MANAGEMENT SCIENCE
Vol. 35, No. 2, February 1989
Printed in U.S.A.

THE SINGLE MACHINE EARLY/TARDY PROBLEM*

PENG SI OW AND THOMAS E. MORTON
Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, Pennsylvania 15213-3890

We examine the problem of scheduling a given set of jobs on a single machine to minimize total early and tardy costs. Two dispatch priority rules are proposed and tested for this NP-complete problem. These were found to perform far better than known heuristics that ignored early costs. For situations where the potential cost savings are sufficiently high to justify more sophisticated techniques, we propose a variation of the Beam Search method developed by researchers in artificial intelligence. This variant, called Filtered Beam Search, is able to use priority functions to search a number of solution paths in parallel. A computational study showed that this search method was not only efficient but also consistent in providing near-optimal solutions with a relatively small search tree. The study also includes an investigation of the impacts of Beam Search parameters on three variations of Beam Search for this problem.
(PRODUCTION/SCHEDULING; DETERMINISTIC JOB SHOP; SINGLE STAGE)

1. Introduction

In this paper, we examine the problem of scheduling a set of jobs on a single machine to minimize total early and tardy costs. More specifically, the objective is to find the schedule that minimizes the total earliness and total tardiness costs of all jobs, subject to the constraints that no preemption of jobs is allowed, no idle time may be inserted, and all jobs are initially available. This problem is a generalization of the weighted tardiness problem. The inclusion of an earliness cost in the objective function may represent the cost of completing a project early in PERT-CPM analyses, as suggested by Sidney (1977), or deterioration in the production of perishable goods. The early/tardy objective is also compatible with the philosophy of just-in-time production to minimize inventory costs by producing goods as close to their due dates as possible. In a "closed shop", scheduling deadlines will be driven by timing and quantities of replenishment for the central plant, final goods inventories giving rise to costs of early and tardy replenishment.

By reference to the special case of weighted tardiness (Karp 1972), the problem can be seen to be NP-complete, although the unit processing case is polynomially bounded since it can be solved as an assignment problem (Ow 1984). The problem also has an irregular objective function, i.e. the cost function may decrease as completion time increases. Thus, many important theorems on job ordering, particularly Emmon's (1969) and Lawler's (1977), cannot be applied to these problems in order to make optimization techniques such as branch-and-bound and dynamic programming more efficient. Hence, we look for heuristics to obtain good solutions to this problem.

We present four dispatch heuristics. In the dispatch method, whenever a machine is free, and there are jobs waiting to be processed, a priority function is used to select the next job. Two of these, the MRV rule (Morton, Rachamadugu, and Vepsalainen 1984) and the Earliest Due Date have been used by others for the weighted tardiness problem, and provide standard comparisons here. We also test a new search method, Filtered Beam Search, which belongs to the class of Beam Search techniques that have been successfully used in speech understanding, image recognition, and in the ISIS and OPIS scheduling systems (Fox 1983, Ow and Smith 1986). Beam Search is essentially a heuristic

* Accepted by Leroy B. Schwarz, former Departmental Editor; received September 1984. This paper has been with the authors 16 months for 3 revisions.

0025-1909/89/3502/0177$01.25

search method that searches β candidate solution paths in parallel (the beamwidth), never backtracks, and saves only the β "best" paths at each stage. The evaluation of "best" at each stage is somewhat expensive, typically requiring one or more evaluations of a dispatch heuristic. In the filtered method, this step is done with some low cost, crude heuristic, which passes some number of nodes, up to $\alpha\beta$ (α is the filterwidth), to the expensive heuristic which in turn selects up to β nodes from them. A preliminary study was performed to provide some insight into the effects of the search parameters, α and β, on solving this problem. A combinatorial experimental design was used to investigate the performance of the dispatch heuristics and Filtered Beam Search using two pairs of parameter settings. The design parameters included two due date ranges, two tardiness factors, three relative earliness costs, and three sizes of job batch. Twenty replications were made, yielding a total of 1440 problems. The Filtered Beam Search methods were not only the best by far, but also the most robust.

The paper is organized as follows. In the next section, prior work on special cases of the early/tardy problem is discussed. The general search methods of branch-and-bound, neighborhood search and beam search are also reviewed. §3 develops the adjacency condition that follows from the local optimality of the globally optimal solution, and identifies special cases of optimal sequences. §4 describes the rationale for the new early/tardy heuristics and the Filtered Beam Search method. The computational study and results are contained in §5. §6 presents the conclusions.

2. Background

The authors know of no other works on the general early/tardy problem although in the last decade some papers have been published on scheduling problems that include both earliness and tardiness penalties. The earliest of these was by Sidney (1977). Sidney showed that the problem of minimizing maximum job penalty (early or tardy), where all jobs have the same early and tardy cost functions and idle time could be inserted, was polynomially bounded. Lakshminarayan, Lakshmanan, Papineau and Rochette (1978) later provided an $O(n \log n)$ algorithm for this problem. Seidmann, Panwalkar and Smith (1981) considered the problem of assigning individual job due dates and identifying a sequence so as to minimize weighted earliness, tardiness and lead times costs. All jobs had the same early, tardy and leadtime weights. Panwalkar, Seidmann and Smith (1982) considered a similar problem except that jobs had to be assigned a common due date. Polynomial algorithms were provided for both, the latter was $O(n \log n)$. Kanet (1981) provided a polynomially bounded algorithm for minimizing absolute lateness on a single machine where jobs have a common due date that is greater than the total makespan of all jobs. Idle times could be inserted in the schedule. Sundaraghavan and Ahmed (1984) extended Kanet's work to the parallel machine case, still polynomially bounded, and to the single machine case where no idle time could be inserted. The latter is a special case of the early/tardy problem where early and tardy costs of all jobs are one. This problem was not shown to be polynomially bounded.

Search techniques may be used in both exact and heuristic procedures. They have mainly been developed and studied by researchers in Operations Research and Artificial Intelligence e.g. Barr and Feigenbaum (1981), Lawler and Woods (1966) and Nilsson (1980). The better known ones are perhaps best-first search and depth-first search. Best-first search involves always selecting the most promising tip nodes of the search tree to explore, while depth-first search explores the most recently sprouted node. Whereas these two techniques tend to develop partial solutions in the process of searching, the neighborhood search methods (Baker 1974) require an initial complete solution to start the search. Neighborhood search methods are typically used to improve solutions obtained through some computationally simple procedures, e.g. to improve schedules obtained through the dispatch method.

There are many variations of the Beam Search method, but they all share a common feature: at any point in time, the search method only explores a limited number of partial solutions in parallel. The Beam Search method was first known to have been implemented by Lowerre (1976) in HARPY, a speech recognition system. It was later used by Rubin (1978) for image recognition in a system called ARGOS. Fox (1983) used Beam Search in a system called ISIS to find schedules in a complex job shop environment. More recently, Ow and Smith (1986) developed another knowledge-based system, OPIS, for scheduling the same job shop model that incorporated Beam Search as one of its knowledge sources. In all these applications, a single expensive evaluation heuristic was used. Further, the research focus was on the problem domain, rather than on understanding the behavior of the search method itself.

3. Analysis of the Early/Tardy Problem

This section gives a formal statement of the early/tardy problem and identifies a necessary condition for an optimal schedule based on local optimality. Conditions under which globally optimal sequences may be found are also discussed. Each job is assumed to have a due date, a penalty defined for each time unit that it is early and a possibly different penalty for each time unit that it is tardy. Thus, the early/tardy objective function is

$$\text{Min.} \sum_{i=1}^{n} (h_i \max(0, d_i - f_i) + w_i \max(0, f_i - d_i))$$

where n is the number of jobs to be scheduled, d_i is the due date of job i, f_i is its completion time, h_i is the early cost rate and w_i is the tardy cost rate.

3.1. *Adjacency Condition*

The Adjacency Condition described below must be satisfied by all optimal sequences. It is based on the requirement that a globally optimal schedule must also be locally optimal so that no improvement can be gained by a pairwise interchange of adjacent jobs.

THEOREM 1. The Adjacency Condition. *All adjacent pairs of jobs in the optimal sequence must satisfy the following condition*

$$w_i p_j - \Omega_{ij}(w_i + h_i) \geq w_j p_i - \Omega_{ji}(w_j + h_j) \tag{1}$$

where job i immediately precedes job j, and Ω_{ij} and Ω_{ji} are defined as

$$\Omega_{xy} = \begin{cases} 0 & \text{if} \quad s_x \leq 0, \\ s_x & \text{if} \quad 0 < s_x < p_y, \\ p_y & \text{otherwise}, \end{cases}$$

where $s_x = d_x - t - p_x$ is the slack of job x, p_x is its processing time x, and t is the earliest time the machine is free.

PROOF[1]. Let $c(ij)$ be the cost of the subsequence (i, j) and $c(ji)$ be the cost of the reversed subsequence (j, i). Positions of jobs i and j can be interchanged without affecting the cost incurred by the other jobs in the sequence. If the adjacency condition holds, then $c(ij) \leq c(ji)$.

[1] We wish to thank one of the referees for this version of the proof.

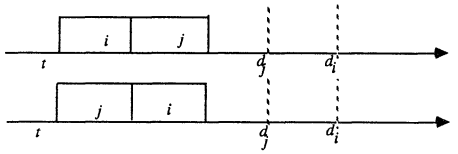


FIGURE 1

Consider the case where jobs i and j are both early in both sequences, as in Figure 1.

$$c(ij) = h_i(d_i - t - p_i) + h_j(d_j - t - p_i - p_j)$$
$$= h_i(d_i - t - p_i) + h_j(d_j - t - p_j) - h_j p_i. \tag{2}$$

Similarly,

$$c(ji) = h_j(d_j - t - p_j) + h_i(d_i - t - p_i) - h_i p_j.$$

Since i is early in either sequence, then $d_i - t - p_i \geq p_j$ and since j is early in either sequence, then $d_j - t - p_j \geq p_i$. Thus $\Omega_{ij} = p_j$ and $\Omega_{ji} = p_i$. Using these facts with inequality (1) gives

$$w_i p_j - p_j(w_i + h_i) \geq w_j p_i - p_i(w_j + h_j) \qquad \text{which reduces to}$$
$$p_j h_i \leq p_i h_j. \tag{3}$$

Substituting (3) into (2) gives

$$c(ij) \leq h_j(d_j - t - p_j) + h_i(d_i - t - p_i) - h_i p_j$$
$$= c(ji).$$

The same procedure is repeated for the remaining cases of i and j early and late to complete the proof:

1. One job is early in the earlier position only while the other is early in either.
2. Both jobs are early in the earlier position and tardy in the later position.
3. One job is tardy in either position, the other is early in either.
4. One job is tardy in either position, the other is early in earlier position only.
5. Both jobs are tardy in either position. □

3.2. *Special Cases of the Early/Tardy Problem*

Given a set of jobs, a schedule that minimizes only the earliness costs provides a lower bound on the total minimum earliness and tardiness costs that can be achieved. If such a schedule results in none of the jobs being tardy then the tardiness cost is zero, and that schedule is also optimal for the early/tardy objective function. The same is true of the tardiness cost minimization. This leads to the following two lemmas. Let $H_i = h_i/p_i$ and $W_i = w_i/p_i$.

LEMMA 1. *If the WSPT sequence (weighted shortest processing time) results in a schedule that does not have any early jobs, then this sequence is optimal. The WSPT sequence is such that* $W_1 \geq W_2 \geq W_3 \geq \cdots \geq W_n$.

PROOF. The WSPT sequence minimizes weighted lateness (see, for example, Conway, Maxwell and Miller 1967, p. 44). If such a sequence does not have any early jobs, then the earliness cost of the sequence is zero and the lower bound cost is also the minimum weighted early/tardy cost.

LEMMA 2. *If the WLPT sequence (weighted longest processing time sequence) results in a schedule that does not have any tardy jobs, then this sequence is optimal. The WLPT sequence is such that* $H_1 \leq H_2 \leq H_3 \leq \cdots \leq H_n$.

PROOF. The WLPT sequence maximizes weighted lateness. In so doing, weighted earliness is also minimized. If no job is tardy, the tardiness cost is zero for all jobs, and hence the minimum total earliness cost is also the minimum weighted early/tardy cost.

4. Heuristics for the Early/Tardy Problem

4.1. *Tardiness Heuristics*

The work of Morton, Rachamadugu and Vepsaleinen (1984) on the weighted tardiness problem provides a paradigm for early/tardy heuristics. They developed a myopic heuristic that attempts to achieve local optimality. A *locally optimal sequence* is defined as one that cannot be improved by interchanging the positions of adjacent pairs of jobs. Local optimality is a necessary but not sufficient condition for global optimality. They found that job i immediately precedes job j in an optimal sequence when $P_{ij}(s_i) \geq P_{ji}(s_j)$ where

$$P_{ij}(s_i) = \begin{cases} W_i & \text{if} \quad s_i \leq 0, \\ W_i(1 - s_i/p_j) & \text{for} \quad 0 < s_i \leq p_j, \\ 0 & \text{otherwise,} \end{cases}$$

and similarly for $P_{ji}(s_j)$. Therefore, if adjacent pairs of jobs in a sequence satisfy this condition, then that sequence is locally optimal.

$P_{ij}(s_i)$ may be taken to be the priority of job i with respect to j at the earliest time the machine is free, t (which in turn determines the slack). A dispatch priority rule was derived by comparing each job's priority to an average job with processing time $\bar{p}$. Hence, a job's priority, $P_i(s_i)$ is

$$P_i(s_i) = \begin{cases} W_i & \text{if} \quad s_i \leq 0, \\ W_i(1 - s_i/\bar{p}) & \text{for} \quad 0 < s_i \leq \bar{p}, \\ 0 & \text{otherwise.} \end{cases}$$

However, the authors found that local optimality was still far from global optimality due to "clashes" between multiple jobs. The following example illustrates the problem. Figure 2 shows a sequence of unit jobs 1, 2, 3, 4, 5, and their respective due dates. For simplicity, assume that the tardy cost rates are equal for all jobs, w. The sequence was obtained using the priority rule above and is locally optimal. At $t = 0$ and $t = 1$, all jobs have zero priority so jobs 1 and 2 were arbitrarily selected. However, at $t = 2$, the remaining jobs 3, 4 and 5 have priority w but only one can be chosen to complete on time. It is clear that jobs 3, 4 and 5 should have been scheduled before 1 and 2, but the priority rule was too myopic to foresee the clash. The problem may be alleviated somewhat by approximating a job's priority with respect to a group of jobs, rather than just one job. In the above example, an optimal sequence may be found if priority is based on a group of three jobs. This insight led to the addition of a lookahead parameter, k to the priority function. The resulting function is:

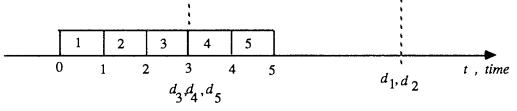


FIGURE 2. A Locally Optimal But Globally Suboptimal Sequence.

$$P_i(s_i) = \begin{cases} W_i & \text{if} \quad s_i \le 0, \\ W_i(1 - s_i/(k\bar{p})) & \text{for} \quad 0 < s_i \le k\bar{p}, \\ 0 & \text{otherwise.} \end{cases}$$

This linear decay function is shown in Figure 3. Morton, Rachamadugu and Vepsaleinen experimented with other functions to find a better approximation to foresee job clashes, thus extending the scope of the local optimality beyond two jobs. They found the exponential function shown in Figure 3, called MRV here, gave the best performance:

$$P_i(s_i) = \begin{cases} W_i & \text{if} \quad s_i \le 0, \\ W_i \exp(-s_i/(k\bar{p})) & \text{otherwise.} \end{cases}$$

4.2. *Early/Tardy Heuristics*

We followed the Morton, Rachamadugu and Vepsaleinen (1984) approach to develop a myopic heuristic for the early/tardy problem that would also attempt to achieve an "extended" local optimality. If inequality (1) is divided by $p_i p_j$, it may be re-written as: $\mathcal{P}_{ij}(s_i) \ge \mathcal{P}_{ji}(s_j)$ where

$$\mathcal{P}_{ij}(s_i) = \begin{cases} W_i & \text{if} \quad s_i \le 0, \\ W_i - s_i(W_i + H_i)/p_j) & \text{for} \quad 0 \le s_i \le p_j, \\ -H_i & \text{otherwise,} \end{cases}$$

and similarly for $\mathcal{P}_{ji}(s_j)$.

$\mathcal{P}_{ij}(s_i)$ may be considered to be the priority of job i with respect to job j. As in the weighted tardiness case, a simple dispatch rule may be obtained by comparing each job's priority to that of a job with average processing time $\bar{p}$, and selecting the highest priority job to schedule next. This rule attempts to find sequences that satisfy the adjacency condition. However, once again, the problem of multiple job clashes can make such a locally optimal sequence far from globally optimal. As in the weighted tardiness case, a lookahead parameter (k) may be used to attempt to extend the scope of optimality beyond two adjacent jobs. The linear priority rule for a job i is:

$$\mathcal{P}_i(s_i) = \begin{cases} W_i & \text{if} \quad s_i \le 0, \\ W_i - s_i(W_i + H_i)/k\bar{p} & \text{for} \quad 0 \le s_i \le k\bar{p}, \\ -H_i & \text{otherwise.} \end{cases}$$

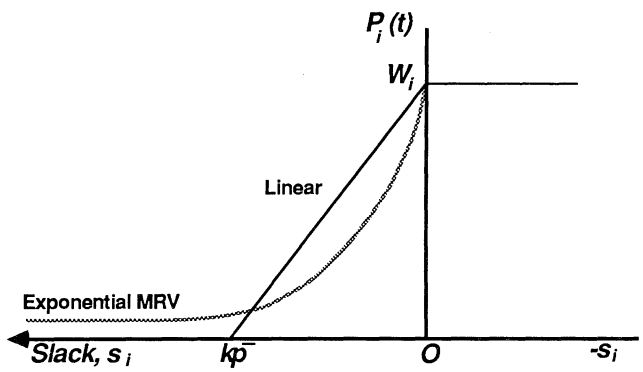


FIGURE 3. Priority Rules for the Weighted Tardy Problem.

This linear function, shown in Figure 4 as LIN-ET, estimates the priority of a job by taking into account its impact on the next k jobs. We also explored other functions giving similar changes in priority as slack decreased to find better estimations. The one that worked best was constructed from two functions and is shown in Figure 4 as EXP-ET:

$$\text{A:}\quad \mathcal{P}_i(s_i) = W_i \exp\left[-\frac{H_i + W_i}{H_i}(s_i/\bar{p})\right] \qquad \text{if} \qquad 0 \le s_i \le \left(\frac{W_i}{H_i + W_i}\right)k\bar{p},$$

$$\text{B:}\quad \mathcal{P}_i(s_i) = h_i^{-2}\left(W_i - \frac{(H_i + W_i)s_i}{k\bar{p}}\right)^3 \qquad \text{if} \qquad \left(\frac{W_i}{H_i + W_i}\right)k\bar{p} < s_i \le k\bar{p}.$$

EXP-ET(A) is similar to the MRV function and reflects a priority that focuses on the tardiness cost of a job as its slack becomes smaller. On the other hand, when the slack is large, it is the early cost that dominates. This is approximated by EXP-ET(B). The two curves terminate where LIN-ET crosses zero priority.

The choice of k should reflect the average number of jobs that may clash in the future each time a sequencing decision is to be made. From the graph of the priority rules in Figure 4, it can be seen that the k parameter controls the time at which a job's priority begins to increase. This may be interpreted to signify that tardiness is imminent. Before this time, only the earliness cost is relevant. Therefore, when job due dates are close together and the lead times of jobs are not very long, a large lookahead k should be used. This causes job priorities to rise earlier to reflect the potential for large numbers of jobs to clash. A decision may then be made early enough to avoid the clash. In the case where due dates are evenly distributed, k should be small as few jobs will clash.

Both LIN-ET and EXP-ET begin with a priority of H_i when jobs are in no danger of being tardy, $s_i \ge kp_i$, gradually increases to a plateau at W_i when jobs are on time or late, $s_i \le 0$. If the WSPT sequence for an early/tardy problem has no early jobs, and, hence, is optimal (Lemma 1), then both rules will be guaranteed to generate the WSPT sequence. This is because some jobs will be tardy or on time whenever a sequencing decision has to be made. For the WSPT job to be optimal, a tardy job will have the largest W_i among the unscheduled jobs, which is also the maximum priority any job can attain. Both rules will determine that job to have priority W_i and select it to be scheduled. Thus a WSPT sequence will be generated. If jobs have large slacks, then both rules will schedule jobs in increasing order of H_i, i.e. WLPT. Such a schedule is optimal if all jobs are early (Lemma 2).

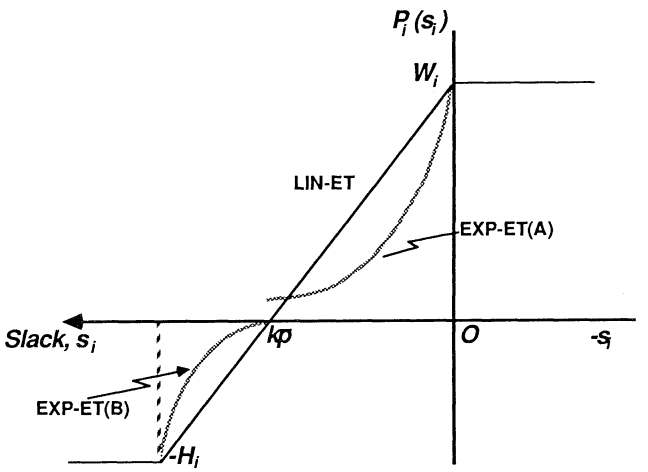


FIGURE 4. Priority Rules for the Weighted Early/Tardy Problem.

4.3. *Beam Search Methods*

The basic idea in all the Beam Search methods is to search a limited number of solution paths in parallel. The following example illustrates a simple Beam Search. Consider a one-machine weighted tardiness problem with five jobs to be scheduled. One job is scheduled at a time. The Beam Search tree for this problem is shown in Figure 5. The nodes represent partial sequences of jobs. The descendants of a node are obtained by extending the sequence by one unscheduled job. The search tree was obtained using a beamwidth of two. Thus, at level one, we begin by enumerating the five different ways a solution sequence could begin. The goodness of each partial sequence is estimated using a function known as an *evaluation function* and the "best" two sequences are selected. These two nodes form the beam, the beginnings of two solution paths to be developed in parallel. Consequently, each of the two nodes are sprouted, generating level 2. The eight new partial sequences, each consisting of two jobs, are then evaluated. Again, the two "best" are selected for the beam and the procedure is repeated until, finally, two complete sequences are found (level 5). The better one is chosen as the solution.

Beam Search methods are not guaranteed to find an optimal solution since there is no guarantee that a node on the optimal path is always selected for the beam. Further, Beam Search does not backtrack so that nodes, once excluded from the beam, are lost forever. The intent of this class of search techniques is to search quickly, hence no backtracking. Effectiveness is achieved by acknowledging the inaccuracy of evaluation functions by pursuing a number of promising paths concurrently so as "to hedge its bets." The more accurate an evaluation function, the narrower the beamwidth that can be tolerated without degrading the quality of the solution.

As with all search techniques, the choice of an evaluation function plays a crucial role in the efficiency and effectiveness of the search. In scheduling problems, nodes may be evaluated in two ways. The priority of the last job added to the sequence may be used, i.e., a *priority evaluation function*. Alternatively, a schedule cost may be estimated for each node, called a *cost evaluation function*. We present three Beam Search methods that use either priority or cost evaluation, or the two together.

Priority Search uses a priority function as its evaluation function and differs from the description above in only one respect. Referring to Figure 5, at Level 1, when nodes 3

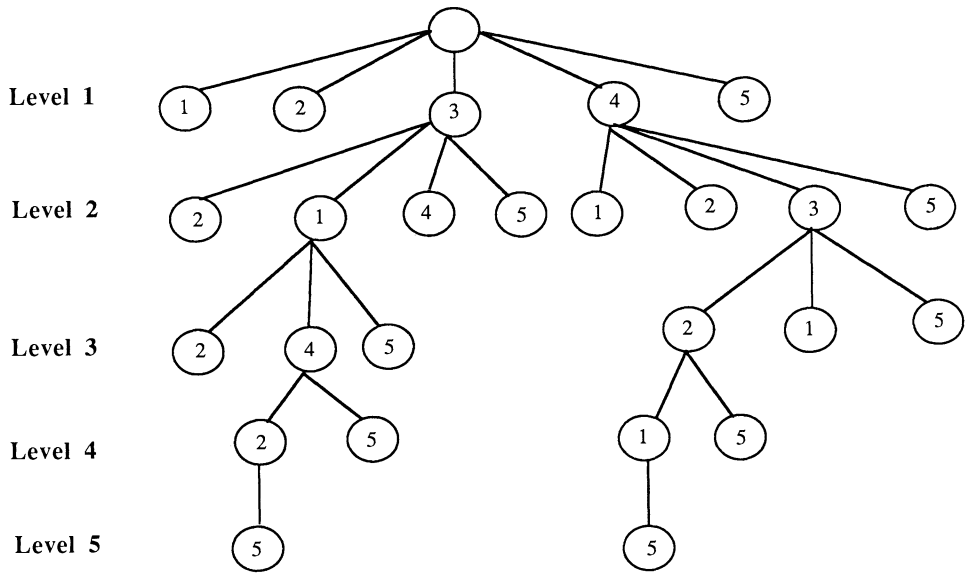


FIGURE 5. The Beam Search Tree (Beamwidth = 2).

and 4 are sprouted, the two sets of four descendant nodes cannot be compared with each other for best since the priorities for one group are calculated under different conditions from the other. The evaluation function is said to be "context-dependent". Therefore, one node is selected from each of the two groups, since the best two out of eight cannot be determined. Earlier systems, such as Lowerre's (1976), Rubin's ARGOS (1978), Fox's ISIS (1983), and Ow and Smith's OPIS (1986), all used variants of priority search, though not always context-dependent evaluation functions. Lowerre and Rubin also used a beamwidth that was dependent on the priority values. Fox, Ow and Smith used a priority function that was the weighted sum of functions of deviations from desired constraints.

In *Probe Search*, an upper bound is calculated to serve as the evaluation function. At any node, where it is desired to have a cost evaluation function, the partial sequence is completed by dispatching the remaining unscheduled jobs using some priority rule. The schedule from this probe can be costed. This produces an upper bound on the schedule cost. In preliminary tests the Probe Search method was found to be good but quite expensive, since for every node sprouted, a full dispatch procedure must be executed.

Filtered Beam Search uses both priority and cost evaluation functions to provide computationally efficient evaluation that does not degrade the search. The cheaper, but possibly less accurate, priority evaluation function is used to eliminate the obviously poorer nodes leaving only the better nodes for the more expensive cost evaluation. Consider again the scheduling problem of Figure 5. The Filtered method has two parameters, the filterwidth and the beamwidth. Let this be three and two, respectively. At level 1, the five nodes are evaluated by the priority function and the best three (the filterwidth) are selected. These three nodes are then further evaluated by the cost function (using a probe) and the best two are selected for the beam. At level 2, each of the two beam nodes are again sprouted. For each node, three of its highest priority descendents are retained for further evaluation. All six nodes are evaluated with the cost function and two are again selected for the beam.[2] The procedure is repeated until two complete sequences are found. Note that the size of the search tree increases linearly with the filterwidth and beamwidth. A formal description and further notes on the Filtered Beam Search method is given in the Appendix.

5. Computational Study

5.1. *Design of the Experiment*

The design of the experiments follow those of previous studies for the weighted tardiness problem: Baker and Martin (1974), Morton, Rachamadugu and Vepsaleinen (1984) and Srinivasan (1971). The primary aim of each study was to investigate the relative performance of its heuristics as certain characteristics of the problem changed. The main control factors were the tardiness factor of the set of jobs to be scheduled, the due date range and the correlation between due dates and processing times.

The tardiness factor, τ, (Srinivasan 1971) is a coarse measure of the proportion of jobs that might be expected to be tardy in an arbitrary sequence. For a given average due date, $\bar{d}$, and average processing time, $\bar{p}$, τ can be derived by $\tau = 1 - \bar{d}/(n\bar{p})$. Wilkerson and Irwin (1971) reported that when the due date range is tight, the priority of all jobs would tend to rise at about the same time, making it difficult to discriminate between urgent and not-so-urgent jobs. In our experiments, a due date range factor, R, was used to control the range of the due date distribution, calculated as $Rn\bar{p}$. Below is a summary of the parameters that were controlled in generating the test problems. All data, processing times, due dates and costs, are integers.

[2] Note that the two selected nodes may be descended from the same parent.

- Processing times and due dates. A bivariate Normal distribution was used which incorporates the variation in processing times, variation in due dates and the correlation between the processing times and due dates. Numbers drawn were rounded to the nearest integer. The parameters were set at the following levels:
  - Population mean for processing times was 15.
  - Coefficient of variation for the processing times, (std. dev./mean), was 0.2.
  - Due dates range factor, R, was set at 0.4 and 1.0.
  - Correlation coefficient between processing times and due dates, ρ, was set at 0 and 0.5.
  - Tardiness Factor, τ, was set at 0.2 and 0.6.
- Tardy cost rate. Tardy cost and holding cost would more frequently, in practice, be related to work content. Therefore, the weights of a job were determined by independently determining the cost per unit processing time, w/p, from a uniform distribution in the range [0, 5]. The tardy cost for a job with processing time, p_i, obtained from the bivariate Normal distribution described earlier, was $w_i = (w/p) \times p_i$.
- Early cost rate. This was expressed as a ratio of the tardy cost rate, h/w. h/w was set at 25%, 10% and 5%. Note that the early cost rate is not an exact proportion of tardy cost rate since the rates are rounded to the nearest integers. This scheme would be consistent with the case where tardy cost rate reflected the value of a job and the early cost rate, reflecting holding cost, is proportionate to job value.
- Number of jobs in each set of tests, n. 8, 15, and 25.

Twenty test problems were generated for each combination of test parameter settings, giving a total of 1440 test problems. A preliminary study of the performances of the three Beam Search methods discussed earlier was conducted using the 25-job problems with early-to-tardy cost rate ratio of 25%. The EXP-ET priority function was used for the priority evaluation and to perform the probe in the cost evaluation. Based on this study, Filtered Beam Search was determined to dominate the others in terms of search efficiency and solution quality. A more thorough investigation of Filtered Beam Search was then carried out together with four priority rules used for dispatching, and neighborhood search. The priority rules were LIN-ET, EXP-ET, the exponential MRV rule and the Earliest Due Date Rule (EDD). For LIN-ET and EXP-ET, k was set at 3 for the 8- and 15-job problems, and 5 for the 25-job problems.[3] Morton, Rachamadugu and Vepsaleinen (1984) have found that the MRV rule outperformed WSPT, EDD and Montagne's method (1969) in the weighted tardiness problem, while EDD gave very good results for low tardiness problems with wide due date ranges. The MRV lookahead was set at 2. Neighborhood search provided an alternative extension of the dispatch method. An initial EXP-ET dispatch schedule was searched by interchanging adjacent pairs of jobs (Baker 1974). Two settings of the Filtered Beam Search parameters were studied. In Beam-3, both filterwidth and beamwidth were set at 3, and in Beam-5, both were set at 5.[4]

5.2. *Results of the Computational Study*

Performance was measured by the difference in cost between the heuristic solution and the optimal or lower bound cost on each problem expressed as a percentage of the optimal or lower bound cost.[5] The optimal solution was found using Branch-and-Bound.

[3] The better performance with a larger lookahead for the 25-job problems is hypothesized to be due to the increased potential for larger number of jobs to clash. The improvement in performance was especially obvious at higher tardiness factors.

[4] In Beam-3, a maximum of 9 nodes are evaluated using the cost function at each level of the tree, while in Beam-5, a maximum of 25 are evaluated "expensively".

[5] The weakness of this method is that if the optimal/lower bound cost is very small, as in the case of lower τ value, then a seemingly small difference, say from one wrong decision, can result in a huge percentage difference.

The lower bound on each partial solution was found by a relaxation involving breaking each job up into unit jobs that can then be solved as a linear assignment problem. Optimal solutions were found for all 8-job problems, but only some 15-job problems. The lower bounds were found to be quite tight when compared to optimal solutions for 60 15-job problems.

A series of tests, graphed in the Appendix, was undertaken to explore the effect of k on the performance of EXP-ET. As expected, a small lookahead ($k = 2$ or $k = 3$) was sufficient at low tardiness factors where few job clashes are expected. When the due date range was wide, larger lookaheads degraded performance. When the range was narrow and tardiness factor was high (i.e. high potential for a large number of jobs to clash) performance improved as k increased.

Figures 6 to 9 show the effects of the filterwidth and the beamwidth on the performance of the Filtered Beam Search method for 25-job problems. The figures also contain comparative performances of Priority Search and neighborhood search (labelled NBhood). Priority Search performances are charted as a vertical bar on each graph representing the range of performance as the beamwidth was varied from one (top of bar) to eight (bottom of bar). The Probe Search method gives identical results to the Filtered Beam Search method with a filterwidth of 25. As can be seen from the graphs, the Filtered version is superior to Priority Search and neighborhood search. The approximately L-shaped graphs indicate an initially large improvement for small increases in the size of the search tree, then little change for further increases. To search efficiently it appears better to have both filterwidths and beamwidths small-to-medium size (between 3 and 5) and relatively balanced, than to have one parameter very large and the other very small. Of course, performances comparable to Probe Search can be obtained with relatively

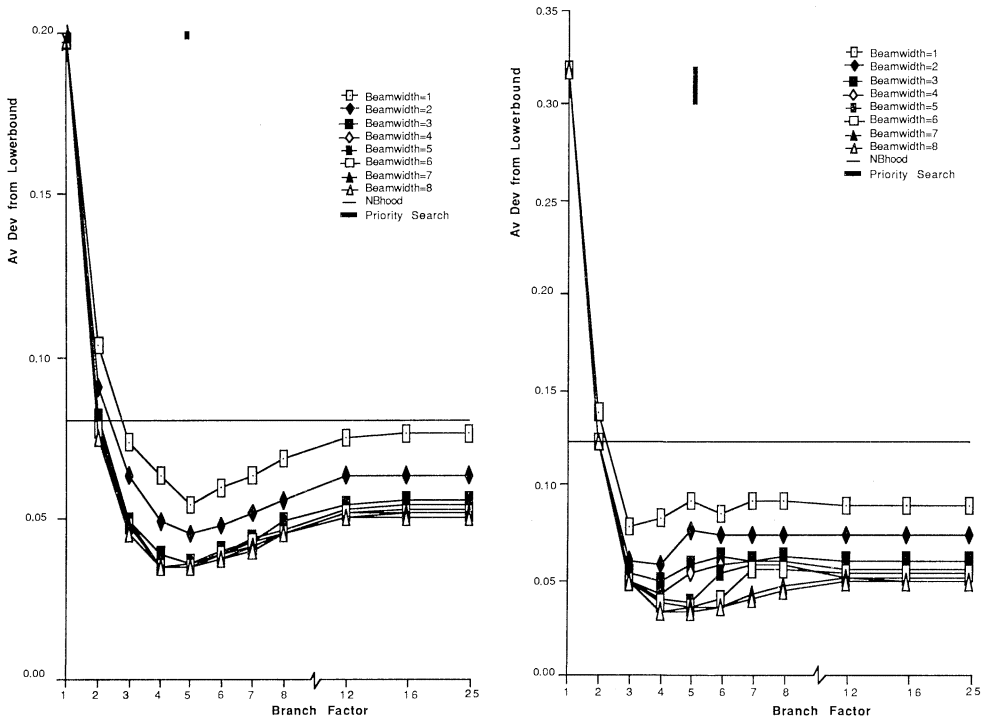


FIGURE 6. Effect of the Filterwidth and Beamwidth on the Beam Search Performance for the 25-Job Single Machine Early/Tardy Problem ($\tau = 0.2$ and $R = 0.4$).

FIGURE 7. Effect of the Filterwidth and Beamwidth on the Beam Search Performance for the 25-Job Single Machine Early/Tardy Problem ($\tau = 0.2$ and $R = 1.0$).

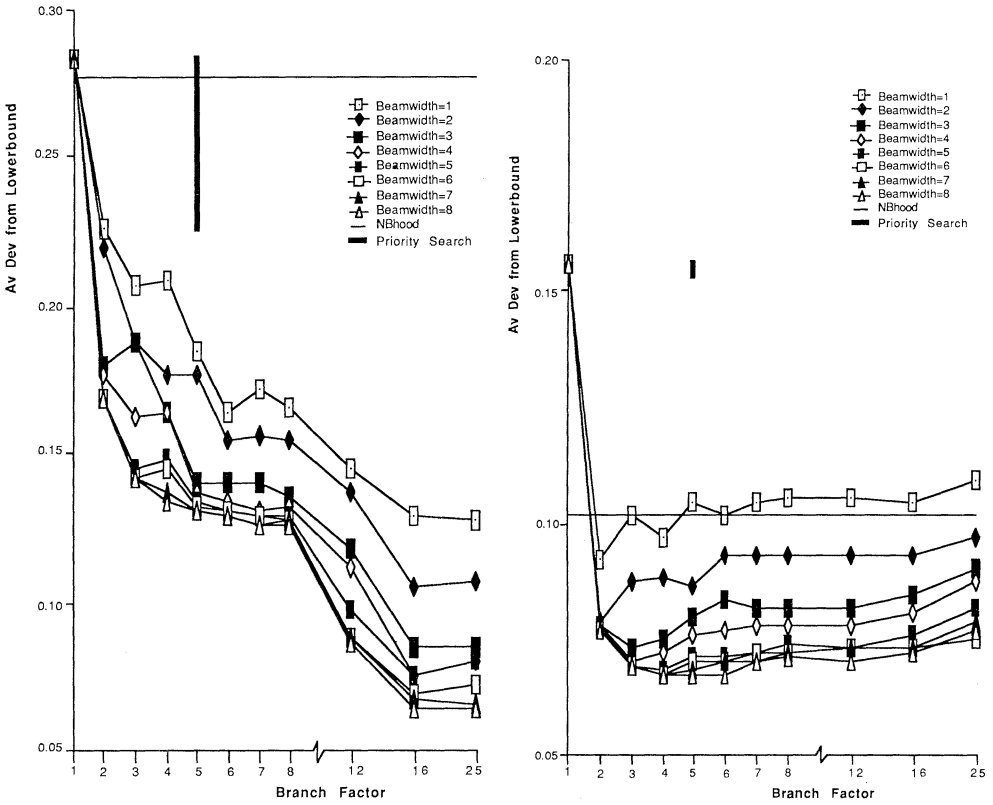


FIGURE 8. Effect of the Filterwidth and Beamwidth on the Beam Search Performance for the 25-Job Single Machine Early/Tardy Problem (τ = 0.6 and R = 0.4).

FIGURE 9. Effect of the Filterwidth and Beamwidth on the Beam Search Performance for the 25-Job Single Machine Early/Tardy Problem (τ = 0.6 and R = 1.0).

small filterwidths and beamwidths. The case where tardiness factor is high, τ = 0.6 and R = 0.4, requires a larger search tree to compensate for the relative inaccuracy of the priority rule for this set of data. In Figures 6, 7 and 9, there is a small but quite consistent deterioration in the performance when the filterwidth is increased beyond a certain size.

Table 1 summarizes the results of the comparative study of four dispatch rules and neighborhood search and Filtered Beam Search. (Neither the correlation coefficient ρ nor the percentage that early costs are of tardy costs had much influence on the nature of these results. Therefore we combined them for presentation.) EDD was disappointing even when the tardiness factor, τ, was very low. MRV performed relatively well at higher tardiness factors, but not at lower factors. In contrast, EXP-ET and LIN-ET were both considerably less sensitive to τ and R. EXP-ET showed a somewhat better overall performance compared to LIN-ET. Using Neighborhood Search (NBhood) with EXP-ET provided an even further improvement on the result. However, using EXP-ET in the Filtered Beam Search method, Beam-3 and Beam-5, gave not only better results but performed consistently over different combinations of tardiness factors and due date ranges.

Table 2 shows the average computation times of the search methods when executed on a VAX 780. The column entitled OPT/LB shows the average time taken to compute an optimal solution using Branch-and-Bound (8-job problems), or a lower bound cost by solving a linear assignment problem. From the data, it seems that NBhood is a worth-

TABLE 1

Average Percentage Deviation from Optimal or Lower Bound

No. of Jobs: 8 Optimal	EDD	R&M	LIN-ET	EXP-ET	NBhood	Beam-3	Beam-5
$\tau = 0.2, R = 0.4$	58.20	119.77	59.33	37.70	15.60	1.03	0
$\tau = 0.2, R = 1.0$	46.73	103.13	38.93	31.07	3.90	0.30	0
$\tau = 0.6, R = 0.4$	86.73	30.83	7.00	3.50	3.03	0.13	0
$\tau = 0.6, R = 1.0$	64.30	17.47	13.33	3.50	0.63	0.03	0
No. of Jobs: 15 Lower Bound	**EDD**	**R&M**	**LIN-ET**	**EXP-ET**	**NBhood**	**Beam-3**	**Beam-5**
$\tau = 0.2, R = 0.4$	93.33	156.67	61.80	54.70	44.57	20.57	15.27
$\tau = 0.2, R = 1.0$	70.67	101.80	26.67	26.93	10.63	7.63	6.67
$\tau = 0.6, R = 0.4$	119.70	13.87	15.13	12.67	11.17	7.47	6.43
$\tau = 0.6, R = 1.0$	98.87	22.23	15.90	9.50	6.57	6.17	6.07
No. of Jobs: 25 Lower Bound	**EDD**	**R&M**	**LIN-ET**	**EXP-ET**	**NBhood**	**Beam-3**	**Beam-5**
$\tau = 0.2, R = 0.4$	115.03	176.93	50.23	34.87	22.03	11.00	7.03
$\tau = 0.2, R = 1.0$	80.83	104.53	29.30	29.63	8.37	6.70	3.40
$\tau = 0.6, R = 0.4$	184.43	13.33	27.97	24.60	21.60	15.97	12.67
$\tau = 0.4, R = 1.0$	165.63	29.80	40.93	13.83	9.03	7.30	6.57

while extension to the basic dispatch procedure while Filtered Beam Search at the two settings are also cheap ways of obtaining consistently good results.

6. Conclusion

This paper examined both heuristics and search methods for the single machine early/tardy problem. The priority function EXP-ET appears to be quite accurate in that when used to schedule jobs using the dispatch method, relatively good schedules are obtained. The results also show that it may be expensive to ignore early costs and just use a heuristic that considers only tardy costs. However, it must be recognized that these results were obtained for a particular set of experiments.

We also investigated a variation of Beam Search that uses a priority function for evaluation purposes. The computational study showed that Filtered Beam Search was able to produce very good solutions with a relatively small search tree. The search is in some ways similar to Neighborhood Search in that both methods attempt to search the region around some good solution or partial solution. The extent of the search can be controlled using the filterwidth and beamwidth so that the search can be tailored to the

TABLE 2

Average Computation Time (seconds) Per Test Problem

	NBhood	Beam-3	Beam-5	OPT/LB
8-job problem	0.01	0.15	0.29	147.49
15-job problem*	0.02	0.67	1.71	80.89
25-job problem*	0.05	2.76	7.36	266.57

* Lower bound computation times.

accuracy of the available evaluation functions for a given problem. There are many other variations of Beam Search not considered here. It is hoped that a demonstration of the flexibility and potential of this class of search methods will encourage further research in this area.[6]

[6] This research was supported in part by the Air Force Office of Scientific Research under contract F49620-82-K0017.

Appendix

Additional Results of Computational Study

Appendix Figure 1 contains the results of the study on the effect of k on EXP-ET for 15-job problems and early to tardy cost rate ratio of 25%.

The Filtered Beam Search Method

Let α and β be the filterwidth and beamwidth, respectively, and be positive integers.

1. (*Initialization.*) Set $R_0 = \{1, \ldots, n\}$; $s_0 = (\ )$; $B = \{s_0\}$. s_0 is an empty sequence; R_0 is the set of jobs that can be scheduled first, i.e. appended to the empty sequence; and B is the set of nodes retained in the beam. α and β are assumed to be pre-defined constants.

2. (*First stage filter.*) Set $A = \varnothing$.

For each sequence s_i in B:

a. Calculate the priority of each job j in R_i at time t_{s_i}, the total completion time of the partial sequence s_i.

b. Let K be the α jobs of highest priority. If there are less than α jobs in R_i, then all jobs are in T. Ties are broken arbitrarily. For each job k in K:

i. Create a new sequence, s_{ik}, by appending k to the end of the sequence s_i.

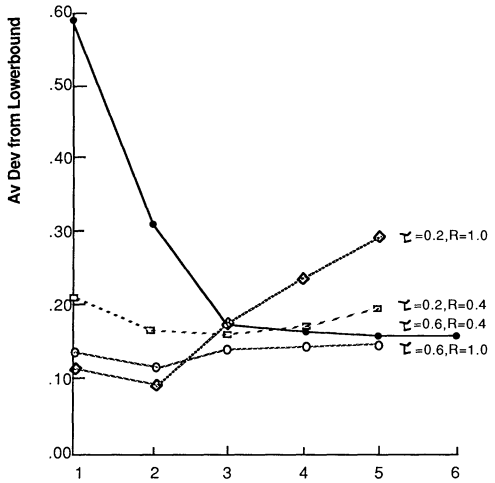
ii. Add s_{ik} to the set A.

iii. Create the set R_{ik} which is the set R_i less job k.

3. (*Second stage filter.*) Set $B = \varnothing$. Calculate the upper bound on each sequence in A. Select the β sequences with the lowest upper bounds to form the new set of active sequences, B. If there are less than β sequences in A, select all sequences. Ties are broken arbitrarily.

4. (*Stopping rule.*) If the new sequences in B are complete sequences of jobs $1, \ldots, n$ then select the lowest total cost (upper bound) sequence as the best solution found, and stop. Otherwise, go to step 2.

The preceding search method develops up to β partial sequences, in B. These sequences are said to be developed *in parallel* because at each iteration of steps 2 through 5, all partial sequences in B are more or less simultaneously extended by one job and evaluated. For each partial sequence in B, at most α extensions are enumerated. Therefore, up to $\alpha\beta$ new sequences are created and added to A. Following the evaluation of the new sequences in A, up to β of the most promising sequences are retained in B as the new sequences to be explored next iteration. The remaining sequences in A are discarded.



APPENDIX FIGURE 1. Effect of k Parameter on ET-EXP for $x = 1$.

A trace of the above search would produce a search tree where the nodes represent partial sequences enumerated in A, except the tip nodes which represent complete sequences. The descendents of a node represent sequences that are extensions of it, i.e. the sequences in A are descendents of the sequences in B at step 3. Since each iteration of steps 2 through 5 adds up to $\alpha\beta$ nodes to this search tree and there are n iterations for an n-job problem, the search tree has less than $n\alpha\beta$ nodes. The size and structure of the tree may be controlled by α and β.

References

BAKER, K. R., *Introduction to Sequencing and Scheduling*. John Wiley and Sons, New York, 1974.

——— AND J. B. MARTIN, "An Experimental Comparison of Solution Algorithms for the Single-Machine Tardiness Problem," *Naval Res. Logist. Quart.*, 21, 1 (January 1974), 187–199.

BARR, A. AND E. A. FEIGENBAUM, *The Handbook of Artificial Intelligence*. Vol. 1, Heuris Tech Press, Stanford, CA, 1981.

CONWAY, R. W., W. L. MAXWELL AND L. W. MILLER, *Theory of Scheduling*, Addison-Wesley, Reading, MA, 1967.

EMMONS, HAMILTON, "One-Machine Sequencing to Minimize Certain Functions of Job Tardiness," *Oper. Res.*, 17 (1969), 701–715.

FOX, M. S., "Constraint-Directed Search: A Case Study of Job-Shop Scheduling," Ph.D. Dissertation, Carnegie-Mellon University, 1983.

KANET, J. K., "Minimizing the Average Deviation of Job Completion Times About a Common Duedate," *Naval Res. Logist. Quart.*, 28, 4 (1981), 643–651.

KARP, R. M., "Reducibility among Combinatorial Problems," In *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, (Eds.), Plenum Press, New York, 1972, 83–103.

LAKSHMINARAYAN, SANKARAN, RAM LAKSHMANAN, ROBERT L. PAPINEAU AND RENE ROCHETTE, "Optimal Single-Machine Scheduling with Earliness and Tardiness Penalties," *Oper. Res.*, 26, 6 (November–December 1978), 1079–1082.

LAWLER, E. L., "A "Pseudopolynomial" Algorithm for Sequencing Jobs to Minimize Total Tardiness," *Ann. Discrete Math.*, 1 (1977), 331–342.

——— AND D. E. WOODS, "Branch-and-Bound Methods: A Survey," *Oper. Res.*, 14, 4 (July–August 1966), 699–719.

LOWERRE, B. T., "The HARPY Speech Recognition System," Ph.D. Dissertation, Carnegie-Mellon University, April 1976.

MONTAGNE, E. R., JR., "Sequencing with Time Delay Costs," Arizona State University, Industrial Eng., January 1969.

MORTON, T. E., R. M. RACHAMADUGU AND A. VEPSALEINEN, "Accurate Myopic Heuristics for Tardiness Scheduling," Working Paper W.P. 36-83-84, Carnegie-Mellon University, February 1984.

NILSSON, NILS J., *Principles of Artificial Intelligence*, Tioga Publishing, Palo Alto, CA, 1980.

OW, PENG SI, "Heuristic Knowledge and Search for Scheduling," Ph.D. Dissertation, Graduate School of Industrial Administration, Carnegie-Mellon University, 1984.

——— AND STEPHEN F. SMITH, "Viewing Scheduling as an Opportunistic Problem-Solving Process," To appear in *Annals of Operations Research, Intelligent Approaches to Decision Support*, R. Jeroslow, (Ed.).

PANWALKAR, S. S., M. L. SMITH AND A. SEIDMANN, "Common Due Date Assignment to Minimize Total Penalty for the One Machine Scheduling Problem," *Oper. Res.*, 30, 2 (March–April 1982), 391–399.

RUBIN, S., "The ARGOS Image Understanding System," Ph.D. Dissertation, Carnegie-Mellon University, April 1978.

SEIDMANN, A., S. S. PANWALKER AND M. L. SMITH, "Optimal Assignment of Due-Dates for a Single Processor Scheduling Problem," *Internat. Production Res.*, 19, 4 (May 1981), 393–399.

SIDNEY, JEFFREY, "Optimal Single-Machine Scheduling with Earliness and Tardiness Penalties," *Oper. Res.*, 25, 1 (January–February 1977), 62–69.

SRINIVASAN, V., "A Hybrid Algorithm for the One Machine Sequencing Problem to Minimize Total Tardiness," *Naval Res. Logist. Quart.*, 18 (September 1971), 317–327.

SUNDARARAGHAVAN, P. S. AND M. U. AHMED, "Minimizing the Sum of Absolute Lateness in Single-Machine and Multimachine Scheduling," *Naval Res. Logist. Quart.*, 31, 2 (1984), 325–333.

WILKERSON, L. J. AND J. D. IRWIN, "An Improved Method for Scheduling Independent Tasks," *AIIE Trans.*, 3, 5 (September 1971), 239–245.