THE HECKMAN BINDERY, INC.
North Manchester, Indiana

KRL

H or V	JUST	FONT	SLOT	TITLE
H	CC	1W	22	BEER
			21	FACULTY
			20	WORKING
			19	PAPER
H	CC	1W	8	1989
			7	NO.1540-1554
H	CC	1W		330 B385<"CV"> no.1540-1554 cop.2
H	CC	7W		<IMPRINT> U. of ILL. LIBRARY URBANA

BINDING COPY

PERIODICAL ☐ CUSTOM ☐ STANDARD ☐ ECONOMY ☐ THESIS
BOOK ☐ CUSTOM ☐ MUSIC ☐ ECONOMY AUTH. 1ST ☐

NO. VOLS. THIS TITLE | LEAD ATTACH.

ACCOUNT 56672 | LIBRARY 001 | NEW | RUB OR SAMPLE | TITLE I.D. 5632 | COLOR: FOIL WHI | MATERIAL 488

ACCOUNT NAME: UNIV OF ILLINOIS

ISSN.

ACCOUNT INTERNAL I.D.: B01912400

I.D. #2: STX3

NOTES | BINDING FREQUENCY | WHEEL 1 | SYS. I.D. 3 | 39256

COLLATING: 35

ADDITIONAL INSTRUCTIONS
Dept=STX3 Lot=#20 Item=142 HNM=[illegible]
1CR2ST3CR MARK BY # B4 911

SEP. SHEETS | PTS. BD. PAPER | TAPE STUBS | CLOTH EXT | FILLER: GUM, STUB | LEAF ATTACH

POCKETS: PAPER | BUCK | CLOTH | SPECIAL PREP.

INSERT MAT. | ACCOUNT LOT NO #20 | JOB NO HU363

PRODUCT TYPE 11 | ACCOUNT PIECE NO 142 | PIECE NO 88

HEIGHT 11 | GROUP CARD 1 | VOL THIS TITLE

COVER SIZE X

0012475

BEBR
FACULTY WORKING PAPER NO. 89-1548

Scheduling a General Flexible Manufacturing System to Minimize Tardiness Related Costs

N. Raman
F. B. Talbot
R. V. Rachamadugu

College of Commerce and Business Administration
Bureau of Economic and Business Research
University of Illinois Urbana-Champaign

BEBR

FACULTY WORKING PAPER NO. 89-1548

College of Commerce and Business Administration

University of Illinois at Urbana-Champaign

April 1989

Scheduling a General Flexible Manufacturing System to Minimize Tardiness Related Costs

N. Raman, Assistant Professor
Department of Business Administration

F. B. Talbot
University of Michigan

R. V. Rachamadugu
University of Michigan

Scheduling a General Flexible Manufacturing System to Minimize Tardiness Related Costs

N. Raman
Department of Business Administration
University of Illinois
1206 South Sixth Street
Champaign, IL 61820

F. B. Talbot
Graduate School of Business Administration
University of Michigan
Ann Arbot, MI 48109

R. V. Rachamadugu
Graduate School of Business Administration
University of Michigan
Ann Arbot, MI 48109

ABSTRACT

We consider the problem of minimizing total tardiness in a dynamic general flexible manufacturing system. While previous investigations of this problem have focused on the relative effectiveness of priority rules, we propose a solution approach which decomposes the dynamic problem into a series of static problems. An implicit enumeration algorithm is constructed for solving the static problem exactly. We also develop a heuristic solution procedure which is based on decomposing the multiple machine problem into several single machine problems. A schedule for the entire FMS is then developed around the sequence generated for the bottleneck machine. Computational studies indicate the efficacy of this procedure for both static and dynamic scheduling problems.

1. INTRODUCTION

This paper addresses the problem of minimizing penalties arising from job tardiness in a flexible manufacturing system (FMS) which produces several part types to specific orders. We consider a dynamic system with random job arrivals. We assume that the operation sequence for each part type establishes a serial precedence relationship among the operations. In addition, the machine required for each operation, operation processing times and travel times are deterministic and known. Preemption of any operation is not permitted. The manufacturing system considered in this paper is the Automated Manufacturing Research Facility at the National Institute of Standards and Technology in Gaithersburg, Maryland.

Much of the prior research on dynamic due date based scheduling deals with the use of priority dispatching rules in job shops. [See, for example, Carroll (1965), Conway (1965), Baker and Bertrand (1982), Kanet and Hayya (1982), Baker and Kanet (1983), Baker (1984) and Vepsalainen and Morton (1987).] One of the facts which emerge from these studies is that the relative effectiveness of a given priority rule depends upon the shop loading conditions such as machine utilization, flow allowance values, etc. However, under balanced machine workloads, Baker (1984) found that the Modified Operation Due Date (MDD) rule yielded lower tardiness values across a wide range of flow allowances. [Raman's (1988) study showed that its effectiveness is not carried forward to the case of unbalanced workloads.]

In this paper, we employ a solution methodology which is an alternative to dispatching rules. We treat the dynamic scheduling problem as a series of static problems. The proposed approach requires solving the static problem entirely and implementing the imminent solution on a rolling basis. In contrast to the local dispatching rules investigated in previous studies, this approach entails solving the global scheduling problem. Rinnooy Kan (1976) shows this problem to be NP-complete. Development of effective algorithms is difficult because of the lack of dominance conditions and efficient bounding mechanisms.

We propose an implicit enumeration based algorithm for solving this problem. In addition, we also develop a heuristic solution procedure which is based on decomposing the multiple machine problem into several one machine problems, and constructing the schedule for the entire FMS around the bottlenecks machine. While this solution approach requires greater computational effort, we show that it results in significant improvement over some of the well-known dispatching rules.

The remainder of this paper is organized as follows. The static problem is formulated in Section 2. The branch and bound procedure used for solving this problem optimally is presented in

Section 3. Section 4 describes the decomposition-based heuristic solution procedure. Experimental investigations of the static problem are given in Section 5. We address the implementation of the static solution procedure within a dynamic framework, and present our computational experience in Section 6. Section 7 gives a summary evaluation of the suggested solution methods. The notation used in this chapter is given in Appendix 1.

2. THE STATIC SCHEDULING PROBLEM

The static problem is generated for the jobs currently available in the system. An integer programming formulation is presented below for this problem. We assume without loss of generality that the operations for each job are numbered such that the successor operation has an index higher than that of its predecessor.

$$\text{Minimize} \quad \sum_j T_j \qquad (1)$$

subject to

$$\sum_t x_{tjk} = 1; \quad j = 1,..,N, \ k = 1,..,N_j \qquad (2)$$

$$\sum_t (t - p_{jl})\, x_{tjl} \geq \sum_t t\, x_{tjk}; \quad j = 1,..,N, \qquad (3)$$

$$k = 1,..,N_j, \text{ and } (k,l) \in S_j$$

$$\sum_j \sum_k \sum_{q=t}^{t+p_{jk}-1} R_{jkm}\, x_{qjk} \leq 1; \quad t = 1,..,T, \ m = 1,..,M \qquad (4)$$

$$\sum_t t\, x_{tjk} + E_j - T_j = d_j\,; \quad k = Nj,\ j = 1,..,N \tag{5}$$

$$x_{tjk} \in \{0,1\};\ E_j\,,\ T_j \geq 0,\ \text{integer; for all } j,\ k,\ t \tag{6}$$

Equation (1) corresponds to the objective of minimizing total tardiness. Constraints (2) ensure that each operation is completed exactly once. Constraints (3) indicate the precedence relationships among the various operations within a job, and ensure that the operation processing times are taken into consideration appropriately. Constraints (4) ensure that each resource (machine and transporter) is assigned to at most one operation at any given time. Constraints (5) measure the tardiness of each job. Finally, constraints (6) specify the integer nature of the variables.

In the above formulation, transportation is treated as a move operation between two machining operations, or the load/unload station and a machining operation. This is a reasonable approximation of the real system for the following reasons. First, in the system modeled, the transporter always returns to the load/unload station after moving parts between machines. Second, there are small, but adequate, input and output buffers at each machine. Third, the time to return the transporter to the load/unload station is small relative to the machining times.

If the transporter did not return to the load/unload station, then the formulation would have to be modified to account for the potentially large number of possible alternative routings. If there were no buffers, or if the buffers were serious bottlenecks, then these conditions would have to be modeled explicitly, otherwise the schedule resulting from (1) - (6) could be infeasible. Also, the above formulation and the proposed solution approach assume that any machining operation does not begin until the transporter returns to the load/unload station. Because of condition three given above, this is a reasonable approximation of reality. As a consequence, travel time can be treated as the sum of the transporter round trip time and the transfer time from one machine to another.

3. EXACT SOLUTION PROCEDURE

The formulation given by equations (1) - (6) results in a large number of variables and constraints for problems of practical size, thereby precluding the use of general-purpose integer programming codes as solution methods. This problem can, however, be viewed as a resource-constrained project scheduling problem (or its subset, the resource-constrained job shop scheduling problem) for which reasonably efficient optimum-seeking codes exist for some objective functions. For example, the procedure developed by Talbot (1982) can be used directly to solve the objective of minimizing makespan, and it has been modified by us to solve the tardiness problem.

Figure 1 illustrates how the problem given by equations (1) - (6) can be viewed as a project scheduling problem. Each job comprises a series of machine operations and transporter movements, each of which is represented by a node in an acyclic network. Each operation or move requires the use of a specific resource for a specified period of time. A due date is associated with the last move (returning the finished part to the load/unload station) of each job.

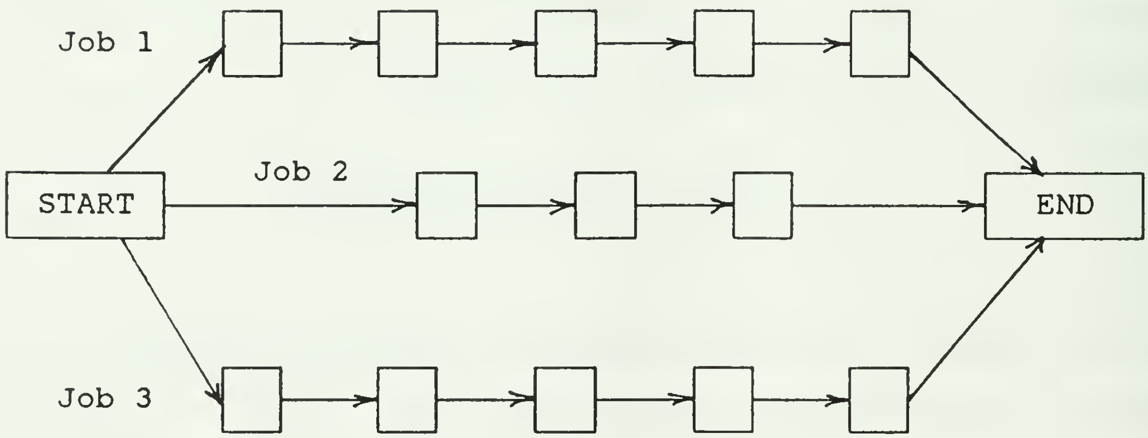

Figure 1 - Network Representation of
the Scheduling Problem

The proposed solution methodology exploits this network structure which obviates the need to explicitly generate the objective function and the constraint set given by (1) - (6).

The procedure uses a depth-first branch and bound algorithm which builds a schedule forward in time. A node at level L in the solution tree has an associated array A_n which contains the indexes of operations which are schedulable at the next level. The precedence relationships restrict the cardinality of A_n to the number of jobs in the system, which reduces computer storage as well as computational time requirements.

Starting with the unique node at level 0, the procedure selects the next operation based on a priority index associated with each operation or move. The priority scheme used in this study is based on the Modified Due Date (MDD) rule. The descendent nodes (operations) of any node are ranked in the nondecreasing order of the modified due date of the job to which the operation belongs. MDD is also used to generate the initial solution. Backtracking rather than skiptracking is employed to keep storage requirements at a minimum.

4. HEURISTIC SOLUTION APPROACH

In view of the computational complexity of the mean tardiness problem and the limited effectiveness of dominance conditions and lower bounding mechanisms, we need to consider heuristic solution methods. As mentioned in Section 1, virtually all heuristic approaches reported in the scheduling literature are based on using local dispatching rules. While these procedures require relatively less computational effort, the solution is of unknown quality. The proposed solution procedure is an improvement heuristic which uses global information. A brief description of the procedure is given below. The individual steps are discussed in detail subsequently.

First, we decompose the job due dates into the due dates for individual operations within each job. Next, we construct the initial solution through a forward scheduling approach starting with the first operation of each job.

The third step attempts to improve upon the initial solution by reassigning operation due dates (ODDs) and rescheduling operations at each machine. The machines are ranked in the non-increasing order of their total workload. At each machine, starting with the most heavily loaded one, we rank the jobs in the order of non-increasing job tardiness. The machines, and the jobs at each machine, are scanned in the order of their ranks. Scanning involves determining the best due date for each operation within each job using a binary search procedure. For each possible ODD value investigated during this search, the entire system is rescheduled, and the value which yields the minimum total tardiness is selected. Because each ODD reassignment and rescheduling step may change the current tardiness of one or more jobs, their ranks are continuously updated.

The efficiency of such a decomposition approach is likely to depend upon the order in which the machines are selected. Because an average job spends the bulk of its total waiting time at the bottleneck machine, it appears reasonable to suppose that the performance of the entire FMS is significantly affected by the sequence of operations at this machine. This explains the rationale behind using relative machine workloads for determining the criticality of machines. The individual steps of the procedure are now discussed.

4.1 Determination of Initial ODDs

The ODDs used for generating the initial solution are derived from the job due dates using the Total Work Content (TWK) rule. Under this rule, the ODD of operation i in job j is given by

$$d_{ji} = d_{j,i-1} + d_j \; p_{ji}/p_j$$

It can be seen that the flow allowance for operation i, $d_{ji} - d_{j,i-1}$, is proportional to its processing time p_{ji}.

4.2 Construction of the Initial Solution

Given the operation due dates, the initial sequence is constructed through a non-delay schedule generation procedure [see, for example, Baker (1974)]. Ties among operations at a given machine are broken using the Modified Operation Due Date

(MOD) rule. This rule selects the operation with the minimum modified operation due date.

The modified operation due date of operation i in job j is given by

$$MOD_{ji} = \max\ (t + p_{ji},\ d_{ji})$$

where t is the time which the scheduling decision needs to be made.

MOD has been found effective in several studies [see, for example, Baker (1984)]. The following result states a possible reason for its effectiveness.

THEOREM 1: For a given set of operation due dates, the total tardiness incurred by two adjacent operations in a non-delay schedule on any given machine does not increase if they are sequenced according to the MOD rule.

PROOF: Refer to Appendix 2.

At the end of this step, if all jobs are completed on time, the algorithm terminates. Otherwise, we proceed to reassign ODDs and reschedule operations.

4.3 ODD Reassignment and Rescheduling of Operations

The initial solution generation procedure has three limitations. First, it considers ODD assignment and operation scheduling sequentially. Note that MOD addresses operation tardiness, not job tardiness. While it provides a locally optimal schedule for a given set of ODDs, the overall solution quality depends upon how effectively job due dates are decomposed into the operation due dates. Because it is possible for a scheduling rule to yield a better solution for a different selection of ODDs, it is desirable to consider ODD assignment and operation scheduling simultaneously.

Second, it ignores the global impact of selecting an operation ahead of another at a given machine. This is so because it is a solution construction procedure, and at the time a scheduling decision is made, its impact on operations to be scheduled later is not known. Third, the MOD rule considers only <u>non-delay</u> schedules. While non-delay schedules are reasonably effective in general, they do not constitute the dominant set. On the other hand, the set of <u>active</u> schedules does contain the optimal solution. The proposed solution method attempts to eliminate these limitations.

We rank the machines according to their workloads. Since the relative ranking of machines remains unchanged, we can number them according to their rank. We start with machine 1, i.e., the most heavily loaded machine, and move down the list until all machines are scheduled. At a given machine, all jobs are scanned in the order of non-increasing tardiness. The rest of the solution procedure is described with the help of the solution tree shown in Figure 2. This tree is similar to a branch-and-bound enumeration tree with the difference that each node represents a complete solution.

Consider machine 1 first. Suppose we are considering job j which requires operations numbered i1, i2,.., I1 on machine 1. Consider operation i1. Let its ODD, as determined by TWK be $d_{j,i1}$, and let X_1 denote its reassignment. Also suppose that X_1 can take any value in the interval (L_1, U_1). A descendant is generated for each value of X_1 in this interval. For a given value of X_1, say x, the ODDs of other operations in j are generated as follows:

$$d_{jk} = d_{j,k-1} + (x - p_{j,i1})\, p_{jk}/P_{j,i1-1}$$

for k = 1,.., i1-1

and $$d_{jk} = d_{j,k-1} + (d_j - x)\, p_{jk}/(p_j - P_{j,i1})$$

for k = i1+1, i1+2,..,N_j

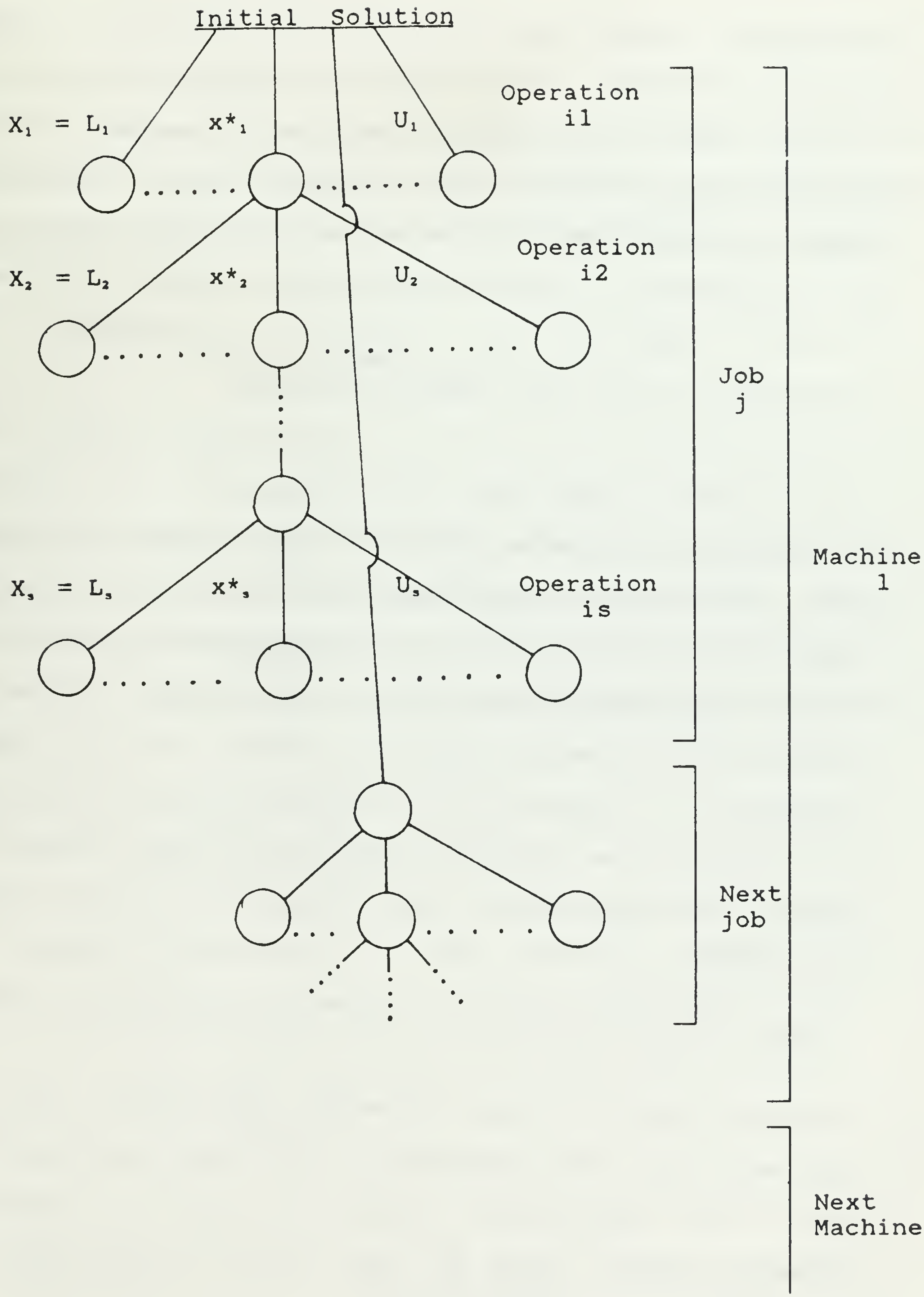

Figure 2 - Solution Tree

In effect, we split job j into three "sub-jobs" j_1 j_2 and j_3, where j_1, consists of all operations prior to i1, j_2 contains only i1, and j_3 comprises all operations subsequent to i1. Due dates of all operations within a sub-job are set independently of other sub-jobs. These due dates are derived from the due date of the corresponding sub-job using the TWK method, due dates of j_1, j_2 and j_3 being $x - p_{j,i1}$, x and d_j respectively. ODDs of operations in other jobs remain unchanged.

The solution value for the descendant is determined by rescheduling all jobs at all machines for the revised set of ODDs. The rescheduling procedure generates an active schedule by considering operations which are expected to arrive at a given machine imminently, in addition to those which are already at the machine at the time the scheduling decision is to be made. This procedure is a revision of the Modified Operation Due Date rule and it selects the operation with the minimum revised modified operation due date (RMOD); RMOD of operation u in job v at time t is given by

$$RMOD_{vu} = \max\,[\max\,(t, r_{vu}) + p_{vu}] + \max\,(t, r_{vu}) \qquad (7)$$

To ensure that no local left-shift is possible, the RMOD rule considers only those operations which can be started before any one of the conflicting operations can be completed. The motivation behind using the RMOD rule is given by the following result which parallels Theorem 1.

THEOREM 2: Suppose operation a in job b is the immediate predecessor of operation c in job d on any machine in a given sequence. Suppose further that operation a starts at time t, and $t \leq r_{dc} < t + p_{ba}$. Then the total tardiness incurred by these two operations does not increase with an interchange of a and c if

$$RMOD_{dc} \leq RMOD_{ba}.$$

PROOF: Refer to Appendix 2.

The branch corresponding to the node with the minimum tardiness is selected for further investigation. Also, the ODD of i1 is frozen at the corresponding value of X_1, say x^*_1. Simultaneously, ODDs of all operations in job j preceding i1 are updated as follows:

$$d_{jk} = d_{j,k-1} + (x^*_1 - p_{j,i1})\, p_{jk}/P_{j,i1-1}$$
$$k = 1,..,i1-1$$

Next, consider operation i2. The interval scanned for the possible reassignment of its ODD X2 is (L2, U2) where

$$L_2 = \sum_{l=i1+1}^{i2} p_{jl} + x^*_1$$

and $$U_2 = d_j$$

For a given value of $X_2 = x$, the due dates of operations in job j excluding i1, i2 and those which precede i1 are generated as follows:

$$d_{jl} = d_{j,l-1} + (x - x^*_1)\, p_{jl}/(P_{j,i2-1} - P_{j,i1})$$

for l = i1+1, i1+2,.., i2-1

and $$d_{jl} = d_{j,l-1} + (d_j - x)\, p_{jl}/(p_j - P_{j,i2})$$

for l = i2+1, i2+2, .., N_j

ODDs of operations preceding and including i1 remain unchanged.

The ODD updating and operation rescheduling steps for i2 are similar to those described earlier for i1. We continue in this manner for the remaining operations of job j. Subsequently, we consider the job with the next highest tardiness and so on until all operations on machine 1 are investigated. This cycle is repeated at machines 2 through M in that order. In the general step, suppose we are considering ODD reassignment of operation k of job j at machine m. Also, suppose that after investigating machines 1 through m-1, and all operations of job j prior to k on machine m, we have frozen the due dates of operations u_1, u_2,.., u_z in job j. Suppose further that operation k is processed between operations u_l and u_{l+1} with frozen due dates of x^*_l and x^*_{l+1} respectively. Then, the ODD of k needs to be searched in the interval (x^*_l, x^*_{l+1}) only. In addition, ODDs need to be generated for only those operations which occur between u_l and u_{l+1}.

It can be seen that as we go down the list of machines and move from one operation to another of a given job at a machine, the search interval reduces. However, near the top of the tree, it can be quite wide resulting in a large number of descendant nodes from a given parent node. We now describe an efficient procedure for improving the search routine.

ODD Search Procedure

Theoretically, while searching for the reassigned ODD value of the first operation il of job j on a given machine, we need to consider the interval $(0, d_j)$ in unit steps. However, the lower limit of this interval can be tightened by noting that the earliest time il can start is $\sum_{l=1}^{il-1} p_{jl}$. Therefore,

$$r_{j,il} \geq \sum_{l=1}^{il-1} p_{jl}.$$

From (7) we have

$$RMOD_{j,il} = \max\left[\max(t, r_{j,il}) + p_{j,il},\ d_{j,il}\right] + \max(t, r_{j,il})$$

$RMOD_{j,il}$ and, therefore, the priority of operation il is independent of $d_{j,il}$ if

$$d_{j,il} \leq \max(t, r_{j,il}) + p_{j,il}$$

The minimum value that $d_{j,il}$ can take is $\sum_{l=1}^{il} p_{jl}$. Hence, the search interval can be limited to $\left(\sum_{l=1}^{il} p_{jl} , d_j \right)$.

The search procedure can be further improved by noting that while X can take many values, an operation can only occupy a given number of positions in any sequence. In a single machine problem, it can only be in n positions in a permutation schedule where n is the total number of operations (or jobs). In the multiple machine case, it is higher because of the interaction effects at different machines. Nevertheless, the number of positions that an operation can occupy at a given machine is usually much smaller than the number of different values that X can take.

Consequently, the operation completion time and, therefore, total tardiness as well, remains unchanged for many sub-intervals within $\left(\sum_{l=1}^{il} p_{jl} , d_j \right)$. [This is true for all operations, not merely il]. Figure 3 illustrates this characteristic by depicting the typical behavior of total tardiness with respect to the reassigned ODD value X for any operation u of job v in the interval (U,L).

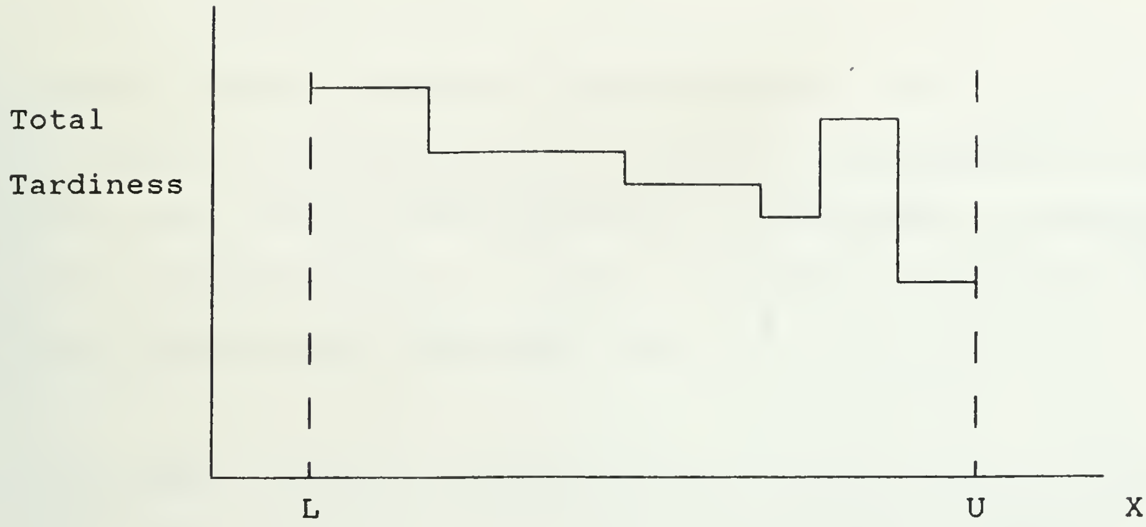

Figure 3 - Graph of Total Tardiness against X

The procedure for searching the best value of X for an operation in a given job employs a modification of the binary search method. As shown in Figure 4, suppose that we need to search in the interval (L_o, U_o). Using TARD(x) to denote the total tardiness of all jobs when X = x, we compute TARD (L_o) and TARD (U_o). Starting with the interval (L_o, U_o) we successively divide each interval into two equal halves and compute the total tardiness value at the midpoint of each half-interval. Within any generated interval, scanning for the next half-interval is initially done to the left. In other words, with reference to Figure 4, we have

$$U_i = \frac{L_o + U_{i-1}}{2}, \quad i = 1,2,3,4.$$

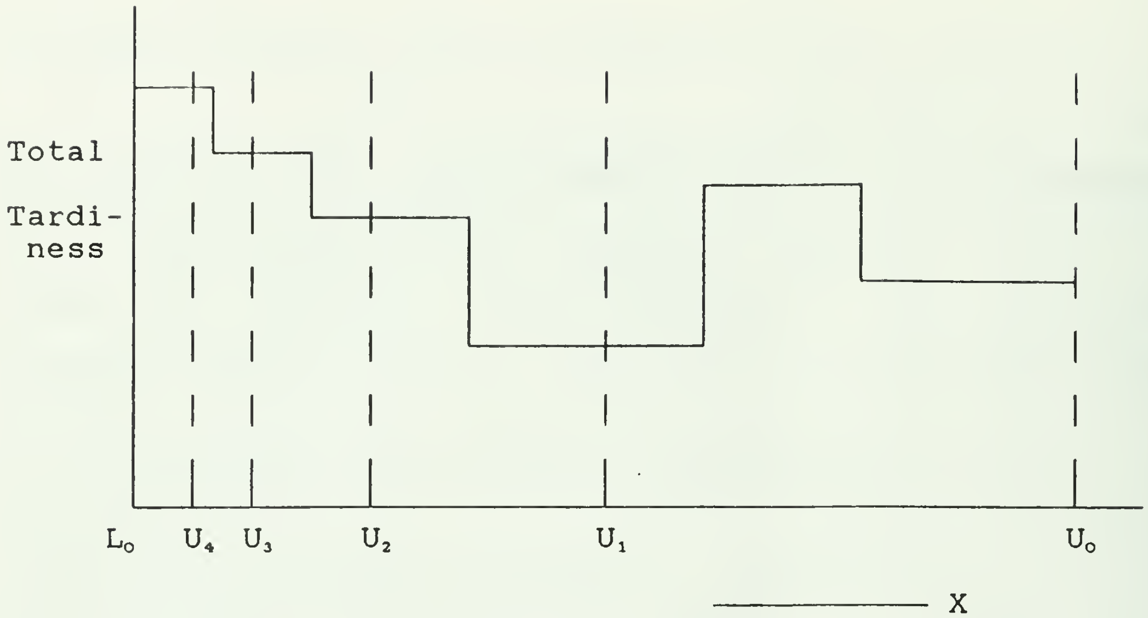

Figure 4 - Search Procedure

Scanning to the left within a half-interval terminates when it is fathomed. An interval is said to be fathomed if it is the most recently generated interval and the total tardiness values at its end-points and mid-point are the same. In Figure 4, for example, the interval (L_0, U_4) is fathomed. Note that the fathoming procedure will ignore changes in total tardiness values within an interval, if in spite of such changes, the <u>same</u> tardiness value is realized at both end points and the mid-point of that interval. While such occurrences are possible, they are somewhat unlikely in most real problems. [We did not observe it in any one of the 50 randomly generated problems.] It should, nevertheless, be noted that while trying to achieve computational efficiency, this search procedure may not always return the best value of X.

At the termination of left-scanning, the procedure next evaluates the most recently generated and unfathomed interval to its right. If the total tardiness values at both its end-points and its mid-point are not the same, another half-interval is generated and left-scanning is resumed. The procedure terminates when all half-intervals are fathomed.

The search procedure yields all or nearly all such values of X which give different values of total tardiness. The due date of the operation under consideration is reassigned to the X value which results in the minimum total tardiness.

Note that it is possible that the position of any operation of a given job which results in the minimum total tardiness may result in that job itself being late (if by doing so, tardiness of other jobs improves significantly). For this reason, it is desirable to increase the upper limit of the search interval for the initial operations of job j from d_j to some arbitrarily large value T. The actual value used for T is of marginal importance because intervals which do not affect total tardiness are rapidly fathomed. In our experimental study, T equaled the makespan of the initial solution generated through the MOD rule.

We now describe our computational experience with this procedure for both static and dynamic problems.

5 EXPERIMENTAL STUDY - STATIC PROBLEM

Two sets of experiments were conducted to assess the relative performance of the scheduling procedure [hereafter referred to as the Global Scheduling Procedure (GSP)] given in Section 4. The first set compared GSP with six other well-known scheduling procedures. The second set evaluated its performance relative to the optimal solution obtained by the enumeration method described in Section 3. The experimental design and test results are now presented.

5.1 Experimental Design

The design of the first set of experiments is described first. The heuristic solution methods selected for comparative purpose were:

1. Shortest Processing Time Rule (SPT): This rule selects the operation with the minimum processing time whenever a tie needs to be broken at any machine. SPT was included in this study because of its reported effectiveness in the case of tightly set due dates.

2. Earliest Due Date Rule (EDD): This procedure breaks ties in favor of the operation with the minimum job due date. Previous studies have shown EDD to be effective when due dates are loose.

3. <u>Critical Ratio Rule (CRIT)</u>: The critical ratio rule resolves conflicts among operations by selecting the operation with the minimum critical ratio, where the critical ratio of operation i in job j is given by

$$CR_{ji} = (d_j - t)/P_{ji}$$

where t is the time at which the scheduling decision is to be made. To prevent anomalies arising when $d_j < t$, this ratio was defined as

$$CR_{ji} = (d_j - t)\ P_{ji}$$

in such cases.

4. <u>Modified Job Due Date Rule (MDD)</u>: The MDD rule breaks ties in favor of the operation with the minimum modified job due date MDD, where MDD of operation i in job j is given by

$$MDD_{ji} = \max\ (t + P_{ji},\ d_j)$$

5. <u>Modified Operation Due Date Rule (MOD)</u>: This rule is described in Section 5.2. It is used for generating the initial solution for GSP. While GSP is a solution improvement procedure, and therefore, is likely to do better, MOD was included primarily to evaluate the degree of improvement achieved. Consistent with the previous studies, MOD was implemented in conjunction with the TWK operation due date assignment procedure.

6. Hybrid Rule (HYB): The hybrid rule is a combination of MOD and MDD, and it recognizes the differences between machine workloads. Under this rule, MDD is used at machines with more than average workload, while MOD is used at non-bottleneck machines.

A comment will be made here regarding the relative computational efforts of the various heuristic solution procedures. Except GSP, all the scheduling rules are dispatching methods. SPT and EDD require $O(MN \log N)$ effort while CRIT, MOD, MDD and HYB require $O(MN^2 \log N)$ effort. Without the binary search procedure, GSP runs in $O[MN^4 (\sum_j p_j) \log N]$ time; the proposed search method reduces this to $O(MN^5 \log N)$. In view of the larger computational effort required, GSP was implemented with a time trap of 20 seconds.

Data Design

For the first set of experiments, various scenarios were generated by varying one or more of the following parameters:

1. System Configuration: Two system sizes - 5 machines and 10 machines, were considered. For each size, three levels of relative machine workloads were generated. The first level simulated a perfectly balanced system by providing equal workloads at all machines. The second level represented systems

with a single bottleneck. In this scenario, the workloads on all machines except the bottleneck were equal; the bottleneck machine had 50% higher workload. The third level simulated systems with a range of workloads. The relative workloads used for the 5-machine system were (0.6, 0.8, 1.0, 1.3, 1.6), and for the 10-machine system were (0.4, 0.6, 0.8, 0.9, 1.0, 1.0, 1.1, 1.2, 1.4, 1.6).

2. Job Configuration: Two ranges were considered for the number of jobs available for scheduling. The first varied uniformly between 10 and 20 for the smaller problems, and the second varied between 30 and 40 for the larger problems. For each range, the number of operations within a job was allowed to vary uniformly between 1 and 5 for the 5-machine system, and between 1 and 10 for the 10-machine system. All jobs had random machine routing although the processing of successive operations on the same machine was prohibited. Operation processing times were selected from a uniform distribution in the interval (5,100). Two parameters were used to control the tightness and the variation of job due dates. The tardiness factor Z measures approximately the proportion of jobs likely to be tardy while R determines the range of job due dates. For given Z and R, the job due dates were sampled from a uniform distribution in the interval

$$[\bar{d}\,(1 - R/2),\ \bar{d}\,(1 + R/2)]$$

where the average job due date $\bar{d}$ is given by

$$\bar{d} = \frac{1}{M} \left(\sum_{j=1}^{N} p_j \right) (1 - Z).$$

Z and R have been used extensively for generating test data in single machine tardiness problems [see, for example, Srinivasan (1971)]. Ow (1985) suggests a modification for a flow shop. Because of the interaction effects among operations, Z is only an approximate measure of the proportion of tardy jobs in multiple machine systems. Nevertheless, it helps to anchor due date tightness at various levels. Four combinations of due date tightness and due date range were used by considering two levels of Z - 0.2, and 0.6, and two levels of R - 0.5 and 1.5.

Scheduling Measures

The performance measure of primary interest is the mean (or total) job tardiness (MT). For better comparison, we used a normalized version of total tardiness (NMT) which is obtained by dividing the sum of job processing times into total job tardiness.

To evaluate the robustness of a given scheduling rule, we also monitored the measures of the proportion of tardy jobs (PT), the standard deviation of tardiness (SDT), and total job flow time (FT).

A total of 48 test scenarios was constructed using different combinations of the system and job configurations. For each scenario, 20 problems were generated by varying the seed values for the random number generator. Performance measure values reported for each scenario in Section 5.2 indicate the average values over these 20 problems.

The second set of experiments considered a 3-machine, 5-job system. The number of operations within each job was allowed to vary between 1 and 3, and the operation processing times were sampled from a uniform distribution in the interval (5,30). As in the case of the first set of experiments, four combinations of the tardiness factor and job due date range, for Z = 0.2 and 0.6, and R = 0.5 and 1.5, were considered. For each combination, 10 problems were generated randomly. The mean tardiness values for these problems were aggregated and the average was recorded. The size of the problems considered in the second set of experiments was deliberately restricted in order to keep the computational costs within reasonable limits. The scheduling procedures used in both sets of experiments were coded in FORTRAN.

5.2 Experimental Results

The experimental results are shown in Tables 1 through 10. For better presentation, the results obtained under the HYB scheduling rule are omitted because they were quite similar to the values obtained under MOD. The relative performance of different scheduling rules with respect to normalized mean tardiness for each test scenario is shown in Tables 1 through 4. The results for the proportion of tardy jobs and the standard deviation of tardiness are given in Tables 5 through 8. For the sake of brevity, the values of these two scheduling measures obtained at different levels of workload balance have been averaged for reporting purposes. Table 9 depicts the total flow time values obtained under different scheduling rules for different combinations of the number of machines and the number of jobs in the system. The values obtained under the other combinations of the experiment parameters are averaged to yield the reported results.

For ease of presentation, we denote the expected number of jobs in a scenario by NJ and the number of machines by NOM. WL1 denotes the case in which machine workloads are balanced, WL2 represents the case with a single bottleneck and WL3 denotes the case with a range of machine workloads.

Table 10 compares the mean tardiness values obtained under GSP with the optimal solution values for the second set of experiments. The number of times GSP found the optimal solution in the 10 problems generated for each scenario is shown in parentheses next to the GSP solution value.

As mentioned in Section 5.1, GSP was implemented with a time trap of 20 seconds. This time trap was never required for the 5-machine, 15-job; 10-machine, 15-job; and 5-machine, 35-job problems. The average solution times for these problems were 0.209 seconds, 1.812 seconds, and 3.105 seconds respectively. For the 10-machine, 35-job problem, however, the time trap was required on many occasions, especially for the case in which Z = 0.6 and R = 0.5. The average solution time for this problem was 19.456 seconds after considering the time trap.

5.3 Analysis of Results

GSP can be seen to provide the best results for the measure of normalized mean tardiness. It yields the lowest values of NMT in 45 out of 48 cases, resulting in improvements of the order of 3%-28% over the next best rule. Its performance relative to other scheduling rules, except MDD, remains robust across the various test scenarios. Also, the improvement achieved over the initial solution provided by MOD is significant. However, for the measures of proportion of tardy jobs and standard deviation of tardiness, it has an average performance. In general, these experiments reveal that rules which are superior for PT give inferior results for SDT. GSP can, therefore, be seen as providing a compromise between these two criteria. For the measure of total flow time, however, GSP is, without exception, the best rule across all scenarios. Its relative performance improves as the number of machines and/or the number of jobs increase.

Among the other rules, MDD is, in general, the best for NMT. It yields the best solutions in the remaining 3 cases. Unlike GSP, however, it has a variable performance relative to other scheduling rules, and in particular, is less effective when the tardiness factor and the job due date range are small. MDD is quite effective for reducing the proportion of tardy jobs and performs reasonably well for the total flow time criterion as well. It can, however, lead to large values of the standard deviation of tardiness.

MOD and HYB yield reasonably good results for NMT, while EDD and CRIT are the best rules for SDT, and SPT is effective for PT and FT.

The results of the second experiment depicted in Table 10, indicate that GSP frequently finds the optimal solution, and in general, gives mean tardiness values close to the optimum for small problems. However, we note that these results cannot be generalized to larger problems.

TABLE 1

NORMALIZED MEAN TARDINESS

5-Machine System; Average Number of Jobs = 15

	SPT	EDD	CRIT	MDD	MOD	GSP
Z=0.2; R=0.5						
WL1	0.329	0.286	0.299	0.297	0.248	0.179
WL2	0.364	0.347	0.387	0.337	0.287	0.235
WL3	0.455	0.459	0.531	0.424	0.390	0.331
Z=0.2; R=1.5						
WL1	0.536	0.417	0.427	0.391	0.431	0.334
WL2	0.560	0.456	0.503	0.442	0.457	0.366
WL3	0.632	0.531	0.599	0.518	0.521	0.440
Z=0.6; R=0.5						
WL1	0.795	0.847	0.900	0.797	0.819	0.671
WL2	0.818	0.898	0.965	0.809	0.838	0.711
WL3	0.898	1.004	1.060	0.871	0.923	0.782
Z=0.6; R=1.5						
WL1	0.884	0.879	0.922	0.830	0.832	0.727
WL2	0.914	0.932	1.023	0.848	0.879	0.753
WL3	0.990	1.042	1.131	0.917	0.946	0.839

TABLE 2

NORMALIZED MEAN TARDINESS

10-Machine System; Average Number of Jobs = 15

	SPT	EDD	CRIT	MDD	MOD	GSP
Z=0.2; R=0.5						
WL1	0.521	0.508	0.512	0.511	0.495	0.426
WL2	0.522	0.520	0.508	0.529	0.484	0.436
WL3	0.601	0.595	0.598	0.600	0.577	0.509
Z=0.2; R=1.5						
WL1	0.609	0.580	0.580	0.595	0.568	0.505
WL2	0.613	0.593	0.592	0.603	0.567	0.514
WL3	0.692	0.655	0.660	0.656	0.649	0.582
Z=0.6; R=0.5						
WL1	0.832	0.836	0.852	0.831	0.817	0.732
WL2	0.839	0.844	0.852	0.826	0.827	0.750
WL3	0.914	0.926	0.937	0.908	0.908	0.822
Z=0.6; R=1.5						
WL1	0.875	0.868	0.888	0.862	0.865	0.781
WL2	0.882	0.878	0.884	0.864	0.864	0.794
WL3	0.958	0.958	0.969	0.951	0.940	0.860

TABLE 3

NORMALIZED MEAN TARDINESS

5-Machine System; Average Number of Jobs = 35

	SPT	EDD	CRIT	MDD	MOD	GSP
Z=0.2; R=0.5						
WL1	0.354	0.180	0.231	0.175	0.194	0.136
WL2	0.443	0.409	0.521	0.346	0.304	0.265
WL3	0.633	0.642	0.849	0.524	0.544	0.440
Z=0.2; R=1.5						
WL1	0.850	0.367	0.411	0.328	0.424	0.345
WL2	0.923	0.510	0.578	0.427	0.509	0.432
WL3	1.061	0.678	0.760	0.575	0.643	0.553
Z=0.6; R=0.5						
WL1	1.201	1.165	1.506	0.991	1.305	0.963
WL2	1.227	1.335	1.693	1.084	1.317	0.995
WL3	1.405	1.661	2.043	1.255	1.475	1.148
Z=0.6; R=1.5						
WL1	1.463	1.122	1.303	0.977	1.268	0.980
WL2	1.507	1.316	1.512	1.067	1.293	1.031
WL3	1.666	1.638	1.902	1.262	1.488	1.194

TABLE 4

NORMALIZED MEAN TARDINESS

10-Machine System; Average Number of Jobs = 35

	SPT	EDD	CRIT	MDD	MOD	GSP
Z=0.2; R=0.5						
WL1	0.416	0.337	0.363	0.344	0.336	0.268
WL2	0.456	0.418	0.478	0.395	0.385	0.305
WL3	0.633	0.624	0.745	0.555	0.608	0.460
Z=0.2; R=1.5						
WL1	0.665	0.424	0.458	0.434	0.501	0.420
WL2	0.692	0.489	0.536	0.480	0.543	0.449
WL3	0.841	0.675	0.716	0.604	0.701	0.565
Z=0.6; R=0.5						
WL1	0.989	0.980	1.124	0.919	1.010	0.881
WL2	1.030	1.055	1.202	0.934	1.043	0.905
WL3	1.202	1.278	1.476	1.075	1.251	1.041
Z=0.6; R=1.5						
WL1	1.100	1.007	1.094	0.960	1.071	0.920
WL2	1.133	1.080	1.184	0.984	1.101	0.957
WL3	1.296	1.302	1.398	1.125	1.298	1.090

TABLE 5

PROPORTION OF TARDY JOBS

5-Machine System

	SPT	EDD	CRIT	MDD	MOD	GSP
NJ=15						
Z=0.2; R=0.5	0.410	0.447	0.563	0.379	0.540	0.399
Z=0.2; R=1.5	0.441	0.490	0.558	0.429	0.472	0.428
Z=0.6; R=0.5	0.654	0.740	0.867	0.612	0.800	0.668
Z=0.6; R=1.5	0.637	0.757	0.851	0.618	0.748	0.650
NJ=35						
Z=0.2; R=0.5	0.651	0.645	0.740	0.575	0.708	0.651
Z=0.2; R=1.5	0.640	0.669	0.715	0.610	0.659	0.622
Z=0.6; R=0.5	0.783	0.822	0.887	0.741	0.841	0.802
Z=0.6; R=1.5	0.796	0.824	0.877	0.744	0.841	0.799

TABLE 6

PROPORTION OF TARDY JOBS

10-Machine System

	SPT	EDD	CRIT	MDD	MOD	GSP
NJ=15						
Z=0.2; R=0.5	0.271	0.267	0.412	0.223	0.342	0.228
Z=0.2; R=1.5	0.353	0.311	0.364	0.262	0.325	0.259
Z=0.6; R=0.5	0.557	0.620	0.849	0.528	0.819	0.524
Z=0.6; R=1.5	0.559	0.675	0.817	0.546	0.750	0.565
NJ=35						
Z=0.2; R=0.5	0.441	0.471	0.700	0.403	0.591	0.478
Z=0.2; R=1.5	0.467	0.499	0.592	0.424	0.528	0.452
Z=0.6; R=1.5	0.687	0.724	0.911	0.602	0.849	0.784
Z=0.6; R=1.5	0.678	0.767	0.885	0.642	0.820	0.752

TABLE 7

STANDARD DEVIATION OF TARDINESS

5-Machine System

	SPT	EDD	CRIT	MDD	MOD	GSP
NJ=15						
Z=0.2; R=0.5	100.0	87.3	87.0	102.3	78.0	78.7
Z=0.2; R=1.5	134.7	93.7	97.0	115.3	108.3	100.3
Z=0.6; R=0.5	150.3	140.7	140.0	174.7	142.7	153.3
Z=0.6; R=1.5	165.0	132.0	139.0	175.7	157.7	163.7
NJ=35						
Z=0.2; R=0.5	173.7	169.0	156.3	201.7	161.7	159.7
Z=0.2; R=1.5	202.0	181.3	176.3	214.0	193.0	186.3
Z=0.6; R=0.5	211.0	207.0	203.0	246.0	203.0	205.7
Z=0.6; R=1.5	222.0	208.3	210.0	254.0	219.0	220.7

TABLE 8

STANDARD DEVIATION OF TARDINESS

10-Machine System

	SPT	EDD	CRIT	MDD	MOD	GSP
NJ=15						
Z=0.2; R=0.5	171.7	134.3	146.0	144.7	134.3	134.3
Z=0.2; R=1.5	263.3	107.3	118.3	139.0	158.3	158.7
Z=0.6; R=0.5	283.0	272.3	281.0	285.0	269.7	279.7
Z=0.6; R=1.5	324.3	226.7	243.7	290.7	294.7	302.0
NJ=35						
Z=0.2; R=0.5	221.7	193.0	186.0	223.0	191.0	190.0
Z=0.2; R=1.5	297.0	173.3	183.3	233.0	239.3	239.0
Z=0.6; R=0.5	321.3	310.0	292.3	366.3	302.7	316.3
Z=0.6; R=1.5	353.7	270.3	276.7	372.3	333.3	343.0

TABLE 9

TOTAL FLOW TIME

	SPT	EDD	CRIT	MDD	MOD	GSP
NOM=5; NJ=15	346.7	375.9	404.4	357.8	389.8	305.0
NOM=5; NJ=35	675.3	737.1	846.3	702.8	821.7	628.6
NOM=10; NJ=15	473.0	479.3	489.9	467.4	482.5	324.0
NOM=10; NJ=35	753.0	782.2	871.2	746.5	840.9	591.9

TABLE 10

MEAN TARDINESS

Comparison with Optimal Solution

	Optimal Solution Value	GSP´s Solution Value	Degree of Suboptimality (%)
Z=0.2; R=0.5	16.6	17.5 (7)	5.4
Z=0.2; R=1.5	23.5	24.8 (6)	5.5
Z=0.6; R=0.5	42.7	44.1 (5)	3.3
Z=0.6; R=1.5	46.4	47.8 (6)	3.0

6. EXPERIMENTAL STUDY - DYNAMIC PROBLEM

Experimental investigation of the dynamic scheduling problem addressed the effectiveness of implementing the solution of the static problem on a rolling basis. In a dynamic environment, a static problem needs to be generated whenever a new job arrives. At that point in time, the network depicted in Figure 1 is generated afresh taking into account the operations already in process. Note that at that point in time, one or more machines or the material transporter can be busy. Since pre-emption is not permitted, such resources are blocked out for the period of commitment. The optimal (or best) solution determined by the solution procedure is implemented until the next job arrives when the process of generating and solving the static problem is repeated. Because of the computational costs involved, the experimental study utilized GSP, instead of the optimum-seeking method described in Section 3, for generating the solution to the static problem.

6.1 Experimental Design

The experimental study addresses the measures of mean job tardiness, proportion of tardy jobs, standard deviation of tardiness and mean flow time. The simulation model considered twenty part types. The number of operations in each part type ranged between 4 and 10; successive operations on any part type were done on different machines. The system comprised five machines and a material transporter. In addition, there was a load/unload station where incoming jobs were received and to which finished jobs were routed. The material transporter was assumed to be located at the load/unload station when not in service. The experiments were designed to yield an overall system utilization of 80%.

Job arrival followed a Poisson process. An incoming job was equally likely to belong to any of the twenty part types. Upon its arrival, a job was assigned a due date based on the Total Work Content (TWK) rule. According to this rule, the due date d_j of job j is given by,

$$d_j = a_j + F\, p_j$$

where a_j is the arrival time of job j, p_j is its total processing time and F is the flow allowance. As seen from the above equation, due date tightness can be controlled by varying the flow allowances.

Three levels of due date tightness were achieved by using job flow allowances of 3, 4, and 5. In addition, another set of simulation runs was conducted in which the job flow allowance was allowed to vary uniformly between 1 and 8. This set was used primarily to assess the robustness of ODD assignment rules with respect to variability in flow allowance. The operation processing times were designed to yield workload imbalance; the actual machine utilization ranged from 66% to 93%, while the transporter utilization was 7%.

We also generated an additional scenario in which the machine workloads were balanced while keeping the overall system utilization fixed. This was achieved by varying the operation processing times while ensuring that the job processing times remain the same as in the case of unbalanced workloads. [The realized utilizations varied between 78% and 82%.]

We compared MOD, MDD and HYB dispatching rules with GSP. For implementing HYB, any machine with more than average workload was treated as a bottleneck. GSP was implemented with a time trap of 1.0 CPU seconds in order to keep computational costs within reasonable limits. In the experiment conducted, the size of the static problem, expressed in terms of the number of operations, varied between 9 and 107. Statistics pertaining to individual jobs were aggregated over a week and recorded as a single observation. The length of the simulation run covered 2650 jobs in the steady state. The method of batching was used to develop the summary statistics.

The scheduling rules were coded in FORTRAN and were interfaced with the simulation model written in SIMAN.

6.2 Experimental Results

The experimental results are shown in Tables 12 through 19. Tables 12 and 13 show the values of mean tardiness obtained under the MOD, MDD, HYB and GSP scheduling rules in the second experiment under the conditions of balanced and unbalanced workloads respectively. Tables 14 and 15 present the values of the proportion of tardy jobs, while the standard deviation of tardiness is shown in Tables 16 and 17. Mean flow time values are shown in Tables 18 and 19.

While implementing GSP, we monitored the number of times GSP was not able to fully solve a static problem generated during the simulation run because of the time trap. In the case of balanced workloads, the proportion of such problems of the total number of problems generated was approximately 19%. It increased to 34% for unbalanced workloads. This was primarily due to the fact that unbalanced workloads increase average job flow time. At any given point in time, therefore, there are more jobs in the system which results in larger static problems. Consequently, in the presence of the time trap, fewer static problems were solved over the length of the simulation run.

TABLE 12

MEAN TARDINESS

Balanced Workloads

Scheduling Rule	Flow Allowance			
	3	4	5	UN(1,8)
MOD	324	115	36	84
MDD	483	164	31	82
HYB	354	141	42	96
GSP	288	100	13	65

TABLE 13

MEAN TARDINESS

Unbalanced Workloads

Scheduling Rule	Flow Allowance			
	3	4	5	UN(1,8)
MOD	869	552	363	508
MDD	901	502	268	361
HYB	855	467	250	325
GSP	791	461	240	370

TABLE 14

PROPORTION OF TARDY JOBS

Balanced Workloads

Scheduling Rule	Flow Allowance			
	3	4	5	UN(1,8)
MOD	0.279	0.125	0.042	0.137
MDD	0.338	0.142	0.050	0.152
HYB	0.272	0.114	0.041	0.144
GSP	0.265	0.113	0.026	0.126

TABLE 15

PROPORTION OF TARDY JOBS

Unbalanced Workloads

Scheduling Rule	Flow Allowance			
	3	4	5	UN(1,8)
MOD	0.361	0.198	0.125	0.233
MDD	0.400	0.231	0.154	0.254
HYB	0.400	0.221	0.144	0.251
GSP	0.386	0.218	0.146	0.260

TABLE 16

STANDARD DEVIATION OF TARDINESS

Balanced Workloads

Scheduling Rule	Flow Allowance			
	3	4	5	UN(1,8)
MOD	357	171	81	118
MDD	523	210	45	82
HYB	357	194	69	105
GSP	346	158	33	69

TABLE 17

STANDARD DEVIATION OF TARDINESS

Unbalanced Workloads

Scheduling Rule	Flow Allowance			
	3	4	5	UN(1,8)
MOD	885	647	472	610
MDD	926	624	352	456
HYB	896	599	342	409
GSP	875	564	321	456

TABLE 18

MEAN FLOW TIME

Balanced Workloads

Scheduling Rule	Flow Allowance			
	3	4	5	UN(1,8)
MOD	2614	2706	2773	2725
MDD	2786	2720	2653	2705
HYB	2627	2664	2686	2685
GSP	2546	2632	2715	2679

TABLE 19

MEAN FLOW TIME

Unbalanced Workloads

Scheduling Rule	Flow Allowance			
	3	4	5	UN(1,8)
MOD	3254	3360	3480	3450
MDD	3254	3226	3229	3228
HYB	3195	3134	3286	3163
GSP	3040	3259	3359	3322

6.3 Analysis of Results

The results indicate that the selection of a different (and appropriate) dispatching rule at the bottleneck machine improves the system performance substantially. In particular, HYB produced results which were both effective and robust across varying levels of due date tightness.

However, among the scheduling rules, GSP yields the best overall results for mean tardiness. In the case of balanced workloads, there is a noticeable difference in the tardiness values obtained under GSP and the next best rule at a given flow allowance. This difference is retained for all flow allowance levels, and for random flow allowance as well. One-tailed tests of paired differences between GSP and the next best rule at a given flow allowance indicate that the null hypothesis concerning the equality of means can be rejected at a significance level of 0.24 when F = 3, and at significance levels in the range of 0.30-0.32 for other F values. [Stronger results, in terms of lower significance levels, are difficult to achieve primarily because of the large values of standard deviation of tardiness obtained as shown in Table 16. This is typical of dynamic problems especially when job arrival follows a Poisson process.]

The other scheduling rules exhibit variable relative performance, with MOD emerging superior at lower F values while MDD gives better results when due dates are loosely set and when they are assigned randomly.

When workloads are unbalanced, GSP continues to yield superior results for deterministic flow allowances. At F = 3, the null hypothesis can be rejected at a significance level of 0.22. However, the difference between GSP and the next best rule HYB is not significant at higher values of F. In fact, HYB emerges superior (at a significance level of 0.26) when due dates are set randomly.

Deterioration in the relative performance of GSP in the experiments with unbalanced workloads is, at least partially, attributable to the fact that, in this case, less than the best solution is returned in many more instances of the static problem within the computational time trap used. This is due to two factors. First, for the reason stated in Section 5.2, the size of the average static problem is larger.

Second, investigations of the static problem reveal that, for the same problem size, greater job due date variability leads to larger solution times. This explains why GSP´s performance is bettered by HYB and MDD when due dates are randomly set. If the available solution time is not adequate to obtain the best solution, GSP is likely to yield results similar to those given by MOD which provides the initial solution, although a comparison of these two rules indicates that significant improvements in tardiness values are achieved within the time trap itself.

GSP also results in the best values of proportion of tardy jobs when workloads are balanced, although when due dates are tightly set, HYB gives results which are not significantly different. However, for unbalanced workloads, MOD is consistently superior, while GSP, HYB, and MDD exhibit similar performance.

GSP is effective for the criterion of standard deviation of tardiness as well for balanced workloads across the range of flow allowances studied. The other three rules show variable relative performance with MOD, and to a lesser extent, HYB giving better results for tight due dates, while MDD is superior for higher as well as variable flow allowances.

For the measure of mean flow time, GSP exhibits a more variable performance. For both balanced and unbalanced workloads, GSP gives the best results when due dates are tight. For higher flow allowances, MDD and HYB yield better results in general.

7. SUMMARY

This study examines the effectiveness of decomposing a dynamic mean tardiness problem into a series of static problems and implementing the solution to the static problem on a rolling basis in a FMS. In doing so, it also evaluates the impact of the solution quality for the static problem within a dynamic framework. We present an implicit enumeration-based optimum-seeking method for the static problem. We also develop a decomposition heuristic which provides efficient solutions with reasonable computational effort.

The results of this study reveal that the efficacy of implementing the optimal or near-optimal solutions to the static problems on a rolling basis reported in Raman et al. (1989) is carried to a multiple machine system as well. Experimental investigations of both static and dynamic problems indicate that by expending a little extra computational effort in developing a global schedule for the entire system, significant improvement over local dispatching rules can be achieved. Note that, for the dynamic problem, GSP was implemented with a time trap of 1.0 CPU second, and therefore, it was not able to solve many static problems completely. In a real system, the CPU time trap would not be needed. Computations could continue until there was a system change (for example, a job arrival) that triggered the need for a new schedule. Generally, this time would be in minutes or hours (not 1.0 second), and hence, more static problems would be solved completely which could possibly lead to further improvement in the performance of GSP. These arguments also imply that, in real systems, optimum-seeking approaches such as the procedure presented in Section 3 merit serious consideration.

However, if for some reason there is no recourse other than to use dispatching rules, this study indicates the need to recognize the difference in the relative machine workloads. In a balanced system, MOD emerges as the best rule when due dates are tightly set while MDD is shown to be the best for larger flow allowances. When workload imbalances exist, HYB is shown to be effective across all levels of due date tightness investigated in this study.

ACKNOWLEDGEMENT

This research was partially supported by the IBE Summer Research Grant at the University of Illinois.

APPENDIX 1

NOTATION FOR THE MULTIPLE-MACHINE TARDINESS PROBLEM

j Job index, $j = 1,.., N$

J Set of available jobs = $\{j\}$

m Machine index, $m = 1,.., M$

t Time period, $t = 1,.., T$, where T is the scheduling horizon

d_j Due date of job j

p_j Processing time of job j

r_j Ready time of job j

c_j Completion time of job j

S_j Set of pairs of adjacent operations in job j, $(k,l) \in S_j$ if operation K immediately precedes operation l in job j

N_j Number of operations in job j

T_j Tardiness of job j = $\max(0, c_j - d_j)$

E_j Earliness of job j = $\max(0, d_j - c_j)$

p_{jk} Processing time of operation k in job j

P_{jk} Remaining processing time for job j at operation k

W_{jk} Remaining waiting time for job j at operation k

d_{jk} Due date of operation k in job j

r_{jk} Ready time of operation k in job j

$$x_{tjk} = \begin{cases} 1, & \text{if operation } k \text{ of job } j \text{ is completed at time } t \\ 0, & \text{otherwise} \end{cases}$$

$$R_{jkm} = \begin{cases} 1, & \text{if operation } k \text{ of job } j \text{ requires machine } m \\ 0, & \text{otherwise} \end{cases}$$

APPENDIX 2

PROOFS OF THEOREMS 1 AND 2

We present the proof of Theorem 2 first. As shown in Figure 5, let σ be a sequence of operations on a given machine in which operation a in job b is the immediate predecessor of operation c in job d. Let σ´ be the sequence of operations formed by interchanging these operations as shown in Figure 6.

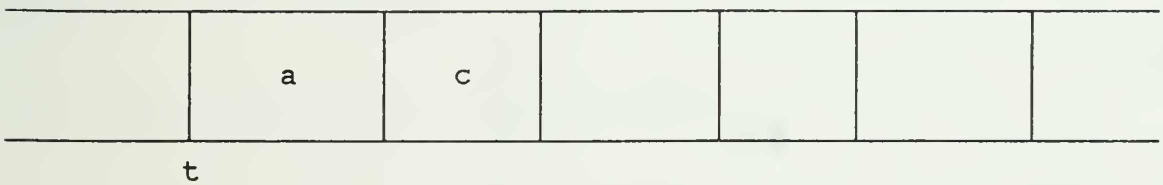

Figure 5 - Sequence σ

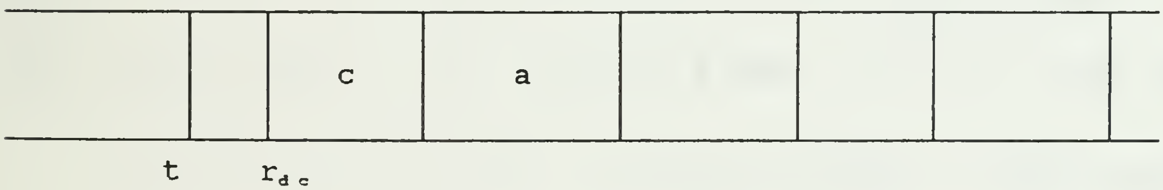

Figure 6 - Sequence σ´

Let T(σ) and T(σ´) denote the total tardiness of operations a and c in σ and σ´ respectively. Given that

$$t \le r_{dc} < t + p_{ba}$$

we need to show that

$$RMOD_{dc} \le RMOD_{ba} \quad \text{implies} \quad T = T(\sigma) - T(\sigma') \ge 0$$

We have

$$T = \max(0,\ t + p_{ba} - d_{ba}) + \max(0,\ t + p_{ba} + p_{dc} - d_{dc})$$

$$- \max(0,\ r_{dc} + p_{dc} - d_{dc})$$

$$- \max(0,\ r_{dc} + p_{dc} + p_{ba} - d_{ba})$$

For simplicity, we drop references to jobs b and d from our notation. Then

$$T = T_a + T_c - T'_c - T'_a \qquad \text{(A1)}$$

where

$T_a = \max(0,\ t + p_a - d_a)$; $T_c = \max(0,\ t + p_a + p_c - d_c)$

$T'_c = \max(0,\ r_c + p_c - d_c)$ and

$T'_a = \max(0,\ r_c + p_c + p_a - d_a)$.

Note that $T'_a \geq T_a$, and $T_c \geq T'_c$.

Because $RMOD_c \leq RMOD_a$, we have

$$\max[\max(t, r_c) + p_c,\ d_c] + \max(t, r_c)$$

$$\leq \max[\max(t, r_a) + p_a,\ d_a] + \max(t, r_a)$$

or $\max(r_c + p_c,\ d_c) + r_c \leq \max(t + p_a,\ d_a) + t$ (A2)

Depending upon the values of $r_c + p_c$, d_c, $t + p_a$, and d_a, the following four cases are possible.

Case I: $r + p_c \geq d_c$; $t + p_a \geq d_a$

From (A2) it follows that

$$(r_c + p_c) + r_c \leq (t + p_a) + t \qquad \text{(A3)}$$

In this case T_a, T_c, T'_a, and $T'_c \geq 0$, and

$$T = T_a + T_c - T'_c - T'_a$$

$$= (t + p_a) - (r_c + p_c) + t - r_c \geq 0 \quad \text{[from (A3)]}$$

Case II: $r_c + p_c \geq d_c$; $t + p_a \leq d_a$

From (A2) we have

$$(r_c + p_c) + r_c \leq d_a + t \qquad \text{(A4)}$$

In this case, $T_c \geq 0$, $T'_c \geq 0$, $T_a = 0$, and $T'_a \geq 0$.
Also, $T = T_c - T'_c - T'_a$. It suffices to consider only the case in which $T'_a > 0$. In this case,

$$T = (t + p_a) - r_c - (r_c + p_c + p_a - d_a)$$

$$= d_a - (r_c + p_c) + (t - r_c) \geq 0 \quad \text{[from (A4)]}$$

Case III: $r_c + p_c \leq d_c$; $t + p_a \geq d_a$

From (A2) we have

$$d_c + r_c \leq (t + p_a) + t \qquad \text{(A5)}$$

In this case, $T_a \geq 0$, $T_c \geq 0$, $T'_c = 0$, and $T'_c > 0$.

It follows that

$$T = T_a + T_c - T'_a$$

$$= t + (t + p_a) - (r_c + d_c) \geq 0 \quad \text{[from (A5)]}$$

Case IV: $r_c + p_c \leq d_c$; $t + p_a \leq d_a$

From (A2), we have

$$d_c + r_c \leq d_a + t \tag{A6}$$

In this case $T_a = T_c = 0$, $T'_a \geq 0$, and $T'_c \geq 0$. Hence,

$$T = T_c - T'_a$$

$$= \max(0, t + p_a + p_c - d_c)$$

$$- \max(0, r_c + p_c + p_a - d_a)$$

$$\geq 0 \quad \text{[from (A6)]}$$

This completes the proof of Theorem 2. Consider Theorem 1 next. In a non-delay schedule generation procedure, we consider only those operations which are currently available at a machine. Therefore, $\max(t, r_c) = t$, and the RMOD rule reduces to the MOD rule. Also, since there is no idle time inserted in the sequence because of an interchange of operations a and c, the tardiness of operations following these remains unchanged. The result stated in Theorem 1 follows immediately. This completes the proof of Theorem 1.

REFERENCES

Baker, K. R. (1974), Introduction to Sequencing and Scheduling, John Wiley and Sons, New York, NY.

Baker K. R. (1984), "Sequencing Rules and Due Date Assignments in a Job Shop", Management Science, Vol. 30, 1093-1104.

Baker, K. R. and J. M. W. Bertrand (1982), "A Dynamic Priority Rule for Sequencing Against Due dates", Journal of Operations Management, Vol. 3, 37-42.

Baker, K. R. and J. J. Kanet (1983), "Job Shop Scheduling with Modified Due dates", Journal of Operations Management, Vol. 4, 11-22.

Carroll, D. C. (1965), "Heuristic Sequencing of Single and Multiple Component Jobs", Ph.D. Dissertation, MIT, Cambridge, MA.

Conway, R. W. (1965), "Priority Dispatching and Job Lateness in a Job Shop", Journal of Industrial Engineering, Vol. 16, 123-130.

Kanet, J. J. and J. C. Hayya (1982), "Priority Dispatching with Operation Due dates in a Job Shop", Journal of Operations Management, Vol. 2, 155-163.

Ow, P. S. (1985), "Focused Scheduling in Proportionate Flowshops", Management Science, Vol. 31, 852-869.

Rachamadugu, R. V., N. Raman and F. B. Talbot (1986),"Real-Time Scheduling of an Automated Manufacturing Center",in Proceedings of the Conference on Real-Time Optimization in Automated Manufacturing Facilities, National Bureau of Standards, Gaithersburg, MD, 293-316.

Raman, N. (1988), "Real Time Scheduling Problems in a General Flexible Manufacturing System", Ph. D. Dissertation, University of Michigan, Ann Arbor, MI.

Rinnooy Kan, A. H. G. (1976), Machine Scheduling Problems Classification, Complexity, and Computations, Nijhoff, The Hague, Netherlands.

Talbot, F. B. (1982), "Resource Constrained Project Scheduling with Time-Resource Tradeoffs: The Nonpreemptive Case", Management Science, Vol. 28, 1197-1210.

Vepsalainen, A. P. J. and T. E. Morton (1987), "Priority Rules for Job Shops with Weighted Tardiness Costs", Management Science, Vol. 33, 1035-1047.