A RESOURCE-CONSTRAINED JOB SHOP SCHEDULING PROBLEM WITH GENERAL PRECEDENCE CONSTRAINTS

Jaekyoung Ahn*

ABSTRACT

In this paper, a rule for dispatching operations, named the Most Dissimilar Resources (MDR) dispatching rule is presented. The MDR dispatching rule has been designed to maximize utilization of resources in a resource-constrainedd job shop with general precedence constraints. It shown that solving the above scheduling problem with the MDR dispatching rule is equivalent to multiple solving of the maximum clique problem. A graph theoretic approach is used to model the latter problem. The pairwise counting heuristic of computational time complexity $O(n^2)$ is developed to solve the maximum clique problem. An attempt is made to combine the MDR dispatching rule with the existing look-ahead dispatching rules. Computational experience indicates that the combined MDR dispatching rules provide solutions of better quality and consistency than the dispatching rules tested in a resource-constrained job shop.

1. Introduction

A traditional job shop consists of a set of machines and a collection of jobs that are to be allocated to machines over a scheduling horizon. The scheduling decisions are generally made under the following assumptions : 1. Each job involves several operations with a linear precedence structure (linear precedence constraints), 2. There is only one scarce resource type, i.e. machines (single resource constraints).

However, typical automated manufacturing system involves alternative routings scarce resources (eg., fixtures, pallets, grippers). The growing need for automated manufacturihng systems requires efficient scheduling schemes reflecting this manufacturing environment. As consequence, a scheduling system in a modern manufacturing should ont only accomodatde itself to the environmental changes, but also provide efficient solutions in a real time.

* Department of Industrial Engineering, Seoul National Polytechnic University

In the past decades, extensive studies have been done in the area of scheduling manufacturing systems. These studies can be divided into two basic categories [Panwalkar and Iskander 1977] : 1. Theoretical research dealing with optimization procedures 2. Experimental research dealing with dispatching rules.

The theoretical research has focused on the development of mathematical models and optimal or suboptimal algorithms (see, for example, Baker [1974], French [1982], Bellman, Esogbue and Nabeshima [1982]). The results of the theoretical approach have not been widely used in industry due to the high computational complexity of the scheduling problem. However, the theoretical approach has its own merit, mainly in capturing the problem structure. It allows an analyst to construct model according to the characteristic of the scheduling environment. The experimental research has been primarily concerned with dispatching rules and heuristic that efficiently solve the scheduling problem. This approach has appealed to both researchers and practitioners. To date, over one hundred dispatching rules have been develped. The research on the dispatching rules has been presented in a number of publications [Gere 1966, Panwalkar and Iskander 1977, Blackstone, Phillips and Hogg 1982, Alexander 1987, Koulamas and Smith 1988, Schultz 1989, and Kusiak 1990].

Panwalkar and Iskander [1977] classified the dispatching rules into three categories : 1. Simple dispatching rules, 2. Combination of simple dispatching rules, and 3. Weighted dispatching indexes. They also recommended that the usefulness of different dispatching rules depend on the specified performance criteria and it is essential to know the scheduling environment. These observations strongly suggest that dispatching rules should be devised to reflect the scheduling environment. In the same context, Koulamas and Smith [1988] presented a simple look-ahead dispatching rule for scheduling two or more parallel semiautomatic machines sharing the same server. Schultz [1989] presented a new dispatching rule that has proven to be effective over a wide range of due dates. Alexander [1987] developed an expert system for selection of dispatching rules for a job shop according to different performance criteria, and different operating conditions.

Although most of the papers published in the literature are rather restricted, attempts have been made to relax the linear precedence constraints and the single resource constraint [Wiest 1967, Fendley 1968, Pritsker et al. 1969, Davis 1973, Davis and Patterson 1975, Hogg, Phillips, and Maggard 1977, Lloyd 1981, Monma 1982, Patterson 1973, 1976, 1984]. Among these, the dual resource constrained (DRC) scheduling problem and the resource constrained project (RCP) scheduling problem need to be mentioned here. In the DRC shceduling problem, the availability of workers is a constraining factor as well as the availability of machines. There has been far too limited studies on the DRC problem to determine whether the more complex dispatching rules are useful for solving it [Blackstone, Phillips and Hogg 1982]. The RCP shceduling problem contains both types of constraints that characterize scheduling decisions [Baker 1974]. One of the major differences between the job shop and the RCP scheduling problems is the continuous work input and flow in the job shop [Davis 1973].

In this paper, a dispatching rule which is able to cope with a multi-resource manufacturing environment is developed. The need to develop such a rule stems from salient features of automated manufacturing systems. One of these features relates to multiple resources used, such as machines, fixtures, pallets, tools, etc. The proposed MDR (Most Dissimilar Resources) dispatching rule considers all these manufacturing resources. The shceduling problem in a resource constrained job shop with gengeal percedence constraints is described and analyzed in section 2. The MDR dispatching rule is defined in section 3. In section 4, the performance of combined MDR dispatching rules relative to the existing rules are evaluated for a static and dynamic case. The conclustions are presented in section 5.

2. THE PROBLEM STATEMENT

In order to formulate a static resource-constrainedd scheduling problem, the follwoing notation in used :

i : part index, $i=1, \cdots, n$

k : operation index, $k=1, \cdots, m$

n : number of parts

m : number of operations

r : resource index, $r=1, \cdots, r_c$

r_c : number of resources r of type c

c : resource type inex

c_r : r^{th} resource of type c

O_i : set of operations of part i

G_i : set of pairs of operations $[k,l]$ of part i, where k precedes l

Q_i : set of pairs of operations $[k,l]$ of part i, where k and l can be performed in any order

$\Pi_k=\{c^r\}$: set of resources used by operation k, where c^r is a resource r of type c

t_{ik} : processing time of operation k of part i

f_{ik} : completion time of operation k of part i

T : makespan

M : arbitrary large number

$$x_{kl} = \begin{cases} 1, & \text{if operation k precedes operation } l \\ 0, & \text{otherwise} \end{cases}$$

The objective of the static scheduling problem is to minimize the makespan.

(P) Min T (1)

subject to

$f_{il} - t_{il} \geq f_{ix}$	$[k,l] \in G_i$ for all i	(2)
$f_{il} - t_{il} \geq f_{ix} - Mx_{lk}$	$[k,l] \in Q_i$ for all i	(3)
$f_{il} - t_{il} \geq f_{jk} - Mx_{lk}$	$l \in O_i$, $k \in O_j$, for all i, j, $i \neq j$, and $\Pi_k \cap \Pi_l \neq \phi$	(4)
$x_{kl} + x_{lk} = 1$	for all k, 1 such that $\Pi_k \cap \Pi_l \neq \phi$, $k \neq l$, or $[k,l] \in Q_i$ for all i	(5)
$f_{ik} \leq T$	for all i, k	(6)
$f_{ik} \geq t_{ik}$	for all i, k	(7)
x_{kl} : 0, 1	for all k, l such that $\Pi_k \cap \Pi_l \neq \phi$ $k \neq l$, or $[k,l] \in Q_i$ for all i	(8)

Constraint (2) ensures that the operations belonging to a part are processed according to the required precedences. Constraint (3) implies that any two operations of the same part cannot be processed at the same tiem. Constraint (4) implies that a resource can be used by only one operation at a time. Constraints (5) imposes a precedence between a pair of operations. Constraint (6) implies that the completion time of each operation is not greater than the makespan. Constraint (7) implies that the completion time of each operation is not less than its processing time. Constraint (8) imposes integrality.

Problem (P) is a variant of the problem of schedduling n operations on m machines discussed in Kusiak [1990, pp.367~381]. It considers simultaeous use of multiple resources for processing each operation.

Lenstra[1977] classified machine scheduling problems using four parameters, such as number of operations, number of jobs, machine environment, and optimality criteria. Graham et al. [1979] introduced three classification schemes for resource constrained scheduling problems in which a problem has a three-field notation that represents machine environment, job characteristics, and optimization criteria. Blazewicz, Lenstra, and Rinnooy Kan[1983] considered an extension of this classification by allowing for the presence of more than one scarce resource.

To clearly envision the computational complexity of the resource constrained scheduling problem with general precedence constraints, it may be useful to represent it using the classification scheme introduced by Blazewicz, Lenstra, and Rinnooy Kan[1983]. The machine environment of the problem is α=J, i.e., the problem is a general job shop. Job characteristics, $\{\beta_1, \beta_2, \beta_3, \beta_4\}$ are as follows :

$\beta_1=\phi$: no preemption is allowed,

β_2=res$\lambda\sigma\rho$ where $\lambda=\cdot$, i.e., the number of resources is a component of the input,

$\sigma=1$, i.e., all resource sizes are equal to 1,

$\rho=1$, i.e., all resource requirements are equal to 1,

β_3=prec : each operation in a part has general precedence constraints (This notation differs from the notation in Blazewicz et al.[1983], where they referred to 'prec' as a relation between jobs rather than operations),

$\beta_4=\phi$: the processing times are arbitrary nonegative integers.

The objective function $\gamma=C_{max}$, is the minimization of the makespan. Thus, the static scheduling problem considered in this paper can be classified as J|res·11, prec|C_{max} in the multi-operation model. A simplified job shop shop version of problem (P), J|res111, $P_{ij}=1$|C_{max}, is proven to be NP-hardd in a strong sense [Blazewicz, Lenstra, and Rinnooy Kan 1983]. Therefore, it is not likely that there exists a polynomial time algorithm for sloving problem(P).

Extensive research focusing on the resource-constrained multiproject (RCMP) scheduling and job shop scheduling problem with multiple resources, has been done over many years [e.g., Wiest 1967, Fendley 1968, Pritsker et al. 1969, Davis 1973, Davis and Patterson 1975, Lloyd 1981, Monma 1982, Patterson 1973, 1976, 1984]. Patterson [1973, 1976] discussed the effectiveness of many of the heuristic extensions to the critical path method which resolve the resource conflicts for a large RCMP scheduling problem. He tested the performance of a number of dispatching rules proposed in the literature, and examined the RCMP problem in an attempt to assess the quality of solutions generated with heuristic methods. The RCMP scheduling problem shares the following parameters with the problem presented in this paper : 1. Resources of different types are considered. 2. Each job (project) consists of a number of operations (takes) with general precedence constraints.

As mentioned in Section 1, the job shop scheduling environment is of dynamic nature. The most promising dispatching rules for solving the RCMP scheduling problem which are based on the critical path method might require a considerable computation time in the job shop environment because of the necessity of periodic recalculation of the critical path and slack data [Wiest 1967]. Moreover, any two operations in a part cannot be processed at the same time in the manufacturing system due to physical constraints, while any two tasks in a project can be processed simultaneously if some resource constraints are met. The physical constraints need some attention in the following aspect. The delay of an activity in a critical path causes an equal delay in the completion of the project (part). In this sense, all the operations in the manufacturing system become critical activities because any delay in completion of an operation results in an equal delay of the job completion. Thus, the physical constraints in the job shop environment may cause dispatching rules based on the critical path method to loose its benefits when applied to manufacturing systems.

The common ground on which the two problems are based, however, indicates that it is beneficial to consider the most promising RCMP dispatching rules for solving the problem considered in this paper. To solve problem (P), a new dispatching rule (the MDR rule) is proposed, and is combined with the existing dispatching rules. The rule allows for dynamic solving of the scheduling problem.

3. THE MDR DISPATCHING RULE

The Most Dissimilar Resources (MDR) dispatching rule has been developed for efficient scheduling of operations in a resource constrained job shop environment (eg., an automated manufacturing system) where the maximization of the utilization rate of manufacturing resources is a major concern because of the following : 1. Considerable capital investment needed to install the system. 2. Production is lost when the manufacturing resources are idle. 3. Scheduling with minimum makespan and minimum tardiness criteria tends implicitly to maximize resource utilization over the (unspecified) scheduling time horizon [Rodammer and White 1988]. The basic idea behind the MDR priority rule is presented next.

3.1 Graph-Theoretical Formulation

1) Operation-resource incidence matrix

Consider a manufacturing system with 3 parts and 7 operations to be machined. The resources required by each operation are given in Table 1.

Table 1. Resource required for manufacturing three parts

Part No.	Operation	Machine	Tool	Fixture
1	o_1 o_2	m_1 m_2	t_1 t_2	f_1 f_2
2	o_3 o_4 o_5	m_3 m_2 m_3	t_3 t_3 t_2	f_3 f_1 f_3
3	o_6 o_7	m_1 m_2	t_2 t_1	f_2 f_3

The data in Table 1 are presented in the form of incidence matrix (9) which is more concenient for further considerations.

$$[m_{kr}] = \begin{array}{ll} & \\ \text{Part 1} & o_1 \\ & o_2 \\ & o_3 \\ \text{Part 2} & o_4 \\ & o_5 \\ \text{Part 3} & o_6 \\ & o_7 \end{array} \overset{\begin{array}{c}\text{Resource} \\ \begin{array}{cccccccccc} & \text{Machine} & & & \text{Tool} & & & \text{Fixture} & & \\ m_1 & m_2 & m_3 & t_1 & t_2 & t_3 & f_1 & f_2 & f_3 & f_4 \end{array}\end{array}}{\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}} \qquad (9)$$

In matrix(9), each entry

$$m_{kr} = \begin{cases} 1, \text{ if operation } k \text{ uses resource } r \\ 0, \text{ otherwise} \end{cases}$$

For any two operations k and j, define a distance measure, d_{kj} :

$$d_{kj} = \sum_{q=1}^{r} d(m_{kq}, m_{jq})$$

$$\text{where } d(m_{kq}, m_{jq}) = \begin{cases} 1, \text{ if } m_{kq}\, m_{jq} = 1,\ k \neq j, \text{ for } k \in Q_i,\ j \in Q_{i'},\ i \neq i' \\ N, \text{ if } k = j \text{ or } (k, j) \in Q_i \\ 0, \text{ otherwise} \end{cases} \qquad (10)$$

In the expression (10), N is any positive number. For the further use in this study, a schedulable operation is defined. An operation o_k is schedulable at time t, if [Kusiak 1990] :

1. No other operation that belongs to the same part is being processed at time t
2. All operations preceding operation ok have been completed before time t
3. All resources required by the process plan to perform operation ok are available at time t.

Assume that the seven operations in matrix (9) are schedulable. The MDR dispatching rule selects from the matrix (9) the largest set of schedulable operations which satisfies the following conditions :

1. Each operation belongs to a distinct part,
   i.e., more than one operation in a part cannot be processed at the same time.

2. All the operations in the set can be processed at the same time, i.e., there is no resource conflict.

2) Operation-resource graph

The problem of selecting operations according to the MDR priority rule can also be represented with an operation-resource graph, G.

Let G(V,E,y) be a graph, where :

V : a set of schedulable operations

E : a set of incidence relations among the schedulable operations

Ψ : $\Psi(e)=v_k v_j$, if $\sum_{q=1}^{z} m_{kq} m_{jq}=0$, and v_k and v_j do not belong to the same part.

A 3-partite operation-resource graph corresponding to matrix (9) is shown in

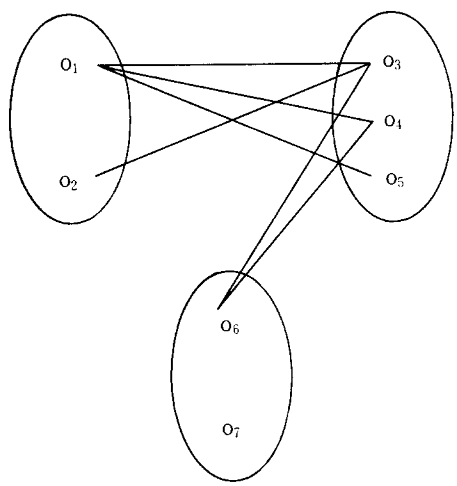

Figure 1. A 3-partite operation-resource graph

For n parts, the operation-resource graph becomes an n-partite graph and is not necessarily complete. In the operation-resource graph, each edge indicates the nodes (operations) that can be processed simultaneously, i.e., the nodes that do not share the same resources. Note that the MDR dispatching rule selects the largest set of nodes such that each node in the set belongs to a distinct part and there is no resource conflict between the nodes in the set.

A clique of a simple graph, G, is defined as a subset S of V such that G[S] is complete [Bondy and Murty 1976]. Thus, a clique in the operation-resource graph satisfies the conditions under which the MDR dispatching rule selects the schedulable operations from the operation-resource incidence matrix, and the maximum clique in the graph corresponds to the largest set of schedulable operations from the incidence matrix. Therefore, selecting operations with the MDR priority rule is equivalent to the problem of finding the maximum clique in the operation-resource graph, G. The preceding concepts are illustrated in Example 1.

Example 1

Given the operation-resource incidence matrix (11).

$$
\begin{array}{ccc}
 & & \text{Resource} \\
 & & \begin{array}{cccccccccc} & \text{Machine} & & & \text{Tool} & & & \text{Fixture} & & \\ m_1 & m_2 & m_3 & t_1 & t_2 & t_3 & f_1 & f_2 & f_3 & f_4 \end{array} \\
[m_{kr}] = & \begin{array}{cc} \text{Part 1} & o_1 \\ \text{Part 2} & o_2 \\ \text{Part 3} & o_3 \\ \text{Part 4} & o_4 \\ \text{Part 5} & o_5 \\ \text{Part 6} & o_6 \end{array} & \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}
\end{array} \qquad (11)
$$

For simplicity, assume that each schedulable operation belongs to a distinct part. The operation-resource graph, G[V,E,Ψ], corresponding to matrix (11) is shown in Figure 2.

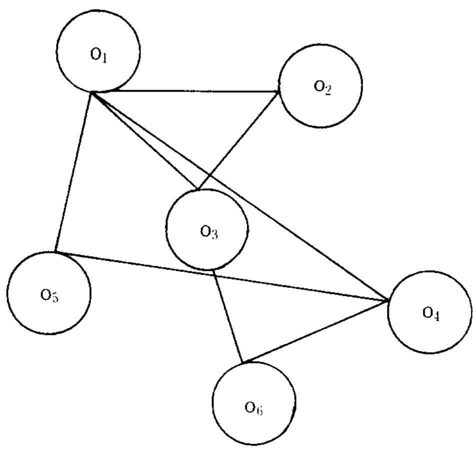

Figure 2. Operation-resource graph of the incience matrix(11)

Two maximum cliques are visible in Figure 2, namely, $\{o_1, o_2, o_3\}$ and $\{o_1, o_4, o_5\}$. The two sets represent the largest set of schedulable operations that can be processed at the same time. The problem of finding a maximum clique in a graph is NP-complete [Garey and Johnson 1979].

3.2 The Pairwise Counting Heuristic

Since the MDR priority rule is to be used in real time scheduling, it is important that the maximum clique problem is solved almost instantaneously. A general integer programming code, such as LINDO, is not likely to meet the latter requirement for the problem sizes encountered in industry. A more efficient way to solve the maximum clique problem is to use a specialized algorithm.

To date, a number of exact and heuristic algorithms [eg., Horowitz and Sahni 1978, Papadimitriou and Steiglitz 1982, Balas and Yu 1986] have been developed for solving the maximum clique problem. A simple pairwise counting heuristic of computational time complexity $O(n^2)$ is presented below.

Before the algorithm will be presented, the following notation is introduced :

P : a set of operations to be dispatched

$D_1(k)$: distance index of operation k, defined as the number of 0's in the k^{th} column in the distance matrix

$D_2(k)$: distance index of operation k, defined as the number of parts in which the number of 0's in the k^{th} column in the distance matrix is greater than 1.

The algorithm constructs a set of operations to be dispatched, P, from a set of schedulable operations, S.

Algorithm

Step 0. Initialize the set of schedulable operations, S, and the set of operations to be dispatched ($P=\phi$).

Step 1. If $S \neq \phi$, then go to Step 2, otherwise, STOP.

Step 2. From the set of schedulable operations, S, construct the operation-resource incidence matrix $[m_{kr}]$.

Step 3. From the operation-resource incidence matrix $[m_{kr}]$, construct the distance matrix $[d_{kj}]$.

Step 4. For each operation $k \in S$, compute the distance index D(k) from the operation-resource incidence matrix $[m_{kr}]$.

Step 5. Find maximum $D(k^*)$.
If a tie occurs, break it, arbitrary.

Step 6. Move operation k^* to P.

Step 7. Update the set of schedulable operations :

a) Delete operation k^* from S.

b) Delete the operations which belong to the same part as the operation k^* from S.

c) Delete the operations which share the same resources with the operation k^* from S.

Step 8. Go to Step 1.

3.3 Two distance indexes

Using a graph theoretic approach, the two distance indexes mentioned can be interpreted as follows :

$D_1(k)$: the number of edges incident to node k in the operation-resource graph

$D_2(k)$: the number of partitioned subsets incident to node k in the n-partite operation-resource graph.

Papadimitriou and Steiglitz [1982, pp.406~407] used the distance index, $D_1(k)$ for solving the minimum node covering problem. The distance index, $D_1(k)$, used in the pairwise counting heuristic has some disadvantages, namely, for an operation k, its value increases by more than one in the following cases :

1. Operation j, and l belong to the same part.
2. Operation k belongs to a different part than the part that includes operations j and l.
3. $d_{kj} = 0$, and $d_{kl} = 0$.

Note that operations k, j, and l cannot be processed at the same time because of the case 1. This is due to the fact that more than one operations in a part cannot be processed at the same time (physical constraints). Therefore, the distance index, $D_1(k)$, for the operation k should be increased by only one under the above conditions.

Based on the graphic representation of the MDR dispatching rule, the advantage of using $D_2(k)$ over $D_2(k)$ is explained next. The MDR dispatching rule attempts to find the maximum clique in the n-partite operation resource graph whenever a dispatching decision is required. It is obvious that any two operations in a part cannot belong to the same clique, so that the maximum number of operations in a part for a clique is 1. Thus, the distance index, $D_2(k)$, takes advantage of the nature of the n-partite operation resource graph.

The distance indexes, $D_1(k)$ and $D_2(k)$, are illustrated in Example 2.

Example 2

Consider the resource-operation matrix (12).

Resource

$[m_{kr}] =$ (12)

		Machine				Tool				Fixture			
		m_1	m_2	m_3	m_4	t_1	t_2	t_3	t_4	f_1	f_2	f_3	f_4
Part 1	o_1	1	0	0	0	0	1	0	0	0	0	1	0
	o_2	0	0	0	1	0	0	0	1	0	0	0	1
	o_3	0	1	0	0	1	0	0	0	0	1	0	0
Part 2	o_4	1	0	0	0	0	0	1	0	0	1	0	0
	o_5	0	0	1	0	0	1	0	0	1	0	0	0
Part 3	o_6	0	0	1	0	0	0	1	0	0	0	0	1

For the mitrix (12), the distance matrix (13) is computed.

$$
\begin{array}{lc|cccccc|}
 & & \multicolumn{6}{c}{\text{Operation}} \\
 & & o_1 & o_2 & o_3 & o_4 & o_5 & o_6 \\
\text{Part 1} & o_1 & M & M & 0 & 1 & 1 & 0 \\
 & o_2 & M & M & 0 & 0 & 0 & 1 \\
 & o_3 & 0 & 0 & M & M & M & 0 \\
\text{Part 2} & o_4 & 1 & 0 & M & M & M & 1 \\
 & o_5 & 1 & 0 & M & M & M & 1 \\
\text{Part 3} & o_6 & 0 & 1 & 0 & 1 & 1 & M
\end{array}
\qquad (13)
$$

The distance index, $D_1(k)$, has the following values :

$D_1(1)=2$, $D_1(2)=3$,

$D_1(3)=3$, $D_1(4)=1$,

$D_1(5)=2$, $D_1(6)=2$.

The maximum value of the distance index, $D_k(k)$ is attained for operations 2 and 3.

If, in step 5 of the pairwise counting algorithm, operation 2 is selected, then the resulting set of operations to be dispatched is $P_1=\{2, 3\}$, $P_2=\{2, 4\}$, or $P_3=\{2, 5\}$. This means that the algorithm finds at most two operations which can be processed at the same time. In the case when operation 3 is selected in step 5 of the algorithm, then the resulting set of operations to be dispatched becomes $P=\{1, 3, 6\}$. This results from the overestimation of $D_2(2)$, i.e., the value of $D_1(2)$ increases by three even though operations 3, 4, and 5 belong to the same part. The use of the distance index, $D_2(k)$ leads to the set of operations to be dispatched in the final solution, $P=\{1,3,6\}$. In the subsequent sections, the distance index, $D_2(k)$ will be used in the pairwise counting algorithm.

3.4 The Combined MDR Rule

In the real-time scheduling environment, schedules are usually determined by dispatching an operation whenever a decision is required. For instance, the MDR rule dispatches an operation which could be processed with as many as possible other operations at the same time while COVERT rule dispatches an operation which has a potential of being late. In a sense, dispatching an operation affects not only the schedulable operations that can be processed simultaneously at a specified time, but also the entire set of operations in the subsequent scheduling horizon. Depending on the information that is used, the dispatching rules can be categorized as following :

1. Myopic dispatching rules dispatch an operation with the local information whenever a decision is made. They include MDR, SPT, LPT, Greatest Total Work, FCFS, and so on.
2. Look-ahead dispatching rules dispatch an operation with the look-ahead information whenever a decision is made. They include MWR, EDD, Minslack, COVERT, and so on. Most of the due date related dispatching rules fall into this category.

It is likely that the most ideal dispatching rule for the problem considered in this paper should use the myopic resource information as well as the look-ahead information. Panwalker and Iskander [1977] indicated that a combination of simple dispatching rules, or a combination of heuristics works better than individual priority rules. However, the combination procedure for dispatching rules is not an easy task. The most frequently used combination methods are based on assigning weights to dispatching rules that are combined. Finding of proper weights (the normalization procedure) requires extensive simulation tests and is difficult to justify. Kusiak [1990] combined seven dispatching rules in a serial way. Since each rule may not provide a tie among operations, the first dispatching rule used predominantly governs the search process.

In this paper, the MDR dispatching rule (myopic) is combined with promising look-ahead dispatching rules. Since the MDR dispatching rule quite frequently provides a tie among operations, the myopic information as well as look-ahead information are combined.

4. COMPUTATIONAL EXPERIENCE

This section involves testing various dispatching rules that determine which operations should have preference whenever there is a potential conflict of resources in the presence of precedence and physical constraints.

4.1 Dispatching Rules

The following notation is used in this section :

t : current time

U_{ik} : set of operations succeeding operation k in G_i

G_i : set of pairs of operations $[k,l]$ of part i, where k precedes l

V_{ik} : set of unprocessed operations of part i except operation k

d_i : due date of part i

sl_k : slack of operation k,

$$sl_k = d_i - t - \sum_{l \in V_{ik}} t_{il}$$

$c_k(\rho_1, \rho_2)$: expected delay penalty for operation k,

$$c_k(\rho_1, \rho_2) = \begin{cases} 1, \text{ if } sl_k < 0 \\ 0, \text{ if } sl_k \geq \sum_{l \in V_{ik}} t_{il} \\ \dfrac{\sum_{l \in V_{ik}} \rho_1 \cdot \rho_2 t_{il} - sl_k}{\sum_{l \in V_{ik}} \rho_1 \cdot \rho_2 t_{il}}, \text{ otherwise} \end{cases}$$

In order to evaluate the equality of schedules, the dispatching rules shown in Table 2 and 3 have been tested.

Table 2. Non-due date related dispatching rules and priority indexes for operation k of part i

Rule number	Rule name	Max /Min	Priority index
R_1	Most Dissimilar Resource	Max	D(k)
R_2	Most Subsequent Work Remaining	Max	$\sum_{l \in U_{ik}} t_{il}$
R_3	Most Subsequent Operations Remaining	Max	$\lvert U_{ik} \rvert$
R_4	Most Work Remaining	Max	$\sum_{l \in V_{ik}} t_{il}$
R_5	Most Operation Remaining	Max	$\lvert V_{ik} \rvert$
R_6	Shortest Processing Time	Min	t_{ik}
R_7	Longest Processing Time	Max	t_{ik}
R_8	Lease Work Remaining	Min	$\sum_{l \in V_{ik}} t_{il}$

Table 3. Due date related dispatching rules and priority indexes for operation k of part i

Rule number	Rule name	Max /Min	Priority index
R_9	Minimum Slack Remaining	Min	sl_k
R_{10}	Remaining Allowance per Operation	Min	$\frac{d_i - t}{\lvert V_{ik} \rvert}$
R_{11}	Slack per Operation	Min	$\frac{d_i - t - t_{ik}}{\lvert V_{ik} \rvert}$
R_{12}	Slack per Remaining Work	Min	$\frac{d_i - t - t_{ik}}{\sum_{l \in V_{ik}} t_{il}}$
R_{13}	COVER 1	Max	$\frac{c_k(2,2)}{t_{ik}}$
R_{14}	COVER 2	MAX	$\frac{c_k(3,2)}{t_{ik}}$

The dispatching rules R_2 and R_3 are equivalent to R_1 and R_5, respectively, under the linear precedence assumption. The dispatching rules R_4 and R_5 are related to the "look-ahead" rules. The SPT dispatching rule, R_6, is one of the most often studied and performs best relative to the mean flowtime criterion. The dispatching rule, R_8, tends to minimize the cycle time of each part.

Among non-due date related dispatching rules in Table 2, there is only one rule (the MDR rule) that takes into account resource utilization. Patterson [1973] introduced four resource related dispatching rules in the RCP scheduling problem. These are : the GTRD (Greatest Total Resource Demand), GRRD (Greatest Remaining Resource Demand), GRU (Greatest Resource Utilization), and MJP (Most Jobs Possible) dispatching rules. Normally, an operation in a job shop environment requires one unit of resource of each resource type (eg., operation 1 is processed on machine 2, clamped in fixture 5, placed on pallet 3). Furthermore, resources in a resource type are not interchangeable (i.e., a drilling operation can be processed only on a drilling machine, not on other machine). Therefore four dispatching rules introduced by Patterson [1973] need some modifications in order to apply them for scheduling of manufacturing systems.

The GTRD dispatching rule selects an operation with the greatest resource usage. For the problem considered in this paper, it becomes a random rule since it is assumed that the number of resource types required by each operation is identical. The GRRD dispatching rule becomes the Most Work Remaining dispatching rule, R4, since the rule selects an operation with the greatest remaining resources required. The GRU dispatching rule selects a set of operations which maximize utilization of available resources, solved as a zero-one integer programming problem. The MJP dispatching rule selects a set of operations which maximize the number of parts being worked on, solved as a zero-one integer programming problem as well (Patterson [1976] showed that the MJP and the GRU dispatching rules perform good for the measures of total project delays and total weighted project delays). The two above dispatching rules can be easily transformed into the maximum clique problem in a resource-constrained job shop (the definition of common units for different types of resources are necessary).

The Minimum Slack Remaining dispatching rule, R_9, utilizes the processing time information but in a way that it actually counteracts benefits from the SPT rule. One of the factors in measuring the urgency of a part is the number of operations remaining for the part. When two parts have the same remaining work, the part with the largest number of operations is intuitively more urgent. This reasoning has led to the dispatching rules, Remaining Allowances per Operation (R_{10}) and Slack per Operation (R_{11}). The Slack per Remaining Work rule aims at taking advantage of the SPT strategy when two parts have the same minimum slack remaining. The most promising dispatching rule when due dates are imposed is the COVERT rule proposed by Carroll [1965]. In this paper, two COVERT dispatching rules with different parameters are tested. The slack parameters, ρ_1 and ρ_2, are fixed throughout the computational experiment. For detailed description of the due date related dispatching rules, see Baker [1974] and Vepsalainen and Morton [1987]. The combinations of the MDR rule and other dispatching rules are listed as $R_{m,i}$ (MDR+R_i).

4.2 Shop Conditions

To test the performance of each of the dispatching rules in the static job shop, 9 different scheduling problems are considered as shown in Table 4.

Table 4. Characteristics of the static problems

Problem No.	Problem Size	No. of Parts	No. of Op'ns	Number of Resources				No. of Runs
				Machines	Fixtures	Pallets	Tools	
1	Small	5	30	3	3	5	5	10
2				3	3	20	15	10
3				3	15	20	15	10
4	Medium	8	80	5	5	8	8	10
5				5	5	60	20	10
6				5	16	60	20	10
7	Large	30	300	16	20	30	20	10
8				16	20	200	70	10
9				16	50	200	70	10

For each problem, 10 instances were solved. The routings are those of a job shop with general precedence constraints. The processing times were generated according to the discrete uniform distribution with parameters [1,10]. Due date for each part is twice the processing time. This follows from the fact that the TWK based due date assignment is a reliable and effective method for setting due dates [Baker 1984]. Static problems are divided into the following categories :

Ⅰ : All the resource types are scarce (problem No. 1, 4, and 7)

Ⅱ : Two of the resource types are scarce (problem No. 2, 5, and 8)

Ⅲ : One of the resource type is scarce (problem No. 3, 6,and 9)

In the dynamic job shop, the number of jobs arriving in an interval of given length (60 unit time) is Poisson-distributed. The arrival rate (λ) of this process is given as follows :

$$\lambda = \frac{\text{No. of machines(bottleneck resources) x Estimated system utilization rate}}{\text{Avg. No. of operations per part x Avg. processing time of an operation}}$$

Nine different problems shown in Table 5 are considered. For each problem, 10 instances were solved, where scheduling was observed until all 100 parts had been completed.

Table 5. Characteristics of the dynamic problems

Load Problem	Light-Load(LL) u=0.6	Normal-Load(NL) u=0.8	Heavy-Load(HL) u=0.9
1	[3,7]/[10,100,15,40]/0.12*	[3,7]/[10,100,15,40]/0.16	[3,7]/[10,100,15,40]/0.18
2	[8,12]/[15,150,23,60]/0.09	[8,12]/[15,150,23,60]/0.12	[8,12]/[15,150,23,60]/0.135
3	[18,22]/[40,400,60,120]/0.12	[18,22]/[40,400,60,120]/0.16	[18,22]/40,400,60,120]/0.18

* $\alpha/\beta/\gamma$

α : parameter of the uniform distribution that represents the number of operations per part

β : number of resources available ([machines, tools, fixtures, pallets])

γ : arrival rate of the Poisson distribution on

The processing times in the dynamic job shop problem were generated with the uniform distribution [5,15]. Due dates were assigned randomly with an average of 3 times the processing time of each part.

Vepsalainen and Morton [1987] simulated a job shop with a fixed number of machine, and suggested normalization procedures of the solutions generated for a set of scheduling problems with different number of parts, average number of operations per part, average processing time of an operation and weighted cost. Besides these factors, number of bottleneck resources of different types in a job shop can be considered. In our computational experiments, average processing time of an operation (in a static and a dynamic job shop, respectively) and weighted cost factors were set to be the same, and machines were assumed to be the bottleneck resource.

Two normalized performance measures for parts and five measures for schedules (runs) are considered :

· Normalized Average Waiting (AW) time

$$AW = \frac{r_{machine}}{n' \cdot m_{avg}} \sum_{i=1}^{R} \frac{AW_i}{R} \tag{14}$$

where : $r_{machine}$ is the number of machines

m_{age} is the average number of operations in a part $n' = \frac{n}{10}$

$$AW_j = \sum_{i=1}^{n} \frac{1}{n} \left(\frac{\text{Sum of Processing Time}}{\text{Work-In-System Time}} \right)_i$$

R is the number of runs

· Normalized Average Flow(AF) time

$$AF = \frac{r_{machine}}{n' \cdot m_{avg}} \sum_{j=1}^{R} \frac{AF_j}{R} \tag{15}$$

where : $AF_j = \sum_{i=1}^{n} \frac{1}{n} (\text{Flowtime})_i$

· Normalized Average Markespan(AM)

$$AM = \frac{r_{machine}}{n' \cdot m_{avg}} \sum_{j=1}^{R} \frac{T_j}{R} \tag{16}$$

where : T_j is the makespan in the j^{th} run

· Normalized Average Percent Tardiness(PT)

$$PT = \frac{r_{machine}}{n' \cdot m_{avg}} \sum_{j=1}^{R} \frac{PT_j}{R} \tag{17}$$

where : PT_j is the perentage tardy parts in the j^{th} run

· Normalized Average Maximum Tardiness(MT)

$$MT = \frac{r_{machine}}{n' \cdot m_{avg}} \sum_{j=1}^{R} \frac{MT_j}{R} \tag{18}$$

where : MT_j is the maximum tardiness in the j^{th} run

· Normalized Average Tarddiness(AT)

$$AT = \frac{r_{machine}}{n' \cdot m_{avg}} \sum_{j=1}^{R} \frac{AT_j}{R} \tag{19}$$

where : AT_j is the average tardiness in the j^{th} run

· Normalized Conditional Average Tardiness(CAT)

$$CAT = \frac{r_{machine}}{n' \cdot m_{avg}} \sum_{j=1}^{R} \frac{CAT_j}{R} \tag{20}$$

where : CAT_j is the conditional average tardiness in the j^{th} run

All the performance measures listed above((14)-(20)) have been used for the static job shop, and the performance measures (14) and (17)-(20), for the dynamic job shop. In the static job shop, as the number of scarce resource types increases, performance of the combined MDR dispatching rule improves. In the dynamic job shop, the combined MDR dispatching rules produce solutions of better quality for the light and heavy shop load. In most of cases solved, the combined MDR dispatching rule outperforms the corresponding single dispatching rule (191 out of 228 cases). Table 6 and 7 summarizes the solutions obtained by the combined MDR dispatching rules for the performance measures considered in each problem category.

Table 6. Number of superior solutions generated by the combined MDR rules for non due date related performance measures

	Static Shop			Dynamic Shop
	AW(14)	AF(15)	AM(16)	AW(14)
$R_{m,2}$	3/3*	3/3	2/3	3/3
$R_{m,3}$	3/3	3/3	2/3	3/3
$R_{m,4}$	2/3	3/3	3/3	3/3
$R_{m,5}$	3/3	3/3	3/3	3/3
$R_{m,6}$	3/3	2/3	3/3	3/3
$R_{m,7}$	2/3	2/3	2/3	3/3
$R_{m,8}$	0/3	3/3	3/3	1/3

* Number of superior solutions generated by the combinedd MDR rule / Total number of solutions

Table 7. Number of superior solutions generated by the combined MDR rules for due datdd related performance measures

	Static Sop				Dynamic Shop			
	PT(17)	MT(18)	AT(19)	CAT(20)	PT(17)	MT(18)	AT(19)	CAT(20)
$R_{m,9}$	2/3	1/3	2/3	2/3	3/3	2/3	3/3	2/3
$R_{m,10}$	2/3	2/3	2/3+	2/3	3/3	3/3	3/3	3/3
$R_{m,11}$	3/3	3/3	3/3	2/3	3/3	3/3	3/3	3/3
$R_{m,12}$	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3
$R_{m,13}$	3/3	2/3	2/3	2/3	2/3	2/3	2/3	2/3
$R_{m,14}$	3/3	2/3	3/3	2/3	1/3	1/2	3/3	1/3

+ Represents a tie

5. CONCLUSIONS

In the paper, the MDR (Most Dissimilar Resources) dispatching rule aimed for the maximum utilization of resources in a resource-constrained job shop with general precedence constraints was developed. The rule considers the usage of multiple manufacturing resources, i.e., machines, tools, and fixtures. Other resources can be easily incorporated. The problem of scheduling operations with the MDR rule was found to be equivalent to a series of maximum clique problems. A graph theoretic formulation was developed for the maximum clique problem. Due to the NP completeness of the problem, a pairwise counting heuristic was developed.

The testing effort was focused on solving a set of 900 static scheduling problems and 90 dynamic problems obtained for randomly generated data sets. The combined MDR dispatching rules provide solutions of better quality and consistency than any other single dispatching rule in a resource-constrained job shop.

REFERENCES

Alexander S. M. (1987) "An Expert System for the Selection of Scheduling Rules in a Job Shop", *Computers and industrial Engineering*, Vol. 12, No. 3, pp.167~171.

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

Baker, K.R. (1984) "Sequencing rules and due-date assignment in a job shop", *Management Science*, Vol. 30, No. 9, pp.1093~1104.

Balas, E. and Yu, C.S. (1986) "Finding a maximum clique in an arbitrary graph", *SIAM Journal on Computing*, Vol. 15, No. 4, pp.1054~1068.

Bellman, R. E., Esogbue, A. O. and Nabeshima, I. (1982) *Mathematical Aspects of Scheduling and Applications*, Pergamon Press, Oxford, U. K.

Blackstone, J. H. Jr., Phillips, D. T. and Hogg, G. L. (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.

Blazewicz, J., Lenstra, J.K., and Rinnooy Kan, A.H.G. (1983) "Scheduling subject to resource constraints : Classification and complexity", *Discrete Applied Mathematics*, Vol. 5, pp.11~24.

Bondy, J. A. and Murty, U. S. R. (1976) *Graph Theory with Applications*, Elservier, New York, N. Y.

Davis, E.W. (1973) "Project scheduling under resource constraints-Historical review and Categorization of procedures", *AIIE Transactions*, December, pp.297~313.

Davis, E.W. and Patterson, J.H. (1975) "A comparison of heuristic and optimum solutions in resource-constrained project scheduling", *Management Science*, Vol. 21, No. 8, pp.944~955.

Fendley, L.G. (1968) "Toward the development of a complete multiproject scheduling system", *The Journal of Industrial Engineering*, Vol. XIX, No. 10, pp.505~515.

French, S. (1982) *Sequencing and Scheduling : An Introduction to the Mathematics of the Job-Shop*, John Wiley, New York, N. Y.

Garey, M. R. and Johnson, D. S. (1979) *Computers and Intractability : A Guide to the Theory of NP-Completeness*, Bell Laboratories, Murray Hill, N. J.

Gere, W. S. Jr. (1966) "Heuristics in Job shop Scheduling", *Management Science*, Vol. 13, No. 3, pp.167~190.

Graham, R.L., Lawler, E.L., Lenstra, J.K., and Rinnooy Kan, A.H.G. (1979) "Optimization and approximation in deterministic sequencing and scheduling : A survey", *Annals of Discrete Mathematics*, Vol. 5, pp.287~326.

Hogg, G.L., Phillips, D.T., and Maggard, M.J. (1977) "Parallel-Channel, dual-resource constrained queueing systems with heterogeneous resources", *AIIE Transactions*, Vol. 9, No. 4, pp. 352~362.

Horowitz, E. and Sahni, S. (1978) *Fundamental of Computer Algorithms*, Computer Science Press, Potomac, MD.

Koulamas, C. P. and Smith, M. L. (1988) "Look-ahead scheduling for minimizing machine interference", *International Journal of Production Research*, Vol. 26, No. 9, pp.1523~1533.

Kusiak, A (1990) *Intelligent Manufacturing Systems*, Prentice Hall, Englewood Cliffs, N. J.

Kusiak, A and Finke, G. (1988) "Selection of Process Plans in Automated Manufacturing Systems", *IEEE Journal of Robotics and Automation*, Vol. 4, No. 4, pp.397~402.

Lenstra, J.K. (1977) *Sequencing by Enumerative methods*, Mathematical Centre Tract 69, Mathematisch centrum, Amsterdam, The Netherlands.

Lloyd E.L. (1981) "Concurrent task systems", *Operations Research*, Vol. 29, No. 1, pp.189-201.

Monma, C.L. (1982) "Linear-time algorithm for scheduling on parallel processors", *Operations Research*, Vol. 30, No. 1, pp.116~124.

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

Papadimitriou C. H. and Steiglitz (1982) *Combinatorial Optimization : Algorithms and Complexity*, Prentice-Hall, Englewood Cliffs, N. J.

Patterson, J.H. (1973) "Alternative methods of project scheduling with limited resources", *Naval*

Research Logistics Quarterly, Vol. 20, No. 4, pp.767~784.

Patterson, J.H. (1976) "Project scheduling : The effects of problem structure on the heuristic performance", *Naval Research Logistics Quarterly*, Vol. 23, No. 1, pp.95~123.

Pritsker, A.A.B., Watters, L.J., and Wolfe, P.M. (1969) "Multiproject scheduling with limited resources : A zero-one programming approach", *Management Science*, Vol. 16, No. 1, pp.93~108.

Rodammer, F. A. and White, K. D. Jr. (1988) "A Recent Survey of Production Scheduling", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 18, No. 6, pp.841~851.

Schrage, L. (1984) *Linear, Integer, and Quadratic Programming with LINDO*, Scientific Press, Palo Alto, CA.

Schultz, C. R. (1989) "An expediting heuristic for the shortest processing time dispatching rule", *International Journal of Production Research*, Vol.27, No. 1, pp.51~41.

Vepsalainen, A.P.J. and Morton, T.E. (1987) "Priority rules for job shops with weighted tardiness costs", *Management Science*, Vol. 33, No. 8, pp.1035~1047.

Wiest, J.D. (1967) "A heuristic model for scheduling large projects with limited resources", *Management Science*, Vol. 13, No. 6, pp.B359~B377.