Production Control, Management and Planning

Constraint-Guided Scheduling— A Short History of Research at CMU

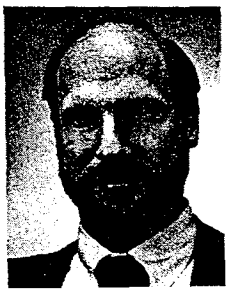
Mark S. Fox
School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, U.S.A.

This paper describes the historical evolution of the ISIS/OPIS/CORTES family of knowledge-based scheduling systems developed at Carnegie Mellon University. At the core of the ISIS/OPIS/CORTES family is an approach to automatic scheduling that provides a framework for incorporating the full range of real-world constraints. Given the conflicting nature of the domain's constraints, the problem differs from typical constraint satisfaction problems. One cannot rely solely on propagation techniques to arrive at an acceptable solution, since no feasible solution may exist. Rather, constraints must be selectively relaxed in which case the problem solving strategy becomes one of finding a solution that best satisfies the constraints. Secondly, constraints on the available capacity of resources forces a scheduling system to divert its attention opportunistically between a job-centered perspective and a resource-centered perspective.

Elsevier
Computers in Industry 14 (1990) 79–88

1. Introduction

The ISIS family of knowledge-based systems have been designed to provide intelligent support in the domain of job-shop scheduling. Job-shop scheduling is a "uncooperative" multi-agent (i.e., each job is to be "optimized" separately) planning problem in which activities must be selected, sequenced, and assigned resources and times of execution. Resource contention is high, hence closely coupling decisions. Search is combinatorially explosive; for example, 85 orders moving through eight operations without alternatives, with a single machine substitution for each and no machine idle time has over 10^{880} possible schedules. The selection of a schedule is influenced by such diverse factors as due rate requirements, cost restrictions, production levels, machine capabilities and sub-



Mark S. Fox received his BSc in Computer Science from the University of Toronto in 1975 and his PhD in Computer Science from Carnegie Mellon University in 1983. In 1979 he joined the Robotics Institute of Carnegie Mellon University as a Research Scientist. In 1980 he started and was appointed Director of the Intelligent Systems Laboratory. He co-founded Carnegie Group Inc. in 1984, a software company which specializes in knowledge-based systems for solving engineering and manufacturing problems. Carnegie Mellon University appointed him Associate Professor of Computer Science and Robotics in 1987. In 1988 he was appointed Director of the new Center for Integrated Manufacturing Decision Systems, which is one of the largest centers in the US for research in intelligent systems to solve engineering and manufacturing problems. Dr. Fox pioneered the application of Artificial Intelligence to factory planning and scheduling problems, project management, and material design. He was the designer of PDS/GENAID, a steam turbine generator diagnostic system, which was a recipient of the IR100, and was the creator of SRL from which Knowledge Craft, a commercial knowledge engineering tool, was derived. Research interests include knowledge representation, constraint directed reasoning and applications of artificial intelligence to engineering and manufacturing problems. Dr. Fox has published over 50 papers, and is a member of AAAI, ACM, IEEE, SME, CSCSI and TIMS.

stitutability, alternative production processes, order characteristics, resource requirements, and resource availability.

At the core of the ISIS/OPIS/CORTES family of systems is an approach to automatic scheduling that provides a framework for incorporating the full range of real-world constraints described above. Given the conflicting nature of the domain's constraints, the problem differs from typical constraint satisfaction problems. One cannot rely solely on propagation techniques to arrive at an acceptable solution, since no feasible solution may exist. Rather, constraints must be selectively relaxed in which case the problem solving strategy becomes one of finding a solution that best satisfies the constraints. Secondly, constraints on the available capacity of resources requires a scheduling system to divert its attention opportunistically between a job-centered perspective and a resource-centered perspective. Thus, the design of ISIS/OPIS/CORTES has focused on

- constructing a knowledge representation that captures the requisite knowledge of the job shop environment and its constraints to support constraint guided search, and
- developing a search architecture capable of exploiting this constraint knowledge to effectively control the combinatorics of the underlying search space.

This results in an ability to generate detailed schedules for production that accurately reflect the current status of the shop floor, and distinguishes ISIS/OPIS/CORTES from traditional scheduling systems that are more myopic. ISIS/OPIS/CORTES is capable of incrementally scheduling orders as they are received by the shop as well as reactively rescheduling orders in response to unexpected events (e.g. machine breakdowns) that might occur.

This paper will describe chronologically the technical evolution and performance of constraint-guided search as embodied in the series of systems we have developed:

- ISIS-1: Constraint guided scheduling.
- ISIS-2: Hierarchical constraint guided scheduling.
- ISIS-3: Multiperspective scheduling.
- OPIS-1: Opportunistic scheduling.
- OPIS-2: Reactive scheduling.
- CORTES: Network-based constraint optimization.

2. Problem Definition

In 1980, I was asked to explore the application of AI techniques to a turbine component plant's job-shop scheduling problem. The primary product of the plant was steam turbine blades. A turbine blade is a complex three-dimensional object produced by a sequence of forging, milling, grinding and finishing operations to tolerances of a thousandth of an inch. Thousands of different styles of blades were produced in the plant, much of them as replacements in turbines in service.

The plant continuously received orders for one to a thousand blades at a time. Orders fell into at least six categories:

(1) Forced outages (FO): Orders to replace blades which malfunctioned during operation. It is important to ship these orders as soon as possible, no matter what the cost.
(2) Critical replacement (CR) and Ship Direct (SD): Orders to replace blades during scheduled maintenance. Advance warning is provided, but the blades must arrive on time.
(3) Service and shop orders (SO, SH): Orders for new turbines. Lead times of up to three years may be known.
(4) Stock orders (ST): Order for blades to be placed in stock for future needs.

The portion of the plant studied has from 100 to 200 orders in process at any time.

Parts are produced according to a process routing. A routing specifies a sequence of operations on the part. An operation is an activity which defines:

- resources required such as tools, materials, fixtures, and machines;
- machine setup and run times, and
- labor requirements.

In the plant, each part number has one or more process routings containing ten or more operations [1]. Process routing variations may be as simple as substituting a different machine, or as complex as changing the manufacturing process. Furthermore, the resources needed for an operation may also be needed by other operations in the shop.

[1] Multiple process routings correspond to a network of activities, each path representing a separate plane.

In AI terms, job-shop scheduling is a planning problem with the following characteristics:

- It is a *time-based* planning problem (i.e., scheduling) in which activities must be selected, sequenced, and assigned resources and time of execution.
- It is a *multi-agent* planning problem. Each order represents a separate agent for which a plan/schedule is to be created. The number of agents to be scheduled is in the hundreds.
- The agents are *uncooperative*. Each is attempting to maximize its own goals.
- *Resource contention* is high, hence closely coupling decisions.
- Search is *combinatorially explosive*. 85 orders moving through ten operations without alternatives, with a single substitutable machine for each operation and no machine idle time has over 10^{880} possible schedules.

An expert systems approach was used to construct the scheduler. This approach assumed that one or more experts could be interviewed to acquire the rules which govern their decision process. During our discussions, we found that orders were not scheduled in a uniform manner. Each scheduling choice entailed side effects whose importance varied by order. One factor that continuously appeared was the reliance of the scheduler on information other than due dates, process routings, and machine availability. The types and sources of this information were found by examining the documents issued by the scheduler. A schedule was distributed to persons in each department in the plant. Each recipient could provide information which could alter the existing schedule. In support of this observation, we found that the scheduler was spending 10–20% of his time scheduling, and 80–90% of his time communicating with other employees to determine what additional "constraints" could affect an order's schedule. These constraints included operation precedence, operation alternatives, operation preferences, machine alternatives and preferences, tool availability, fixture availability, NC program availability, order sequencing, setup time reduction, machine breakdowns, machine capabilities, work-in-process time, due dates, start dates, shop stability, cost, quality, and personnel capabilities/availability.

From this analysis, I concluded that the object of scheduling is not only meeting due dates, but satisfying the many constraints found in various parts of the plant. Scheduling is not a distinct function, separate from the rest of the plant, but is highly connected to and dependent upon decisions being made elsewhere in the plant. The added complexity imposed by these constraints leads schedulers to produce inefficient schedules. Indicators such as high work-in-process, tardiness, and low machine utilization support this conclusion [2]. Hence, any solution to the job-shop scheduling problem must identify the set of scheduling constraints, and their effect on the scheduling process.

Consequently, taking an expert systems approach to scheduling appeared inappropriate. There are two problems with the expert systems approach:

(1) Problems like factory scheduling tend to be so complex that they are beyond the cognitive capabilities of the human scheduler. Therefore, the schedules produced by the scheduler are poor; nobody wants to emulate their performance.
(2) Even if the problem is of relatively low complexity, factory environments change often enough that any expertise built up over time becomes obsolete.

Expert systems appear to be appropriate only when the problem is both small and stable.

Once the issue of designing a constraint-guided scheduling system was identified, I decided to solve the problem by constructing a family of systems. The purpose being to investigate the performance of successively more sophisticated search architectures. At each stage, experiments were run to measure the effectiveness of the architecture.

3. ISIS-1: Constraint-Guided Scheduling [2,6]

AI views problem solving as search in a problem space guided by heuristics: solutions to problems are represented as symbol structures. A

[2] It is unfair to measure a scheduler's preformance based on the above measures alone. Our analysis has shown that scheduling is a complex constraint satisfaction problem, where the above indicators illustrate only a subset of constraints that the scheduler must consider. Schedulers are expert in acquiring and "juggling" the satisfaction of constraints.

physical symbol system exercises its intelligence in problem solving by search, i.e., by generating and progressively modifying symbol structures until it produces a solution structure [10]. The problem space operationalizes the concept of a physical symbol system. A problem space is composed of

- *states* which are collections of features that define some situation;
- *operators*, that transform one state into another; and
- an *evaluation function*, that rates each state in the problem space.

Search begins at an *initial* state, and the problem is solved when a path is found from it to *a goal* state.

Scheduling can be viewed as a search through a problem space, where states represent partial schedules, operators extend a partial schedule defined by a state into a new state, and the evaluation function rates each state in the problem space according to the known constraints. Constraint-guided search is a form of search where constraints can be used to specify operators (e.g., operation precedence constraints specify the next operations) and terms of the evaluation function (e.g., a due date constraint measures slack in the schedule). The efficacy of this approach depends on the ability of the constraints to identify the more profitable paths to pursue. Experience has shown that this tends not to be the case. In our experiments [3], tardiness and work-in-process were both high except for high priority jobs.

4. ISIS-2: Hierarchical Constraint-Guided Scheduling [2–5,15]

Since the problem cannot be solved using either expert systems or constraint-guided search, more sophisticated search techniques are required. One approach is to reformulate the problem as a simpler problem whose solution can be used to guide the solution of the original problem. Hierarchical constraint-guided search is one method that embodies this approach.

Search is divided into four levels: order selection, capacity analysis, resource analysis, resource assignment. Each level is composed of three phases: a pre-search analysis phase which constructs the problem, a search phase which solves the problem, and a post-search analysis phase which determines the acceptability of the solution. In each phase, ISIS uses constraints to bound, guide, and analyze the search.

Level 1 is responsible for selecting the next unscheduled order to be added to the existing shop schedule. Its selection is made according to a prioritization algorithm that considers order type and requested due dates. The selected order is passed to Level 2 for scheduling.

Level 2 represents the simpler reformulation of the original problem. It simplifies the problem by removing both resources and constraints from consideration. It performs a dynamic programming analysis of the plant based on current capacity constraints. It determines the earliest start time and latest finish time for each operation of the selected order, as bounded by the order's start and due date, and available resources. The times generated at this level are codified as operation time bound constraints which serve to influence the search at the next level by constraining the times during which operations can be performed.

Level 3 solves the original scheduling problem. It selects a particular routing for the order and assigns reservation time bounds to the resources required to produce it. Pre-search analysis begins with an examination of the order's constraints, resulting in the determination of the scheduling direction (either forward from the start date or backward from the due date), the creation of any missing constraints (e.g. due dates, work-in-process), and the selection of the set of search operators which will generate the search space. A beam search version of constraint-guided search is then performed using the selected set of search operators. The search space to be explored is composed of states which represent partial schedules.

Once a set of candidate schedules have been generated, a rule-based post search analysis examines the candidates to determine if one is acceptable (a function of the ratings assigned to the schedules during the search). If no acceptable schedules are found, then diagnosis is performed. Intra-level repair may result in the re-instantiation of the level's search. Pre-analysis is performed again to alter the set of operators and constraints for rescheduling the order. Inter-level repair is initiated if diagnosis determines that the poor solutions were caused by constraint satisfaction decisions made at a higher level.

Level 3 outputs reservation time bounds for each resource required for the operations in the chosen schedule. Level 4 then establishes actual reservations for the resources required by the selected operations which minimize the work-in-process time.

This approach performs well when there exists adequate capacity in the factory. Relative to ISIS-1, tardiness was reduced by 80% and work-in-process by 33%. But in situations where contention for resources was high, the performance of the system was no better than the human scheduler.

5. ISIS-3: Multiple Perspective Scheduling [16,17]

Work began on ISIS-3 during the summer of 1984. Though ISIS-2 made significant headway in satisfying its constraints in the presence of a high degree of resource contention, it was still believed that better use of the resources could be made resulting in higher constraint satisfaction.

In situations where resources are highly contended for, experience has shown that optimizing resource allocation by scheduling operations incident with the resource produces better results than

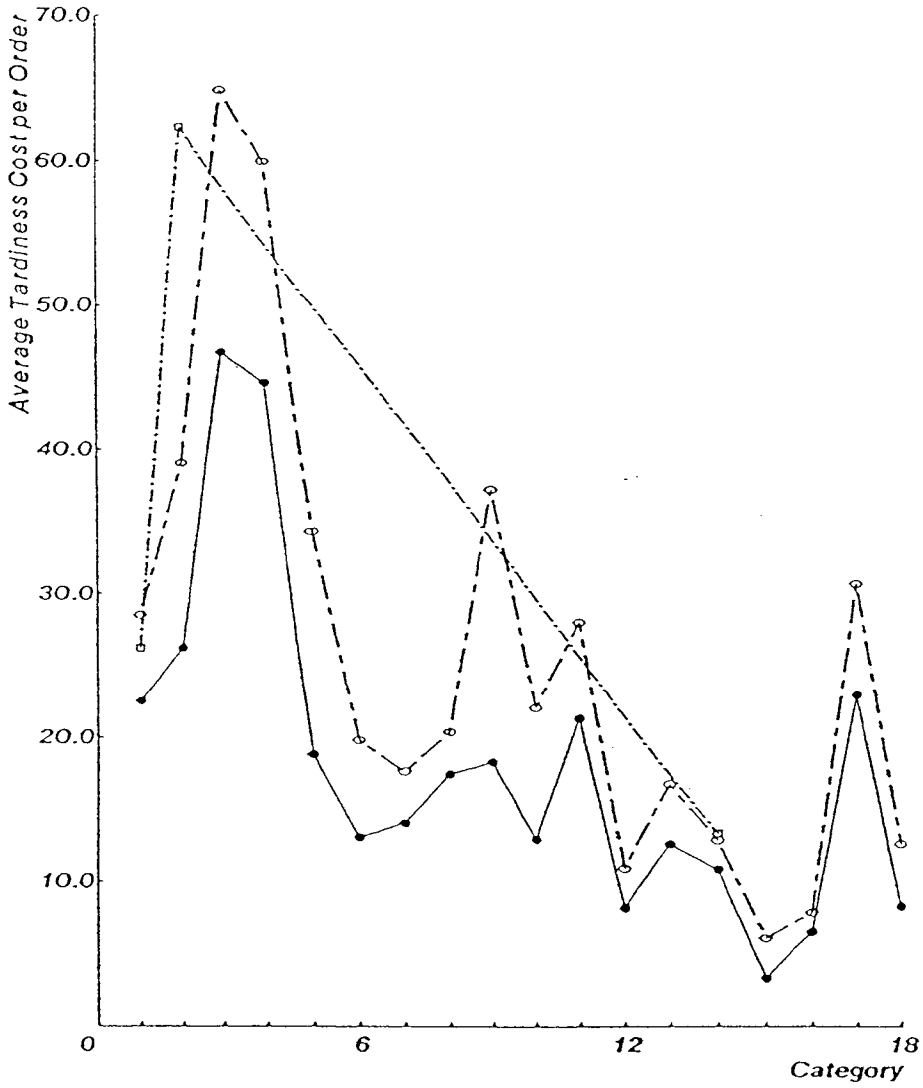


Fig. 1. ISIS-3 experiments. Average tardiness per order. (— · —) ISIS; (— – —) COVERT; (———) OPIS.

scheduling jobs one at a time. Multiple perspective scheduling [16,17] extends hierarchical search by first analyzing the capacity requirements of all jobs in order to measure the amount of resource contention. If contention exists for a resource, then a resource-centered scheduler is chosen to schedule the operations incident with it (usually a dispatch rule simulation [11]). The job-centered scheduler (i.e., ISIS-2) is used to schedule the jobs out from the resource.

The approach was to mix order scheduling with resource scheduling. (See [16] for more details.) This was accomplished as follows:

- An additional level was added to ISIS-2. This level generated a factory schedule approximated schedule using a focus approach dispatch rule [11]. Based upon this schedule, the largest bottleneck was identified and the operations associated with it were scheduled.
- The bottleneck and the schedules of orders through it were passed down to the resource analysis level. This level was modified to perform "island driving", similar to that found in Hearsay-II [1]. The bottleneck was designated an "island", and the highest priority order was selected and scheduled out (forward and backward) from the island using the original beam search with the added constraints of this level.
- The rest of the ISIS-2 architecture remained the same.

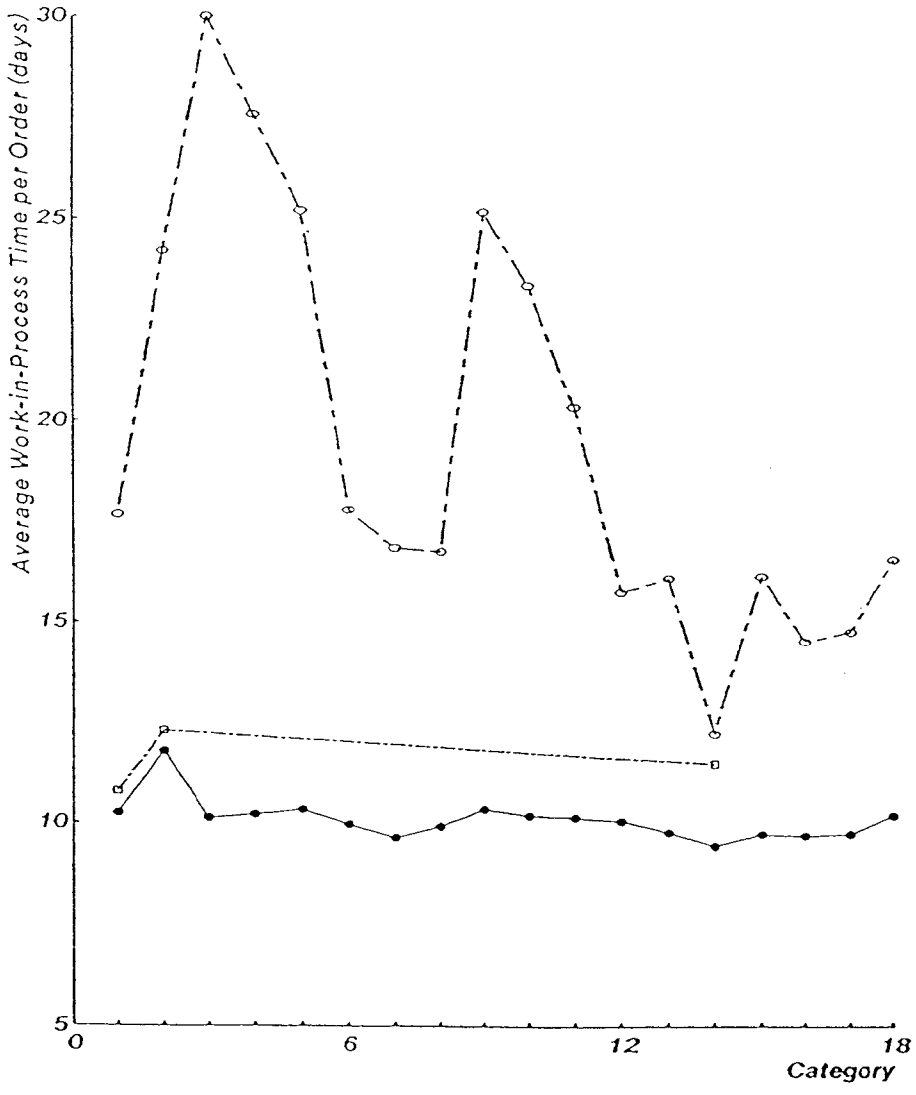


Fig. 2. ISIS-3 experiments. Average work-in-process time per order. (— · —) ISIS; (— – —) COVERT; (———) OPIS.

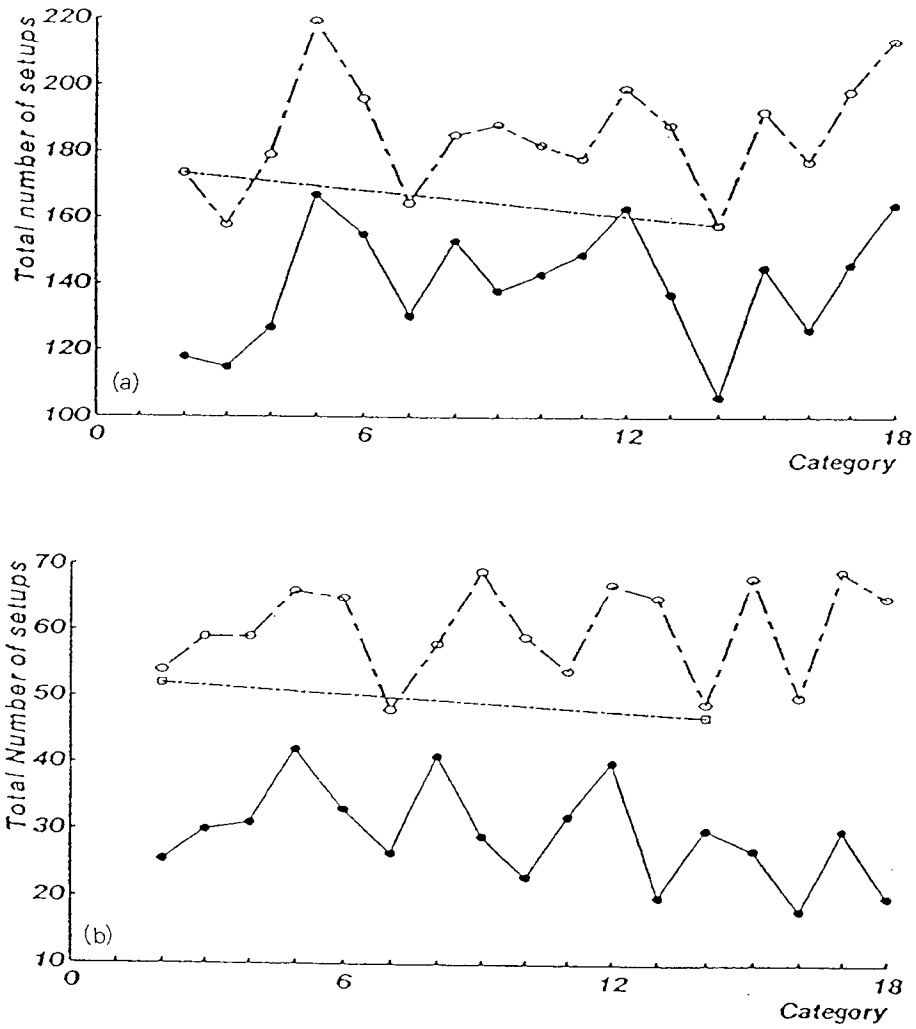


Fig. 3. ISIS-3 experiments. Total number of setups: (a) for all resources; (b) for bottleneck resources. (— · —) ISIS; (— - —) COVERT; (———) OPIS.

A new set of experiments were composed of: 120 orders, six priority classes, varying lead times, 33 machines, and varying load on bottlenecks. Each experiment varied the: product-mix (2), load on the bottleneck resources (70%, 95%, some 130%), distribution of lead times, and distribution of intervals between order releases. The experiments tested two types of manufacturing environments. The job shop environment described earlier, plus a computer board assembly and test line configured as a flow shop.

Figures 1–3 shows that the multi-perspective version of ISIS-3 minimizes tardiness, work-in-process and number of setups better than ISIS-2 and the COVERT dispatch rule.

In summary, compared to COVERT, ISIS-3 demonstrated a 25% improvement in average tardiness costs in 70% of the experiments, and a 10% improvement overall. Average work in process time over all experiments improved by 50% with variances of 4–9 days versus 14–225 days. Lastly, the number of setups at bottleneck was reduced by 50%.

6. OPIS-1: Opportunistic Scheduling [8,9,18]

The ISIS-3 architecture is inflexible in the sense that once it identifies the bottleneck resource, it schedules it and then schedules each job out from

the bottleneck using the agent centered scheduler (i.e., ISIS-2). Depending on the state of the factory, this sequence may be inappropriate; it may be more appropriate to schedule the highest priority job first. In the summer of 1985 work began on OPIS, the beginning of a new series of planning/scheduling systems in which opportunism in search plays a greater role. In particular, the first version of OPIS, focuses on opportunistic selection of the scheduling perspective. Opportunism arises out of the system's ability to dynamically determine at any point during the construction of schedules, primary/secondary bottleneck and take the opportunity to schedule them rather than pursue a strictly job centered approach.

OPIS-1 is implemented as a blackboard architecture [1]. The blackboard contains a model of the factory's resources, including parts, machines, tools, etc., activities such as manufacturing operations, constraints and generated schedules. Whenever scheduling decisions are made, or updates of the factory status are received, the blackboard manages the propagation of temporal constraints.

Scheduling decisions are made by knowledge sources (KS). There are two classes of knowledge sources: Analysis KSs and Decision KSs. OPIS-1 contains single analysis KS for analyzing capacity. It constructs a rough schedule using a line balancing heuristic and then determines the bottlenecks from computed demand/supply ratios. The two decision KSs are the resource scheduler and the order schedule described in ISIS-3.

Search is controlled by the Search Manager. Each cycle, temporal demands for resources are propagated both horizontally across operations and vertically across aggregations of resources. Significant events such as time and capacity conflicts, machine breakdowns, and scheduling decisions are identified and posted on an agenda. The Search Manager creates a subtask for each event, prioritizes them and selects the highest priority task to execute. Upon completion of the subtask by a KS, the cycle begins again.

Performance data on this system is still being gathered.

7. OPIS-2: Reactive Scheduling [12]

Manufacturing environments are dynamic; schedules, once generated, are seldom adhered to. This is due to unanticipated events such as resource limitations, e.g., machine breakdowns, and errors in production processes. Due to the tight coupling of scheduling decisions, the effect of an unanticipated event may spread problems to other parts of the schedule. It is therefore necessary that a scheduling system perform incremental revision of existing schedules in response to unanticipated changes in the environment.

The event driven opportunistic control strategy of OPIS-1 provides the basis upon which the reactive response to events may be constructed. Given a schedule represented via operation time bound and resource available capacity descriptions, external changes are mapped into this representation. Constraints resulting from schedule changes are combined with model-defined constraints to update these descriptions. Conflicts detected during constraint propagation, such as time conflicts and capacity conflicts, result in events being placed upon the agenda. Due to the ripple effect of these events on the schedule, elementary conflicts are aggregated according to commonality of the resources involved so that they can be dealt with by a single algorithm.

The Search Manager then selects a knowledge source to deal with each event. In OPIS-2 [12], two more decision knowledge sources are introduced:

- *Right shifter*—forward "push" of designated scheduling decisions (along with those conflicting with the push).
- *Demand Swapper*—exchange of schedule components of two orders of the same type (redirecting orders to fill each other's respective demands).

The introduction of these knowledge sources represents a shift in the OPIS approach. In particular, OPIS-2 extends the multi-perspective view of scheduling to include alternative scheduling algorithms within a view. That is, within an order centered view, there is more than one scheduling algorithm available, such as the "right shifter" and the "order scheduler". The search manager has to be able to differentiate among the events in order to choose the appropriate decision knowledge source. In order to accomplish this, OPIS-2 has a decision tree that defines the conditions under which a knowledge source is selected.

OPIS-2 was tested on an actual model of a computer board assembly and test line containing 10 sectors (aggregate resources), planned and un-

planned revisits to various sectors, and approximately 20 sector-level operations per board. A set of 26 reactive problems, such as an operation failure (implying extra repair operations) or a loss in resource capacity, were defined relative to a precomputed schedule. An analysis was performed comparing OPIS-2's selected actions to the random selection of actions. The performance criteria compared were *Schedule Quality*: change in tardy orders, change in tardy time, change in completion time (WIP), and *Schedule Disruption*: number of order schedules changed, number of resource schedules changed, average time change per rescheduled operation. In 60% of the case, OPIS-2 produced the best results, in 88% of the cases, it produced with 0.07 of the best.

8. CORTES: Micro-Opportunistic, Constraint Guided Scheduling [7,13,14]

The CORTES system [14] represents the next step in the evolution of the ISIS/OPIS family of scheduling systems. The ISIS systems are primarily order centered scheduling systems, while the OPIS systems dynamically switch between being order and resource centered. The CORTES system takes an activity centered view of scheduling where activities are chosen opportunistically to schedule.

In an activity-based approach, each activity is treated as an aggregate variable, or decision point, that comprises the activity's start time, its resources, and possibly its duration. The schedule is built incrementally by iteratively selecting an activity to be scheduled and a reservation for that activity (i.e., start time, resources and possibly duration). Every time a new activity is scheduled, new constraints are added to the initial scheduling problem, and propagated. If an inconsistency is detected during propagation, the system backtracks. The process stops either when all activities have been successfully scheduled or when all possible alternatives have been tried without success.

In order to reduce the combinatorics of backtracking based search, it is necessary to have strong knowledge of where to focus one's attention. We achieve this by defining a set of heuristics that we call problem space textures [7,13] which are used to identify features of the problem space that differentiate one state from another.

Scheduling is performed according to the following steps.

Step 1. A problem space topology is constructed where each activity for each order is represented as a state. States are constrained by the temporal relations that exist between activities and resources. Preferences for the start times of activities are specified as utilities over a temporal interval.

Step 2. Constraint propagation is performed. Propagation of temporal constraints in the presence of temporal preferences is performed as described in [13]. The result is a function that defines the preferences for an activity's start time.

Step 3. Compute texture measures. Texture measures such as looseness, contention, and elasticity [7] are computed for the states in the constraint network. Texture measures are used to focus search so that backtracking is minimized. For example, the contention measure is used to identify temporal intervals during which a resource is a bottleneck. It is during this interval that the scheduler focuses on developing a schedule for the activities incident at the resource.

Step 4. A state is selected based upon the texture measures. For example, an activity that demands a large fraction of a highly contended for resource and is of high priority may be selected.

Step 5. Select rule. A rule is chosen which may assign a value to a state's variable, or add a new state or constraint to the network. In this case, a resource is assigned to an activity over some time interval. The system then iterates from Step 2.

The activity based approach to scheduling provides greater flexibility at the cost of greater search complexity. Our initial experiments [14] have shown that both state and value ordering texture measures can generate quality solutions with low search complexity.

9. Conclusion

This paper chronicles our approach in using constraint-guided search to solve the job shop scheduling problem. The ISIS systems explored a hierarchical architecture where successive levels constrained decisions at levels below them. The OPIS systems explored both multiple perspective and opportunistic scheduling. CORTES explores ac-

tivity based scheduling with probabilistic constraint satisfaction. Each represents a more sophisticated and powerful approach whose performance is demonstrably better than other approaches. Though the results are very good, our exploration continues!

Acknowledgements

The work reported here are the results of many of may colleagues at CMU. Steve Smith and Peng Si Ow have been major contributors to both ISIS and OPIS. Norman Sadeh has been a major contributor to CORTES.

This research has been supported by numerous agencies and companies including, the Air Force Office of Scientific Research under contract F49620-82-K0017, the Defence Advance Projects Agency under contract #F30602-88-C-0001, Westinghouse Electric Corporation, International Business Machines, Schlumberger Co., Boeing Computer Services Co., and McDonnell Douglas Corporation.

References

[1] L.D. Erman, F. Hayes-Roth, V.R. Lesser and D. Raj Reddy, "The Hearsay-II speech understanding system: integrating knowledge to resolve uncertainty", *ACM Comput. Surv.*, Vol. 12, 1980, pp. 213–253.

[2] M.S. Fox, "Constraint-directed search: a case study of job-shop scheduling", Technical Report CMU-RI-TR-85-7, Intelligent Systems Laboratory, The Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, 1983.

[3] M.S. Fox, *Constraint-Directed Search: A Case Study of Job-Shop Scheduling*, Morgan Kaufmann, Los Altos, CA, 1987.

[4] M.S. Fox and S. Smith, "ISIS: a knowledge-based system for factory scheduling", *Int. J. Expert Syst.*, Vol. 1, No. 1, 1984, pp. 25–49.

[5] M.S. Fox and S.F. Smith, "The role of intelligence reactive processing in production management", in: *Proc. CAM-I's 13th Annual Meeting and Tech. Conf.*, 1984.

[6] M.S. Fox, B. Allen and G. Strohm, "Job-shop scheduling: an investigation in constraint-directed scheduling", in: *Proc. Natl. Conf. on Artificial Intelligence*, William Kaufmann, Los Altos, CA, 1982, pp. 155–158.

[7] M.S. Fox, N. Sadeh and C. Baykan, "Constrained heuristic search", in: *Proc. 11th Int. Joint Conf. on Artificial Intelligence*, 1989.

[8] C. Le Pape and S.F. Smith, "Management of temporal constraints for factory scheduling", Technical Report, The Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, 1987; also in: C. Rolland, F. Bodart and M. Leonard (eds.), *Temporal Aspects in Information Systems*, (Proc. IFIP TC 8/WG 8.1 Working Conf., Sophia-Antipolis, France, May 1987). North-Holland, Amsterdam, 1988.

[9] N. Muscettola and S. Smith, "A probabilistic framework for resource-constrained multi-agent planning", in: *Proc. 10th Int. Conf. on Artificial Intelligence*, 1987, pp. 1063–1066.

[10] A. Newell and H.A. Simon, "Computer sciences as empirical inquiry: symbols and search", *Commun. ACM*, Vol. 19, No. 3, 1976, pp. 113–126.

[11] P.S Ow and T.E. Morton, "The single machine early/tardy problem", *Manage Sci.*, Vol. 35, No. 2, 1989, pp. 177–191.

[12] P.S. Ow, S.F. Smith and A. Thiriez, "Reactive plan revision", in: *Proc. 7th Natl. Conf. on Artificial Intelligence*, 1988.

[13] N. Sadeh and M.S. Fox, "Preference propagation in temporal constraint graphs", Technical Report, Intelligent Systems Laboratory, The Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, 1988.

[14] N. Sadeh and M.S. Fox, "Focus of attention in an activity-based scheduler", in: *Proc. NASA Conf. on Space Telerobotics*, NASA, 1989.

[15] S.F. Smith, "Exploiting temporal knowledge to organize constraints", Technical Report, Intelligent Systems Laboratory, The Robotics Institute, Carnegie-Mellon University, Pittsburgh, PA, 1983.

[16] S.F. Smith and P.S. Ow, "The use of multiple problem decompositions in time constrained planning tasks", in: *Proc. Int. Joint Conf. on Artificial Intelligence*, William Kaufmann, Los Altos, CA, 1985.

[17] S. Smith, M.S. Fox and P.S. Ow, "Constructing and maintaining detailed production plans: investigations into the development of knowledge-based factory scheduling systems", *AI Mag.*, Vol. 7, No. 4, Fall 1986, pp. 45–61.

[18] S.F. Smith, P.S. Ow, C. Lepape, B. McLaren and N. Muscettola, "Integrating multiple scheduling perspectives to generate detailed production plans", in: *Proc. 1986 SME Conf. on AI in Manufacturing*", 1986, pp. 123–137.