. A fundamental issue in designing scheduling systems that integrate these three types of decision-making is effective reconcilia-

tion of manufacturing system performance optimization and scheduling system responsiveness objectives. Alternative approaches adopt different perspectives relative to this tradeoff, placing more or less emphasis on the importance, influence, and nature of each of these scheduling activities. The goal of this session was to explore the implications of different approaches to the overall production management and control problem. To focus the discussion, a number of specific issues about the integration of predictive and reactive decision-making were identified including:

1) What types of scheduling techniques, schedule representations, and organizational structures are required to achieve a balanced integration of predictive and reactive decision-making?

2) What are the advantages and disadvantages of different integration approaches in attending to interacting scheduling objectives like optimization of factory performance, stability of factory operations, and system responsiveness?

3) How should characteristics of the manufacturing environment like process characteristics, demand patterns, and uncertainty influence the level of emphasis placed on optimization, the nature of schedules that are constructed in advance, and the nature of the coupling between predictive and reactive decision-making?

4) How do we measure the utility of a given approach to integrating production management and control?

The session speakers focused their remarks primarily on the first and second of these issues, perhaps the most concrete from the standpoint of scheduling system design, and presented a variety of positions relative to the realization of an integrated scheduling framework. In each case, these positions were related to specific scheduling mechanisms. The remaining issues were considered in the subsequent discussion period.

The Contributions

Anne Collinot and Claude Le Pape of Stanford University, in their paper titled "Testing and Comparing Reactive Scheduling Strategies", argued that the processes of predictive schedule generation and reactive schedule revision are not fundamentally separable. The decisions made in each case must be based on common knowledge like preference constraints, performance criteria, and scheduling heuristics, if they are to be compatible. Reactive scheduling processes are useful in predictive scheduling contexts, allowing the incremental generation of schedules without worrying about all problem constraints at each step. Detected inconsistencies can be "reactively" resolved when they arise. Moreover, predictive scheduling processes are useful in reactive scheduling contexts where regenerating a schedule from scratch can be viewed as a worst case reaction. Given this commonality, an important issue is integration of predictive and reactive scheduling components within the same scheduling system. One difficulty in this regard is providing a schedule maintenance subsystem that effectively supports both types of decision-making. In particular, experimental results were presented which demonstrated that constraint propagation requirements are different in predictive and reactive contexts. A blackboard-style scheduling system architecture that makes use of a "programmable" constraint propagation system was presented as a framework for solution.

Y. Huang, L. Kanal, and S. Tripathi of the University of Maryland, in a paper titled "Dynamic Scheduling: Problem Definition, Analysis and Solution", adopted the opposing perspective that global predictive scheduling can be separated from operational decision-making and focused specifically on the problem of efficient execution-time decision-making in the face of machine failure. The issue at hand, stated another way, is how to maintain a good schedule without forcing the factory to wait for schedules to be revised. They described a dynamic scheduling heuristic that, in the case of a single machine problem, is capable of "continuously" maintaining an optimal schedule in terms of minimizing the maximal tardiness as the length of a machine breakdown increases. Such an approach can be contrasted with a reactive scheduling approach, wherein either time would be wasted while the schedule is revised once an indication has been received that the machine is again operational, or execution proceeds according to the previous possibly outdated schedule while revision takes place.

Patrick Prosser of the University of Strathclyde, discussing his paper titled "Reactive Scheduling", based his remarks on the general observation that optimization is an ill-conceived objective for scheduling. First, optimality is hard, if not impossible, to define in a realistic scheduling environment. Second, the computation of optimal schedules is a futile enterprise, given the unpredictability of factory operations. Thus, it is preferable to produce and maintain satisfactory schedules over time. He argued that this implies a view of scheduling as a search for one and only one solution satisficing some goals, in which case scheduling systems can be built without making any distinction between predictive and reactive scheduling since it is one and the same process. He then described such a scheduling system organized as a hierarchy of scheduling agents. Agents at each level have well defined responsibilities, at the lowest level managing the schedule of a single resource, at the next level managing the distribution of jobs among sets of resources, and so on. Constraint propagation and truth maintenance techniques are employed to maintain a consistent global schedule. When a conflict occurs, the lowest priority agent among those in conflict uses dependency-directed backtracking to revise its solution. If it cannot find a new solution, the conflict rises to the next level in the hierarchy. The attractiveness of the approach is that the same dependency-directed backtracking search can be used in both predictive and reactive scheduling contexts.

The Discussion

The apparent rejection of any attempt to optimize the performance of the manufacturing system in Prosser's approach gave rise to considerable initial discussion. To be sure, optimality in an analytic sense is not a useful concept. But if there is no concern about how the factory performs then why bother scheduling at all (except in the case of an totally automated factory)? In fact, metrics for gauging the desirability of different schedules do exist and all schedulers make some attempt to exploit such guidance, either explicitly as evaluation criteria for focusing a search or implicitly in the form of heuristics which restrict the search.

The approach described by Prosser is not altogether different in this regard. Agents at higher levels, like the global agent responsible for releasing jobs to the factory, are assumed to operate with heuristic models, and the priorities assigned to agents at lower levels would seem to encode similar information.

At the same time, the interpretation given to the notion of a "satisfactory schedule" does provide a basis for distinguishing between different approaches to integrating predictive and reactive scheduling. Within Prosser's system, a satisfactory schedule is defined as the first solution found under a general dependency-directed backtracking scheme. Thus, the approach trades off performance optimization concerns in favor of some insurance of system responsiveness. Apparently, the hierarchical division of labor between scheduling agents and the assignment of agent priorities makes use of this general backtracking scheme computationally efficient. The approach of Collinot and LePape takes a different position relative to the tradeoff between performance optimization and system responsiveness. These approaches also concentrate on producing a single solution for efficiency reasons, but rely instead on heuristic models reflecting performance optimization objectives to control the search and decide how to backtrack. These approaches also make a further distinction between predictive scheduling contexts where more computation time is available and reactive scheduling contexts. In the approach of Collinot and LePape, for example, the amount of constraint propagation performed is tailored to the time available.

With respect to the issue of how the characteristics of the manufacturing environment should influence the coupling between predictive and reactive decision-making, there was general agreement that the types of uncertainties encountered should dictate both the amount and nature of predictive scheduling that should be performed. Ideally one would like a predictive schedule that retains the executional flexibility to absorb operational uncertainties. The trick is to accomplish this while still providing constraints that lead to good factory performance. MRP-style frameworks provide an extreme example here. Predictability in the context of the consistent meeting of due dates is often achieved under these approaches. However, this is a misleading indicator of good factory performance because the use of inflated lead times in setting due dates allows (and in some sense sanctions) gross production inefficiencies. It was suggested that a distinction could be made between "normal uncertainties", which are in some sense expected (like variability in processing time) over the scheduling horizon and "catastrophic events", which happen less frequently and are in some sense unexpected (like a surprise machine breakdown). A good predictive schedule should be robust to normal uncertainties, whereas the occurrence of catastrophic events warrant schedule revision. However, techniques for determining which decisions to close on in advance and which to leave open until execution time so as to achieve sharper yet sufficiently flexible constraints on factory operations remain a subject of current research.

With respect to the issue of measuring the utility of various approaches, it is clear that this requires relative evaluation of total systems, perhaps via simulation. This raised the subject of defining a set of benchmark problems against which the performance of various approaches could be contrasted. Since the topic of benchmarks was to be considered in detail in the last session, this part of the discussion was deferred.

Maintaining Convenient Schedule Descriptions

Claude Le Pape, Moderator

The Issues

Many scheduling systems are decomposed into a general system that maintains a "convenient'' description of a schedule being developed or executed, and a more specific decision-making system, for example, a human user, a tree search algorithm, or a set of schedule modification operators. Basically, the first component provides a characterization of choices that remain to be made to complete the schedule and a description of noticeable constraint violations while the second component makes and retracts scheduling decisions accordingly.

There is considerable commonality in the functionality provided by alternative schedule maintenance components. However, developers of scheduling systems have not converged on a particular scheme. In particular, time maps, explicit timetables, and logic/algebraic formalisms are often considered as exclusive alternatives, each of which provides some advantages and includes some drawbacks. Consequently, this session was designed to make progress with respect to the following questions:

1) What are the goals of the separation between schedule maintenance and decision-making components?

2) What is a "convenient'' schedule description?

3) How can we efficiently update such a description as decisions are made and as events occur on the factory floor?

4) What makes one formalism more appropriate than another for a given application?

Speakers advocated a variety of approaches to schedule maintenance, ranging from general purpose constraint solving techniques, to problem independent techniques for managing time and capacity constraints, to more specific, but potentially more efficient, representations and propagation techniques. They also presented the separation between schedule maintenance and decision-making from very different perspectives.

The Contributions

Stephen P. Smith of Northrop, in a paper titled "Scheduling Using Schedule Modification Operators", considered the separation as a natural database-like decomposition. The schedule maintenance component is viewed as a database management system provided with a number of "constraint violation noticers'' allowed to look over the current schedule and to determine if the corresponding constraints have been violated. Decision-making then consists of selecting 1) the most important of the violated constraint and 2) the best "schedule modification operator'' applicable to fix this constraint. According to the author, the main difficulty in building a solution maintenance component is that the schedule representation cannot be independent of its use. For instance,

a "general'' schedule maintenance component must allow its user to choose either a continuous or a discrete model of time.

In the paper "Application of CHIP to Industrial Scheduling Problems" by Pascal Van Hentenryck of the European Computer Industry Research Center, the separation concerns the efficiency of the scheduling process and the ease of formulating constraints, either at the beginning of a scheduling session (definition of the scheduling problem) or after a schedule has been generated (incremental modification of the problem). This work aims at combining the advantages of logic programming to naturally state combinatorial problems and the efficiency of state-of-the-art constraint solving techniques to solve these problems. The schedule maintenance component is a "constraint solver'' that uses a collection of efficient algorithms, such as boolean unification algorithms, to solve equations on boolean terms, constraint propagation algorithms to solve numeric and symbolic constraints on finite domains, and an incremental version of the simplex algorithm to solve linear optimization problems. The decision-making component is a logic programming inference engine with which a programmer can do anything he wants including tree search, problem modification, and user interaction. According to the author, the resulting system enables an efficiency comparable to special-purpose codes for a large class of combinatorial problems, while the burden of tree-search and constraint propagation programming are abstracted away from the programmer.

Jacques Erschler, Pierre Lopez, and Catherine Thuriot of LAAS, in their paper "Scheduling under Time and Resource Constraints", distinguish different types of knowledge involved in solving scheduling problems: theoretical knowledge about time and resource management, empirical knowledge coming from previous experimentation with heuristic decision procedures, and practical knowledge concerning a particular application. Theoretical knowledge constitutes the generic kernel of the overall scheduling system. As in previous scheduling systems, a time and resource management system makes use of constraint propagation rules to characterize admissible schedules. Constraints represent the need for consistency between tasks as well as time and resource limitations. For instance, bounds on durations appear in a graph of bounded differences ($(x - y) >= c$) and the availability of resources over time is represented through supplier and consumer intervals. The main difficulty is to constitute a set of constraint propagation rules which allow efficient generation of new constraints from existing constraints. The use of new rules (relating to original concepts such as "energy'') is advocated to increase the computing power of the management system.

Yoshio Tozawa, Shin-Ichi Hirose, and Michiharu Kudo of IBM, in a paper entitled "Sub-Assembly Scheduling System", focus on the difference between types of knowledge and consider the separation between schedule maintenance and decision-making as a way to facilitate the interaction with human experts in cases in which all the constraints cannot be provided a priori. Important constraints are identified and given at the beginning to a scheduling engine able to satisfy them. Unlisted constraints are given afterwards by human experts. Then the scheduling engine modifies the schedule and the process continues until no new constraint is given to the system.

Constantine Spyropoulos and Stavros Kokkotos, in their paper "Interactive Fuzzy Scheduling using the Time Graph System TGS", concentrate on efficiency issues. They consider the design of a schedule maintenance component with guaranteed response-time to information requests. They consequently advocate the use of indexed structures to efficiently access information. A multi-root time graph with index nodes enables efficient time management and guarantees reaction response-time to information requests. Access time is nearly independent of the graph size. However, complicated requests, like determining all of the possible insertions of a process of operations into a schedule, seem to remain computationally intractable.

The Discussion

The comparison of these five approaches raised issues concerning the representation of schedules within a schedule maintenance component, and consideration of these issues constituted the bulk of the discussion.

The convenience of the internal schedule description language to the system user needs attention. Indeed, one can always exploit an existing schedule maintenance component and interface it with an external language. When the schedule maintenance component is principally considered as a database management system, the ease of accessing information in a suitable form is extremely important. When the schedule maintenance component is considered as an inference system or as a deductive database management system, the efficiency of the inference process indubitably becomes the main concern.

Another issue concerned the generality of the representation. For instance, some schedule maintenance components do not allow continuous processes or pre-emptions between manufacturing operations while others do not allow continuous or discrete models of time. This raises the question of the design of a universal model of factory operations. The expressive power of such a model would probably have to be equivalent or superior to the expressive power of first-order logic. Another (simpler?) question is to determine which representations are good in which factories. For example, the availability of a resource is often represented through a set of reservations, which means that individual resources of the group are unavailable throughout an interval of time. The question is to determine when such a representation allows 1) description of the possible behaviors of the factory and 2) implemention of an efficient constraint propagation process.

An important distinction between the five approaches centers around the efficiency of the schedule maintenance process. In some sense, the time graphs of Constantine Spyropoulos and Stavros Kokkotos are related to Thomas Dean's time maps [Dean 86] with domain specific indexing structures built on top for more efficient manipulation and access. Stephen P. Smith advocates the use of constraint violation noticers to check constraints individually, so that the system does not get "out of control''. On the contrary,

Jacques Erschler, Pierre Lopez, and Catherine Thuriot advocate an extensive use of constraint propagation techniques in order to reduce the exploration of the search space. In general, it appears suitable to perform certain propagation steps and certain checks at specific times.

In some cases, knowledge on the problem solving strategy can be exploited to reduce the computational burden of schedule maintenance without sacrificing the deduction of important information. Relative to this point, the notion of a parameterizable propagator, giving the problem solver control over the amount of work that is done in specific decision-making contexts, was brought up as one means of addressing the efficiency/generality tradeoff. The paper by Anne Collinot and Claude Le Pape presented in the previous session provides an example of such an approach. The schedule maintenance component is explicitly defined as a controllable inference system in order to facilitate its adaptation to different problem solvers and problem solving contexts.

The programmer of Pascal Van Hentenryck's language can decide not to express constraints which are likely to be costly to propagate. Similarly, Thomas Dean [Dean 85] discusses the use of disjunctive constraints to represent possible choices: "The objective in representing choices is to avoid backtracking. Backtracking of any sort is potentially expensive but so is dealing with an exponential number of choice combinations. The (system described) allows the planner to use choices where appropriate''. These approaches are not as general as controlling the computational behavior of the constraint propagation system. Indeed, they do not allow the planner to decide what the constraint propagation system does with each constraint at each point in time.

Difficult issues include identifying when specific types of inference can reasonably be avoided and defining an appropriate language for "programming'' the propagator. Here as in Pascal Van Hentenryck's system, the notion of "programming'' raises the question of generic systems that can be customized with a small amount of effort. Existing parameterizable tools are not simple enough to be customized by the user of the scheduling system. In particular, the efficiency of the constraint propagation process is still an issue for system programmers. An interesting avenue of research is to make the adaptation of a generic system manageable by its users. Another is to provide the system with the ability to learn from its experience and adapt itself to the manufacturing environment in which it works. Since the topic of learning was to be considered in detail in the last session, this part of the discussion was deferred.

Advanced Topics

Karl Kempf, Moderator

Three separate advanced topics were covered in this session, two of which had been extracted from the submitted papers, one of which was suggested by the session moderator as newly elected SIGMAN Benchmarks Secretary.

Chaos

The first part of the session addressed the idea that the concepts of "chaos" as formalized for natural systems by mathematicians and physicists over the past 25 years may be applicable (in spirit if not in detail) to artificial systems in general, and manufacturing systems specifically. Two participants presented a micro-tutorial and a research agenda to alert members of the AI and scheduling community to the opening of this new area of work.

Karl Kempf of Intel Corporation began the session with highlights from a paper titled "The Concepts of Chaos Applied to Manufacturing Production Scheduling". The idea was to explore the analogy between determining time-dependent behavior a) by modeling natural systems with, for example, sets of differential equations, and b) by modeling artificial systems with, for example, discrete event simulators. In either case, solutions can only be found practically by computer. The general goal is to determine whether the concepts of chaos can provide problem-solving guidance to those working on artificial systems in the same way that we derive guidance from the concepts of NP-completeness. The specific research agenda will address three questions. 1) Is it theoretically possible for a manufacturing facility to achieve a chaotic regime? 2) Is it practically demonstrable that an actual manufacturing facility has achieved a chaotic regime? 3) Can or should the chaotic regime be avoided or utilized? Answers to these and related questions should be of great interest to AI practitioners involved in manufacturing scheduling.

H. Van Dyke Parunak of the Industrial Technology Institute ended the session with comments from a paper titled "The Challenge of Chaos to Manufacturing Scheduling". He first described one of the simplest natural systems which exhibits chaotic behavior, and drew analogies between this system and manufacturing systems. If the analogies hold, it is clear that none of the scheduling systems discussed during this workshop will be sufficient to ensure stable performance. It was then suggested that the best possible way to handle chaos is to avoid it. Two broad classes of approach seem promising, AI having much to offer in both. The first has to do with the intelligent design of manufacturing systems with appropriate structural constraints to preclude chaotic behavior. The second includes intelligent monitoring and interpretation of real-time manufacturing data to appropriately adjust operating parameters to keep out of chaotic regimes.

In the brief discussion that followed, it was imaginable to most of the participants that, given a single scheduling problem and a single schedule, two slightly different initial conditions could lead to radically different execution histories. This kind of behavior is, of course, the crux of chaos.

Learning

The second part of the session addressed the idea of applying methods developed in the area of AI and learning to problems encountered in manufacturing scheduling. Since these ideas had not been deeply explored in previous SIGMAN workshops, the first goal was simply to provide a perspective on the state-of-the-art for the participants. But the obvious questions included: what would be learned (how to predict, how to react, when to predict/react, scheduling rules, and so on): what is a plausible research agenda: what can be expected in what timescales? Again no conclusions were expected, but there was time for a small amount of discussion.

The first paper, entitled "FMS Scheduling: A Machine Learning Approach", by Alok Chaturvedi and George Hutchinson, described work done at the University of Wisconsin-Milwaukee. The learning technique applied, goal-directed conceptual aggregation (GDCA), evolved from the well-known conceptual clustering paradigm. GDCA relies on a database of decisions made in the past and a simulator for generating future states, both valid for the FMS being scheduled. Three inputs are given to the system including 1) the local decision to be made, which job to load next for example, 2) a global context in which the decision is to be made, reduction of work-in-progress for example, and 3) sample data from the FMS which the system uses to deduce its current status. After resolving any uncertainty in the sample data, the system classifies and aggregates relevant historical data, and passes the resulting selected parameters on for simulation. System performance measures are recorded for future reference.

The second paper, written by David Ruby and Dennis Kibler of the University of California-Irvine, was titled "Learning to Plan in Scheduling Domains". The learning problem solver described consists of three basic components including a problem solver, a memory of problem-solving knowledge, and a learning component able to compile problem-solving experience into problem-solving knowledge. The problem solver depends primarily on means-ends analysis, a technique which can have difficulty with some types of subgoal interaction. The memory of problem-solving knowledge is called whenever an impasse is encountered which requires the undoing of some previously solved subgoal. If a new subgoal sequence can not be found which solves the problem, the system resorts to a brute force search of task re-ordering. The resulting sequence of moves leading to an improved solution is used by the learning component to generate a new sequence of subgoals to be stored in memory for reuse.

The third paper, titled "Sharing Scheduling Knowledge Between Intelligent Agents", was submitted by Wen Ling Hsu, Allen Newell, Michael Prietula, and David Steier of Carnegie Mellon University. The paper describes MERL-SOAR, the application of the SOAR architecture, with its subgoaling approach to problem solving and chunking approach to learning, to scheduling problems. The focus is distributed problem solving agents and the idea that one intelligent agent can directly benefit from the knowledge accumulated by another problem solver. Four MERL-SOAR schedulers were exercised in task environments which differed only in the volume of tasks to be scheduled. The learned chunks generated by one agent while solving its local problem were then transferred to another agent. The receiving agent was reinitialized and proceeded to solve its own local problem again with a different set of initial knowledge. The general conclusion was that there is an overall benefit from sharing learned knowledge, and specifically, importing learned knowledge from agents solving more difficult problems is especially useful.

In the discussion that followed, it was generally agreed that schedulers that learn are certainly intriguing, both as a mechanism for constructing predictive schedules (learning how to avoid difficulties) and as a mechanism for reactively executing schedules (learning how to repair failures), especially since feedback is easily available in each case. However, some concern was voiced that there is a certain naivety on the part of the learning approaches as to the complexity of the manufacturing scheduling problem. On one hand, analytical approaches to learning are predicated on an ability to explore the entire search space (in the worst case) to find something to compile, but this is unreasonable given the size of practically interesting manufacturing scheduling problems. On the other hand, learning approaches which remember solutions for subsequent use may not be productive either since there is a perpetual novelty in problem solving situations encountered by actual manufacturing schedulers over time. While abstraction mechanisms were given as partial solutions to both of these difficulties, generalizing over problems with the scale of combinatorics exhibited by scheduling was agreed to be a non-trivial exercise.

Benchmarks

The third part of the session was motivated by an extract from the SIGMAN Charter: "SIGMAN is interested in the precise definition of paradigmatic manufacturing problems, the clear and complete description of AI solutions, and the unambiguous evaluation of implementations. It is particularly interested in the availability of well-defined problem sets and performance benchmarks as aids to communication and progress." The initial purpose of discussion of this statement was to decide whether a set of benchmark scheduling problems were both possible and desirable. If the answer to both parts of this question were yes, then the secondary purpose was to brainstorm concerning ways to generate, collect, maintain, distribute, and utilize such benchmarks.

The single contribution in this session came from Karl Kempf of Intel Corporation representing the SIGMAN Executive Committee as Benchmark Secretary. It was pointed out that within the AI scheduling community there is no problem description vocabulary or classification scheme and hence no map of the problem space. Neither do we have a solution description language, classification scheme, or evaluation procedure. These deficiencies make communication among those interested in working on scheduling very difficult indeed, and therefore make progress more difficult than it should be.

Ideally we would have an abstraction hierarchy of scheduling problems with sets of benchmark problems as leaf nodes. Furthermore, there would be a complementary abstraction hierarchy over scheduling solutions with sets of literature references as leaf nodes. A mapping between problem leaf nodes and solution leaf nodes would be especially interesting.

Many benefits would result from the realization of such a scheme. There would be a framework in which to store our collective experiential assessment of the detailed difficulties involved with each problem type. The framework would also help us evaluate the qualities of our ideas based on which benchmarks a particular solution could handle and its performance on each. The scheme would clearly identify which problems could be classified as solved and which solutions could be accepted as standard. Unsolved problems would be more obvious and trial solutions

could be more easily evaluated. Holes in both the problem set and solution set might be recognized. Such a framework would support collaboration within the community as well as allowing those new to the field to come up to speed more rapidly.

Unfortunately, there are a large number of excuses which can be raised to explain why this goal cannot be reached. Benchmarks never really work as they should. They always turn out to be subjective, biased, and prejudiced, reflecting the viewpoint of the person who developed them. Besides, the topic of manufacturing scheduling is too large with far too many problems to classify and too many solutions to catalog. Furthermore the problems are too ill-defined and the solutions not well enough described.

It was strongly proposed that these are nothing more than excuses, and the imagined pitfalls are far outweighed by concrete benefits. Subjective, biased, prejudiced benchmarks from a large and diverse set of experts can be very valuable. Performance measured against such a set of benchmarks is far less misleading than the results on someone's favorite single example problem. While it is true that scheduling is a large area, it is also true that there are a large number of us interested in the problem. Ill-defined problems remain so until we devise a way to define them. Poor solution descriptions persist until we adopt the discipline to describe them well.

Generally, these remarks were well received by the participants and discussion continued until well past the official ending time of the workshop. There was little (if any serious) discussion about whether or not this proposal was either possible or desirable as originally set in the statement of issues for the session. The participants moved immediately to implementation suggestions.

The opinion was expressed that assembling a set of benchmark problems is the obvious starting place. Trying to build a problem description language at this point would be too difficult a task. But given an expanding set of benchmark problems, it would be necessary to find a useful format in which to collect and disseminate the set. This would force the groundwork to be laid for the problem description language.

It was pointed out that an initial set of benchmarks might be gathered from the Operations Research literature, although it was emphasized that this could not be considered anything more than a starting point for either problems or their description. More complex and/or more realistic examples would be added when necessary. At this point there were several participants who volunteered to try to construct benchmark contributions from their current work. In fact, it was made known that a research report was nearing completion at Carnegie Mellon University that included descriptions of many of the test problems that had been utilized there.

It was thought to be very important to try for good coverage of the problem space from the very beginning. This was stressed since support from the AI-based scheduling community was thought to be vital for this effort to gain the momentum needed to succeed over the course of time. Part of the basic idea is that the benchmarks should begin to be included in publications as the normal measure of solution coverage and performance. But if the benchmark set does not serve a large percentage of the community, regular use will not be achieved.

Participants were anxious to define "good coverage" for "regular use", but not many concrete remarks were made. There should be problems from many different problem domains, that is problems containing many different kinds of entities and constraints. There should be a range of problem sizes in terms of the number of entities and constraints involved. And the problems should include various metrics by which to measure the quality of the solution.

A related topic of discussion was problem generators. It was thought that the benchmark problem should actually encode a group of actual problems. The benchmark would contain descriptions of the resources, jobs, constraints, and so on, with ranges of parameters. Actual problems for system testing would result from picking specific parameters. For example, the benchmark might contain a parameter for the quantity of each product to make and another parameter for the due dates of each product. A specific problem might then be 1500 of product A due at a rate of 300 on the 16th of each month and 2500 of product B due at a rate of 500 on the 25th of each month. Another problem might require 5000 of A and 1000 of B. This means that the benchmark would not only include the problem description, but also the problem generator.

The discussion and workshop closed with the SIGMAN Benchmark Secretary calling for assistance on two topics. The workshop participants were encouraged to submit sample problems from their past or present research. This would allow a beginning to be made on the mechanism for representing and disseminating benchmarks. The participants were also asked to supply reference lists so that a complete bibliography of the AI manufacturing literature can be compiled. This might also serve as a source of test problems as well as allow a beginning to be made on the goal of mapping the solution space. This bibliography would also be disseminated by the Benchmark Secretary.

References

Dean, T.L. 1985. Temporal Reasoning Involving Counterfactuals and Disjunctions. In Proceedings of the Ninth International Joint Conference on Artificial Intelligence (Los Angeles, California, USA), 1060-1062. Menlo Park, Calif.: International Joint Conference on Artificial Intelligence.

Dean, T.L. 1986. Temporal Imagery: An Approach to Reasoning about Time for Planning and Problem Solving. Ph.D. diss., Dept. of Computer Science, Yale University.

Contributors

Barry Fox received his Ph.D. in Computer Science from the University of Missouri in 1987. He served as a research scientist at McDonnell Douglas Research Laboratories from 1986 until 1989, doing research in sequencing and scheduling. During this time he produced POINT, a system for representing, analyzing, and sequencing activities that are subject to complex ordering constraints, and WEDGE, an interactive system for job-shop scheduling. In 1989, he moved to McDonnell Douglas Space Systems where he leads the development of COMPASS, an interactive scheduling system produced for NASA/JSC intended for Space Shuttle and Space Station Freedom applications.

Stephen F. Smith is a Research Scientist in the Robotics Institute at Carnegie Mellon University, and Director of the Production Control Laboratory within the Institute's Center for Integrated Manufac-

turing Decision Systems. He holds a B.S. degree in Mathematics from Westminster College, and M.S. and Ph.D. degrees in Computer Science from the University of Pittsburgh. His research interests include constraint-based planning and scheduling, integration of predictive and reactive decision-making, distributed problem solving, temporal reasoning, machine learning, and knowledge-based production management. He has been a principal architect of several knowledge-based scheduling systems for complex manufacturing and space applications.

Claude Le Pape is a visiting researcher in the Robotics Laboratory at Stanford University. He received a Ph.D. in Computer Science from University Paris XI in 1988. He also attended "Ecole Normale Superieure de Paris" from 1982 to 1987 and received a management degree from "College des Ingenieurs" in 1988. His main interests are constraint satisfaction, knowledge base consistency checking and the application of symbolic reasoning techniques to manufacturing and engineering problems. His current worked is aimed at determining to what extent various constraint propagation techniques allow autonomous robots to construct and execute efficient schedules with minimal centralized control.

Karl Kempf is Principal Scientist in the Knowledge Application Laboratory at Intel Corporation located in Santa Clara, California, and is a Mentor in the Center for Integrated Systems at Stanford University. He was the founding co-chairman (with Mark Fox of Carnegie Mellon University) of the AAAI Special Interest Group in Manufacturing (SIGMAN), and currently serves as its benchmark secretary. He is also on the editorial board of IEEE Expert. Karl received a B.S. in Chemistry and a B.A. in Physics from Otterbein College (as a mathematics major), and did graduate work at Stanford University and The University of Akron, receiving his PhD in Computer Science. His research interests center on spatial and temporal reasoning systems applied to robots specifically and to manufacturing systems in general. Karl has been involved in the design, development, and delivery of six successful AI-based manufacturing applications, and has produced over 40 publications in the area of AI and Manufacturing.

AI-Based Schedulers in Manufacturing Practice: Report of a Panel Discussion

Karl Kempf, Bruce Russell, Sanjiv Sidhu, and Stu Barrett

Abstract

There is a great disparity between the number of papers which have been published about AI-based manufacturing scheduling tools and the number of systems which are in daily use by manufacturing engineers. It is argued that this is not a reflection of inadequate AI technology, but is rather indicative of lack of a systems perspective by AI practitioners and their manufacturing customers. Case studies to support this perspective are presented by Carnegie Group as a builder of scheduling systems for its customers, by Texas Instruments and Intel Corporation as builders of schedulers for their own use, and by Intellection as a consulting house specializing in scheduling problems.

Introduction

Given the impact of manufacturing on the gross national product of any developed nation, it is not surprising that an increasing number of AI practitioners are becoming involved in the manufacturing domain. (Although the AAAI SIGMAN (Special Interest Group in MANufacturing) held its first formal business meeting at AAAI-88, its membership already includes roughly one-third that of the AAAI parent organization.) From an optimistic viewpoint, this blossoming of interest could bring about two important results. One is the revitalizing effect that strong solutions for outstanding problems would have in manufacturing. The other is the validating effect that successful solution of large scale problems would have on AI theories.

The pessimistic reality is that these results have not yet been realized in spite of long-term efforts by talented people. In this paper, we try to examine some of the reasons for this limited progress. While there are clearly technical problems encountered in applying AI techniques to manufacturing problems, our experience is that it is more likely that "people problems" block the march of progress. Although specifying and implementing the things which go on inside the computer are difficult, handling the things which go on outside of the computer are even more troublesome. While a few comments on technological matters will inevitably slip into the discussion, our intended focus is the manufacturing personnel and AI technologists involved in the projects with which we are familiar.

Artificially intelligent schedulers of manufacturing production serve as our example application. A variety of scheduling problems, including flowshops and job-shops, have been shown to be NP-complete [Garey and Johnson 1979]. A wide selection of AI-based solutions have been proposed including at least the 100 referenced in a recent overview article [Kempf 1989a]. But very few AI-based schedulers have found their way into daily manufacturing practice. Even the most publicized of AI-based manufacturing schedulers, the ISIS/OPUS systems produced at Carnegie Mellon University starting in 1980, have yet to enter productive service and are not considered to be on the verge of doing so [Smith 1989].

A panel was convened at the AAAI co-sponsored Third International Conference on Expert Systems and the Leading Edge in Production and Operations Management with the charter of addressing the disparity in manufacturing production scheduling between the number of papers in print and the number of systems in service. Under the auspices of the Management Sciences Department in the College of Business Administration of the University of South Carolina, a number of speakers were invited to participate. With direction from Timothy Fry, one of the Program Chairmen, the aerospace, automotive, and electronics industries were targeted. Systems were sought which had been built from scratch and/or using commercial tools, and