SCHEDULING WITH INSERTED IDLE TIME: PROBLEM TAXONOMY AND LITERATURE REVIEW

JOHN J. KANET and V. SRIDHARAN

Department of Management, Clemson University, 101 Sirrine Hall, Clemson, South Carolina 29634-1305
kanet@clemson.edu • suhas@clemson.edu

(Received March 1993; revisions received August 1997, November 1998; accepted December 1998)

In the context of production scheduling, inserted idle time (IIT) occurs whenever a resource is deliberately kept idle in the face of waiting jobs. IIT schedules are particularly relevant in multimachine industrial situations where earliness costs and/or dynamically arriving jobs with due dates come into play. We provide a taxonomy of environments in which IIT scheduling is relevant, review the extant literature on IIT scheduling, and identify areas of opportunity for future research.

1. INTRODUCTION

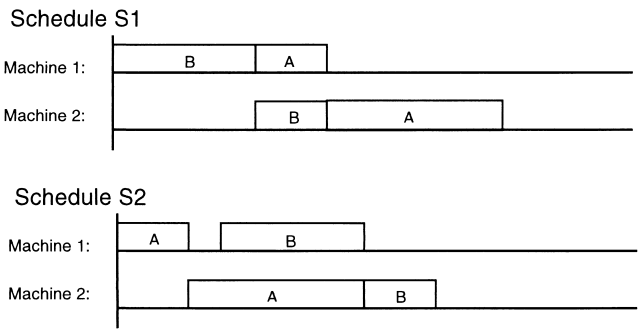
The design of production scheduling systems historically has been based on forward-pass construction methods in which no resource is deliberately kept idle in the presence of waiting work. While intuition indicates that such methods favor timely completion of required activities, specific anomalies that result from such practice have been reported in the literature, especially when preemption is not allowed. Deliberately holding resources idle may well be desirable in many situations. For example, it may be desirable to hold a resource idle when there are arriving jobs and the resource is a bulk processor (e.g., an oven), or when there is an urgent activity that cannot immediately start but that would be adversely delayed if other less critical work were allowed to begin. Likewise, it may be desirable to delay the start of an activity when there are significant penalties for early completion or when there are incentives for just-in-time (JIT) delivery. With better information systems, rapidly advancing computer capabilities, and a clear call toward JIT in production scheduling, there is good reason to investigate more sophisticated classes of schedules.

The great majority of research in scheduling has concentrated on the construction of nondelay schedules. However, we are interested in a more general class of schedules, namely the *inserted idle time* (IIT) schedules. A nondelay schedule has been defined by Baker (1974, p. 185) as a feasible schedule in which no machine is kept idle at a time when it could begin processing an operation. We define an IIT schedule as a *feasible* schedule in which a machine is kept idle at a time when it *could begin processing an operation*, i.e., the complement of nondelay schedules. The words "could begin processing an operation" are important. There are many scheduling problems in which idle time is necessary to preserve feasibility. For example, consider the *no wait* flowshop problem. Figure 1 shows the two possible schedules for a two-machine, two-job instance. Observe that both schedules are nondelay schedules because neither machine could begin processing an operation any earlier.

Conway et al. (1967) have shown that for some problems it is unnecessary to consider idle time. There they proved that (1) for the single machine problem, (2) with all jobs simultaneously available, and (3) for a regular performance measure (nondecreasing function of job completion times), it is unnecessary to consider schedules with inserted idle time. Later they relaxed the assumption of simultaneous job arrival and showed that when a *preempt-resume* scheduling regime is enforced, the situation is essentially the same. Baker (1974, p. 13–14, 81–84, 137–138, 181) provided an essentially similar but somewhat more thorough treatment. We can immediately extend the result of Conway et al. to the case of multiple machines. To see this, consider any schedule that has an occurrence of IIT. Then some job j was deliberately delayed. Move job j (or a part of it) into the IIT period and observe no degradation in the objective function value.

These early works provide inspiration for a taxonomy of problems settings where IIT scheduling may be required. We next review the literature with the view to identify such

Figure 1. The two possible schedules for a two-machine two-job *no-wait* makespan problem.



Subject classifications: Production/scheduling: Single and multiple machine scheduling.
Area of review: MANUFACTURING OPERATIONS.

problem settings. Following the literature review we provide a taxonomy for IIT scheduling situations and map the extant literature to that taxonomy. We follow with conclusions regarding what is known about IIT scheduling and present our thoughts on where the best opportunities lie for further research.

Unless otherwise specified, we assume throughout the remainder of this paper that preemption is not allowed.

2. THE LITERATURE OF IIT SCHEDULING

In reviewing the literature we restrict our focus to only those papers that acknowledge that an IIT schedule may be required and deal with the development of solution procedures (optimum or heuristic), dominance properties, or bounds. We first review studies dealing with regular performance measures, then follow with a review of studies dealing with nonregular performance measures. In reviewing the literature we adapt the notation of Blazewicz et al. (1993, p. 47–50).

2.1. Literature on Problems with a Regular Objective Function

In the earliest known work on IIT schedules, Giffler and Thompson (1960) limited the scope of consideration to the set of active schedules. They defined an active schedule, as "... a feasible schedule having the property that no operation can be made to start sooner by permissible left shifting". Said simply, an active schedule is one in which no task's completion time can be reduced without increasing some other task's completion time. They showed that active schedules are important because they comprise a dominant set for scheduling situations in which the performance measure is regular and provided an algorithm for generating active schedules. Their focus was on the static job shop scheduling problem ($J|\ |$reg), but their results can be extended easily to the more general case of $J|r_j|$reg.

Minimizing Maximum Lateness. Several papers dealing with $1|r_j|$reg have appeared in the literature. Early work focused on designing efficient enumeration schemes for solving the problem of minimizing maximum lateness on a single machine with job ready times ($1|r_j|L_{\max}$). Lenstra (1977) showed this problem to be NP-hard and equivalent to the so-called delivery time model ($1|r_j$, delivery times$|C_{\max}$). The works of McMahon and Florian (1975), Lenstra (1977), Carlier (1982), Erschler et al. (1983), and Larson et al. (1985) are particularly relevant here because their algorithms permit inserted idle times in the schedule.

McMahon and Florian (1975) presented a novel forward scheduling procedure. Their search procedure defines a complete schedule at each node and derives a lower and an upper bound. Using a jumptracking strategy the search expands the node with the lowest lower bound value. They labeled the job that realizes maximum lateness in a schedule as a *critical job.* Their procedure holds the machine idle for the critical job j by delaying the start of jobs whose due dates are greater than that of j, and that precede j in the block containing j. The tree search stops when the lateness of the critical job at the current node is less than or equal to the least lower bound of all open nodes. Realizing that the McMahon and Florian algorithm is efficient only when $r_{\max} - r_{\min} > d_{\max} - d_{\min}$, Lenstra (1977) proposed an inversion scheme to reclaim the efficiency when $r_{\max} - r_{\min} < d_{\max} - d_{\min}$. The scheme exchanges each job's ready time with its due date to form an inverted problem (in which the ready time and due date ranges are reversed). The optimum solution of the inverted problem is reversed to obtain the optimum solution to the original problem.

Carlier (1982) presented an improvement to Schrage's algorithm (see Blazewicz et al. 1993, p. 60) to permit inserted idle times in the schedule, without any added computational burden, for the $1|r_j$, delivery times$|C_{\max}$ problem. He developed also two dominance properties and a lower bounding scheme. He used these ideas in a branch-and-bound method, which solved problems of up to 1000 jobs. Note that, as already mentioned, this problem is equivalent to the $1|r_j|L_{\max}$ problem.

Erschler et al. (1983) presented an attractive dominance property that is independent of job processing times. By ordering the jobs on the basis of their ready times and due dates it enables a smaller set of schedules to be considered. Larson et al. (1985) presented improvements to the McMahon and Florian algorithm in terms of how the sequences are constructed, how to test for optimality, and how to generate new nodes. Simons (1978) presented as sophisticated approach for solving the problem $1|r_j, p_j = p|L_{\max}$, where p is an arbitrary integer.

Minimizing Flowtime Related Measures. In what is probably the first of few papers recognizing the need to consider IIT for flowtime related measures, Bratley et al. (1971) studied the so-called deadline problem, $1|r_j, \tilde{d}_j|C_{\max}$. They devised a branch-and-bound algorithm that constructed schedules by choosing at each node a job to attach to the end of the current partial schedule. They defined a *block* as a group of jobs with the first job starting at its earliest start time and all other jobs following without delay until the end of the schedule. When a schedule has a block with the property that the earliest start time of all jobs after the first in the block have start times greater than or equal to the earliest start of the first job, then the schedule, if feasible, is optimum for $1|r_j, \tilde{d}_j|C_{\max}$. Bratley et al. (1971) used this property to test complete feasible solutions for optimality and thus, when successful, enable their algorithm to end the search.

Bianco and Ricciardelli (1982) studied the $1|r_j|\Sigma w_j C_j$ problem. They provided six different dominance properties, two tests for optimality, and a lower bounding procedure. They reported computational experience of a branch-and-bound algorithm that incorporated these ideas. Their results for problems of up to 10 jobs are encouraging. From their work it is clear that inserted idle time is important also for

other weighted measures such as total weighted flow time and maximum weighted flow time.

An interesting problem variation occurs when jobs are not permitted to leave the scheduling system early. Then the flow time for a job becomes $\max\{C_j, d_j\} - r_j$, a regular performance measure. When arrival times are not identical, it becomes necessary to consider inserted idle time to find a minimum total flow time schedule. To illustrate, consider the simple problem of two jobs (A, B) with respective ready times (0, 2), processing times (4, 6), and due dates (12, 9). The two possible schedules are obviously AB and BA. AB is a nondelay schedule with a total flowtime of 20; BA is an IIT schedule with a total flowtime of 19. Clearly, deliberate idle time can be beneficial for such problems. Kanet and Christy (1984) have shown that this problem is equivalent to the single machine tardiness problem.

Minimizing Tardiness. Numerous researchers have studied the tardiness problem. Only a few have considered IIT in their analysis. One of the earliest published methodologies for inserting deliberate idle time was provided by Carroll (1965) with his so-called *hold-off* and *sneak-in* heuristics, which he tested as an augmentation to his COVERT dispatching method for job shop scheduling ($J|r_j|\Sigma\, \tau_j T_j$). These heuristics work as follows. At time t the decision to hold off a machine (insert idle time) is made by considering the estimated cost of delay, c_j, for jobs in queue as well as those yet to arrive already tardy jobs. Let h_j be the hiatus time for job j ($h_j = \max\{0, r_j - t\}$). Select the job with largest $c_j/(p_j + h_j)$. If the selected job is a yet to arrive job, then schedule the machine to start its processing upon arrival. Search the list of considered jobs (in descending order of c_j) for the possibility of starting and completing a job before the arrival of the selected job (i.e., look for possible sneak-ins). Schedule all possible sneak-ins to start as soon as possible. Carroll's heuristics turn out to be a rather circuitous way of gauranteeing an active schedule. We describe below a more straightforward approach.

Carroll's simulation results comparing COVERT with and without hold-off and sneak-in show that the heuristics marginally but significantly (in the statistical sense) improve schedule performance. Moreover, his results give some indication that the added benefit of heuristics for inserting idle time declines with the allowance level and increases with the utilization rate. That is to say, the percent improvement in mean tardiness will be most marked in cases of loose due dates and high utilization.

In the same spirit as Carroll, heuristic methods for inserting idle time developed by Morton and Ramnath (1992) have been reported in Morton and Pentico (1993, p. 164–168). Their procedure is somewhat more elegant than Carroll's in that it is explicitly connected to utilization. They defined a *soon-to-arrive* job as one whose arrival time r is less than $(t + p_{\min})$, where $p_{\min}$ is the smallest required processing time among the waiting jobs at time t. Calculate a priority π_j for each job j in the queue and each soon-to-arrive job. Reduce the priority of a soon-to-arrive job j by application of the following: $\pi'_j = \pi_j[1 - B(r_j - t)/p_{\min}]$, where B is a constant directly proportional to the utilization level. Then choose the job with highest priority to next seize the idle machine. Their preliminary results show this procedure provides notable improvement in weighted tardiness. Their results seem to show that the marginal improvement in using hold-off heuristics is more marked in cases of lower utilization and tighter due dates—in apparent direct contrast to Carroll's observation. One explanation may be that Carroll reported raw tardiness figures, whereas Morton and Ramnath reported normalized relative tardiness values. The recent results of Sridharan and Zhou (1996a) suggest that the value of inserting idle time is indeed a function of utilization, with marked improvement when the machine is not heavily loaded. Under high utilization, there were fewer attractive opportunities to insert idle time. However, the few instances in which idle time was inserted produced substantial improvement in tardiness. This is consistent with Carroll's earlier results. Due date tightness appears unimportant. Due data range (arbitrariness) appears to have a significant and substantial effect on the improvement, with higher improvement when due date range is increased. A more detailed explanation of these interactions awaits further investigation.

We can improve the procedures used by Carroll and Morton and Ramnath to determine a soon-to-arrive jobs by directly applying the Giffler and Thompson specification for an active schedule.

PROPOSITION. *Assume a machine is idle with at least one waiting job at time t. For any regular performance measure, it is unnecessary to consider inserted idle time for any job with arrival time greater than* $\min\{r'_j + p_j\}$, *where* $r'_j = \max\{t, r_j\}$.

PROOF. Assume to the contrary, namely that some schedule S was constructed with a delay longer than $\min\{r'_j + p_j\} - t$. Then one could schedule the job with $\min\{r'_j + p_j\}$ in the idle period without delaying the completion of any other job, yieldng a schedule no worse than S. □

The implication of the proposition is that we could redefine a soon-to-arrive job as one arriving before $\min\{r'_j + p_j\}$, and obtain a smaller set than that provided by Carroll's or Morton and Ramnath's definition. Their definition unnecessarily permits considering the scheduling of jobs with arrival times in the interval $(\min\{r'_j + p_j\},\ t + p_{\min})$, whenever $\min\{r'_j + p_j\} < t + p_{\min}$. The procedure suggested here would be faster, never permit a worse solution, and would guarantee an active schedule. Using this definition, Sridharan and Zhou developed a decision theory based heuristic for the $1|r_j|\Sigma\, T_j$ problem. Via a set of simulation experiments, they demonstrated the importance of permitting inserted idle times when the due dates are arbitrary.

Chu and Portmann (1992) presented a priority rule called PRTT (Priority Rule for Total Tardiness) for the $1|r_j|\Sigma\, T_j$

problem: PRTT $(j,t) = \max\{r_j, t\} + \max\{\max\{r_j, t\} + p_j, d_j\}$. They then defined a *T-active* schedule as an active schedule in which for any pair of adjacent jobs i and j (i followed by j) either $\max\{r_i, \Delta\} < \max\{r_j, \Delta\}$ or $\text{PRTT}(i, \Delta) < \text{PRTT}(j, \Delta)$, where $\Delta = C_k$ if some job k immediately precedes i; $\Delta = -\infty$, if i is the first job in the sequence. They then proved that the set of T-active schedules is dominant for the criterion unweighted tardiness. Their priority function PRTT represents an important extension of the Modified Due Date rule studied by Baker and Bertrand (1982) and Baker and Kanet (1983) in so much as it uses a job's arrival time in computing its priority.

We can envision the utility of Chu and Portmann's result for constructing search procedures for tardiness problems. For example, consider a branch and bound algorithm which constructs schedules in a forward direction. At any stage k we have a T-active partial schedule PS_k. Branch from PS_k only with jobs for which the T-active property holds.

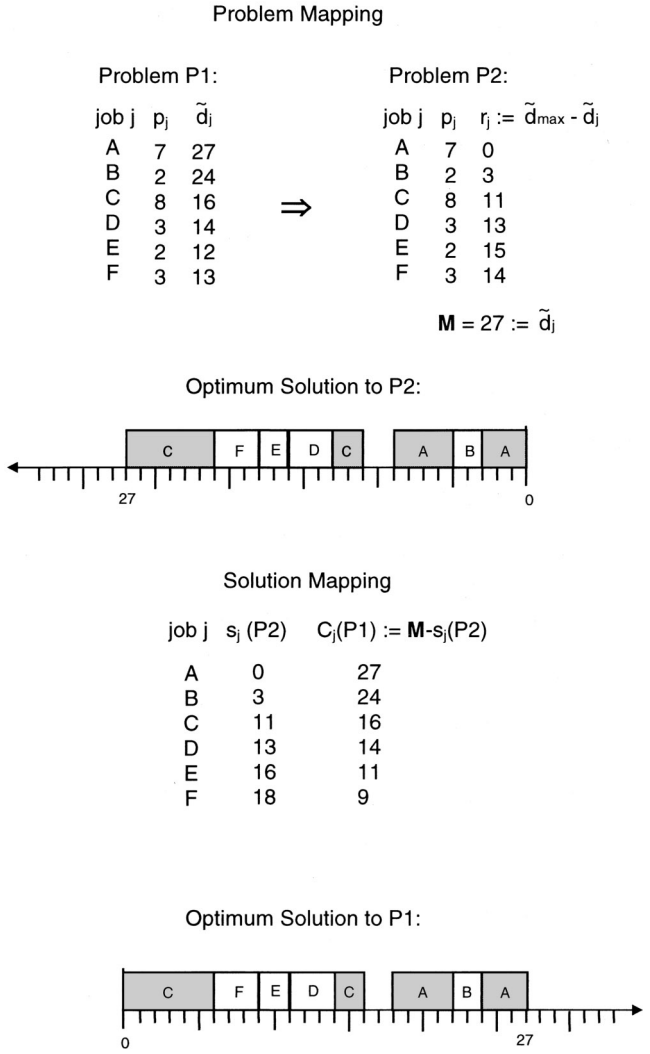
2.2. Literature on Problems with a Nonregular Objective Function

There are important scheduling problems in which the performance measure is not regular. The most obvious case is when there are penalties incurred for *earliness*. Then we see that IIT may be beneficial. A special case here is $1|\ |\Sigma E_j$, already shown to be NP-hard and equivalent to $1|\ |\Sigma T_j$ (Du and Leung 1990). A variant of this is the problem of minimizing total weighted earliness when each job must be completed by a deadline $\tilde{d}_j$ (problem $1|\tilde{d}_j|\Sigma E_j$). A simple solution procedure for the special case of $1|\text{pmtn}, \tilde{d}_j|\Sigma E_j$ can be developed as follows. Call an instance of this problem P1. Because preemption is allowed, jobs may be interrupted and split into processing segments. Now consider P2, and instance of $1|\text{pmtn}, r_j|\Sigma F_j$ subject to the constraint $C_{\max} \leqslant \mathbf{M}$. P1 is equivalent to P2, after making the following substitutions: r_j in P2 $:= \tilde{d}_{\max} - \tilde{d}_j$ in P1; $\mathbf{M}$ in P2 $:= \tilde{d}_{\max}$ in P1. Solve P2 and make the substitution: C_j in P1 $:= \mathbf{M} - s_j$ in P2, where s_j is the start time for the first segment of job j in P2. Figure 2 illustrates these substitutions.

For P2, we know from Smith (1956) that an optimum schedule is one in which the machine is kept busy with the available job (segment) with minimum remaining processing time. Because this results in a nondelay schedule, then if $C_{\max} \leqslant \mathbf{M}$ we have an optimum schedule, else no feasible solution exits. Notice that the times in P2 when the machine awaits the arrival of a job correspond to the IIT periods in P1.

For the weighted version of this problem ($1|\text{pmtn}, \tilde{d}_j|\Sigma w_j E_j$) the results are not as encouraging. Using the same reduction algorithm as above, we get $1|\text{pmtn}, \tilde{d}_j|\Sigma w_j E_j$ reduces to $1|\text{pmtn},\ r_j|\Sigma w_j F_j$. But $1|\text{pmtn}, r_j|\Sigma w_j F_j$ reduces to $1|\text{pmtn}, r_j|\Sigma w_j C_j$ which is strongly NP-hard (Labetoulle et al. 1984). So $1|\text{pmtn}, \tilde{d}_j|\Sigma w_j E_j$ is NP-hard. Similarly, $1|\tilde{d}_j|\Sigma w_j E_j$ reduces to $1|r_j|\Sigma w_j F_j$. But $1|r_j|\Sigma w_j F_j$ reduces to $1|r_j|\Sigma w_j C_j$ which is known to be NP-hard (Lenstra et al. 1977). So $1|\tilde{d}_j|\Sigma w_j E_j$ is NP-hard.

Figure 2. Reduction of $1|\text{pmtn}, \tilde{d}_j|\Sigma E_j$ to $1|\text{pmtn}, r_j|\Sigma F_j$.



Earliness/Tardiness Problems. There is a growing body of literature on the earliness/tardiness (E/T) problem. Raghavachari (1988) and Baker and Scudder (1990) have already provided reviews of that literature. However, most of the E/T work that has been reported avoids the issue of inserted idle time either by restricting the solution to be a nondelay schedule or by assuming a common due date for all jobs. For the $1|d_j = d|\Sigma g_j(E_j) + h_j(T_j)$ problem (the so-called common due date problem), Cheng and Kahlbacher (1991) proved that it is unnecessary to consider schedules with inserted idle time except prior to the first job in the schedule. Their result holds for any cost function of the form $\Sigma_{j=1}^{n} f(C_j - d)$, where $f(\cdot)$ is nonincreasing in the interval $[-\infty, 0)$, nondecreasing in the interval $(0, \infty]$ and $f(0) = 0$. The reader can assume that any study mentioned by Raghavachari or Baker and Scudder and not included here either makes the restricting nondelay assumption or deals with a problem for which the Cheng-Kahlbacher result holds. In the first case, such papers are not directly relevant to the issue of inserted idle time. We agree with Baker and

Scudder's observation that the essence of the E/T problem lies in its nonregular performance measure and to impose the arbitrary restriction that there be no idle time diminishes the importance of this objective. Because Baker and Scudder have provided an extensive review of the literature on the common due date problem, and in light of Cheng and Kahlbacher's result, we refrain from repeating such a review here. After taking all this into consideration the IIT-E/T literature is scanty. We can characterize the available literature into four broad groups of papers dealing with (1) optimizing procedures (2) special purpose E/T heuristics, (3) heuristic search procedures, and (4) timetabling algorithms.

Optimizing Procedures. Mixed integer programming formulations have been presented by Fry et al. (1987), Coleman (1992) and Balakrishnan et al. (1997). Branch-and-bound schemes have been developed by Fry et al. (1986, 1987). Fry et al. considered the single machine problem of minimizing a weighted mixture of flow time, earliness, and tardiness. Coleman (1992) formulated a single machine problem with sequence dependent setup times and earliness/tardiness penalties.

Balakrishnan et al.'s formulation extends the models of Fry et al. and Coleman to include multiple parallel uniform machines, *sequence dependent* setups, and job ready times ($Q|r_j, \text{setups}|\Sigma\, \varepsilon_j E_j + \tau_j T_j$). They assumed that processing times on a machine m are scaled by a factor $\alpha_m \leqslant 1$. With their formulation, they were able to solve eight-job, two-machine problems with an average of 8175 pivots in about 30 seconds, while 10-job, two-machine problems required an average of about 50,000 pivots and about four minutes on a 333-Mhz Pentium processor. Recognizing the discouraging nature of these results, the authors described a Bender's decomposition approach for separating the problem into an integer master problem that focuses on finding the machine assignments and the sequence in which jobs are processed, and a continuous valued linear subproblem that focuses on finding the exact completion time of each job.

E/T Heuristics. Special purpose E/T heuristics have been proposed by Mannur and Addagatla (1993), Nandkeolyar et al. (1993), and Sridharan and Zhou (1996b).

Mannur and Addagatla developed two heuristics for E/T problems with machine "vacations," one of which permits schedules with inserted idle time. Their limited results show the nondelay heuristic to be superior, but their problem instances were all such that the utilization was so high as to always cause a nondelay schedule to be optimum.

Nandkeolyar et al. studied the single machine $\Sigma\, w_j(E_j + T_j)$ problem with dynamically arriving jobs and proposed a two-step modular approach. In the first step, a marginal cost analysis is performed in order to decide whether or not to keep the machine idle in anticipation of an important soon-to-arrive job. In the second phase, they deployed and tested the performance of various dispatching rules to select a job to next occupy the machine. They also optionally used a so-called "balancing routine" to timetable the final schedule. It is difficult to assess the quality of their approach because no comparison to optimum solutions was made available.

Sridharan and Zhou presented a *nearly online* (Sanlaville 1995) scheduling heuristic of complexity $O(n^2)$ for the $1|r_j|\Sigma\, \varepsilon_j E_j + \tau_j T_j$ problem. Their heuristic identified soon-to-arrive jobs and kept the machine deliberately idle for them. At each decision epoch t, their heuristic looked ahead to $\max\{t + p_j, d_j\}$ to identify arriving jobs. Thus, the candidate job set at t included all jobs in the queue and soon-to-arrive jobs. The heuristic, based on a decision theoretic approach, proceeds to select the best job to schedule next as follows. First, C_j, the best completion time of job j, is determined as if it were processed next. Assuming all remaining jobs follow j in a nondelay mode, the average completion time of the remaining jobs is estimated using their average processing time. If the average completion time of unscheduled jobs is greater than their average due date, then C_j is adjusted accordingly, provided it is feasible and economical. Upon determining the best completion time of job j, the completion times of remaining jobs are estimated using their individual processing times and the average processing time of all unscheduled jobs. Using these estimates the total cost of scheduling j next is obtained. Repeating this process for each job in the candidate job set at time t, they obtain an estimate of the cost consequence of scheduling each job next and select the job that produces the lowest cost to process next. They tested their heuristic on the 116 published static problems in Davis and Kanet (1993) and Yano and Kim (1991). Their heuristic was found faster than the heuristics of both Yano and Kim and Davis and Kanet, and it produced superior results. In additional tests involving dynamic problems with up to 5000 jobs, and under a variety of conditions, their heuristic was found to consistently outperform adapted (by incorporating the above described look-ahead feature) version of EXP-E/T (Ow and Morton 1989) and EDD to handle dynamic E/T problems.

Heuristic Search. In this category of papers the focus has been on either neighborhood search method development or application of genetic algorithms.

Fry et al. (1990) proposed an adjacent pairwise exchange heuristic for solving $1|\ |\Sigma\, E_j + T_j$. Using a set of nine precedence relationship rules to reduce the number of candidates for interchanging jobs and a straightforward linear programming formulation to timetable the resulting sequences, they were able to solve problems of up to 16 jobs, finding an optimum solution in 122 of 192 test problems.

Yano and Kim (1991) and Kim and Yano (1994) considered two cases of the $1|\ |\Sigma\, \varepsilon_j E_j + \tau_j T_j$ problem: when $\varepsilon_j = \tau_j$ for all j; and when ε_j and τ_j are proportional to the job processing times and the restriction that $0 \leqslant \varepsilon_j \leqslant \tau_j$. They provided a branch-and-bound method and a pairwise interchange heuristic and demonstrated their use in solving problems of up to 30 jobs. They were able to obtain the optimum solution for 99 of the 100 problems considered.

Keyser and Sarper (1991) also developed a pairwise interchange heuristic. They presented a target start time heuristic

to minimize the sum of earliness, tardiness, and waiting time costs. The heuristic permits machine idle times between jobs in the schedule produced. The heuristic solution is improved using an adjacent pairwise interchange algorithm. They formulated and solved the problem as a mixed integer program and compared their heuristic for problems of up to six jobs. Their results are encouraging, albeit limited.

Both Kanet and Sridharan (1991) and Lee and Choi (1995) developed genetic-based algorithms for E/T problems. Kanet and Sridharan investigated the problem of n jobs with nonidentical ready times and sequence-dependent setup times to be scheduled on m uniform machines, with the convex objective function $\Sigma_{j=1}^{n} \varepsilon_j E_j + \tau_j T_j + \sigma_j SU_j$, where SU_j represents setup time for job j. Their algorithm creates successive *generations* of schedules, with each generation inheriting the characteristics of a subset of the prior generation. To avoid convergence to a local optimum, the algorithm has a *mutation* feature, regulated by the algorithm's rate of convergence. That is, as the successive improvement in schedule populations begins to diminish, the probability of the appearance of *mutant* schedules is increased. It is difficult to assess the quality of their procedure because they made no comparisons to optimum solutions.

Lee and Choi (1995) presented a search procedure for $1|\ |\Sigma \varepsilon_j E_j + \tau_j T_j$ problems to generate near-optimum sequences using crossover and mutation operators and linear scaling of the fitness function. They used an embedded timetabling procedure to determine the optimum starting times of jobs in a sequence by inserting idle times when necessary. They solved up to 80 job problems with both proportional and general penalty weights. Compared to Yano and Kim's heuristic, their algorithm produced 12% to 33% lower total cost, for a set of random problems, especially when the problem size is increased and penalty weights are general, albeit at increased computational times.

Timetabling Algorithms. The issue of finding best ways for timetabling a given job sequence has attracted the attention of a number of researchers. Starting with Sidney in 1977, timetabling procedures have been proposed by Lakshminarayan et al. (1978), Garey et al. (1988), Davis and Kanet (1993), Lee and Choi (1995), and Szwarc and Mukhopadhyay (1995). Fry et al. (1984) developed linear programming formulation to timetable jobs. Faaland and Schmitt (1987, 1993) formulated the timetabling problem as a maximum network flow model and described a real-life implementation.

Sidney's (1977) work is possibly the first appearance of a study involving E/T problems. He studied the $1\|\max\{g(\max\{E_j\}), h(\max\{T_j\})\}$ problem, where both g and h are monotonically nondecreasing continuous functions such that $g(0) = h(0) = 0$. For each job j there is a target start time a_j and a target completion time (due date) $b_j > a_j$. These parameters have the property that if $a_i < a_k$, then $b_i \leqslant b_k$. This condition assures there is at least one optimal schedule with the property that the jobs are simultaneously ordered by nondecreasing a_j and nondecreasing b_j, making it trivial to obtain an optimum permutation. Given the permutation, he then computed an upper bound for E_j and T_j and used these bounds to timetable the jobs using a simple two-step procedure. Sidney's algorithm was refined by Lakshminarayan et al. (1978), who improved the complexity from $O(n^2)$ to $O(n \log n)$.

The work of Garey et al. (1988) is probably the most comprehensive treatment of timetabling algorithms for E/T problems. They, in fact, addressed two problems: $1\|\Sigma E_j + T_j$, and $1\|\max\{E_j, T_j\}$ and several of their variants. In addition to showing that the $1\|\Sigma E_j + T_j$ problem is NP-hard, they also provided an $O(n \log n)$ timetabling algorithm for the case when a sequence is given. They showed that the variant $1|p_j = p|\Sigma E_j + T_j$ can be solved by first sorting the jobs in nondecreasing order of due date and then applying the timetabling procedure (still $O(n \log n)$). They showed also that the timetabling algorithm can be altered, without added complexity, to $1\|\Sigma w_j(E_j + T_j)$ and to the cases when *window constraints* or *consecutive task constraints* are present. Window constraints occur when each job j is given a window of time $[u_j, v_j]$ in which the job must start, with the restriction that $v_j + p_j \leqslant v_{j+1}$. Consecutive task constraints occur when for sets of jobs $\{J_j, J_{j+1}, \ldots, J_k\}$, job J_i is constrained to start immediately after job J_{i-1} for $i = j+1, \ldots, k$.

Note that the $1\|\max\{E_j, T_j\}$ problem is a reduction of the objective defined by Sidney (1977) in problem $1\|\max\{g(\max\{E_j\}), h(\max\{T_j\})\}$. Garey et al. (1988) were able to find a *pseudopolynomial* time algorithm for the $1\|\max\{E_j, T_j\}$ problem without Sidney's restriction on target start and target completion times (if $a_i < a_k$, then $b_i \leqslant b_k$). The algorithm of Garey et al. is of complexity $O(n(\log n + \log p_{\max}))$. It remains open whether or not a polynomial time (in n) algorithm can be found for the unrestricted version of the $1\|\max\{g(\max\{E_j\}), h(\max\{T_j\})\}$ problem.

Davis and Kanet (1993) tackled the case where the penalties are general convex functions of earliness and tardiness $(1\|\Sigma g_j(E_j) + h_j(T_j))$ and proposed a pseudopolynomial algorithm, i.e., complexity $O(nH)$, where H is the number of units of time in the planning horizon.

Szwarc and Mukhopadhyay (1995) provided an efficient timetabling algorithm for the $1\|\Sigma \varepsilon_j E_j + \tau_j T_j$ problem. They showed that the solution will be composed of $m \leqslant n$ clusters of uninterrupted jobs, possibly separated by idle periods. They observed and proved that the cluster partitions can be determined in advance (i.e., before actually deciding the size of the idle periods). Clusters can be identified by observing that within the sequence of n jobs, for any two adjacent jobs a, b, to be in a cluster: $d_b - d_a \leqslant p_b$ must hold. They showed that for any cluster, the tardy jobs are always preceded by the early jobs, that the earliness of consecutive jobs in a cluster is nonincreasing, and that the tardiness of consecutive jobs in a cluster is nondecreasing. They then provided an efficient two-stage procedure for first identifying clusters and then timetabling them. An essentially equivalent algorithm has been independently developed by Lee and Choi.

Considering the $1\|\Sigma\,\varepsilon_j E_j + \tau_j T_j$ problem, Fry et al. (1984) described a straightforward linear program formulation that produces an optimum timetable for a given sequence in one pivot. Faaland and Schmitt (1987, 1993) developed and tested a two-phase sequencing-timetabling procedure for a multimachine job shop problem ($J|r_j|\Sigma\,\varepsilon_j E_j + \tau_j T_j$). In the first phase, the jobs are forward loaded according to precedence relationships to create the dispatching sequence at the work centers. Given the Phase 1 sequence, Phase 2 of the procedure formulates the problem as a maximum network flow model and iteratively reschedules (timetables) the tasks to minimize total cost. In a subsequent paper they reported application of their approach to a real factory with over 26,000 tasks and 52 work centers. Theirs is the first reported case of acknowledging the importance of inserted idle time in the design of a real-life production scheduling system.

3. A TAXONOMY FOR IIT SCHEDULING PROBLEMS

The literature leads us to three major situations (problem parameters) in which it may be sensible to deliberately introduce idle time into a schedule:

SITUATION 1: When there is more than one processor.

SITUATION 2: When there are jobs with nonidentical ready times.

SITUATION 3: When the scheduling performance measure is nonregular.

Notice that the union of these situations forms the complement of the intersection of the three special conditions of Conway et al. (1967) in describing when IIT is *not* required. Figure 3 presents a Venn diagram describing the relationship of these three classes of scheduling problems.

At the core of the diagram in Figure 3 is the problem specification of Conway et al. (1967), namely a single machine, jobs with identical ready times, and a regular performance measure. The remaining sets, numbered 1 through 7, identify cases where inserted idle time may be required. The relevant problem sets are:

Group 1 ($1|r_j|$reg): Single machine, nonidentical ready times, regular performance measure.

Group 2 ($m\,|\,|$reg): Multimachine, identical ready times, regular performance measure.

Group 3 ($1|\,|$nonreg): Single machine, identical ready times, nonregular performance measure.

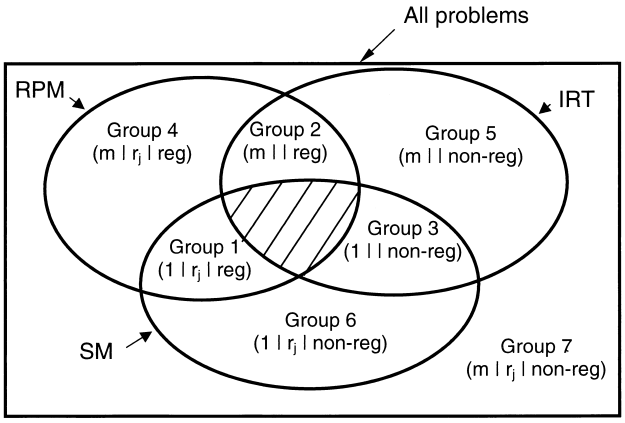
Group 4 ($m|r_j|$reg): Multimachine, nonidentical ready times, regular performance measure.

Group 5 ($m\,|\,|$nonreg): Multimachine, identical ready times, nonregular performance measure.

Group 6 ($1|r_j|$nonreg): Single machine, nonidentical ready times, nonregular performance measure.

Group 7 ($m|r_j|$nonreg): Multimachine, nonidentical ready times, nonregular performance measure.

Figure 3. Venn diagram showing groups of scheduling problems where inserted idle time may be required.



RPM : Regular Performance Measure Problems
IRT : Problems Where Jobs Have Identical Ready Time
SM : Single Machine Problems
▨ : Problems where inserted idle time is not necessary

Table 1 maps the extant IIT literature according to the group structure defined in Figure 3.

4. CONCLUSIONS

Figure 4 illustrates the relationship of timetabled, active, and nondelay schedules and shows what is known about the search space for various problem groups. The innermost set (nondelay schedules) dominates for $1|\,|$reg, $1|$pmtn, $r_j|$reg, and $m|$pmtn, $r_j|$reg problems. The set of active schedules, which includes nondelay schedules, dominates for $m|r_j|$reg, and may contain IIT schedules. The outermost set, timetabled schedules, dominates for $m|r_j|$nonreg problems. We define a timetabled schedule as a schedule in which no local shift (left or right) can reduce the objective function value. Note that the set of timetabled schedules is in fact a generalization of the set of "semi-active" schedules described by Giffler and Thompson (1960). A semi-active schedule is achieved by removing all superfluous idle time appearing to the left of every job in the schedule (i.e., a special case of timetabling). The descriptive work of Kanet (1981) has shown that the set of active schedules, although dominant over the set of nondelay schedules, is significantly larger. This is where contributions like those of Bianco and Ricciardelli (1992), Carlier (1982), Erschler et al. (1983) and Chu and Portmann (1992) play a role. In each case the results serve to reduce the required search space within active schedules for a specific scheduling objective.

4.1. Research Opportunities

We see several areas where further research in inserted idle time scheduling might prove beneficial. These areas

Table 1. A Mapping of the Extant IIT Literature.

Problem Group	Investigator(s)
Group 1 ($1\|r_j\|$reg)	
$1\|r_j,\ p_j = p\|L_{\max}$	Simons (1978)
$1\|r_j\|L_{\max}$	McMahon and Florian (1975); Lenstra (1977); Carlier (1982); Erschler, Fontan, Merce, and Roubellat (1983); Larson, Dessouky, and Devor (1985)
$1\|r_j$, delivery times$\|C_{\max}$	Bratley, Florian, and Robillard (1971)
$1\|r_j,\ \tilde{d}_j\|C_{\max}$	Bianco and Ricciardelli (1982)
$1\|r_j\|\Sigma w_j C_j$	Chu and Portmann (1992); Sridharan and Zhou (1996a)
$1\|r_j\|\Sigma T_j$	
Group 2 ($m\|\|$reg)	
$J\|\|$reg	Giffler and Thompson (1960)
Group 3 ($1\|\|$non-reg)	
$1\|$pmtn$, \tilde{d}_j\|\Sigma E_j$	Kanet and Sridharan (2000)
$1\|d_j = d\|\Sigma g_j(E_j) + h_j(T_j)$	Cheng and Kahlbacher (1991)
$1\|\|\max\{g(\max\{E_j\}), h(\max\{T_j\})\}$	Sidney (1977); Lakshminarayan, Lakshmanan, Papinou, and Rochette (1978)
$1\|\|\max\{E_j, T_j\}$	Garey, Tarjan, and Wilfong (1988)
$1\|\|\Sigma E_j + T_j$	Garey, Tarjan, and Wilfong (1988); Kim and Yano (1994)
$1\|\|\Sigma \varepsilon_j E_j + \tau_j T_j$	Fry, Armstrong, and Blackstone (1984); Fry, Darby-Dowman, and Armstrong (1986); Fry, Leong, and Rakes (1987); Fry, Armstrong, and Rosen (1990); Yano and Kim (1991); Lee and Choi (1995); Szwarc and Mukhopadhyay (1995)
$1\|\|\Sigma g_j(E_j) + h_j(T_j)$	Davis and Kanet (1993)
$1\|$setups$\|\Sigma \varepsilon_j E_j + \tau_j T_j$	Coleman (1992)
Group 4 ($m\|r_j\|$reg)	
$J\|r_j\|\Sigma w_j T_j$	Carroll (1965); Morton and Ramnath (1992)
Group 5 ($m\|\|$non-reg)	No known work
Group 6 ($1\|r_j\|$non-reg)	
$1\|r_j\|\Sigma E_j + T_j$	Mannur and Addagatla (1993)
$1\|r_j\|\Sigma w_j(E_j + T_j)$	Nandkeolyar, Ahmed, and Sundararaghavan (1993)
$1\|r_j\|\Sigma \varepsilon_j E_j + \tau_j T_j$	Keyser and Sarper (1991); Sridharan and Zhou (1996b)
Group 7 ($m\|r_j\|$non-reg)	
$J\|r_j\|\Sigma \varepsilon_j E_j + \tau_j T_j$	Faaland and Schmitt (1987, 1993)
$Q\|r_j,$ setups$\|\Sigma \varepsilon_j E_j + \tau_j T_j$	Kanet and Sridharan (1991); Balakrishnan, Kanet, and Sridharan (1997)

can be organized according to the following interrelated categories:

1. Further development of algorithms and dominance properties.
2. Integration of timetabling into search procedures.
3. Development of heuristic methods for constructing inserted idle time schedules.

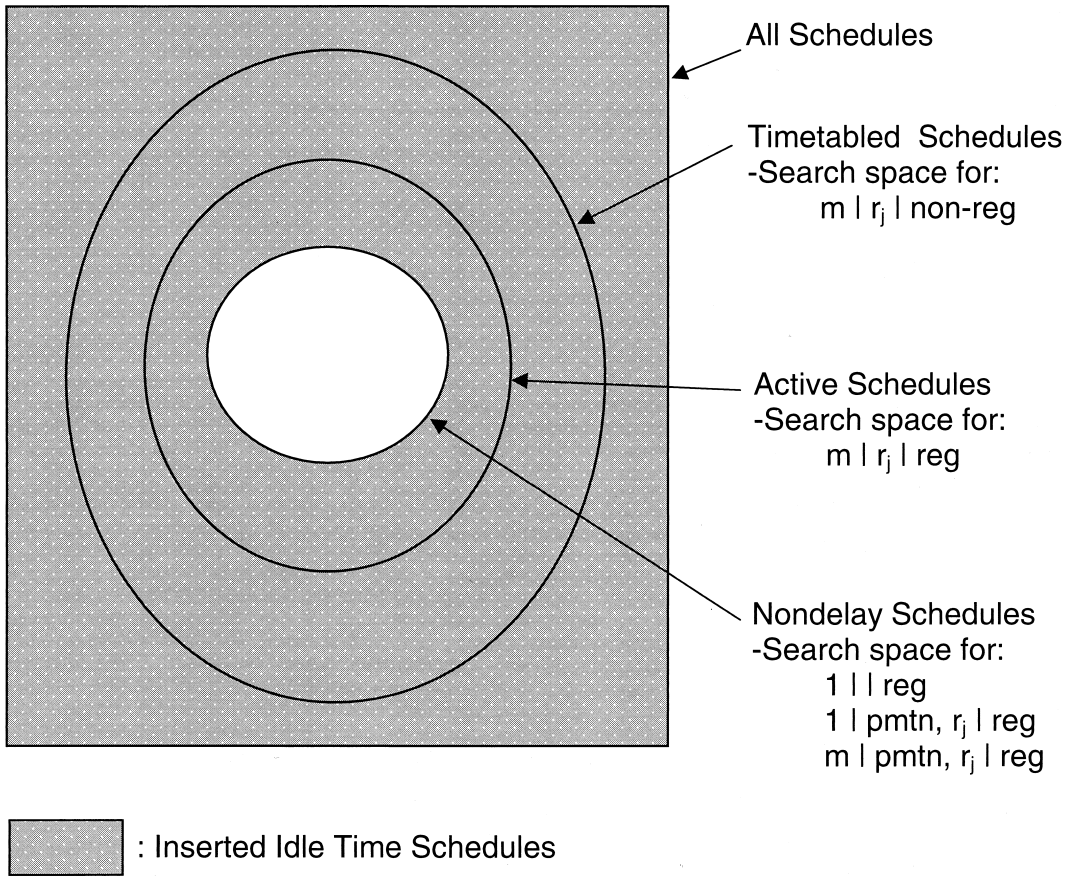
Further Development of Algorithms and Dominance Properties. The separation of scheduling into sequencing and timetabling has obvious implications on strategies for the construction of schedules. (This point shall be developed further in the paragraphs to follow.) Aside from this, however, the availability of pure timetabling procedures may have practical advantages. For example, consider the scheduling of preventive maintenance in a factory. We want to schedule such maintenance when it causes the least disruption to the production schedule, i.e., when machines are idle. Given any current schedule of production, a timetabling algorithm could be deployed to reassign the idle time of machines to greatest advantage, allowing the maintenance to occur when least disruptive.

When the objective function is *well behaved* (i.e., piecewise linear), then the algorithm of Szwarc and Mukhopadhyay (1995) could be adapted to efficiently re-timetable jobs. However, for more general cost functions, the available algorithm is that of Davis and Kanet (1993) with complexity $O(nH)$. Here may be an opportunity for further development along two avenues: either by exploiting the properties of a specific type of function (e.g., quadratic), or by deployment of general line search methods such as interval bisection or golden section (e.g., see Wagner 1977, p. 539). So, there seem to be a number of opportunities for further development of timetabling procedures.

In the area of complexity analysis, the unrestricted problem $1\|\max\{g(\max\{E_j\}),\ h(\max\{T_j\})\}$ and its variants, $1\|\max\{\varepsilon_j E_j,\ \tau_j T_j\}$, $1\|w_j\ \max\{E_j,\ T_j\}$, and $1\|\max\{E_j,\ T_j\}$, remain open for analysis. It is yet to be shown whether these problems have polynomial (in n) time algorithms or if they belong to the class of NP-hard problems.

There seem to be a number of opportunities also for developing dominance properties. For example, it may be possible to exploit the results of Bianco and Ricciardelli (1992) when addressing the $1|\tilde{d}_j|\Sigma \varepsilon_j E_j$ problem. Such an

Figure 4. Venn diagram illustrating the relation of timetabled schedules to active and nondelay schedules.



extension would be analogous to the procedure we outlined earlier for mapping the constrained weighted earliness problem $(1|\text{pmtn}, \tilde{d}_j|\Sigma \varepsilon_j E_j)$ to the weighted completion time problem $(1|\text{pmtn}, r_j|\Sigma w_j C_j)$. Bianco and Ricciardelli's theorems for establishing dominance properties between adjacent jobs for $1|r_j|\Sigma w_j C_j$ may have a possible counterpart for $1|\tilde{d}_j|\Sigma w_j E_j$. In a similar vein, the work of Chu and Portmann (1992) on establishing dominance properties for $1|r_j|\Sigma T_j$ might be extendible to other related problems. An obvious first step might be the $1|r_j|\Sigma \tau_j T_j$ problem. For this problem it should be possible to build on the dominance property for $1\|\Sigma \tau_j T_j$ already developed by Rachamadugu (1987). For E/T problems, similar extensions may be possible. We know, for example, from Du and Leung (1990) that $1\|\Sigma T_j$ and $1\|\Sigma E_j$ are equivalent. So, analogs of Chu and Portmann's results to $1|r_j|\Sigma E_j$ or even $1|r_j|\Sigma \varepsilon_j E_j + \tau_j T_j$ may be possible. This would serve to reduce the search space for certain E/T problems to a smaller set than the set of timetabled schedules (refer to Figure 4).

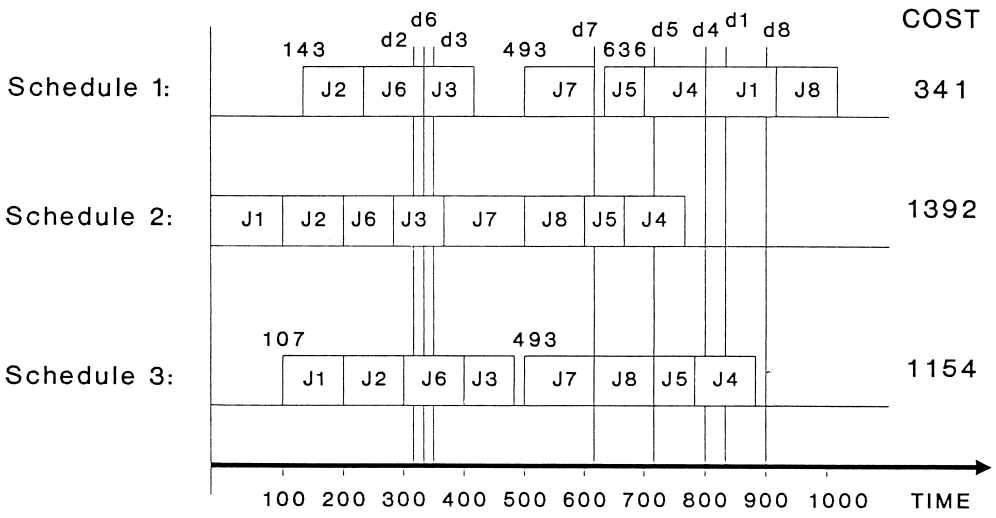
Integration of Timetabling into Search Procedures. There is a growing body of knowledge in the area of advanced computer search methods for scheduling. The types of approaches we refer to here include methods such as heuristic branch and bound, simulated annealing, beam search, tabu search, etc. For a review of these methods see Morton and Pentico (1993). Other approaches that appear promising include application of genetic algorithms (see Kanet and Sridharan 1991 and Lee and Choi 1995), application of basic decision theory (see Chryssolouris et al. 1988, Kanet and Zhou 1993, Sridharan and Zhou 1996a and 1996b), and application of neural networks (see Johnston and Adorf 1992). All these methods distinguish themselves from simple forward simulations in that they may include a (limited) capability for backtracking and/or the feature of dynamically changing the search path.

Our earlier discussion regarding the separation of the scheduling task into sequencing and timetabling leads to the tempting conclusion that the application of such search approaches to problems involving inserted idle time might be quite simple. For example, a tempting heuristic might be to first ignore any inserted idle time and deploy a search procedure for identifying a good permutation. Then, having identified the final sequence, to timetable it using a timetabling algorithm to insert idle times. This strategy of wanton separation of sequencing and timetabling can be dangerous as illustrated by the example depicted in Figure 5 of an eight-job instance of $1\|\Sigma E_j + T_j$.

The figure shows three schedules for the sample problem. Schedule 1 is an optimum schedule; its total cost is 341. Schedule 2 is an optimum schedule under the constraint that no inserted idle time is permitted; i.e., Schedule 2 is an

Figure 5. Illustration of the effect of separating sequencing and timetabling.

Job:	J1	J2	J3	J4	J5	J6	J7	J8
Process time:	116	94	80	103	57	91	115	104
Due date (d_i):	835	318	354	797	714	329	609	909
Schedule 1 (C_i):	912	237	408	796	693	328	608	1016
Schedule 2 (C_i):	116	210	381	760	657	301	496	600
Schedule 3 (C_i):	223	317	488	872	769	408	608	712



Schedule 1: An optimal schedule for the stated problem
Schedule 2: An optimal schedule when delays are forbidden
Schedule 3: Schedule 2 after optimal allocation of idle time

Objective: minimize $\text{cost} = \sum_{i=1}^{8} |C_i - d_i|$

optimum nondelay schedule and costs 1392. Schedule 3, obtained by timetabling Schedule 2, is far from optimum with a cost of 1154. Note the drastic difference in sequence for Schedule 1 and Schedule 2. This example serves to illustrate that a simple strategy of first ignoring timetabling to find a sequence when the performance measure is nonregular can lead to significantly suboptimum performance. Yet this is the procedure deployed by Faaland and Schmitt and Yano and Kim. We know that Lee and Choi embedded a timetabling algorithm in their genetic-based search procedure and obtained significantly better results in a direct comparison with the procedure of Yano and Kim. The explanation may well lie in their integration of timetabling into the search procedure. An important direction for research would be to more thoroughly investigate this phenomenon. In the application of search methods to E/T problems there appears to be a need to develop procedures for embedding timetabling into the fabric of the search procedure. One area of development, for example, rests in the observation that all the timetabling procedures discussed here index through the *complete* set of jobs, starting with the last job in the sequence. Efficient timetabling procedures operating on *partial* schedules that comprise a subsequence of either the first jobs or the last jobs in a schedule, however, would seem worthy of development. Such procedures could prove valuable in branch-and-bound approaches where it is necessary to be able to calculate a lower bound for a given partial schedule.

Development of Heuristics for Construction of IIT. The development of heuristics for construction of inserted idle time schedules is important for two reasons: (1) IIT scheduling problems are extremely complex, so exact solution methods may never be practical; and (2) In practice, the problem definition is under constant revision because of the dynamic nature of real-life scheduling environments, so that quick solutions are an absolute necessity.

The works of Morton and Ramnath (1992) and Sridharan and Zhou (1996a) indicate the potential fruitfulness of this research theme for single machine tardiness problems. An interesting follow-up would be to examine the effects of redefining soon-to-arrive jobs as suggested here and report the computational experience. Another extension would be to investigate the behavior of these types of procedures to problems with sequence dependent

setup times. Yet another interesting extension would be to see how such idle time insertion procedures might be designed/adapted for situations when the penalty function is nonregular (e.g., when earliness costs also come into play). We know that for such problems active schedules do not dominate. So a rethinking of the concept soon-to-arrive might well be warranted. (For an initial effort along this line of research see Sridharan and Zhou 1996b.) Likewise, as discussed earlier, the effect of environmental variables such as utilization, due date tightness, and due date range on the improvement in performance with the use of such hold-off heuristics seems to warrant further clarification. Finally, from the practitioner's point of view, nearly on-line algorithms seem to hold the maximum potential for use, whereas virtually all published research has focused on *off-line* algorithms. In this context, the works of Nandkeolyar et al. (1993) and Sridharan and Zhou (1996b) are worth noting because they both presented heuristic procedures which assumed minimum forward visibility and thus may be considered nearly on-line. Additional research extending and improving their heuristics by incorporating queuing theory based busy period analysis to determine the look-ahead window for nearly on-line algorithms may prove extremely fruitful and valuable.

In addition to further study of combining hold-off and sneak-in heuristics to priority dispatching methods a' la Carroll (1965) and Morton and Ramnath (1992), a decision theory approach as described by Chryssolouris et al. (1988) and Kanet and Zhou (1993) might be sucessfully adapted to include inserted idle time as one of the alternatives. In the decision theory approach a schedule is constructed in a forward direction (as with a dispatching approach). A decision point corresponds to the event that a resource has become available. A decision alternative corresponds to the selection of a waiting job from the queue. At each decision point, the total cost of an extended schedule corresponding to each decision alternative is estimated. The most favourable (least estimated cost) alternative is then chosen and the construction program advances to the next decision point. An interesting question from here is to what extent might system performance be improved by including the additional alternative *leave machine idle*, i.e., including the possibility of inserting idle time. (For an initial effort along this line of research see Sridharan and Zhou 1996a, 1996b.) A related issue concerns estimating job completion times to obtain an estimate of the total cost of an extended schedule.

4.2. Summary

We have defined here an inserted idle time schedule as a feasible non-nondelay schedule. IIT schedules are relevant when there is more than one processor, or when there are jobs with nonidentical ready times, or when the objective function is not regular. Considering the importance of this topic, the amount of work reported to data is meager, and there appears to be ample opportunity and need for further research. We see research opportunities in the further development of algorithms and dominance properties, in the integration of timetabling into search procedures, and in the development of heuristic methods for the construction of inserted idle time schedules.

ACKNOWLEDGMENTS

The authors thank an anonymous reviewer for many insightful comments and suggestions on earlier versions of this paper. This research was supported in part by contracts 916T4152 and 926T11255 from the McDonnell Douglas Space Division, Houston, Texas.

REFERENCES

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

——, J. W. M. Bertrand. 1982. A dynamic priority rule for scheduling against due-dates. *J. Oper. Management* **3**(1) 37–42.

——, J. J. Kanet. 1983. Job shop scheduling with modified due dates. *J. Oper. Management* **4**(1) 11–22.

——, G. D. Scudder. 1990. Sequencing with earliness and tardiness penalties: a review Oper. Res. **38**(1) 22–36.

Balakrishnan, N., J. J. Kanet, V. Sridharan. 1999. Early/tardy scheduling with sequence dependent setups on uniform parallel machines. *Comput. Oper. Res.* **26** 127–141.

Bianco, L., S. Ricciardelli. 1992. Scheduling of a single machine to minimize total weighted completion time subject to release dates. *Naval Res. Logist. Quart.* **29**(1) 151–167.

Blazewicz, J., K. Ecker, G. Schmidt, J. Weglarz. 1993. *Scheduling in Computer and Manufacturing Systems.* Springer-Verlag, Berlin.

Bratley, P., M. Florian, P. Robillard. 1971. Scheduling with earliest start and due date contrainsts. *Naval Res. Logist. Quart.* **18**(4) 511–517.

Carlier, J. 1982. The one-machine sequencing problem. *European J. Oper. Res.* **11** 42–47.

Carroll, D. C. 1965. Heuristic sequencing of jobs with single and multiple components. Ph.D. Dissertation. Sloan School of Management, MIT, Cambridge, MA.

Cheng, T. C. E., H. G. Kahlbacher. 1991. A proof of the longest-job-first policy in one-machine scheduling. *Naval Res. Logist.* **38** 715–720.

Chryssolouris, G., K. Wright, J. Pierce, W. Cobb. 1988. Manufacturing systems operation: dispatch rules versus intelligent control. *Robotics & Computer-Integrated Manufacturing* **4**(3/4) 531–544.

Chu, C., M.-C. Portmann. 1992. Some new efficient methods to solve the $n/1/r_i/\Sigma T_i$ scheduling problem. *European J. Oper. Res.* **58** 404–413.

Coleman, B. J. 1992. A simple model for optimizing the single machine early/tardy problem with sequence dependent setups. *Production and Oper. Management* **1**(2) 225–228.

Conway, R. W., W. L., Maxwell, L. W. Miller. 1967. *Theory of Scheduling*. Addison-Wesley, Reading, MA.

Davis, J. S., J. J. Kanet, 1993. Single machine scheduling with early and tardy completion costs. *Naval Res. Logist.* **40** 85–101.

Du, J., J. Y.-T. Leung. 1990. Minimizing total tardiness on one machine is NP-hard. *Math. Oper. Res.* **15** 483–495.

Erschler, J., G. Fontan, C. Merce, F. Roubellat. 1983. A new dominance concept in scheduling *n* jobs on a single machine with ready times and due dates. *Oper. Res.* **31**(1) 114–127.

Faaland, B., T. Schmitt. 1987. Scheduling tasks with due dates in a fabrication/assembly process. *Oper. Res.* **35** 378–388.

——, ——. 1993. Cost-based scheduling of workers and equipment in a fabrication and assembly shop. *Oper. Res.* 253–268.

Fry, T. D., R. D. Armstrong, J. H. Blackstone. 1984. Minimizing weighted absolute deviation in single machine scheduling. IIE *Trans.* **19** 445–450.

——, ——, L. D. Rosen. 1990. Single machine scheduling to minimize mean ablosute lateness: a heuristic solution. *Computers and Oper. Res.* **17**(1) 105–112.

——, K. Darby-Dowman, R. D. Armstrong. 1986. Single machine scheduling to minimize mean absolute lateness. *Computers and Oper. Res.* **23**(2) 171–182.

——, K. Leong, T. Rakes. 1987. Single machine scheduling: a comparison of two solution procedures. *OMEGA* **15** 277–282.

Garey, M., R. Tarjan, G. Wilfong, 1988. One-processor scheduling with symmetric earliness and tardiness penalties. *Math. Oper. Res.* **13** 330–348.

Giffler, B., G. L. Thompson. 1960. Algorithms for solving production scheduling problems. *Oper. Res.* **8**(4) 487–503.

Johnston, M. D., H.-M. Adorf. 1992. Scheduling with neural networks — the case of the Hubble telescope. *Computers and Oper. Res.* **19**(3/4) 209–240.

Kanet, J. J. 1986. Tactically delayed versus non-delay scheduling: an experimental investigation. *European J. Oper. Res.* **24**(1) 99–105.

——, D. P. Christy. 1984. Manufacturing systems with forbidden early order departure. *Internat. J. Production Res.* **22**(1) 41–50.

——, V. Sridharan. 1991. PROGENITOR, a genetic algorithm for production scheduling. *Wirtschaftsinformatik* **33**(4) 332–336.

——, Z. Zhou. 1993. A decision theory approach to priority dispatching for job shop scheduling. *Production and Oper. Management* **2**(1) 2–14.

Keyser, T. K., H. Sarper. 1991. A heuristic solution of the E/T problem with waiting costs and non-zero release times. *Computers and Industrial Engrg.* **21**(1–4) 297–301.

Kim, Y.-D., C. A. Yano. 1994. Minimizing mean tardiness and earliness in single-machine scheduling problems with unequal due dates. *Naval Res. Logist.* **41**(7) 913–933.

Labetoulle, J., E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan. 1984. Preemptive scheduling of uniform machines subject to release dates. W. R. Pulleyblank ed., *Progress in Combinatorial Optimization.* Academic Press, New York.

Lakshminarayan, S., R. Lakshmanan, R. Papinou, R. Rochette. 1978. Optimum single-machine scheduling with earliness and tardiness penalties. *Oper. Res.* **26** 1079–1082.

Larson, R. E., M. I. Dessouky, R. E. Devor. 1985. A forward-backward procedure for the single machine problem to minimize maximum lateness. *IIE Trans.* **17**(3) 252–260.

Lee, C. Y., J. Y. Choi. 1995. A genetic algorithm for job sequencing problems with distinct due dates and general early-tardy penalty weights. *Comput. Oper. Res.* **22**(8) 857–869.

Lenstra, J. K. 1977. *Sequencing by Enumerative Methods*. Mathematical Centre Tracts, Mathematisch Centrum, Amsterdam.

——, A. H. G. Rinnooy Kan, P. Brucker. 1977. Complexity of machine scheduling problems. *Ann. Discrete Math.* **1** 343–362.

Mannur, N. R., J. B. Addagatla. 1993. Heuristic algorithms for solving earliness-tardiness scheduling problem with machine vacations. *Comput. Industrial Engrg.* **25**(1–4) 255–258.

McMahon, G. B., N. Florian. 1975. On scheduling with ready times and due dates to minimize maximum lateness. *Oper. Res.* **23**(3) 475–482.

Morton, T. E., D. Pentico. 1993. *Heuristic Scheduling Systems.* John Wiley & Sons, Inc., New York.

——, P. Ramnath. 1992. Guided forward TABU/beam search for scheduling very large dynamic job shops. Working Paper, Carnegie Mellon University, Pittsburgh, PA.

Nandkeolyar, U., M. U. Ahmed, P. S. Sundararaghavan. 1993. Dynamic single machine weighted absolute deviation problem: predictive heuristics and evaluation. Internat. J. Production Res. **31**(6) 1453–1466.

Ow, P. S., T. E. Morton. 1989. The single machine early/tardy problem. *Management Sci.* **35** 177–191.

Rachamadugu, R. M. V. 1987. A note on the weighted tardiness problem. *Oper. Res.* **35**(3) 450–452.

Raghavachari, M. 1988. Scheduling problems with non-regular penalty functions: a review. *Opsearch.* **25** 144–164.

Sanlaville, E. 1995. Nearly on line scheduling of preemptive independent tasks. *Discrete Appl. Math.* **57** 229–241.

Schrage, L. E. 1971. Obtaining optimal solutions to resource constrained network scheduling problems. *Proc. AIEE Systems Engineering Conference.* Pheonix, AZ.

Sidney, J. 1977. Optimum single-machine scheduling with earliness and tardiness penalties. *Oper. Res.* **25** 62–69.

Simons, B. 1978. A fast algorithm for single processor scheduling. *Proc. 19th Annual IEEE Symposium on Foundations of Computer Sci.* 50–53.

Smith, W. E. 1956. Various optimizers for single-stage production. *Naval Res. Logist. Quart.* **3** 59–66.

Sridharan, V., Z. Zhou. 1966a. Dynamic non-preemptive single machine scheduling. *Comput. Oper. Res.* **23**(12) 1183–1190.

——, ——. 1996b. A decision theory based scheduling procedure for single machine weighted earliness and tardiness problems. *Euro. J. Oper. Res.* **94** 292–301.

Szwarc, W., S. K. Mukhopadhyay. 1995. Optimal timing schedules in earliness-tardiness single machine sequencing. *Naval Res. Logist.* **42**(7) 1109–1114.

Wagner, H. 1977. *Principles of Operations Research.* Second ed. Prentice-Hall of India Private Ltd., New Delhi, India.

Yano, C. A., Y. Kim. 1991. Algorithms for a class of single machine weighted tardiness and earliness problems. *Euro. J. Oper. Res.* **52** 161–178.