A JOB SHOP SCHEDULING APPROACH TO PART TYPE SELECTION IN FLEXIBLE MANUFACTURING SYSTEMS

YEONG-DAE KIM
Department of Industrial Engineering
Korea Advanced Institute of Science and Technology
Yusong-gu, Daejon
Korea

CANDACE ARAI YANO
Department of Industrial & Operations Engineering
University of Michigan
Ann Arbor, MI 48109-2117

Technical Report 91-31

October 1991

Revised May 1992

A Job Shop Scheduling Approach to Part Type Selection in Flexible Manufacturing Systems

Abstract

The literature on part type selection for flexible manufacturing systems has focused on maximizing throughput. We present an approach for part type selection in which meeting due dates is the primary objective. The approach is based on the idea of using information from the solution of an approximate, aggregate scheduling problem as the basis for determining release priorities. The need for the approximation and aggregation arises because the exact machine configuration (partitioning of identical machines into groups, and loading of tools) cannot be decided until the parts are selected.

We develop and evaluate several different policies for setting release priorities in a context where the approximate, aggregate schedule is accomplished using list processing (dispatching) rules. The results indicate that using information from such a schedule to set release priorities performs far better than using simpler procedures.

1. Introduction

Flexible manufacturing systems have several characteristics that make scheduling difficult. First, many different types of parts can be processed on the system at any given time if the machines are tooled appropriately. Second, changeovers between different types of parts are negligible, so parts need not be processed in batches and each unit within an order can be scheduled separately. Finally, in some cases, an operation can be performed by more than one machine, so routing decisions also must be made.

As a consequence of these complicating factors, many FMSs are "scheduled" by controlling the overall rate of release of items to the system, then possibly using simple priority rules to determine which item (from the queue) should be processed next on each machine. Some such policies may be effective when due dates are not a major consideration (see, for example, Buzacott and Shanthikumar 1985). On the other hand, when due dates are tight, it may be desirable to use a more sophisticated approach to scheduling.

The problem of determining the pool of parts eligible for release to the system is commonly called the part type selection problem. Tool magazine constraints, if any, are incorporated in this problem to ensure that it is feasible for any part within the pool to be processed without tool changes between operations. (Some FMSs are able to change tools on-line, but many flexible assembly systems do not have this capability.)

There are several approaches to the part type selection problem in the literature: the group technology approach; the sequential decision approach; and the constraint-directed approach (Hwang 1986). The *group technology approach* uses the concept of group technology in grouping parts; that is, parts with similar characteristics are grouped together. Kusiak (1984), Kumar *et al.* (1986), and Chakravarty and Shtub (1984) use this type of approach. In the *sequential decision approach,* parts are included sequentially to maximize the probability of a desirable outcome or to maximize dollar savings. This approach is discussed in Whitney and Gaul (1984) and Whitney and Suri (1984). The *constraint-directed approach* considers due dates, processing capacities of the machine tools, and tool magazine capacities. The approaches of Kiran and Tansel (1986), Rajagopalan (1986), O'Grady and Menon (1984, 1985, 1987), Chakravarty and Shtub (1987), Bastos (1988), Afentakis *et al.* (1989), Stecke (1989), Hwang and Shogan (1989), and Kim and Yano (1991) can be classified as constraint-directed. Stecke and Kim (1986) suggest a particular constraint-directed method called the *flexible approach,* which accommodates changes in the part mix and machine breakdowns. This approach is compared with the approaches of Whitney and Gaul (1984) and Hwang (1986) in Stecke and Kim (1988).

Various objectives are used in these papers. For example, the various constraint-directed approaches use the following as objectives: minimizing setup and inventory costs; maximizing the number of selected orders; minimizing number of setups; minimizing total time (processing and setup time) needed to produce all the parts; meeting daily requirements for the part types; balancing workloads of the machines; and minimizing the total idle time on the machines.

Another related problem is called the *grouping problem*, which involves deciding how to partition machines of a given type into groups, where each group has identical tooling. The *loading problem* involves assigning operations (and related tools) to these groups. These decisions obviously influence the variety of operations that can be performed at any time and routing options. However, the grouping and loading decisions cannot be made without first deciding the parts eligible for release. Moreover, nearly all of the literature on the grouping and loading problems deal with objectives in which due dates are not considered. See Lashkari *et al.* (1987), Sarin and Chen (1987), Han *et al.* (1989), Shanker and Srinivasulu (1989), Wilson (1989), Ram *et al.* (1990), and Kim and Yano (1990) for recent papers on this topic.

We consider the part type selection problem with the objective of minimizing the total tardiness for a given set of orders. Here, an *order* is defined by an order quantity, a due date, and a part type. We restrict our attention to setting priorities that specify the sequence in which part types are considered for inclusion in the pool of parts eligible for release to the system. We assume that all orders selected for one tool setup are released simultaneously. These orders are completed and a tool changeover period occurs before another set of orders is released. Thus, dynamic tool changes are not considered.

2. The Job Shop Scheduling Approach

As mentioned earlier, the part type selection and grouping/loading problems are closely related to one another. When due dates are considered, these problems are related even to scheduling problems. (See Figure 1.)

>> *Insert Figure 1 here* <<

Our problem can be viewed as a general job shop problem with additional complications due to the possibility of alternative routings that are a function of the machine grouping and loading decisions. Additionally, due to tool magazine capacity constraints, there is a restriction on the number of jobs (orders) that can be processed in a single tool setup. Thus, we need to partition the jobs into subsets of jobs (batches) and to sequence these subsets for release to the system.

Because of the difficulty of the problem, we develop a heuristic approach to specify priorities for the parts. The parts are then partitioned into batches and released to the system consistently with these priorities. The objective is to minimize total tardiness of the orders. Since the grouping and loading problems cannot be solved in advance of the part type selection problem, we must necessarily make some approximations to solve the scheduling problem. These approximations are described in more detail in the next section.

This paper presents our method in two steps. In the first step, which is described in Section 3, the problem is converted into an approximate job shop scheduling problem, and several different list scheduling procedures are used to construct heuristic solutions for this approximate problem. In the next step (described in Section 4), information from these schedules is used to determine priorities of the orders. Several methods are presented for this purpose. Computational results are reported in Section 5, and conclusions appear in Section 6.

3. The Approximate Job Shop Scheduling Problem

Recall that our objective is to minimize tardiness, and scheduling problems with tardiness objectives are notoriously hard to solve. Even the single machine problem with the objective of minimizing total tardiness is NP-complete (Karp 1972 and Lenstra *et al.* 1977). Therefore, various heuristics have been developed for use in real manufacturing systems.

One commonly-used class of heuristic algorithms is list scheduling. Such an algorithm lists the operations waiting (or available) for a given machine and assigns them to the machine in some order as the machine becomes available. The ordering of operations is commonly done using priority (or dispatching) rules. Various priority rules have been devised and tested for many different objectives. See Panwalkar and Iskander (1977) and Blackstone *et al.* (1982) for extensive surveys. For detailed descriptions of various rules, see for example Baker and Kanet 1983, Berry and Rao 1975, Dar-El and Wysk 1982, Elvers and Taube 1983, Kim 1987, Kim 1990, O'Grady and Harrison 1985, Russel *et al.* 1987, Scudder and Hoffmann 1985, and Vepsalainen and Morton 1987.

We use the term 'priority' in two different ways in this paper. In the context of list scheduling algorithms, the priority of an operation specifies its (logical) position in the queue in front of a machine. (Note that its physical position may be different.) We address the problem of determining release priorities, which specify the sequence in which jobs (or orders for part types) are released to the system. When needed for clarity, we will refer to the former as "processing priority," and the latter as "release priority."

In our problem, each part may require several processing steps on various types of machines. Since we have not yet decided how to group the machines and load tools, it is difficult to anticipate exactly how each type of part will be processed. On the other hand, it is likely that steps that can be performed *consecutively* on one machine type will be performed by a single machine, rather than being split between two or more machines, unless a capacity constraint is violated by doing so. Thus, we aggregate such steps, and refer to them collectively as an *operation*. In our procedure, we schedule these operations.

Although it is possible to consider tool magazine capacity constraints in defining feasible operations, we cannot explicitly consider them in determining the schedule since the loading problem has not been solved at this stage. Thus, we solve a relaxed version of the problem where the tool magazine capacity constraints are ignored and, consequently, each machine can perform all operations associated with its machine type. We also ignore tool changeovers in the relaxed scheduling problem. Thus, the exact timing of events will differ from those in the approximate schedule. Moreover, because we have relaxed the tool magazine capacity constraints, in practice, it may be possible to process the orders on only a portion of the machines to which they have been assigned.

The sequence of machines on which a part is processed is called a *routing*, and there may be several routings for a part if there are two or more machines of the same type that are tooled identically. To reflect the fact that individual units of an order may follow different routings, we index operations so that each operation for each unit of an order has its own operation index. For example, a 20-unit order for a part type with five operations will have 100 operations for the entire order.

In some dispatching rules, priorities depend on the number of successor operations and/or the amount of remaining work. Therefore, precedence definitions may affect the performance of these rules. Special precedence relationships among operations are imposed by the facts that each order may consist of more than one unit and each unit can be routed independently. One example of such a precedence relation is that the first unit of an order must precede the second unit. It is clear that we can eliminate from consideration schedules in which any unit in an order overtakes a lower-indexed unit, since there is an effectively identical schedule without such overtaking.

If there are multiple machines of the same type, the precedence relationships become a little more complicated. For example, if there are two identically tooled machines, the second unit in an order can be started before the first is complete. However, the third unit can be started only after the first unit has finished processing on the first machine on its route.

Figure 2 illustrates a precedence diagram for the operations within an order. In this example, the part type has four operations. There are two, one, three, and two machines available to process these four operations, respectively. The order quantity is eight. Each node in the figure denotes an operation, and each arc represents a precedence relation between operations.

>> *Insert Figure 2 here* <<

Although, in principle, it would be necessary to consider the detailed operation precedence information described above, the main reason for considering operations at such a fine level of detail is to allow interleaving of orders on the various machines. Earlier work by Kim (1990) indicates that interleaving adversely affects both flow time and due date performance. Although flow time is not our primary objective, long flow times lead to schedules in which many orders are processed simultaneously. Such schedules would generally be infeasible in contexts where part type selection must be performed because of tool magazine capacity constraints. Consequently, schedules with short flow times are likely to provide more realistic guidelines than schedules with long flow times. For these reasons, we chose to develop a method which assigns the same processing priority to all identical operations in an order. This tends to facilitate continuous processing of identical operations within the same order, unless none is currently available (due to delays at upstream machines). Different operations are assigned different processing priorities. We next explain how these priorities are computed.

We use list scheduling algorithms to generate schedules for the approximate scheduling problem, from which we then determine release priorities. We employ the best performing list scheduling algorithms (based on results of other research) for problem instances with alternative routings, where an order can be split across several machines operating in parallel. Several methods have been proposed by Kim (1990) to deal with this class of job shop problems. He reports that the modified due date (MDD) rule and the modified operation due date (MOD) rule outperformed several others (SLACK, Slack Per Remaining Operation, Slack Per Remaining Work, COVERT, and ATC, for example) in both due date and flow time performance. We selected MOD and MDD based on these results.

In MDD, the highest priority is given to the operation with the minimum modified due date, which is defined as the earlier of the actual due date of the job in which operation i is included and the earliest time the job could be completed if there were no delay for that job. In MOD, due dates are assigned to each operation (as progress milestones), with any available slack being allocated to all operations in proportion to the work remaining after that operation. Then, the highest priority is given to the operation with the minimum modified operation due date, which is the earlier of the operation due date and the earliest time the operation can be completed.

More formally, let p_i = processing time of operation i, d_i = due time of operation i (= due time of the order in which operation i is included), t = time for which the priorities are computed, and rk_i = total work remaining on i and its successors. Successors of an operation can be identified from the precedence data. In the case of MDD, the available operation with the minimum value of

$$\max \{ d_i , t + rk_i \}$$

is assigned to the next available machine that can process the operation. Similarly, in the case of MOD, the available operation with the minimum value of

$$\max \{ d_i - b (rk_i - p_i) , \ t + p_i \}$$

is assigned.

In the MOD rule, the parameter b must be specified. It can be viewed as a flow time allowance factor to account for the anticipated waiting time between operations. In this paper, we use b = 1.0 which performed the best among several values of b in the computational results obtained by Kim (1990). (This value roughly corresponds to a situation in which there is no waiting time.)

Rules based on remaining work (rk_i) were devised with the intent of using it as the basis for estimating the completion time of the job. When there are multiple parallel machines, units from a given order can be processed on more than one machine at the same time. Thus, it might be logical for completion time estimates to reflect this fact. On the other hand, since the loading and grouping problems have not been solved at this stage, we do not know exactly how many machines will be tooled so that they are capable of performing a given operation.

We employ two different methods to compute rk_i, leading to two variations for each of the rules. One method for computing rk_i is based on the assumption that all machines that are technically capable of performing an operation (if appropriately tooled) share equally in the processing of that operation (used in MDD2 and MOD2). The other method for computing rk_i is based on the assumption that only one machine will perform that operation on all units within an order (used in MDD1 and MOD1). In the first case, rk_i is calculated as $\sum_{j \in S_i} p_j / m_j + p_i$, where m_j is the number of machines capable of processing operation j and S_i is the set of successors of operation i, while in the latter case it is simply calculated as

$\sum_{j \in S_i} p_j + p_i$. To ensure that all identical operations within an order are assigned the same priority, we define the set of successors of operation i to include the successors for that unit only, not all successor operations in the same order.

The (approximate, aggregate) schedule is developed as follows. Each unit within an order is assigned a priority according to the selected rule, and as time proceeds, the priority is updated when applicable. We construct the schedule forward in time. Whenever a machine becomes available, the operation with the smallest value of the priority index at that time (t) is selected from among the operations waiting for the machine type. The minimum slack ($d_i - t - rk_i$) rule and the first-come first-service rule are used, in that order, to break ties.

4. Release Priorities for Orders

In this section, we present several methods for setting release priorities using information from the approximate schedule constructed in the first step. We consider measures of urgency that are functions of the due date, and the starting, completion, and flow times in the approximate schedule. In each method, orders are prioritized in increasing order of the measure of urgency. Let d_i, f_i, C_i, and S_i be the due date, the flow time, the completion time, and the starting time of order i in the approximate schedule. Here, flow time is defined as the difference between the completion time of the order and the start of processing of the first unit of the order. In other words, it is the duration of time that the order spends in the system. Also, let mf_i (minimum flow time) be the flow time of order i when it alone is processed on the system. The priority measures that we consider are:

EDD (earliest due date) : d_i
MIDDS (midpoint of due date and starting time) : $d_i - mf_i \,/\, 2$
DOMF (due date over minimum flow time) : $d_i \,/\, mf_i$
DMMF (due date multiplied by minimum flow time) : $d_i \cdot mf_i$
ECT (earliest completion time) : C_i
EST (earliest starting time) : S_i
MIDSC (midpoint of starting and completion times) : $(S_i + C_i) \,/\, 2$
COF (completion time over flow time) : $C_i \,/\, f_i$
CMF (completion time multiplied by flow time) : $C_i \cdot f_i$
COMF (completion time over minimum flow time) : $C_i \,/\, mf_i$
CMMF (completion time multiplied by minimum flow time) : $C_i \cdot mf_i$

SL (slack time) : $d_i - C_i$ ($\equiv sl_i$)

SOF (slack over flow time) : $sl_i \,/\, f_i$

SMF (slack multiplied by flow time) : $sl_i \cdot f_i$

SOMF (slack over minimum flow time) : $sl_i \,/\, mf_i$

SMMF (slack multiplied by minimum flow time) : $sl_i \cdot mf_i$

RANDOM (random) : a random number

In the EDD (earliest due date) method, orders are selected in increasing order of their due dates. The next three methods (MIDDS, DOMF, and DMMF) consider the minimum flow time of each order. The flow time of an order depends on the combination of orders that are scheduled together. Therefore, the value of f_i in the approximate schedule may not be a good estimate of the actual flow time of i within a different set of orders. As an alternative to f_i, we use mf_i scaled by an adjustment multiplier to provide an estimate of the actual flow time. Note, however, that if we set the adjustment multipliers to be equal for all orders, the multipliers can be deleted from the measures of urgency. By using mf_i instead of f_i, priorities can be decided without the approximate schedule. Note that the same is true of EDD. Thus, these four methods can serve as benchmarks to evaluate the benefit of constructing the approximate schedule.

The remaining twelve methods use information from the approximate schedules. ECT considers the completion times of the orders and EST considers the starting times in the resulting schedule, while MIDSC considers the midpoint of the completion and starting times for each order as the basis for determining priorities. These rules were selected since starting and completion times in the approximate schedule might be indicative of the urgency of an order, or of the (relative) time in the schedule where the order fits best in relation to other orders.

COF, CMF, COMF, and CMMF incorporate flow times as well as completion times of orders in the approximate schedules, but do not consider due dates explicitly. On the other hand, SL, SOF, SMF, SOMF, and SMMF are variations of slack-based rules that also consider flow times. Since the true effect of flow time is not well understood, we use both mf_i and f_i. in constructing these rules. Finally, in RANDOM, orders are selected randomly for release. This method serves as a very simple benchmark. It considers only resource constraints (processing time and required tools, for example), and ignores due dates and flow times.

5. Computational Experiments

The sixteen methods presented in the previous section were tested on 200 problems. The problems have the following characteristics: the number of orders ranges from 20 to 30, the order quantities range from 3 to 20 units, the total number of machines ranges from 10 to 20, and the number of machine types ranges from 4 to 8. The maximum number of orders that can be processed together in a single tool setup ranged from 5 to 10, and the tool setup (changeover) time was generated randomly based upon the number of tool slots in the tool magazines, since in real systems, the actual tool changeover time usually depends on the number of tools to be replaced.

Processing times and due dates were generated following a method used by Potts and Van Wassenhove (1982) for scheduling problems with due dates. The processing times for each part on each machine type was generated from a uniform distribution. Then, due dates of the orders were generated using two parameters, R and T, which are referred to as the *relative range* of due dates and the *tardiness factor*. Each due date was generated from a uniform distribution on $[P(1-T-R/2), P(1-T+R/2)]$, where P is the total processing time of all the orders divided by the number of machines in the system. (P denotes the absolute lower bound on the makespan of the problem.) T ranged from 0.1 to 0.5 and R ranged from 0.8 to 1.8 in the test problems.

Recall that the primary concern of our study is to determine how to assign release priorities. However, in order to evaluate the various methods described earlier, in practice, we would also have to select a part-type selection procedure (to decide how many orders to batch together in a single tool setup), and procedures for grouping machines and assigning operations and associated tools to groups. Since optimal solution procedures for these problems do not exist, the quality of the release priority decisions could easily be obscured by these latter decisions. Consequently, we have attempted to isolate the effects of the release priorities by standardizing the other decisions. We do so as follows. For each problem, the number of orders to be selected in each tool setup (n_s) is specified exogeneously. The machine grouping is generated as described earlier. We assume that for each machine group, all tools associated with the operations assigned to that group will fit into a single tool magazine for that machine type. Thus, it is not necessary to partition the machines into groups, nor to partition the operations and associated tools among them.

For each problem, schedules are constructed using a deterministic simulation for each combination of the four methods to construct approximate schedules and the seventeen release priority rules. Each schedule is constructed as follows. First, priorities of orders are determined by applying list scheduling algorithms to construct heuristic solutions for the approximate scheduling problem. Next, the n_s orders with the smallest values of the priority measure are released to the system (ties are broken using the EDD rule) and we construct a schedule for those orders using the same method as was used in the generation of the approximate schedule. Then, at the completion of these orders, a tool changeover is scheduled. Using the completion time of the tool changeover as the new starting point, we solve the approximate scheduling problem again to obtain new release priorities and identify the orders in the second tool setup. This procedure is repeated until all orders are released to the system. Note that in the first four methods and the RANDOM method, the release priorities of the orders need not be recomputed at each tool setup, since these values do not change. For the remaining 12 methods, however, priorities of the remaining orders are recomputed prior to each tool setup.

The various alternatives described above also include benchmarks that represent what would occur if there were no priority determination scheme, and list processing procedures were applied directly, except that the number of orders in each tool setup is constrained to be less than or equal to n_s. In particular, the schedules with EST as the release priority rule are the same as those that would have been obtained by the list processing procedure alone.

Since optimal solutions cannot be obtained in a reasonable amount of time, we compare the solutions by using a *relative deviation index*, which, for some method a, is defined as $(T_a - T_B) / (T_W - T_B)$ where T_a, T_B, and T_W are the objective value obtained from method a, the best objective value among the various methods, and the worst objective value among the various methods, respectively. This index has a value between 0 and 1 and reflects how well a particular method performs relative to the range of objective function values obtained for the problem.

Table 1 reports the average relative deviation index and number of best solutions found by each algorithm. There was little difference among the approaches for constructing the approximate schedule. However, some release priority determination methods (especially ECT and MIDSC) outperformed the others. Note that some procedures, although logically designed, are not much better than the RANDOM procedure. Consequently, careful design of procedures to specify release priorities is important. Also note that several of the priority determination methods produced much better results than EST, which (as mentioned above) is equivalent to applying list processing without the first step of determining release priorities. Consequently, careful and intelligent use of information from the approximate schedule can provide significant improvements beyond what can be accomplished through direct application of list processing procedures without this pre-processing step.

>> *Insert Table 1 here* <<

Noticeably better results were obtained by the simpler release priority determination methods. In other words, considering slack time or flow time in the release priorities may have had a negative impact on performance. Since due dates, and consequently also slack times, have already been considered in the generation of the approximate schedule, it appears to be unnecessary, and perhaps even inappropriate, to consider them again in determining release priorities. Similarly, while approximate schedules with short flow times may be closer to what can be achieved after the batching of orders and tool changeovers are considered , flow times do not provide much guidance on which orders should be released first. Recall, however, that the approximate schedules were constructed using procedures that tend to yield short flow times. Thus, the approximate schedules from which the release priorities were determined already had relatively short flow times.

Since EDD, ECT, MIDSC performed similarly, we generated 800 additional problems to evaluate them further. Summary statistics for all 1000 problems are shown in Table 2. ECT performed best for all four approaches for generating approximate schedules. This suggests that using an approximate schedule rather than simple due date information (cf. EDD) is advantageous. Note that the completion times in the approximate schedule reflect contention for resources at various times during the horizon, and thereby more accurately capture when each job *can* be completed given the resource contention, or when it *should* be completed in view of tardiness penalties. EDD, on the other hand, simply indicates the most desirable completion time, which may be far from feasible.

>> *Insert Table 2 here* <<

Tables 3 and 4 show how tightness of due dates affects the performance of the various methods. ECT consistently dominates the other methods for all tardiness factors and most due date ranges. The performance of EDD, on the other hand, deteriorates as due dates become tighter and as the due date range decreases (i.e., as due dates become more dense). When due dates are sparse, orders with earlier due dates usually precede those with later due dates in most good schedules. Therefore, EDD results in priorities similar to those obtained from ECT. Also, when due dates are not tight, most methods result in very low tardiness, and thus perform similarly.

>> *Insert Tables 3 and 4 here* <<

In conclusion, our results suggest that good release priorities can be determined by sequencing jobs in increasing order of their completion times in an approximate schedule generating using MOD1. Note that MOD2 also performed well in conjunction with ECT. Thus, it appears that using operation due dates rather than order due dates leads to approximate schedules that provide better scheduling guidelines. ECT performed well when used in conjunction with all four methods to generate approximate schedules, and perhaps surprisingly, outperformed EST. Although the reasons for this are not completely clear, there are some possible explanations for this phenomenon. In some instances, some orders may be started early, simply to take advantage of resources that might otherwise be idle, even if their due dates are not imminent. It is possible that such an order could have been started much later (and perhaps another order much earlier) with essentially the same results with regard to total tardiness. ECT might be viewed as a due date adjusted for resource availability, with some orders finishing before their due dates, and some possibly after, in order to smooth the utilization of resources over time. When put in this perspective, it appears to be a very logical policy.

6. Concluding Remarks

In this paper we presented an approach to part type selection where the primary objective is to minimize the total tardiness for a given set of orders. We presented a two-step approach to the problem. In the first step, we construct an approximate schedule in which tool magazine capacity constraints are relaxed and tool setup times are ignored. We evaluated several different list processing (dispatching) rules for constructing the schedule. In the second step, we use information from the schedule to determine release priorities for the orders. We devised and tested 17 priority rules for this purpose.

In experimental results on randomly generated problems, we found that a variation of the modified operation due date (MOD) rule for schedule construction and the earliest completion time (ECT) for release determination, provided the best results. Further research is needed to develop approaches that can more accurately incorporate tool magazine capacity constraints and tool changeover times. Also, further research is needed to better integrate the part type selection decisions with machine loading and grouping decisions. Some recent work along these lines appears in Kim and Yano (1991).

Acknowledgement

Yeong-Dae Kim was supported by the Korea Science and Engineering Foundation under grant 923-0900-017-1.

Table 1. Test results for part priority determination (17 methods)

Priority Determination Methods	MDD1		MOD1		MDD2		MOD2	
	RDI†	NBS‡	RDI†	NBS‡	RDI†	NBS‡	RDI†	NBS‡
EDD	.151	31	.096	40	.133	31	.105	32
MIDDS	.167	26	.110	35	.149	25	.122	27
DOMF	.403	0	.352	0	.387	0	.361	0
DMMF	.144	4	.124	10	.138	4	.130	5
ECT	.058	34	.042	57	.049	36	.049	34
EST	.248	1	.212	5	.236	0	.224	2
MIDSC	.066	30	.060	47	.058	40	.057	26
COF	.524	0	.498	0	.523	0	.490	0
CMF	.162	3	.160	9	.157	4	.163	5
COMF	.339	0	.313	1	.332	0	.311	0
CMMF	.184	2	.169	3	.176	1	.171	3
SL	.587	1	.509	2	.572	1	.529	1
SOF	.645	0	.551	2	.624	0	.567	0
SMF	.518	3	.436	4	.507	3	.452	4
SOMF	.596	0	.520	0	.581	0	.523	0
SMMF	.541	2	.479	7	.532	2	.502	3
RANDOM	.699	0	.661	0	.685	0	.667	0

† Mean relative deviation index

‡ Number of problems for which the method found the best solution among those obtained from the various methods tested in this experiment

Table 2. Test results for part priority determination (three methods)

List Scheduling Algorithms	Priority Determination Methods	Relative Deviation Index (mean)	Number of Best Solutions
MDD1	EDD ECT MIDSC	.640 .314 .376	154 209 185
MOD1	EDD ECT MIDSC	.409 .235 .298	224 294 260
MDD2	EDD ECT MIDSC	.599 .301 .357	155 211 176
MOD2	EDD ECT MIDSC	.467 .258 .320	176 218 201

Table 3. Test results for different tardiness factors

Approach	Tardiness Factors (number of problems tested)									
	0.1 (300)		0.2 (250)		0.3 (200)		0.4 (150)		0.5 (100)	
	RDI	NBS	RDI	NBS	RDI	NBS	RDI	NBS	RDI	NBS
MDD1*EDD	.41	106	.55	44	.79	4	.85	0	.92	0
MDD1*ECT	.30	107	.32	56	.34	22	.32	10	.28	14
MDD1*MIDSC	.36	105	.43	41	.38	18	.34	11	.34	10
MOD1*EDD	.35	140	.38	58	.48	18	.53	7	.61	1
MOD1*ECT	.23	132	.26	69	.23	40	.23	39	.21	14
MOD1*MIDSC	.30	122	.36	66	.29	36	.26	22	.23	14
MDD2*EDD	.39	106	.53	39	.73	7	.79	1	.86	2
MDD2*ECT	.29	105	.31	53	.34	23	.31	16	.25	14
MDD2*MIDSC	.36	100	.41	36	.36	17	.30	16	.28	7
MOD2*EDD	.32	122	.43	47	.55	4	.61	3	.62	0
MOD2*ECT	.27	106	.28	54	.26	26	.26	16	.19	16
MOD2*MIDSC	.34	102	.37	41	.30	30	.28	16	.24	12

Table 4. Test results for different relative ranges of due dates

Approach	Due Date Ranges (number of problems tested)											
	0.8 (250)		1.0 (250)		1.2 (200)		1.4 (150)		1.6 (100)		1.8 (50)	
	RDI	NBS	RDI	NBS	RDI	NBS	RDI	NBS	RDI	NBS	RDI	NBS
MDD1*EDD	.83	6	.76	18	.62	30	.48	44	.29	42	.34	14
MDD1*ECT	.31	37	.28	47	.30	40	.32	41	.36	32	.46	12
MDD1*MIDSC	.36	26	.34	41	.35	41	.43	31	.43	33	.47	13
MOD1*EDD	.53	17	.50	26	.39	56	.29	51	.17	50	.23	24
MOD1*ECT	.21	54	.22	39	.24	64	.24	51	.27	37	.32	19
MOD1*MIDSC	.27	52	.29	49	.25	57	.36	46	.32	41	.42	15
MDD2*EDD	.77	9	.71	20	.59	31	.44	44	.29	38	.31	13
MDD2*ECT	.29	37	.27	51	.30	42	.32	39	.33	31	.45	11
MDD2*MIDSC	.33	27	.32	29	.32	48	.41	32	.43	28	.49	12
MOD2*EDD	.59	12	.57	20	.45	35	.34	49	.22	43	.26	17
MOD2*ECT	.24	35	.23	55	.27	45	.27	41	.30	31	.38	11
MOD2*MIDSC	.30	42	.30	41	.26	45	.39	31	.37	29	.44	13

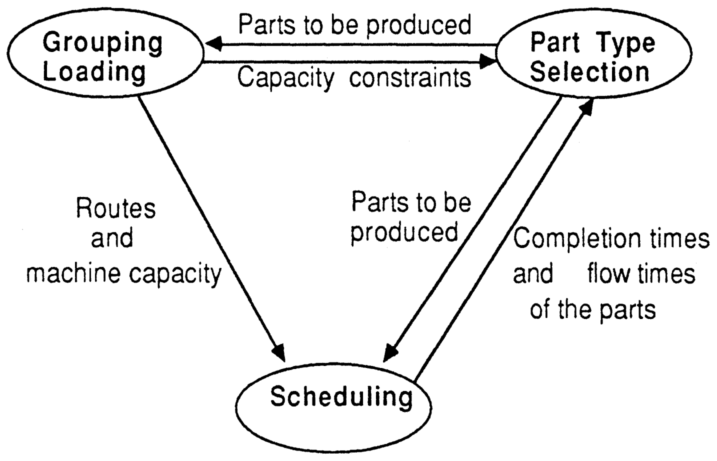

Figure 1. Relationships among the problems

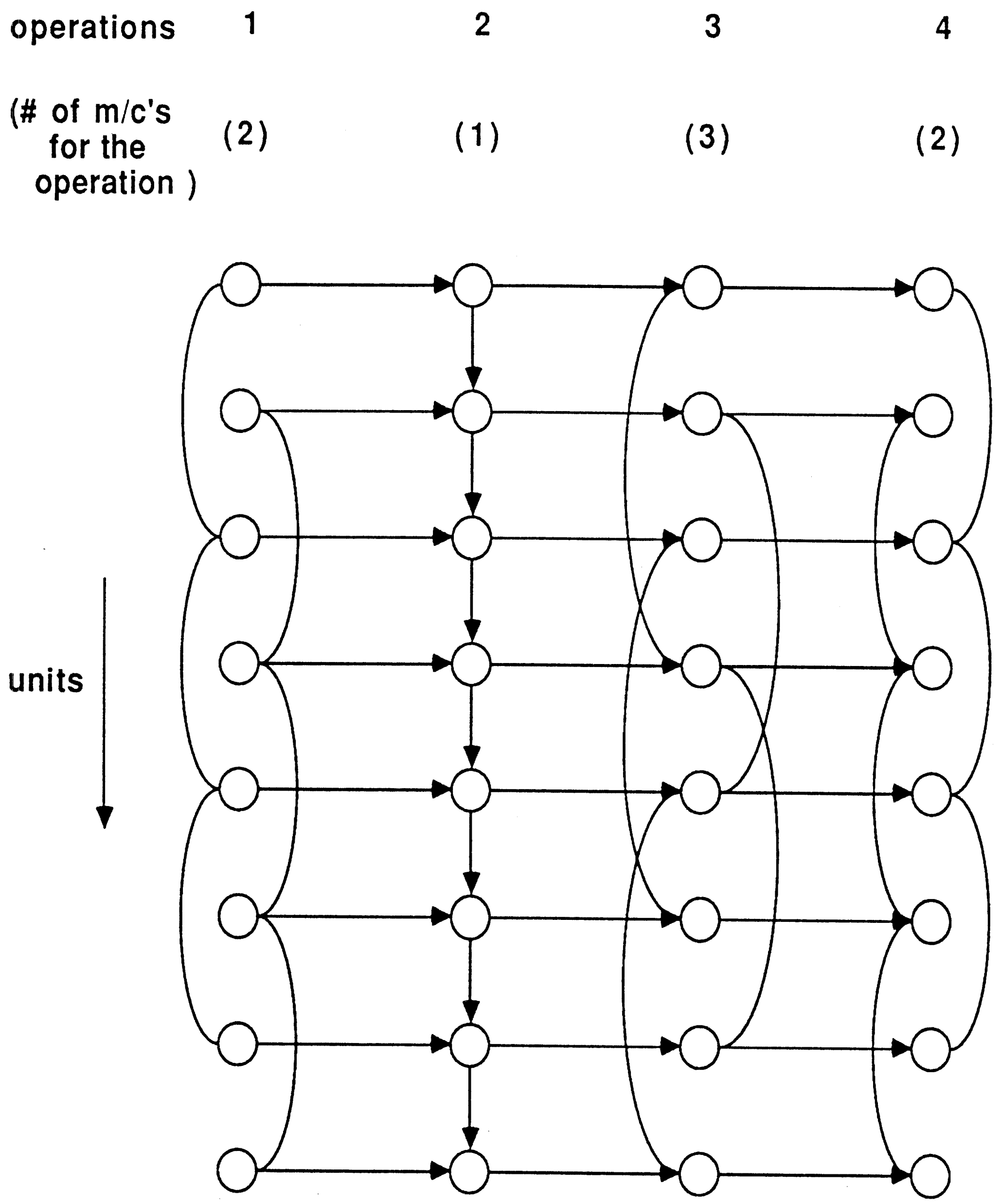

Figure 2. Precedence relationship diagram (an example)

References

Afentakis, F., M. M. Solomon, and R. A. Millen, 1989. The Part-Type Selection Problem. *Proc. of the 3rd ORSA/TIMS Conf. on Flexible Manufacturing Systems: Operations Research Models and Applications*, K.E. Stecke and R. Suri (editors), Elsevier Science Publishers B.V., Amsterdam, pp.141-146.

Baker, K. R. and J. J. Kanet, 1983. Job Shop Scheduling with Modified Due Dates. *Journal of Operations Management*, Vol.4, No.1, pp.11-22.

Bastos, J. M., 1988. Batching and Routing: Two Functions in the Operational Planning of Flexible Manufacturing systems. *European Journal of Operational Research*, Vol.33, pp.230-244.

Berry, J. B. and V. Rao, 1975. Critical Ratio Scheduling: An Experimental Analysis. *Management Science*, Vol.22, No.2, pp.192-201.

Blackstone, Jr., J. H., D. T. Phillips, and G. L. Hogg, 1982. A State-of-the-Art Survey of Dispatching Rules for Manufacturing Job Shop Operations. *International Journal of Production Research*, Vol.20, No.1, pp.27-45.

Buzacott, J.A. and J.G. Shanthikumar, 1985. On approximate queueing models of dynamic job shops. *Management Science* Vol. 31, No. 7, pp. 870-887.

Chakravarty, A. K. and A. Shtub, 1984. Selecting Parts and Loading Flexible Manufacturing Systems. *Proceedings of the 1st ORSA/TIMS Special Interest Conference on Flexible Manufacturing Systems*, Ann Arbor, MI, pp.284-289.

Chakravarty, A. K. and S. Shtub, 1987. Capacity, Cost, and Scheduling Analysis for a Multiproduct Flexible Manufacturing Cell. *International Journal of Production Research*, Vol.25, No.8, pp.1143-1156.

Dar-El, E. M. and R. A. Wysk, 1982. Job Shop Scheduling – A Systematic Approach. *Journal of Manufacturing Systems*, Vol.1, No.1, pp.77-88.

Elvers, D. A. and L. R. Taube, 1983. Time Completion for Various Dispatching Rules in Job Shops. *OMEGA*, Vol.11, No.1, pp.81-89.

French, S., 1982. *Sequencing and Scheduling: An Introduction to the Mathematics of the Job-Shop*. Halsted Press, New York.

Han, M-H., Y. K. Na, and G. L. Hogg, 1989, Real-Time Tool Control and Job Dispatching in Flexible Manufacturing Systems. *International Journal of Production Research*, Vol.27, No.8, pp.1257-1267.

Hwang, S., 1986. A Constraint-Directed Method to Solve the Part Selection Problem in Flexible Manufacturing Systems Planning Stage. *Proc. of the 2nd ORSA/TIMS Conf. on Flexible Manufacturing Systems, Ann Arbor, MI*, K.E. Stecke and R. Suri (editors), Elsevier Science Publishers B.V., Amsterdam, pp.297-309.

Hwang, S., and A. Q. Shogan, 1989. Modelling and Solving an FMS Part Type Selection Problem. *International Journal of Production Research*, Vol.27, No.8, pp.1349-1366.

Karp. R.M. , 1972. Reducibility Among Combinatorial Problems. *Complexity of Computer Computations*. R.E. Miller and J.W. Thatcher (eds.), Plenum Press, New York.

Kim, Y-D., 1987. On the Superiority of a Backward Approach in List Scheduling Algorithms for Multi-Machine Makespan Problems. *International Journal of Production Research*, Vol.25, No.12, pp.1751-1759.

Kim, Y-D., 1990. A Comparison of Dispatching Rules for Job Shops with Multiple Identical Jobs and Alternative Routings. *International Journal of Production Research*, Vol.28, No.5, pp.953-962.

Kim, Y-D. and C. A. Yano, 1990. Heuristic Approaches to Loading Problems in Flexible Manufacturing Systems. To appear in *IIE Transactions*..

Kim, Y-D. and C. A. Yano, 1991. An Iterative Approach to System Setup Problems in Flexible Manufacturing Systems. To appear in *International Journal of Flexible Manufacturing Systems.*

Kiran, A. S. and B. C. Tansel, 1986. The System Setup in FMS: Concepts and Formulation. *Proc. of the 2nd ORSA/TIMS Conf. on Flexible Manufacturing Systems, Ann Arbor, MI*, K.E. Stecke and R. Suri (editors), Elsevier Science Publishers B.V., Amsterdam, pp.321-332.

Kumar, K. R., A. Kusiak, and A. Vannelli, 1986. Grouping of Parts and Components in Flexible Manufacturing Systems. *European Journal of Operational Research*, Vol.24, No.3, pp. 387-397.

Kusiak, A., 1984. The part Families Problem in Flexible Manufacturing Systems. *Proc. of the 1st ORSA/TIMS Special Interest Conf. on Flexible Manufacturing Systems, Ann Arbor, MI*, pp. .237-242.

Lashkari, R. S., S. P. Dutta, and A. M. Padhye, 1987, A New Formulation of Operation Allocation Problem in Flexible Manufacturing Systems: Mathematical Modeling and Computational Experience. *International Journal of Production Research*, Vol.25, No.9, pp.1267-1283.

Lenstra, J.K., A.H.G. Rinnoy Kan, and P. Brucker, 1977. Complexity of Machine Scheduling Problems. Annals of Discrete Mathematics, Vol. 1, pp. 343-362.

O'Grady, P. J. and C. Harrison, 1985. A General Search Sequencing Rule for Job Shop Sequencing. *International Journal of Production Research*, Vol.23, No.5, pp.961-973.

O'Grady, P. J. and U. Menon, 1984. A Flexible Multiobjective Production Planning Framework for Automated Manufacturing Systems. *Engineering Costs and Production Economics*, Vol.8, pp.189-198.

O'Grady, P. J. and U. Menon, 1985. A Multiple Criteria Approach for Production Planning of Automated Manufacturing. *Engineering Optimization*, Vol.8, No.2, pp.161-175.

O'Grady, P. J. and U. Menon, 1987. Loading a Flexible Manufacturing System. *International Journal of Production Research*, Vol.25, No.7, pp.1053-1068.

Panwalkar, S. S. and W. Iskander, 1977. A Survey of Scheduling Rules. *Operations Research*, Vol.25, No.1, pp.45-61.

Potts, C. N. and L. N. Van Wassenhove, 1982. A Decomposition Algorithm for the Single Machine Total Tardiness Problems. *Operations Research Letters*, Vol.1, No.5, pp.177-181.

Rajagopalan, S., 1986. Formulation and Heuristic Solutions for Parts Grouping and Tool Loading in Flexible Manufacturing Systems. *Proc. of the 2nd ORSA/TIMS Conf. on Flexible Manufacturing Systems, Ann Arbor, MI,* K.E. Stecke and R. Suri (editors), Elsevier Science Publishers B.V., Amsterdam, pp.312-320.

Ram, B., S. Sarin, and C. S. Chen, 1990, A Model and a Solution Approach for the Machine Loading and Tool Allocation Problem in a Flexible Manufacturing System. *International Journal of Production Research*, Vol.28, No.4, pp.637-645.

Russell, R. S., E. M. Dar-El, and B. S. Taylor, III, 1987. A Comparative Analysis of the COVERT Job Sequencing Rule Using Various Shop Performance Measures. *International Journal of Production Research*, Vol.25, No.10, pp.1523-1540.

Sarin, S. C. and C. S. Chen, 1987, The Machine Loading and Tool Allocation Problem in a Flexible Manufacturing System. *International Journal of Production Research*, Vol.25, No.7, pp.1081-1094.

Scudder, G. D. and T. R. Hoffmann, 1985. Composite Cost-based Rules for Priority Scheduling in a Randomly Routed Job Shop. *International Journal of Production Research*, Vol.23, No.6, pp.1185-1195.

Shanker, K. and A. Srinivasulu, 1989, Some Solution Methodologies for Loading Problems in a Flexible Manufacturing System. *International Journal of Production Research*, Vol.27, No.6, pp.1019-1034.

Shanker, K. and Y-J. J. Tzen, 1985. A Loading and Dispatching Problem in a Random Flexible Manufacturing System. *International Journal of Production Research*, Vol.23, No.3, pp.579-595.

Stecke, K. E., 1983. Formulation and Solution of Nonlinear Integer Production Planning Problems for Flexible Manufacturing Systems. *Management Science*, Vol.29, No.3, pp.273-288.

Stecke, K. E., 1985. Design, Planning, Scheduling, and Control Problems of Flexible Manufacturing Systems. *Annals of Operations Research*, Vol.3, pp.3-12.

Stecke, K. E., 1989. Algorithms for Efficient Planning and Operation of a Particular FMS. *International Journal of Flexible Manufacturing Systems*, Vol.1, No.4, pp.287-324.

Stecke, K. E. and I. Kim, 1986. A Flexible Approach to Implementing the Short-Term FMS Planning Function. *Proc. of the 2nd ORSA/TIMS Conf. on Flexible Manufacturing Systems, Ann Arbor, MI,* K.E. Stecke and R. Suri (editors), Elsevier Science Publishers B.V., Amsterdam, pp.283-295.

Stecke, K. E. and I. Kim, 1988. A Study of FMS Part Type Selection Approaches for Short-Term Production Planning. *International Journal of Flexible Manufacturing Systems*, Vol.1, No.1, pp.7-29.

Vepsalainen, A. P. J. and T. E. Morton, 1987. Priority Rules for Job Shops with Weighted Tardiness Costs. *Management Science*, Vol.33, No.8, pp.1035-1047.

Whitney, C. K. and T. S. Gaul, 1984. Sequential Decision Procedures for Batching and Balancing in Flexible Manufacturing Systems. *Proc. of the 1st ORSA/TIMS Special Interest Conf. on Flexible Manufacturing Systems, Ann Arbor, MI*, pp.243-248.

Whitney, C. K. and R. Suri, 1984. Decision Aids for FMS Part and Machine Selection. *Proc. of the 1st ORSA/TIMS Special Interest Conf. on Flexible Manufacturing Systems, Ann Arbor, MI*, pp.205-210.

Wilson, J. M., 1989, An Alternative Formulation of the Operation-Allocation Problem in Flexible Manufacturing Systems. *International Journal of Production Research*, Vol.27, No.8, pp.1405-1412.