Stephen F. Smith, Mark S. Fox and Peng Si Ow

Constructing and Maintaining Detailed Production Plans: Investigations into the Development of Knowledge-Based Factory Scheduling Systems

Introduction

One of the major deterrents to productivity in industry today is the inability to effectively manage and control production. The problem is particularly acute in job shop environments where plant operation is routinely characterized by high work-in-process (WIP) inventories, tardy orders, poor resource utilization, and other shop floor inefficiencies. Perhaps the single most significant obstacle to improved factory performance is the complexity associated with constructing and maintaining good production schedules. Good schedules must reflect both the full detail of the operating environment and the influence of a conflicting set of preferences that range from global organizational objectives to specific operational idiosyncrasies. Existing computer-based techniques for production scheduling are capable of incorporating only a small fraction of this scheduling knowledge and, as a result, typically produce schedules that bear little resemblance to the actual state of the factory. At best, these scheduling techniques provide the plant scheduler or shop floor supervisors with high-level guidelines for developing detailed schedules. These guidelines, however, do little to reduce the complexity associated with the detailed scheduling decisions that must be made.

For the past several years, we have been engaged in work aimed at providing a better solution to the job shop scheduling problem. Viewing the problem as a complex constraint-directed activity, we sought to employ knowledge-based techniques to represent and operationalize the constraint knowledge utilized in actual factory environments. This led to the construction of the Intelligent Scheduling and Information System (ISIS), a series of experimental job shop scheduling systems (Fox, Allen, and Strohm 1982; Fox 1983; Fox and Smith 1984a, 1984b). ISIS-2 was demonstrated in the context of the Westinghouse Turbine Components Plant (WTCP) in Winston-Salem, North Carolina. More recently, we initiated work on a new system called OPIS (Smith and Ow 1985; Ow and Smith 1986; Smith et al. 1986), which continues to generalize from the ISIS experience and provide greater system flexibility in approaching various scheduling tasks. We feel we have made significant progress in this work, but there are difficult issues that remain to be addressed. This article attempts to coalesce what we have learned thus far and to reflect on the potential of knowledge-based approaches to this difficult problem.

Stephen F. Smith is a research associate at The Robotics Institute, Carnegie-Mellon University, Pittsburgh, Pennsylvania 15213 Mark S. Fox is the director of the Intelligence Systems Laboratory, The Robotics Institute, Carnegie-Mellon University; and Peng Si Ow is a visiting assistant professor in the Graduate School of Industrial Administration at Carnegie-Mellon University

Abstract To be useful in practice, a factory production schedule must reflect the influence of a large and conflicting set of requirements, objectives and preferences. Human schedulers are typically overburdened by the complexity of this task, and conventional computer-based scheduling systems consider only a small fraction of the relevant knowledge. This article describes research aimed at providing a framework in which all relevant scheduling knowledge can be given consideration during schedule generation and revision. Factory scheduling is cast as a complex constraint-directed activity, driven by a rich symbolic model of the factory environment in which various influencing factors are formalized as constraints. A variety of constraint-directed inference techniques are defined with respect to this model to provide a basis for intelligently compromising among conflicting concerns. Two knowledge-based factory scheduling systems that implement aspects of this approach are described.

The Factory Scheduling Problem

In broad terms, the factory scheduling problem concerns the allocation over time of a finite set of resources to specific manufacturing operations such that the orders for parts received by the factory are produced in a timely and cost-effective fashion. The production of a given order typically involves the execution of a sequence of operations, each of which possesses a specific and, for the most part, a distinct set of resource requirements. In more detailed terms then, the scheduling problem consists of (1) the determination of an appropriate sequence of operations, or *process routing,* for each order and (2) the assignment of required resources and time intervals to the operations selected.

To some extent, these two aspects of the problem are separable, and historically, this has been the case in most manufacturing organizations. Process-routing selection is viewed as a planning task that is carried out during part design, and the allocation of resources to particular orders over time is viewed as the role of the scheduler. In actuality, however, there is much greater interplay between these seemingly distinct functions. There are often several ways in which a given part can be produced (for example, alternative machines or production processes can be utilized) and although a particular routing might be designated as preferred, an a priori commitment to it ignores the dynamic nature of the actual factory floor. The feasibility of a given operation depends on the availability of its required resources, and consequently, many process-selection decisions cannot be intelligently made without consideration of the current status of the factory (for example, current order mix, current resource levels, and so on.)

The problem is further complicated by the unpredictability inherent in factory operation. Machines break down, in-process orders fail to pass intermediate quality control inspections, engineering changes are introduced, operators call in sick, and so on, all of which quickly force changes to previously planned activities. As uncertainty in the performance of activities on the shop floor increases, the usefulness of precise schedules decreases. The precision of schedules produced by a scheduler must be determined by the uncertainty of the information used in making the decision. Thus, we can identify two general goals in approaching the factory scheduling problem:

1. An ability to effectively predict shop behavior through the generation of schedules that accurately reflect the full detail of the environment and the stated objectives of the organization
2. An ability to reactively revise and maintain the schedule in response to changing shop conditions

These activities are not viewed as distinct; rather it is felt that the same types of knowledge and methods are relevant to both. From a system engineering perspective, however, it is important to provide a system organization with the flexibility to selectively focus on specific aspects of the current schedule.

The job shop scheduling problem in many idealized forms is known to be NP-hard (belonging to a class of inherently intractable problems) (Garey and Johnson 1979). The situation in real-world scheduling environments is considerably more complex. Much of the complexity stems from the need to attend to a large and diverse set of objectives, requirements, and preferences that originate from many different sources in the plant. These scheduling influences are often in direct conflict with one another, wherein lies the crux of the problem. The production schedule must reflect a satisfactory compromise with respect to these competing influences.

Scheduling Constraints

We can partition the range of factors that influence factory scheduling decisions into two broad classes of constraints: (1) scheduling restrictions that serve to delineate the space of possibilities in developing a schedule and (2) scheduling preferences that provide a basis for differentiation among possible choices. This distinction is useful to make at the outset because existing computer-based scheduling systems typically give limited attention to both, and each offers distinct opportunities for improving the quality of the schedules generated.

Scheduling restrictions constrain the alternatives that can be considered in selecting and ordering operations, binding resources to operations, and designating temporal intervals during which selected operations are to take place. Collectively, these restrictions serve to define the space of admissible schedules that the scheduler must search. Scheduling restrictions include the following:

- Causal Restrictions: Typically, there are precedence constraints associated with the operations that must be performed to produce a given part, restricting the manner in which orders for this part can be routed through the factory. A precedence constraint on an operation states that another operation must be performed before (or after) it. These causal relationships between operations are further qualified by interoperation travel times, transfer quantity sizes (that is, the portion of the order which must be completed before any subsequent operations can be initiated), maximum allowable times between operations, and so on. In addition, each individual operation possesses a well-defined set of resource requirements that must be satisfied for specific periods of time either before or during the execution of the operation. For example, a milling operation might require the possession of a milling machine, an operator with the necessary skills, specific tools and fixtures, an appropriate NC tape (if the machine is a numerically controlled or NC machine), and so on.

- Physical Constraints: Each machine in a job shop has specific capabilities that restrict the types of operations which can be performed on it. For example, the size of a machine's work bed might prohibit its use for operations on a particular class of parts. Likewise, each machine has particular operating characteristics (for example, cutting speed, setup procedures) which limit the amount of work that can be performed over a certain period of time. Generally speaking, physical constraints define the functional limitations of specific resources in the factory.
- Resource Unavailability: There are also dynamic restrictions on the availability of resources that limit the scheduling alternatives available. Here, we are speaking of events such as machine breakdowns that occur asynchronously and are outside the control of the scheduler.

An understanding of the full range of scheduling alternatives is essential to the development of a realistic model of the job shop environment. The simplifications introduced in most existing computer-based scheduling systems (for example, the designation of a single routing for each part) reduce the flexibility with which the system can respond to different scheduling problems, which results in a divergence of the schedules generated from the actual situation. At the same time, however, any attempt to embrace the full complexity of the environment requires the ability to explicitly represent and reason with the imposed scheduling restrictions during the generation of candidate schedules.

In many problem domains addressed within the field of artificial intelligence (AI), the restrictions imposed by the problem constrain the set of admissible solutions to the extent that least commitment and constraint-propagation techniques are sufficient to converge on an acceptable result (Stefik 1981; Sussman and Steele 1980; Waltz 1975). This is not the case in the factory scheduling domain. Adherence to the scheduling restrictions identified earlier still leaves the problem severely underconstrained, and knowledge of various preferential concerns must be considered to focus the scheduler on good solutions. These scheduling preferences fall into several categories:

- Organizational Goals: All manufacturing facilities are driven by a set of organizational goals. These goals reflect global concerns and objectives with respect to the operation of the factory and imply general criteria against which prospective schedules can be compared. Organizational goals are established along several distinct performance dimensions:
  - *Meeting due dates:* A major concern of a factory is meeting the customer due dates that are established as orders are received. A late order affects customer satisfaction and the likelihood of future business.
  - *Minimizing WIP time:* WIP inventory represents a substantial investment in raw materials and added value. Because these costs are not recoverable until delivery of the final product, minimizing WIP time is an important goal.
  - *Maximizing resource utilization:* Maximizing the amount of time that critical machines in the shop are actually operating (as opposed to being prepared for operation or standing idle waiting for parts) can greatly increase the overall throughput of the plant. Also, typically there are fixed costs associated with maintaining and operating the machines in a factory that can be minimized if resources are used efficiently.
  - *Maintaining shop stability:* The concern here is minimizing the amount of disruption to shop operations caused by revisions to the schedule. Last-minute changes to the schedule can lead to increased periods of machine idle time as the preparation (or machine setup) performed in anticipation of the previously scheduled operations is undone and as preparation for the newly scheduled operations is carried out.

  These performance concerns can all be viewed as approximating the overall concern of the organization: a desire to make scheduling decisions that maximize profits. These concerns are addressed as part of the organizational planning process and lead to the establishment of specific operating expectations. For example, production levels are designated for various areas in the plant, a forecast of the number of work shifts that will be run in each area is made, and preliminary resource maintenance schedules are developed. These preferences all influence the shop schedule that is subsequently developed.
- Operational Preferences: These constraints express preferred choices among alternatives at the level of individual scheduling decisions (that is, the selection of specific operations, resources, and time intervals) and reflect the heuristic knowledge present in a given scheduling environment. In many cases, these preferences provide a tactical basis for accomplishing specific global objectives. For example, an ability to effectively exploit order-sequencing preferences to minimize the amount of time spent setting machines up for operations contributes directly to maximizing resource utilization. In other cases, however, such knowledge reflects an understanding of the operational characteristics of the plant that cannot be captured in predictive estimates of how well various scheduling objectives have been met; for example, a decision to avoid a particular machine when possible, based on knowledge of its susceptibility to breakdowns, cannot be properly assessed until the schedule is actually executed on the shop floor. Such decisions can in fact have detrimental effects with respect to predictive measures of schedule quality.
- Resource Unavailability: In contrast to the resource unavailability restrictions mentioned ealier, resource un-

availability preferences refer to constraints that are introduced by the scheduler. As resources are allocated to specific operations during generation or revision of the schedule, constraints declaring the resources unavailable during the allocated time periods must be generated. Such decisions must be viewed as preferences, because they can later be retracted in the face of other overriding factors (for example, the receipt of a high-priority order).

It is clear from this discussion that an effective solution to the real-world factory scheduling problem requires an ability to reason intelligently with an amalgam of diverse constraints. Our initial discussions with plant schedulers at WTCP underscored this point. It was found that schedules were not developed in any uniform fashion but rather through an iterative process of distributing a proposed schedule to various departments in the plant, collecting additional constraints, and attempting to alter the schedule accordingly. However, although it was clear that scheduling decisions were based on the full range of factors identified here, the lack of a methodology for balancing these concerns consistently led to the generation of schedules that resulted in less than satisfactory factory performance. The schedulers were simply overburdened by the complexity of the task.

Related Research

Scheduling research to date has had relatively little impact on the real-world factory scheduling problem. Operations management (OM) research has long been concerned with the scheduling problem but in relation to two rather restrictive perspectives. The first approach, centered around a desire to obtain optimal results, has sought to formulate mathematical models of the problem that are tractable by linear-programming techniques. Attention has been focused on simplified scheduling problems (for example, the single machine case) that unfortunately have little in common with actual factory environments. A second branch of OM research has been concerned with the development of priority decision rules to provide a heuristic basis for order sequencing. These rules, although useful in making local dispatching decisions, are typically responsive to specific types of concerns (for example, meeting due dates), ignoring all others. This restricted emphasis limits their effectiveness in global (that is, plantwide) decision-making contexts.

In recent years, AI researchers in planning have also turned their attention to scheduling issues. Recognizing the limitations of reasoning with implicit notions of time, several researchers have focused on extending existing planning paradigms to include the assignment of time intervals to activities. Vere (1981) describes a technique used to schedule activities aboard a spacecraft which associates time windows and durations with the various activities in a plan and which propagates refinements to this temporal information as the plan crystallizes. A similar approach is adopted in Fukumori (1980) for generating train schedules. Others have sought to reformulate the planning process within an explicit temporal framework (Allen 1981; Allen and Koomen 1983; McDermott 1982). Expansion of the planning problem to explicitly address temporal concerns has also necessitated the establishment of criteria for differentiating between alternative plans. Overall plan duration has been the most common consideration in AI scheduling systems. However, activity cost estimates are included in Daniel (1984), and a scheduling framework described in Miller (1983) proposes the use of special-purpose critics to detect specific undesirable characteristics (for example, deadline violations). In relating these efforts to the factory scheduling problem, the chief point of divergence is the absence of conflicting constraints. This is due in large part to the emphasis on planning the activities of a single agent and the consequential lack of emphasis on efficient allocation of shared resources (that is, resource availability is the sole concern). One exception is the NUDGE system (Goldstein and Roberts 1977), which compromises between the conflicting preferences of distinct individuals in producing a schedule for a given individual's weekly activities and appointments.

Scheduling as a Constraint-Driven Search Process

Our approach to the factory scheduling problem has been predicated on the assumption that the key to successfully managing complexity lies in appropriate use of knowledge about the constraints of the specific operating environment. This view raises three broad issues. The first concerns the development of an appropriate framework for structuring this knowledge. We have seen from the WTCP scheduling example that knowledge of constraints is accumulated in an unstructured, asynchronous fashion by polling different parts of the organization. A factory model that makes explicit the scheduling restrictions in the environment and appropriately organizes knowledge about preferential concerns can, in itself, provide a sufficient amount of leverage. The second issue concerns an ability to differentiate amongst alternative sets of scheduling decisions. It is clear that development of good schedules requires exploration of alternatives, and different alternatives variably attend to various preferential concerns. Knowledge and mechanisms for estimating how well a given set of decisions satisfies the constraints must be identified. The third issue concerns control of the search process: Only a very small portion of the underlying solution space can feasibly be explored. Techniques for exploiting knowledge about constraints and their interdependencies to intelligently control the search must be defined.

These three issues all relate to the representation and use of knowledge about scheduling constraints. In this section we address these issues, presenting what we believe to be the fundamental ideas that underlie our approach to factory

scheduling. In sections following, we turn our attention to the software systems that have arisen out of this work.

Modeling of the Factory Environment

A fundamental prerequisite to effective scheduling is an accurate model of the production environment. The model provides a framework for representing and organizing the constraint knowledge in the operating environment and imposes structure that can be exploited in the development of schedules. Our approach to modeling the production environment draws on frame-based representation techniques and emphasizes the definition of a set of structural and relational primitives for describing manufacturing organizations. Kernel descriptions provide the basic concepts of states, objects, and activities and a set of temporal and causal relations for describing their interactions. Higher-level descriptions refine these general semantic primitives into concepts germane to the scheduling environment. For example, in specifying the set of process routings associated with a particular product, manufacturing operations are defined as activities, and precedence constraints are composed of basic temporal and causal relations. The resources required by specific operations are expressed as objects, with their allocation represented as a collection of possession states spanning particular intervals of time. Individual components of the model are linked to composite entities (for example, machines are aggregated into work areas, detailed operations are aggregated into abstract operations, and so on) to provide multiple levels of description.

To a large extent, the scheduling restrictions present in the factory environment and hence the set of alternatives relevant to specific scheduling decisions are directly reflected in the resulting model. This provides a structural framework for organizing the preferential concerns that influence various choices. This knowledge is encoded within a general constraint representation, and specific instances are attached directly to the relations-attributes in the model that they constrain. This is illustrated in figure 1, which graphically displays a portion of a factory model centered around the description of a particular machining operation called "P1 root grinding." The constraint representation formalizes the sources of heuristic knowledge found to be useful in developing constraint-satisfying schedules. The specific types of knowledge included are identified in the following subsections. Detailed accounts of this approach to modeling the factory environment and its constraints, as well as its implementation within Schema Representation Language (SRL), a frame-based, knowledge representation system (Wright and Fox 1983), can be found in Fox 1983; Fox and Smith 1984a; Smith 1983; Sathi, Fox, and Greenberg 1985.

Relaxable Constraints

The constraint representation extends the factory model to include the notion of relaxable constraints and provides an explicit basis for injecting preference knowledge into the reasoning process. Scheduling is a synthesis task in which the total set of alternatives cannot be enumerated. The role of constraints must therefore extend beyond the concept of "winnowing" an enumerated set in order to reduce the combinatorics of the search process. It is necessary for constraints to play a greater role in the generation of alternatives. Rather than create rules that embody this knowledge in an ad hoc form, we took the approach of expanding the semantics of the constraint representation to include specification of the following:

- Alternatives: Acceptable alternatives for satisfying the constraint are made explicit. For example, a due date constraint would not only specify the actual date but also a continuum of acceptable dates possibly spanning a four-week period.
- Utility: Each alternative might not be equally acceptable. The utility specifies the degree of acceptability of each alternative. In the example of a due date, the utility function could be normal in form, with the maximum at the actual due date and the tails at the earliest and latest alternatives.
- Elasticity: In the specification of a shift constraint, we found that even though running more than one shift might be acceptable, it was not easy to implement the alternative. Consequently, a measure of a constraint's elasticity is required as a measure of "ease of change." This can then be used to determine which constraints should be relaxed and which should be held constant during the search process.

This extended representation of relaxable constraints enables the creation of general search operators that interpret constraints. In particular, knowledge of relaxation supports reasoning about the complexity of the search space, enabling the search to be bounded by restricting the set of generated alternatives.

Quantifying the Notion of Constraint Satisfaction

We have already observed that exploration of alternatives is an integral part of generating constraint-satisfying schedules. For the most part, the alternatives we speak of are partial solutions, subsets of the total set of scheduling decisions that make up the factory schedule. Candidate partial solutions under consideration might, for example, represent alternative sets of decisions that could be taken with respect to a particular order. This emphasis on the exploration of alternative partial solutions is due to the combinatorics of the underlying search space, which makes it necessary to focus incrementally on specific aspects of the schedule and make commitments prior to seeing a complete schedule. We use the terms hypothesis, solution component, and partial schedule interchangeably to refer to a candidate partial solution.

Central to the search that must be undertaken, of course, is a means of evaluating partial schedules; this is the issue we

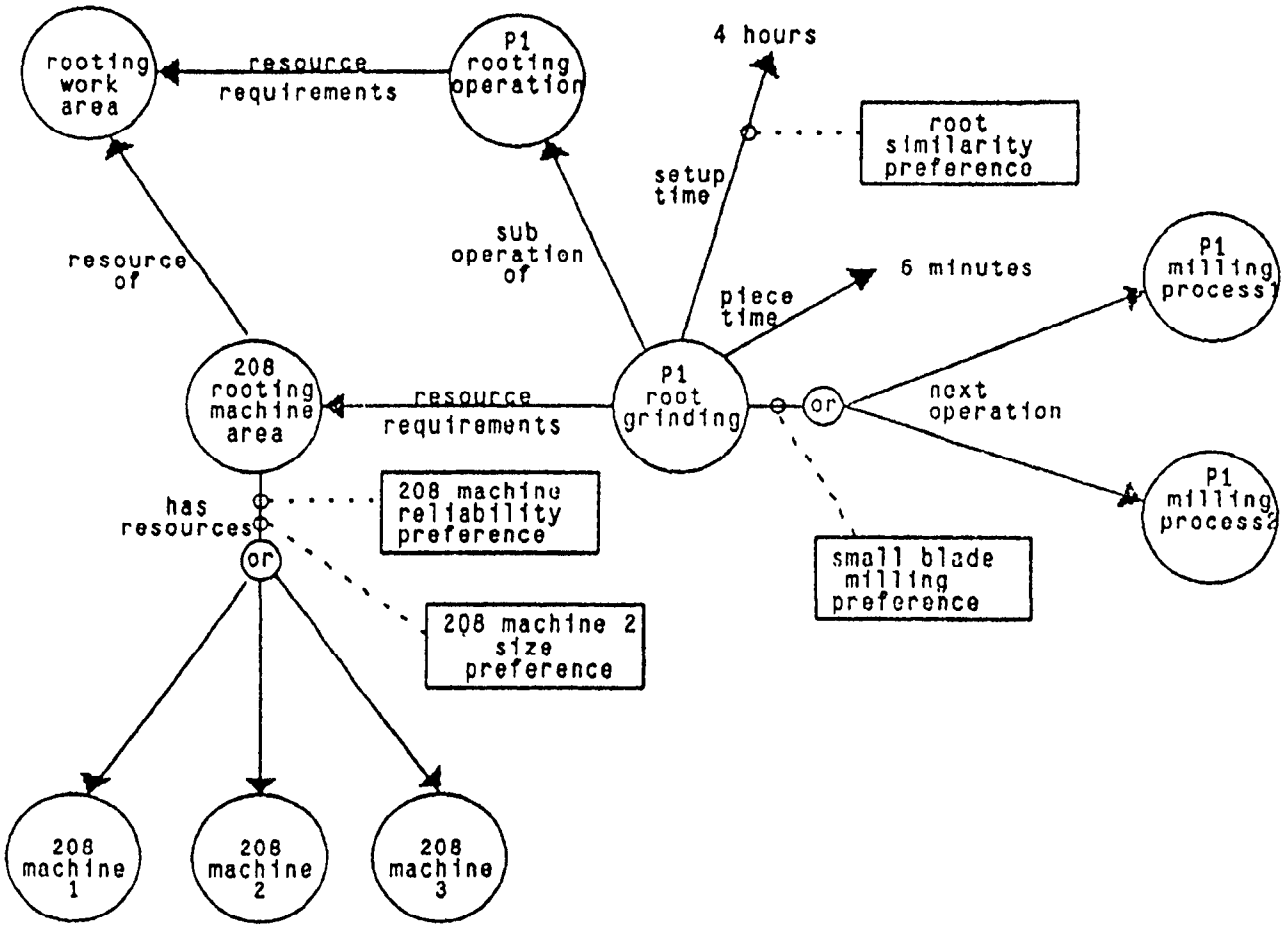


Figure 1. A Portion of a Factory Model with Attached Constraints.

address in this section. Given our view of the scheduling problem, we would like an evaluation of a specific partial schedule to intuitively reflect how well its constituent scheduling decisions attend to, or satisfy, the various problem constraints. Operationally, such an evaluation scheme requires constraint knowledge relating to three basic notions:

- Relevance: Individual constraints are typically relevant to specific types of scheduling decisions. For example, a constraint that expresses a preference for a given machine under certain circumstances should only be considered in the context of scheduling decisions which involve the potential allocation of the machine. Also, different shift constraints might apply at different times of the year, for example, holidays.

- Degree of Satisfaction: Typically, there are preference relationships among the set of choices that exist with respect to a given constraint. Specific choices compromise (or relax) the constraint to various extents. These relationships can be made explicit by attributing a specific utility (that is, a degree-of-satisfaction estimate) to each alternative.

- Importance: Different constraints do not typically exert the same influence on the final schedule. In cases where all constraints cannot be satisfied, some are more important to satisfy than others, and knowledge of these importance relationships must be taken into account.

Our approach to representing and using this knowledge is sketched in the following paragraphs.*

Determination of Relevant Constraints. As indicated earlier, a given constraint is typically relevant to a particular class of scheduling decisions. Thus, the first step in evaluating a given hypothesis must be to identify the set of constraints that should participate in the evaluation. The under-

* An alternative approach to evaluation is to focus on the costs associated with relaxing various constraints and construct evaluations that estimate the total cost of partial schedules Cost is, in fact, the bottom line in differentiating between candidate schedules. However, our experience has found actual costs to be difficult to quantify in many cases The use of importance and degree-of-satisfaction estimates appears to provide a more tractable common denominator in considering the influence of multiple constraints

lying factory model outlined in Modeling of the Factory Environment, which emphasizes the direct attachment of constraints to the objects they constrain, provides considerable guidance in this task. A given hypothesis to be evaluated designates specific objects in the model (that is, the scheduling decisions composing the hypothesis concern specific operations, resources, orders, and so on), and all potentially relevant constraints can be located by examining these objects. We say the constraints collected in this manner are potentially relevant because there might be further conditions on their applicability. These conditions might specify particular attributes of the hypothesis (for example, the relevance of a machine preference might depend on dimensional characteristics of the product being scheduled) or identify specific types of scheduling decisions of interest (for example, an order's due date and WIP constraints are relevant to any decision that commits to a particular interval of time). Each individual constraint possesses a specific applicability predicate that is used as a final filter in determining relevancy.

Producing Constraint-Satisfaction Estimates. Our approach to hypothesis evaluation centers around the notion of each relevant constraint imparting a utility indicative of how well it is satisfied by the hypothesis. These individual "ratings" are then combined to produce an overall measure of constraint satisfaction. Utility values are defined to range from zero to one, with zero interpreted as complete dissatisfaction and one interpreted as complete satisfaction.

The expression of preference relationships among possible choices, as well as methods for attributing specific utility values, varies according to the type of constraint involved. Given the diversity in the types of constraints encountered, this variation is not surprising. This diversity is accommodated in the model by defining a taxonomy of constraint types, each of which provides an appropriate representation and associated methods for interpreting it. In many cases, a constraint expresses preference relationships over a set of choices that are explicitly defined in the model. For example, operations are defined to take place in certain work areas of the factory, and machine preferences promote specific choices from the sets of machines in these areas. Specification of the preference relationships in these cases entails an association of specific utility values with each of the defined alternatives.

Many organizational goals, however, express preferences over a continuous and often infinite range of possible values. A due date constraint, for example, must associate a degree of satisfaction with each point along the time line. Constraints of this nature require an implicit mapping of degree of satisfaction to possible alternatives. This mapping is accomplished by defining a characteristic function that, when evaluated with respect to a particular alternative yields a specific utility value. This approach to specifying the preference relationships amongst alternative relaxations (choices) is not unlike the techniques employed in mathematical programming and, in fact, allows advantage to be taken of OM heuristic priority rules that emphasize specific organizational goals.

Combining Estimates. The utilities assigned by the constraints relevant to a particular hypothesis must be combined to produce its final evaluation. Here, it is necessary to consider the relative amount of influence that should be exerted by each contributing constraint. This entails a specification of weights for the utilities assigned. The weights are derived from both absolute and relative importance knowledge. Absolute importance measures, which are predefined and associated with the constraints themselves, provide a basis for expressing the importance relationships that exist among constraints of a given type (for example, machine preferences). Importance relationships among different types of constraints are typically dynamic in nature. For example, satisfying a due date constraint is likely to be a much more important concern than satisfying the set of relevant resource preferences in the context of a high-priority order, but the opposite might be true in the case of an order generated to build inventory. Such relationships are established through the introduction of scheduling policies, which partition constraint types into distinct importance classes and which associate with each partition a fraction of the total importance to be distributed among the constraints belonging to the partition. This fraction is distributed in proportion to the absolute importance measures associated with these constraints. The weights derived are restricted to values between zero and one, such that the total set of weights sums to one. The final evaluation of a given hypothesis then is simply the weighted sum of the utilities assigned by the relevant constraints.

This evaluation scheme provides a framework for selective evaluation of the evolving solution at different levels of aggregation. For example, a subtask concerned with the generation of a particular solution component, such as decisions relating to a particular order, can be driven by constraint-satisfaction assessments local to that portion of the solution. As sets of scheduling decisions are combined to form more encompassing partial solutions (for example, as schedules for individual orders are integrated into a shop schedule), corresponding aggregate measures of constraint satisfaction can be produced by appropriately merging previously collected sets of constraints. The scheme also provides a means for measuring the extent to which specific influences have been attended to at different levels of aggregation. With respect to meeting deadlines, for example, measures reflecting "how well order 10's due date constraint is satisfied," "how well the end-time constraints in the milling work area are satisfied," and "how well the due date constraints of orders of priority class x are satisfied" are all obtainable by focusing on different cross sections of the previously determined set of relevant constraints. These distinct levels of evaluation

can all provide useful information upon which to base search-control decisions.

Controlling the Search for a Constraint-Satisfying Schedule

Perhaps the most difficult questions concern how to carry out the search for a constraint-satisfying schedule. The scheduling restrictions defined in the factory model specify the range of admissible scheduling alternatives and provide the basis for deriving a basic set of operators for conducting a search. For example, causal restrictions provide the grist for generating alternative operation and resource selections. Likewise, work shift alternatives, resource availability restrictions, and various physical constraints (for example, machine cutting speed) combine to provide a basis for generating alternative intervals of time during which resources can be allocated to specific operations. However, the ability to define such operators in and of itself provides little leverage in achieving an effective compromise with respect to the domain's conflicting preferential concerns. To be sure, the application of these operators results in the generation and extension of hypotheses that variably satisfy various scheduling preferences, and a comparison of alternatives can be made on this basis. Each local commitment made during the search, however, has unforeseen global consequences on the final solution that cannot be accurately predicted short of an exhaustive search of all possibilities. Indeed, it is these highly nonlinear interactions amongst preferential concerns that effectively preclude hope of obtaining an optimal global compromise. Thus, the goal is to heuristically impose enough structure on the search process to make the generation of good solutions tractable. Wc argue that knowledge of constraint interactions can be exploited to this end.

In stating a desire to heuristically structure the search, we speak of controlling the generation of alternative partial schedules. It is necessary to make commitments regarding specific scheduling decisions early on in the search and, for the most part, assume that these decisions will be part of the final schedule generated. Of course, there might be problems with these commitments that only become apparent later in the search, but even in these circumstances it is desirable (if not computationally necessary) to retain as much of the current partial schedule as possible. Thus, the control problem is largely one of determining the order in which various commitments should be made (that is, what subset of decisions should we focus on first, and so on). Because each decision that is made affects the extent to which specific preferential concerns can be satisfied, the control problem can be equivalently viewed as one of determining which constraints should be compromised (or relaxed).

In considering these issues, it is useful to define the notion of a conflict. We use the term broadly to refer to a situation in which scheduling decisions do not exist that mutually satisfy a given set of constraints. Resolution of a conflict can be thought of as reaching a compromise amongst the constraints involved in the conflict. Much of what we say in the following subsections is concerned with seeking out important conflicts (or types of conflicts) to resolve and reacting to unsatisfactory compromises that become apparent. An unsatisfactory compromise can be the result of earlier control decisions to deemphasize particular conflicts, or it can be the result of changing circumstances in the factory environment.

Conflict-Directed Problem Decomposition. To provide a framework for controlling the search, it is necessary to specify a set of control alternatives. We have identified several levels at which such alternatives exist:

- Scheduling Perspective Level: There are distinct scheduling perspectives that might be adopted as a means of decomposing the overall scheduling task into more tractable subtasks. For example, an order-based perspective views the shop schedule as a collection of order schedules and restricts attention at any point in time to the development of a particular order's schedule. Alternately, a resource-based perspective views the shop schedule as a collection of resource schedules (for example, work area or machine schedules) and restricts system attention at any point in time to the development of a particular resource's schedule. Each perspective advocates a specific local and incomplete view of the overall problem that is appropriate for resolving specific types of conflicts.
- Strategy Level: There are alternative scheduling strategies that might be adopted when operating within a given scheduling perspective, each varying in the types of constraints that are considered. For example, consider the reactive adjustment necessary to the current schedule in response to an external indication that a particular machine has broken down. Reaction could vary in scope from a simple reservation shifting strategy where no preferential concerns are considered to an extensive rescheduling of the encompassing work area where all relevant preferences are addressed.
- Tactics Level: Although a given strategy defines a specific range of conflicts that are attended to, it need not predetermine the specific sequence of actions to be taken in exploring alternative compromises. Determination of which scheduling decisions to consider next might instead be a function of the current-problem solving state.

The extent to which alternatives are defined at any level reflects the trade-offs that exist in approaching a given scheduling task. We consider the nature of these trade-offs in the following paragraphs, along with the types of knowledge about constraint interactions that provide a basis for these control decisions.

Decisions about how to decompose the scheduling task into subtasks affect the system's ability to resolve conflicts. To optimally resolve a given conflict (that is, to achieve the best possible compromise), it must be possible to simultaneously consider all of the constraints involved in the conflict. The subtasks that result from adopting different scheduling perspectives isolate different sets of constraints and hence emphasize different types of conflicts. An order-based decomposition groups together the constraints that surround a particular order and provides the opportunity to effectively resolve order-centered conflicts (for example, conflicts involving a WIP constraint and operation preferences). A resource-based decomposition isolates the constraints that surround the allocation of a particular resource (for example, order-sequencing preferences, and promotes the resolution of a different class of conflicts).

Given the implications of decomposing the problem in different ways, the choice of scheduling perspective requires knowledge about which conflicts are most important to resolve. In reactive contexts, where specific problems have been introduced by unanticipated external events, there is often a direct mapping to the conflicts of interest. For example, a machine breakdown clearly suggests a focus on the set of conflicts surrounding this resource. Similarly, an indication from quality inspection that a particular order requires rework suggests a specific set of order-centered conflicts. Alternately, in predictive contexts where substantial portions of the shop schedule are being generated, both resource-centered and order-centered conflicts are likely to be of concern. Here, it is advantageous to selectively address different aspects of the problem from different perspectives. One useful heuristic in this regard is the observation that the most important resource-centered conflicts are those which surround the bottleneck resources in the shop. This leads to a partitioning of effort, wherein a resource-based perspective is used to construct schedules for those resources which are predicted to be in high demand, and an order-based perspective is used to schedule the remaining operations required by each order (see The Opis Project).

Control decisions at the scheduling perspective level yield a specific set of focal points around which to generate or revise scheduling decisions. Strategy-level control decisions concern the approach that is adopted in carrying out these tasks. Alternative scheduling strategies differ in the types of constraints they consider and hence provide an opportunity to further restrict the range of conflicts that are explicitly addressed. In initially generating partial schedules relative to a given focal point, the control decision at this level is clear; we are interested in the schedule that offers the best possible compromise among all relevant preference constraints and want a strategy which takes all constraints into account. However, in reacting to problems that arise, there are two reasons why specialized strategies might be desirable. First, it makes sense to take advantage of those aspects of the original compromise (that is, the compromise which originally led to the solution component under revision) which remain unchanged. For example, if an operation is reported to have finished 10 minutes after its scheduled finish time, it is unlikely that a simple shift of the affected portion of the schedule can dramatically alter the degree to which relevant preference concerns are satisfied, and nothing sophisticated is warranted. The second reason for preferring specialized strategies concerns the pragmatics of having to produce a result within specific response-time constraints. It is often necessary to sacrifice the potential of achieving a better compromise to accommodate the urgency of the problem at hand.

Generally speaking, the determination of which aspects of the original compromise are most cost effective to revise is a difficult problem. However, guidance can be provided by (1) knowledge of how the scheduling restrictions on the solution have changed (for example, the start-time constraint on an operation has been pushed forward in time by a significant amount), (2) knowledge of how different types of constraints generally interact with one another (for example, the ability to meet deadlines is influenced by work shift and order-sequencing decisions), and (3) knowledge of the amount of search effort required to relax specific types of constraints (for example, exploration that considers alternative work shift assignments versus exploration which assumes preferred shift assignments).

Control decisions at the tactics level concern the actual generation and extension of candidate partial schedules. Here, we are talking about making decisions that satisfy or relax specific constraints. In the most desirable case, we might envision a compromise process involving specialized operators that function as lobbyists for specific constraints. Each proposes scheduling decisions that satisfy its benefactor, operators are applied on the basis of the relative importance of various concerns, and compromises are made only when it is recognized as necessary. In practice, this type of dynamic search control at the lowest level has been difficult to realize, and it has been necessary to consider scheduling decisions in a well-defined order. For example, in generating alternative partial schedules for a particular order, we use the precedence constraints on operations, or, generally, the set of process plans they define, to dictate the order in which decisions are considered.

A Hierarchy of Problem Descriptions. One device commonly used within AI planning systems to reduce complexity is abstraction. A hierarchy of problem descriptions is defined, each level omitting or aggregating specific lower-level details. The levels of aggregation defined in the underlying factory model provide one example of how the scheduling problem can be abstracted. Machines can be grouped functionally into work areas, which in turn can be grouped into larger work areas, and so on. The resulting hierarchy provides increasingly less precise descriptions of

both the resources that must be allocated and the allocation decisions themselves. Stated another way, the hierarchy abstracts away specific constraints. Generally, we can view the exclusion of any set of constraints as an abstract formulation of the actual problem.

Solutions to abstract forms of the problem can provide useful guidance in solving the detailed problem. They can yield insights about where the detailed scheduling effort would best be focused next. For example, computation of an aggregate-level schedule might serve to highlight particular areas of high resource contention. They can also be used to reduce the search in carrying out a detailed scheduling task. We might, for example, assume that machines are the most important resource to be allocated and defer consideration of the preferences defined over other types of resources until machine assignments have been made.

There is another important reason for working with higher-level descriptions of the scheduling problem at hand. Given the unpredictability inherent in the factory environment and the likelihood of change, it does not make sense to generate detailed scheduling decisions over more than a short-term time horizon. To do so not only requires additional computational effort but is also likely to complicate later efforts to revise the schedule.

Summary

In this section we advocated an approach to schedule construction and maintenance based on the application of heuristic knowledge about the various restrictions, goals, and preferences present in the actual factory environment under consideration. We began by considering the issue of representing and organizing this knowledge. A model of the domain was outlined, designed to explicitly capture the full range of scheduling alternatives and provide a framework for structuring preferential knowledge. We then presented a technique for estimating the quality of a given set of scheduling decisions based on the degree to which the decisions satisfy various preferential concerns. Finally, we examined the role of constraint knowledge in controlling the generation of alternatives. We now turn attention to the prototype scheduling systems whose construction has given rise to these principles of constraint-directed scheduling.

ISIS: An Intelligent Scheduling and Information System

The Intelligent Scheduling and Information System (ISIS) project was initiated in the fall of 1980 in conjunction with Westinghouse Electric Corporation to investigate the applicability of knowledge-based approaches to the problem of job shop scheduling. WTCP was selected as a test site for the work. During the period from 1980 through 1984, several versions of an experimental job shop scheduling system called ISIS were constructed and tested with an evolving model of the WTCP manufacturing facility. Each iteration reflected a better understanding of both the WTCP scheduling problem and the types of constraint knowledge required to enable a flexible solution.

Much of the early work centered around a categorization of the scheduling constraints in the environment and the development of an appropriate knowledge representation. This led to an initial SRL model of the factory, which included an explicit representation of scheduling preferences and proposed the knowledge-structuring ideas described in Modeling of the Factory Environment. These ideas were extended and refined over the course of the project as the problem became better understood and as additional aspects of the problem were addressed. The basic approach regarding comparison of alternatives discussed in Quantifying the Notion of Constraint Satisfaction, was also formulated fairly early in the project. Experimentation with different constraint evaluation schemes led to the specific mechanisms previously described. Later versions of ISIS focused on techniques for carrying out the search for a constraint-satisfying schedule. A nonhierarchical heuristic search technique was initially formulated and subsequently enhanced through the introduction of additional levels of analysis. Although some difficulties with the final search architecture have subsequently been recognized (see The OPIS Project), we feel that the contributions made by ISIS toward a general solution to real-world scheduling problems are significant. In this section we briefly describe the WTCP environment, the operation of the final version of the ISIS system, and its current status. Further details can be found in Fox, Allen, and Strohm 1982; Fox 1983; Fox and Smith 1984a.

Problem Domain

The WTCP scheduling problem addressed during the ISIS development effort was restricted to the portion of the plant responsible for the production of steam turbine blades, which constituted approximately one-third of the total shop floor area. A turbine blade is a complex three-dimensional object produced by a sequence of forging, milling, and grinding operations to tolerances of one-thousandth of an inch. Thousands of different blade styles are produced in the plant, primarily in batches of 1 to 200 blades. Orders released to the floor fall into distinct priority classes, which range from replacement orders for malfunctioning blades in currently operating turbines to blade orders that accompany orders for new turbines to orders for blades to be placed in stock for future use. There are typically 100 to 200 blade orders on the shop floor at any time.

Each style of turbine blade produced in the plant has one or more possible process routings associated with it, each ranging in length from 10 to 15 operations. Distinctions between alternative routings can be as simple as substituting a different machine or as complex as changing the manufacturing process. In-process orders in the shop must share the use of approximately 50 machines and human-manned work

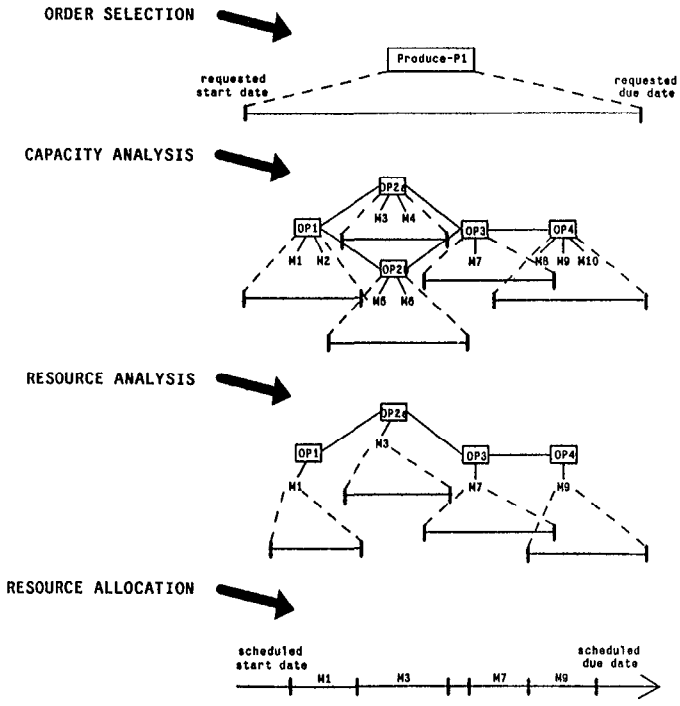


Figure 2. Successive Refinement of an Order's Schedule within ISIS.

centers as well as a full array of supporting resources (for example, operators, tooling, NC tapes, box gauges, and so on). Shop floor scheduling at WTCP is a formidable task, and decisions are influenced by the full range of concerns outlined earlier.

Constraint-Directed Generation of Factory Schedules

In constructing a job shop schedule, ISIS assumes an order-based scheduling perspective. An initial decomposition of the problem is performed, wherein the orders that require scheduling are prioritized, and the shop schedule is then derived by incrementally determining a schedule for each individual order. The generation of a given order's schedule is cast as a hierarchical, constraint-directed search. Different levels of the search operate with different abstractions of the problem, each a function of the types of constraints which are considered at that level. Control generally flows in a top-down fashion, moving through successively more detailed levels of analysis. This is illustrated in figure 2. The following subsections summarize the major components of this search architecture.

Problem Decomposition. The order-selection level of analysis constitutes the system's global problem-decomposition strategy. It collects the set of orders that require scheduling—for example, newly released orders, partially scheduled orders, and previously scheduled orders whose schedules have been affected by unanticipated events or decisions imposed by the user—and assigns a priority to each. This prioritization of the orders to be scheduled provides a high-level, order-by-order plan for completing the shop schedule. The ISIS search manager carries out this plan by selecting orders for scheduling in priority order. The scheduling of a given order can disrupt existing schedules for lower-priority orders, in which case the affected lower-priority orders are queued for rescheduling. The particular prioritization schemes that have been tested presume a grouping of orders into distinct priority classes and base the order-priority calculation on both priority class and the closeness of the requested due date.

Constructing an Order's Schedule. Construction of the shop schedule proceeds by repeatedly selecting and scheduling the unscheduled order with the highest priority. We found that order-centered search in the presence of bottlenecked resources resulted in a significant horizon effect. The need became obvious for a hierarchical search in which an abstracted version of the problem, focusing on critical resource capacity, was solved. Consequently, a level of analysis based on critical resource capacity is first applied to propagate the temporal consequences of the requested start and due dates assigned to the selected order. The result is a coarse schedule that reflects due date considerations in the context of the current shop load. Propagation is carried out by means of a dynamic programming analysis that elaborates the set of possible routings for the selected order and associates an earliest start time and a latest finish time with each operation. This information is embodied in a set of preference constraints which serve to influence the decisions that are made during the subsequent detailed resource analysis. If these constraints are satisfied by the final schedule produced, then the order is completed within its externally imposed release and due dates. The constraints are cast as preferences, however, to enable compromise with respect to other conflicting concerns.

The detailed resource-analysis level considers the full range of restrictive and preferential constraints that surround the production of the current order. Again, operating over the set of possible routings, a heuristic search is performed that proceeds either forward from the order's requested start date or backward from its requested due date. Alternative schedules for the order are explored incrementally: On each iteration of the search, the current set of candidate partial schedules is expanded by considering one additional scheduling decision, for example, the selection of an operation to perform, the selection of a resource for an operation, or the selection of a time interval for an operation. Using a beam search, ISIS retains and extends at each iteration only the n best partial schedules. Constraints are collected and applied, as described in Quantifying the Notion of Constraint Satisfaction, to assess how well each candidate satisfies relevant preferences; this method provides the basis for pruning.

Upon completion of the search, a commitment is made to the highest-rated hypothesis. Constraints reflecting these decisions are posted to restrict the final determination of the order's schedule. The inclusion of the operation time-bound constraints from the capacity level has what we call a *periscope effect* on the search: Their consideration in the local evaluation of a partial schedule provides a look ahead into the possible consequences of the decisions.

The schedule produced during detailed resource analysis significantly refines the coarse schedule generated at the capacity-analysis level of the search. A specific process routing has been selected for the order under consideration, resources have been selected for each operation in that routing, and resource time-bound constraints have been associated with each selected resource. Complete specification of the order's schedule at this stage requires only the refinement of the imposed resource time bounds. The resource-assignment level of the search carries out this refinement, leading to final allocation decisions for each resource required in the order's schedule. Operating within the time bounds imposed by detailed resource analysis, ISIS makes allocation decisions that attempt to minimize the order's WIP time. Once finalized, these decisions are added to the existing shop schedule and serve to further constrain any subsequent scheduling that is required.

Deviating from the High-Level Plan. This top-down approach to generating an order's schedule constitutes the system's default order scheduling-plan. The ISIS search architecture provides a framework for deviating from this plan in problematic situations by associating presearch and post-search analysis phases with each level of the search. Post-search analysis is concerned with the detection of unacceptable search results (that is, poorly satisfied constraints) and the identification of prior decisions that are likely to have caused the problem. If problems are encountered, the diagnosis identifies the appropriate level at which to redirect the search. Presearch analysis responds to diagnosed problems by altering the set of assumptions under which the targeted level of the search will proceed (that is, relaxing specific constraints which would otherwise be considered nonnegotiable). In practice, these aspects of the ISIS search architecture have not been extensively explored. This has been due primarily to difficulties in mapping appropriate prescriptive actions into the specific levels of analysis conducted by ISIS. In particular, an ability to adopt different scheduling perspectives (as suggested in Controlling the Search for Constraint-Satisfying Schedule) is necessary in many reactive contexts.

Dealing with Shop Floor Plan Deviations. A rudimentary facility was provided for handling problems on the shop floor that forced deviations from plan, (for example, machine breakdowns). Because a schedule is viewed as a set of constraints on the availability of resources, deviations were viewed as constraint violations. Our approach implemented a policy that the repaired schedule deviate as little as possible from the original in order to reduce shop instability. This was accomplished by turning the original schedule's resource-reservation constraints into preference constraints to be used in the rescheduling of the affected orders.

Interaction with the User

The ISIS user interface is viewed as a medium for communicating constraints to the system. The user specifies what the constraints are, and the schedules produced are responsive to these concerns. To facilitate acquisition and refinement of this constraint knowledge, a number of high-level interfaces are provided. The constraint editor is used to formulate preference constraints. Driven by knowledge of the underlying constraint representation, the editor provides guidance to the user in specifying or revising the necessary information relating to alternatives, relevance, importance, and partial satisfaction. Once specified, a new constraint is automatically integrated into the existing knowledge base. Similar editors are provided to facilitate changes to other aspects of the factory model. A status-update interface is used to communicate new scheduling restrictions that result from factory operation.

The interactive scheduling subsystem provides a graphic interface through which the user can manually perform some portion of the scheduling task. The user can elect to make specific scheduling decisions prior to involving the automatic scheduler (for example, manually schedule a critical area of the plant), or can manually adjust the automatically generated schedule after the fact. Scheduling decisions imposed by the user are treated as additional constraints during subsequent automatic scheduling.

As individual scheduling decisions are made by the user, the system uses its constraint knowledge in an advisory capacity. Relevant scheduling restrictions are checked for constraint violations (for example, the operation being scheduled cannot be performed on the machine indicated), and if any are found, feasible alternatives are suggested. If a proposed scheduling decision is found to satisfy all scheduling restrictions, the decision is evaluated with respect to relevant preferential concerns. Once again, constraints are collected and applied in the manner described in Quantifying the Notion of Constraint Satisfaction, and the satisfaction estimates returned are used to provide the user with an indication of the desirability of the decision. A sample commentary is shown in figure 3. In this case, five distinct preference constraints were found to be relevant to the decision in question. The partitioning of these considerations reflects the particular scheduling policy associated with the order being scheduled. These assessments make the user aware of all constraints that the system knows to be relevant to specific scheduling decisions. In doing so, the assessments also pro-

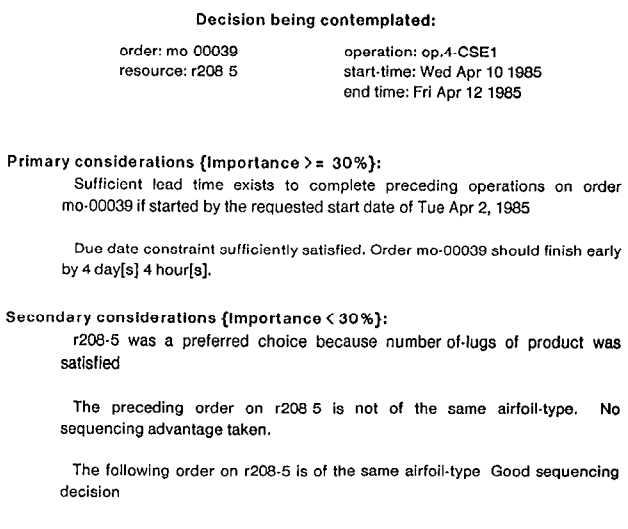


Figure 3. Evaluating a Scheduling Decision.

vide a context for identifying constraints that are incorrectly specified or currently unknown to ISIS.

Current Status

In December 1984 a production prototype of ISIS was installed at the Winston-Salem facility and demonstrated for Westinghouse management. The effort was considered successful by all involved, and all prespecified objectives were met. Nonetheless, the next step—putting ISIS into field test in the plant—did not take place. This decision was made for two reasons. First, the ISIS prototype was implemented in SRL, a noncommercial, experimental knowledge representation language embedded in Franz Lisp. In the absence of commercially available counterparts at the time, SRL provided a useful and necessary tool for conducting our research. However, its inadequacy as a delivery vehicle became painfully apparent as the ISIS factory model was scaled up to capture the full detail of the tapered blade scheduling environment. Problems with both SRL and the underlying LISP implementation in dealing with such a large system (the ISIS knowledge base alone now occupied over 10 megabytes of disk space) made it extremely difficult to operate the prototype. It was clear that a translation of ISIS to robust, commercially supported software (which was now available) was a prerequisite to an actual field test.

A second reason concerned the need for integration with existing information systems, an issue that was not previously addressed. It was felt that the development of an ability to directly interact with the order-entry system, engineering databases, and so on, should also precede field testing. Given these considerations, Westinghouse decided to suspend work at Winston-Salem. ISIS was instead taken over by the Westinghouse Productivity and Quality Center (WPQC), the technology transfer arm of the corporation. Since that time, WPQC has been engaged in transforming the ISIS prototype into a production package.

The OPIS Project

As experience was accumulated with the ISIS search architecture, limitations stemming from its strict reliance on an order-based decomposition strategy were perceived. As discussed in Controlling the Search for a Constraint-Satisfying Schedule, it was suspected that any single-perspective scheduling system would run into difficulties in effectively attending to the full range of preference constraints. These perceptions were later confirmed in an experimental study (summarized later). The Opportunistic Intelligent Scheduler (OPIS) Project grew out of the recognition of these problems and the desire to explore the potential benefits of a dynamic, conflict-directed approach to problem decomposition. Here, we describe the results we have obtained thus far regarding the use of multiple scheduling perspectives, summarize the current OPIS architecture, and briefly describe our current directions. The reader is referred to Smith and Ow (1985), Ow and Smith (1986), and Smith et al. (1986) for further details of this work.

A Comparative Analysis of Multi-Perspective and Single-Perspective Scheduling

To provide experimental justification for the claims put forth regarding the use of multiple scheduling perspectives, an initial multiperspective scheduling system was configured. A resource-scheduling strategy based on the focused-approach priority rule developed by Ow (1985) was implemented, and the scheduling strategy of ISIS was adapted for use as the order scheduler. To simplify issues of coordination, the following tightly controlled pattern of interaction between these two scheduling perspectives was imposed:

- The resource scheduler was first applied to a single, prespecified bottleneck resource (a work area of the plant consisting of some number of machines).
- The order scheduler was then applied to work outward from this established portion of the shop schedule to complete the schedules for each individual order.

The performance of this system, designated OPIS 0, was then contrasted with that of ISIS and a dispatch system using the COVERT priority rule for minimizing tardiness cost as formulated in Vepsaleinen (1984). The latter system represents a well-known and well-regarded approach to job shop scheduling and was included to provide a benchmark for the experimental study. The results obtained in this comparative analysis are summarized here. Complete details can be found in Ow (1986).

A scaled-down model of the Winston-Salem job shop was used to provide an environment for the experimental study. Six product types were included in the model, with associated process plans that utilized over 30 machines. Machines were functionally grouped into 11 work areas. The bottleneck area contained 7 machines. The orders to be scheduled were known in advance, and predetermined due dates and tardiness cost rates were used. For purposes of

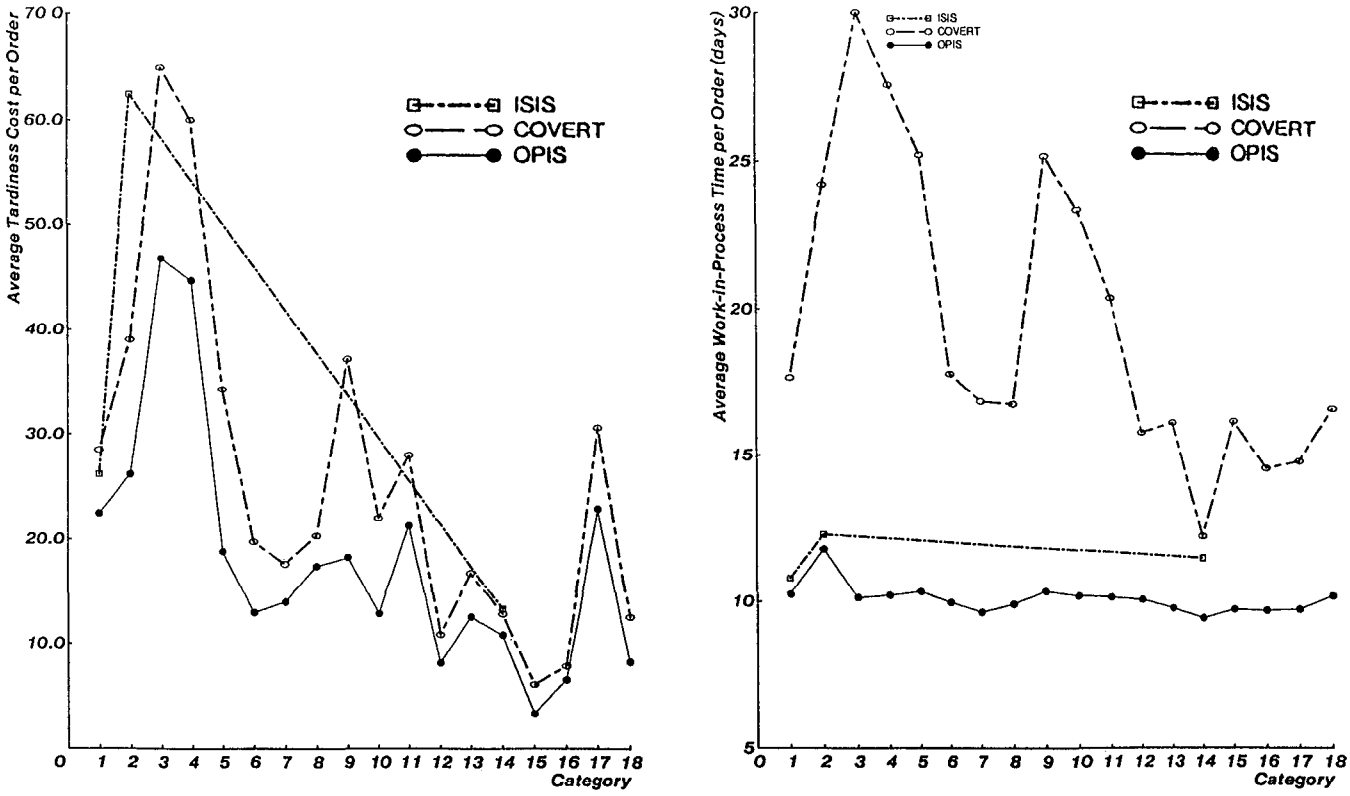


Figure 4. Average Tardiness Cost per Order and Average WIP time per Order over Different Problem Categories.

comparison, the schedules generated by each system were evaluated with respect to tardiness costs, WIP time, and the number of machine setups.

The comparative analysis was carried out over a total of 22 test problems. Twenty of these test problems required 120 orders to be scheduled, and the remaining 2 problems involved only 85 orders. Individual problems were generated by manipulating 4 parameters: the pattern of order releases (daily, weekly, and exponential rates), the number of orders released in each batch of releases, the product mix, and the setting of due dates. The set of problems was grouped into 18 categories, representing different shop conditions and load factors ranging from 70% to 120% of the capacity of the bottleneck area.

Figure 4 summarizes the performance of each system with respect to tardiness cost and WIP time. On each account, OPIS 0 outperformed both ISIS and COVERT. Only four test problems were solved using ISIS largely because of the length of time taken to complete each task. Furthermore, it became clear from a detailed analysis of the ISIS schedules that the shortcomings predicted in using a purely order-based perspective were experienced. ISIS performed well with respect to minimizing WIP time because achievement of this objective depends primarily on an ability to resolve order-based conflicts. However, its performance with respect to tardiness cost suffered because of its inability to effectively handle resource-based conflicts. This fact is underscored by examining the number of setup changes in the schedules produced by OPIS 0 and ISIS. The ISIS schedules contained close to twice as many setup changes for bottleneck machines as did the schedules generated by OPIS 0. The time required to perform these setup changes appeared to account for much of the discrepancy in tardiness cost performance. The variances in average tardiness costs were also smaller for OPIS 0 than for ISIS and COVERT. WIP time variances for OPIS 0 and ISIS were between 4 and 9 days, and those for COVERT ranged from 14 to 225 days because of the strictly local nature of the dispatch-based decision making.

A System Architecture for Conflict-Directed Problem Decomposition

OPIS 0 was constructed for the purpose of assessing the relative advantages of multiperspective scheduling. From the standpoint of system flexibility, some rather severe limitations were imposed:

- The system operates with a fixed plan for partitioning effort between its two scheduling perspectives that requires a priori specification of a single bottleneck work area. In an actual shop, the situation is often much more complex. Several bottleneck resources can exist, either independently of one another or in a specific primary bottleneck-secondary bottleneck relationship. Furthermore, the bottleneck resource in the shop often "floats"

over time, in which case specific resources need not be considered critical for the entire duration of the schedule. A priori specification of these complex resource requirements is unreasonable, because many of the specific relationships emerge only after some amount of scheduling is performed. An ability to dynamically predict high-contention areas of the shop schedule is necessary to fully exploit the resource-based scheduling perspective.

- The system assumes that the resource-based subproblem at the bottleneck resource completely dominates the order-based subproblems that involve nonbottleneck operations. The order scheduler is obliged to make scheduling decisions that are consistent with the bottleneck schedule which is initially constructed by the resource scheduler. In the general case, it must be possible to reconsider and revise the bottleneck schedule when conflicting concerns surface in subsequent subproblems.
- The system implements a particular schedule-generation strategy and provides no facilities for reactively revising a portion of the shop schedule in light of changing circumstances in the shop.

Having experimentally confirmed our intuitions regarding multiperspective scheduling, we designed and implemented a scheduling architecture sensitive to these concerns. The current OPIS architecture draws from principles of standard blackboard style architectures (Erman et al. 1980) and similarly assumes a system organization comprised of a number of knowledge sources (KSs) that extend and revise a global set of one or more hypotheses. In this case, the KSs implement alternative scheduling strategies, and the hypotheses being manipulated are candidate shop schedules. A framework for conflict-directed control is provided by specializing system activity in the following two ways:

1. Centralized Schedule Management: An underlying schedule-management component assumes sole responsibility for maintaining an accurate characterization of the current state of a given candidate schedule. All revisions or extensions to a given schedule resulting from either KS execution or external circumstances pass through this component, which in turn propagates the temporal consequences of these decisions (that is, operation time-bound constraints and their origins) to other affected parts of the schedule. Thus, the schedule-management component serves as the system's mechanism for communicating constraints among interdependent subproblems. It also enables straightforward detection of any constraint violations that are introduced. The techniques employed are descended from but significantly extend those described in Smith (1983).
2. Event-Based Control: Problem decomposition and subsequent coordination of the scheduling effort is carried out by a designated KS called the manager. The manager operates in an event-driven fashion, formulating and initiating subtasks to be performed in response to changes in the current schedule. A hierarchy of event types is defined to characterize the specific kinds of changes that are of interest. One subclass of events concerns the reporting of scheduling progress; for example, a resource schedule has been added to the shop schedule. Other subclasses designate specific types of inconsistencies and preference compromises. The hierarchy of event types provides a basis for structuring the manager's control knowledge. Associated with each specific type of event is a set of event-processing heuristics that map characteristics of the event to specific courses of action (that is, sequences of subtasks to assign to designated KSs). Generally, they encode specific pieces of constraint-interaction knowledge. (These mechanisms are described in more detail in Smith et al. forthcoming.)

Within this control framework, we augmented the previous configuration of resource-based and order-based scheduler KSs with two additional KSs. The first, referred to as the capacity analyzer*, implements a shop-level scheduling perspective. It provides a basis for dynamic problem decomposition by generating predictions of likely areas of high resource contention. In contrast to the detailed schedulers, the capacity analyzer operates with aggregate descriptions of resources, operations, and resource-allocation decisions. It constructs a predictive shop schedule that satisfies the time-bound constraints posted with each aggregated operation, using a general line-balancing heuristic. The demand for capacity reflected by this schedule is then compared with the actual capacity of the required aggregate resources, and likely bottlenecks are predicted. The second KS added to the configuration implements a simple right-shifting strategy and is employed to resolve minor inconsistencies that might arise. The manager's current body of control heuristics generalizes the OPIS 0 control strategy to one where effort is initially focused on scheduling any number of predicted bottlenecks (as dynamically determined by the capacity analyzer), and the schedule is then completed on an order-by-order basis. Contingencies are also included for revising decisions made by the resource scheduler in response to inconsistencies that arise later on in the search.

Current Directions

The current thrust of our work centers on the issues surrounding reactive management of existing schedules. We see several open research problems. The first concerns the

* Despite the name, this KS bears no relationship to the capacity-analysis level of the ISIS search architecture In fact, the ISIS capacity analysis level has been subsumed by the propagation techniques of the schedule-management component and removed from the order scheduler.

use of specialized reactive scheduling strategies in response to inconsistencies that are introduced. We need to gain a better understanding of the conditions under which local changes can be expected to adequately preserve the global compromise reflected by the current schedule. A more difficult problem concerns the detection of weaknesses in the current schedule (that is, situations where preference concerns are unsatisfactorily compromised). Although we currently have an ability to detect individual preference violations, we do not yet have a methodology for estimating the ramifications of individual compromises on the global solution (or some global aspect of it). A final issue concerns an ability to maintain predictive accuracy as schedules move farther out in time. A failure to somehow account for the unpredictability inherent in the scheduling environment can result in overly optimistic predictions.

Final Remarks

In this article we presented work aimed at providing a knowledge-based solution to real-world factory scheduling problems. We focused specifically on the problem of generating and maintaining detailed production schedules in job shop environments. We began by examining the characteristics of this activity and identified the large and conflicting set of factors that influence scheduling decisions in real-world environments. This led to a view of scheduling as a complex constraint-directed process, and an approach to the problem centered around the representation and use of knowledge about these diverse scheduling concerns. We then focused attention on determining the types of knowledge necessary to make intelligent job shop scheduling decisions. Issues relating to the use of constraint knowledge in decomposing the problem, generating alternative partial solutions, assessing the quality of alternative solutions, and compromising amongst conflicting constraints were all considered, and a set of constraint-directed reasoning mechanisms was defined.

We next described a series of software systems that have been constructed during the course of our work. These systems provided a basis for experimenting with and refining our ideas and illustrate a progression toward increasingly more sophisticated constraint-based reasoning techniques. We first examined ISIS-2, which emphasized the use of a rich symbolic model of the factory environment and an initial methodology for constructing schedules that reflect the full complexity of a given production environment. Recognition of a need for greater system flexibility in addressing different scheduling tasks led to the development of OPIS, which provides constraint-based techniques for opportunistically focusing the scheduling effort. This remains the current focus of our efforts.

This work was initiated on the premise that a practical solution to factory scheduling problems requires an ability to represent and reason with knowledge of the specific operating environment. We feel that the results presented in this article have made important strides toward the realization of this ability and demonstrate the viability of this perspective. To be sure, there are many important issues that remain unresolved, and the ultimate potential of this research must await further investigation. Nonetheless, we feel that knowledge-based factory scheduling systems will soon provide an attractive alternative to the scheduling techniques currently employed in manufacturing environments.

Acknowledgments

This research has been sponsored in part by the Air Force Office of Scientific Research under contract F49620-82-K-0017; Westinghouse Electric Corporation; and The Robotics Institute, Carnegie-Mellon University.

References

Allen, J F. 1981. A General Model of Action and Time. Technical Report 97, Computer Science Dept , Univ of Rochester.

Allen, J F., and Koomen, J A 1983. Planning Using a Temporal World Model In Proceedings of the Eighth International Joint Conference on Artificial Intelligence, 741–747

Daniel, L 1984 Planning and Operations Research *In Artificial Intelligence Tools, Techniques, and Applications,* eds. T. O'Shea and M Eisenstadt, New York: Harper & Row 423–452.

Erman, L D ; Hayes-Roth, F ; Lesser, V R ; and Reddy, D R 1980 The HEARSAY-II Speech Understanding System: Integrating Knowledge to Resolve Uncertainty *Computing Surveys* 12: 213–253.

Fox, M. S 1983 Constraint-Directed Search: A Case Study of Job Shop Scheduling Ph D diss , Computer Science Dept , Carnegie-Mellon Univ , 1983

Fox, M S ; Allen, B P ; and Strohm G A 1982 Job Shop Scheduling: An Investigation in Constraint-Directed Reasoning In Proceedings of the Second National Conference on Artificial Intelligence, 155–158

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

Fox, M. S , and Smith, S F. 1984b The Role of Intelligent Reactive Processing in Production Management In Proceedings of the Thirteenth Annual CAMI Technical Conference, 6,13–6.17.

Fukumori, K 1980 Fundamental Scheme for Train Scheduling (Application of Range-Constriction Search), A I Memo 596, AI Lab, Massachusetts Institute of Technology

Garey, M. R and Johnson, D. S. 1979 *Computers and Intractability.* San Francisco: Freeman

Goldstein, I P , and Roberts, R B 1977. NUDGE: A Knowledge-Based Scheduling Program In Proceedings of the Fifth International Joint Conference on Artificial Intelligence, 257–263.

McDermott, D 1982 A Temporal Logic for Reasoning About Processes and Plans *Cognitive Science* 6: 101–155.

Miller, D 1983 Scheduling Heuristics for Problem Solvers Research Report 264, Yale Univ , Dept of Computer Science

Ow, P. S. 1986 Experiments in Knowedge-Based Scheduling. Technical Report, The Robotics Institute, Carnegie-Mellon Univ

Ow, P. S 1985 Focused Scheduling in Proportionate Flowshops *Management Science* 31: 852–869

Ow, P S , and Smith, S. F. 1986. Toward an Opportunistic Scheduling System. In Proceedings of the Nineteenth Hawaiian International Conference on System Sciences, 345–353

Sathi, A ; Fox, M S ; and Greenberg, M 1985 Representation of Activity Knowledge for Project Management. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 7: 531–552

Smith, S F 1983 Exploiting Temporal Knowledge to Organize Constraints Technical Report, CMU-RI-TR-83-12, The Robotics Institute, Carnegie-Mellon Univ

Smith, S. F., and Ow, P S. 1985 The Use of Multiple Problem Decompositions in Time-Constrained Planning Tasks In Proceedings of the Ninth International Joint Conference on Artificial Intelligence.

Smith, S. F.; Ow, P. S.; LePape, C ; and Muscettola, N. Forthcoming Reactive Management of Factory Schedules. Technical Report, The Robotics Institute, Carnegie-Mellon Univ.

Smith, S F ; Ow, P. S.; LePape, C ; McLaren, B ; and Muscettola, N 1986. Integrating Multiple Scheduling Perspectives to Generate Detailed Production Plans In Proceedings of the SME Conference on Artificial Intelligence in Manufacturing, Forthcoming

Stefik, M 1981 Planning with Constraints (MOLGEN Part 1) *Artificial Intelligence* 16: 111-140

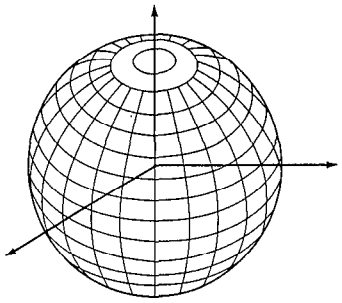
Sussman, G. J., and Steele, G L , Jr 1980 CONSTRAINTS—A Language for Expressing Almost Hierarchical Descriptions. *Artificial Intelligence* 14: 1-40.

Vepsaleinen, A. 1984 State-Dependent Priority Rules for Scheduling Ph D diss , Graduate School of Industrial Administration, Carnegie-Mellon Univ

Vere, S 1981. Planning in Time: Windows and Durations for Activities and Goals Techmical Report, Jet Propulsion Laboratory.

Waltz, D. 1975. Understanding Line Drawings of Scenes with Shadows. In *The Psychology of Computer Vision,* ed P H Winston, 19-92 New York: McGraw-Hill.

Wright, M , and Fox, M S. 1983 SRL/1 5 User Manual, The Robotics Institute, Carnegie-Mellon Univ.



call for participation

Workshop on Spatial Reasoning and Multi-Sensor Fusion

sponsored by AAAI

Spatial reasoning is central to the interaction of an intelligent robot with its environment. Although the problems are somewhat different for mobile and stationary robots, the basic need for correlating perceived information -- which due to viewpoint limitations in most cases constitutes only partial evidence about scene entities -- with the stored world knowledge remains the same. Also common to both cases are the problems of integrating incoming information through various sensors, such as photometric, range, tactile and force/torque. Such issues will form the focus of this workshop. In particular, the topics that will be highlighted at the meeting include

- **Reasoning about shape from partial evidence**
- **Fusion of photometric and range data for mobile robots**
- **Fusion of 2D, 3D, tactile and F/T sensing for assembly robots**
- **Evidential reasoning for verification vision**
- **Reasoning architectures for spatial data**
- **Programming paradigms for spatial reasoning**
- **Representation of world knowledge for assembly and mobile robots**
- **Formal theories of spatial reasoning**
- **Spatial planning and problem solving**

Papers on these topics are invited for consideration. Three copies of an extended abstract or a full-length paper should be sent to either of the following two addresses prior to March 15, 1987.

Su-shing Chen
Dept of Computer Science
University of North Carolina
Charlotte, NC 28223

Avi Kak
Robot Vision Lab
EE Building, Box 121
Purdue University
W. Lafayette, IN 47907

This workshop will be held August 20-22 immediately preceding the IJCAI conference which is to be held in Milan, Italy Aug 23-30, 1987. The workshop site is Hilton International Milano on Via Galvani in the heart of the city.