工業經營學會誌 第 20卷 第 43輯 1997年 8月 229

Implementation of Cell Scheduling Algorithm
-셀 스케줄링 알고리즘의 실행-

Leem, Choon Woo*
임춘우
Lee, Noh Sung**
이 노성

Abstract

셀 생산방식에서 생산시스템은 생산셀로 나누어 지기 때문에 셀 스케줄링 문제의 범위는 일반적인 스케줄링 문제에서 작은 셀들의 스케줄링 문제로 줄어든다. 각각의 생산셀의 경우에서는 부품과 머신들의 수가 일반적인 경우보다 훨씬 줄어든다.

본 논문은 생산셀에서의 작업순서를 결정하기 위해 exchange-heuristic 알고리즘을 사용한다. 셀 스케줄링 문제에 exchange-heuristic 알고리즘의 전과정들이 적용된다. 셀 exchange-heuristic에서의 성능은 두가지 경우, 즉, 두 기계 머신인 경우와 다 머신인 경우에 대하여 분석된다. 이때, 셀 exchange-heuristic 문제는 일괄처리 작업방식으로 하며, 성능지표는 생산기간의 단축으로 한다.

1. Introduction

Cell scheduling is a function of determining an optimal or near-optimal time schedule in which parts assigned to a cell should be processed to optimize measures of effectiveness. General scheduling problems are too complex and difficult to optimize. The cellular manufacturing approach based on manufacturing cells for scheduling simplifies the problems. that is, in cell scheduling, the scope of the problem is reduced from that of large portions of the shop to a small cell of machines. Due to the appropriateness of cell scheduling, the following advantages can be obtained.[1,2]

(1) Reduction of total throughput time, setup time and cost, and WIP inventory,
(2) Reduction of flow line,
(3) Optimization of cell layout, and
(4) Provision of overall economic benefits.

In most cases, the developed scheduling algorithms can be used in cell scheduling problems. In the cell scheduling problem, Gupta and Szwarc explained the fundamental assumptions as follows:[3,4]

(1) Parts to be processed are classified into several cells, and parts within the same cell are processed in succession,
(2) Cell processing time required for completion of a cell consists of cell setup time and the sum of processing times contained for parts in each cell,
(3) Cell setup time needed to process a cell is independent of the sequence of cells and

---

* LG생산기술원 책임연구원
** 특허청 심사관

(4) Job setup time needed to process a part is independent of the sequences of cells and parts, and it is included in the part processing time.

There are two basic models of optimum sequencing for cell scheduling:

(1) scheduling for a single part family and

(2) scheduling for a set of part families.

Whether the scheduling is for single or multiple families, the cell scheduling problem can be either unidirectional or nonunidirectional. Unidirectional means that the sequence of machines is completely identical for all jobs. If the technological sequence of machines for each job is mostly different, it is termed nonunidirectional. The most important requirement for cell scheduling is to have part families and machine cells to process the given parts. However, the machines/operations for these parts do not necessarily have to be grouped as machine cell. The two different cases for cell scheduling are functional layout and cellular layout. In cell scheduling, there two different methods of transferring the parts from one machine to the next: 1) by successive transfer(lot by lot) and 2)by overlap transfer (piece by piece).[1,2,5]

2. Flow Shop vs. Job Shop Cell Scheduling

Cell scheduling problems can be divided into the flow shop problem and the job shop problem according to the flow shop of work. If the flow of work is unidirectional, this shop is called a "flow shop". In situations where there is no common pattern for the flow of work through the shop, it is called a "job shop".

2.1 Cell scheduling for Flow Shop

Cell scheduling problem for flow shop is used to determine the sequence of jobs based on a specific criterion such as the minimization of makespan. Since Johnson's work on the two machines and n job flow shop scheduling problem, a number of the sequencing algorithms have been developed.

In general, flow shop scheduling problems including three machines or more are known as NP complete; no simple rule has been offered for determining the optimal schedule. Therefore, for flow shop scheduling problems of four or more machines, a general type of schedule that has different orders of jobs on each machine must be considered to find an optimal schedule in the true sense of the word.[1]

2.2 Cell Scheduling for Job Shop

Cell scheduling problem for job shop is concerned with the allocation of tasks to resources. The resources are called machines and the tasks are called operations subject to precedence constraints. The objective of the cell scheduling problem for job shop is to find the sequence and starting times of operations, while satisfying precedence constraints and optimizing the given criteria.

Cell scheduling in a job shop is completely different from that in a flow shop. In a job shop, the flow of work is nonunidirectional. Most of the developed cell scheduling algorithms are concerned with flow shop problems, since the job shop scheduling problem is much more complicated. Even despite the complexity of job shop scheduling, discrete manufacturing systems in mechanical industries have been converting to job shop to gain more flexibility in recent years.

In the designed manufacturing cell, there may be parts with an omission or absence of some machines in cells. For example, there are four machines in shop and six parts processed. From the process planning module, the operation sequence and the processing times for machines are determined. Then, the manufacturing routing information can be represented as a machine-part incidence matrix as shown in Table 1. In addition, the processing time for each machine is obtained in Table 2.

Table 1 Initial Machine-part Incidence Matrix

Machine	Part					
	P1	P2	P3	P4	P5	P6
M1	0	1(1)*	0	1(2)	1(2)	0
M2	1	0	1(1)	0	0	1(2)
M3	0	1(2)	0	1(1)	1(1)	0
M4	0	0	1(2)	0	0	1(1)

* The number in parentheses denotes the sequence of the operation

Table 2 Processing Time

Machine	Part					
	P1	P2	P3	P4	P5	P6
M1	0	2	0	3	2	0
M2	5	0	2	0	0	2
M3	0	3	0	2	4	0
M4	0	0	3	0	0	4

From the machine cell formation algorithm, the result obtained is depicted in Table3. In the first manufacturing cell, all three parts require two machines, M1 and M3. However, the flows of those two machines are different because of the sequence of their operations. In the second manufacturing cell, part P1 skips machine M4. Therefore, machine M2 is required for all parts, while machine M4 is used for only two parts, P3 and P4. The flows of these two machines are different, too.

Table 3 Final Machine-part Incidence Matrix

Machine	Part					
	P1	P2	P3	P4	P5	P6
M1	1(1)	1(2)*	1(2)	0	0	0
M2	1(2)	1(1)	1(1)	0	0	0
M3	0	0	0	1	1(1)	1(2)
M4	0	0	0	0	1(2)	1(1)

3. Exchange-Heuristic Algorithm in Cell Scheduling

An essential requirement of cell scheduling is that parts are grouped into part families. Thus, both the optimal cell and the optimal sequence can be determined such that the given criterion is satisfied. A number of different scheduling methods have been developed such as the branch and bound methods, the heuristic methods, and others. It is well known that the job shop problem is one of the most difficult optimization problems, and it cannot be optimally solved. Thus, the heuristic approach is one of the desirable approaches.

The exchange-heuristic algorithm is a heuristic approach for multiple resource constrained project scheduling to job shop scheduling problems with alternative machines. It is a numerical search method that is initiated with a starting feasible solution. The role of the exchange heuristic is to find a schedule that exhibits improvement over one previously obtained in terms of the minimization of the makespan.[6] Compared to traditional dispatching rules, these exchange-heuristic algorithm possesses different features as follows:

(1) It is a multiple pass heuristic dispatching procedure which considers the local minimum,

(2) It reaches to the final solution via s series of rearrangements, and

(3) It starts with some initial schedule.

3.1 Problem Statement

The problem of cell scheduling is similar to that of static job shops, but it has several different features because of the inherent characteristics of cellular manufacturing. the problems with which the exchange-heuristic algorithm in cell scheduling is concerned can be summarized ad follows:

(1) A set of parts in a cell is to be scheduled,

(2) Each part performs a set of predetermined operations,

(3) Each operation has a known processing time. They cannot be interpreted during execution and require certain numbers of different machines,

(4) The number of machines available for each type is known and limited, and

(5) The objective of the cell scheduling is to minimize the makespan.

Each part belonging to a part family herein after will also be referred to as a job in order to be consistent with the existing literature on cell scheduling. In a cell, there are p different types of jobs and m machines, and R_{jk} is the number of machines of type j available at time k. For jobs, n is the possible number of operations that can be performed with different machines. Each operation has a processing time d_j and has immediate preprocessors b_i. If operation I requires the use of machine j, the indicator function a_{ij} takes "1". Otherwise, it is expressed by "0". The cell scheduling matrix can be expressed as shown in Table 4.

Table 4 Cell Scheduling Matrix

Job	Operation (i)	Machine			Processing Time d_j	Immediate Preprocessors
		1	⋯	j		
1	1				d_1	b_1
	⋮		a_{ij}			
p	⋮		a_{ij}			
	n				d_n	b_n
Availability		$R_{jk}(k=1)$	⋯	$R_{jk}(k=1)$		

3.2 Procedure for the Exchange-Heuristic Algorithm

The procedure for the exchange-heuristic algorithm is summarized as follows:

Step 0) Initialization

Let S^0 be a starting solution.

Set ITER=0 and N=0.

Step 1) Collection of Noncritical Operations

Collect the non critical operations C and save them.

If C is empty, go to Step 7. Otherwise, go to the next step. These operations are candidates for a new TARGET operation.

Step 2) Selection of Search Region

Set ITER=ITER+1.

Select a target operation k, the start time of which operation is the latest in C. If not, go to Step 7.

Determine the boundaries the search region.

Step 3) Backward Move Operation (BMO)

Remove the TARGET from the current schedule.

Collect operations for B and for D. If B is empty, go to Step 2.

Perform BMO for operations in B.

Step 4) Forward Move Operation (FMO)

Schedule the Target as early as possible inside the search region.

If a new start time for the TARGET is not found, return to Step 2.

Step 5) Improvement

Perform REARRG for the operations in B and in D. If no operation in D is left-shifted, return to Step 2. Otherwise, proceed to the next step.

Step 6) Update the Current Schedule

Perform REARRG for the operations in E. If an improvement is obtained, then N=N+1 and update the schedule, i.e., $S=S^{N.}$

Return to Step 1.

- **Backward Move Operation (BMO)** : Select the operation k, which has the latest finish time, from the operations in B. the operation k should not be the one formerly selected in this procedure. If no operation is selected, then terminate this operation. Otherwise, move the operation k to the time period as close as possible to the start time of the successor unless the move violates the resource restriction.

- **Forward Move Operation (FMO)** : Schedule the target operation as early as possible inside the search range.
- **Rearrange Operation (REARRG)** : Select the operation k, which has the earliest start time, from the operations in B and D or in E, The operation k should not be the one formerly selected in this procedure. If no operation is selected, terminate this operation. Otherwise, move the operation k to the time period as close as possible to the finish time of the predecessor unless the move violates the resource restrictions.
- **TARGET** : An operation selected for forward move operation. A target operation is chosen among noncritical operations.
- **B** : A set of operations, which are scheduled to start inside the search region formed by LHB(Left-hand boundary) and RHB(Right-hand boundary) except the target operation, i. e., $B=\{j/LHB \le t^s_j \le RHB\}$
- **D** : A set of operations scheduled after RHB without any slack, i. e., $D=\{i/t^s_i=RHB+1\}$.
- S : Current schedule: $S=\{...,(i,t^s_i),\ (j,t^s_j),...\}$
- S^N : Nth feasible schedule.
- t^s_i : Scheduled start time of operation I.

4. Implementation and Illustration

Consider the problem given in Table 3 to demonstrate the implementation of the exchange-heuristic algorithm to the cell scheduling problem. The problem includes three jobs and two machines in each cell. The information in this example is given in Table 5 and Table 6. For instance, in the manufacturing cell-1, operation 4 takes three unit times to process via machine M1, and it has the precedent operation 3.

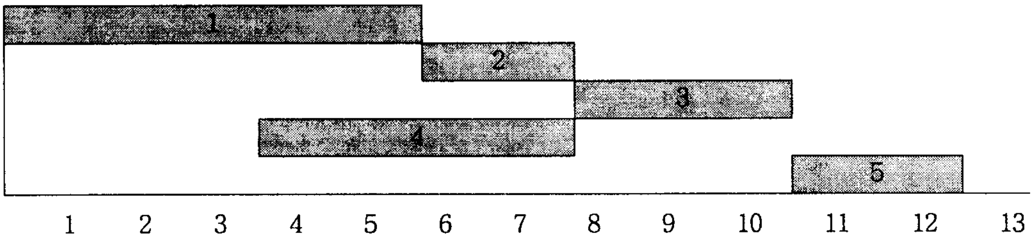
Table 5 Cell Scheduling Matrix for Manufacturing Cell-1

Job	Operation	Machine		Processing Time	Immediate Preprocessors
		M1	M3		
1	1	1	0	2	-
1	2	0	1	3	1
2	3	0	1	2	-
2	4	1	0	3	3
3	5	0	1	4	-
3	6	1	0	2	5
Availability		1	1		

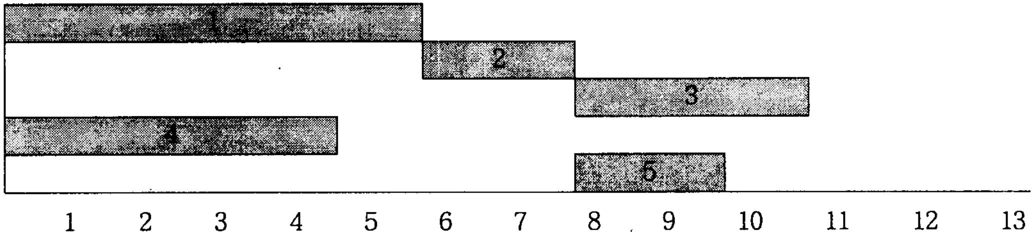
Table 6 Cell Scheduling Matrix for Manufacturing Cell-2

Job	Operation	Machine		Processing	Immediate
		M1	M3	Time	Preprocessors
1	1	1	0	5	-
2	2	1	0	2	-
2	3	0	1	3	2
3	4	0	1	4	-
3	5	1	0	2	4
Availability		1	1		

Applying the procedure discussed in previous section, the given problem is solved. Figure 1a shows the initial schedule of the manufacturing cell-2 and the final schedule is summarized in Figure 1b. The makespan at the initial step is 12. After iterating the steps, the makespan has been reduced to 10 without violating any constraints.



(a) Initial Schedule



(b) Final Schedule

Figure 1 Initial and Final Schedules

5. Performance Analysis

The performance of the exchange-heuristic algorithm is analyzed for two cases: the two machine case and the multiple machine case. In the two machine case, Jackson's algorithm(1956) is compared to the exchange-heuristic algorithm. Jackson's algorithm is a generalization of Johnson's algorithm(1954) for the job shop with the constraint that each job can have two operations at the most. In the multiple machine case, the exchange-heuristic algorithm is implemented to determine the schedule of each cell. First, the problem is converted into the format of this algorithm, then the solution is obtained.

Two Machine Case

Four examples for the two machine case are shown in Table 7. Four examples have been compared the makespans using the two algorithms. The comparison of the makespans is shown in Table 8.

Table 7 Four Examples

	Example 1		Example 2			Example 3			Example 4		
	P1	P2	P1	P2	P3	P1	P2	P3	P1	P2	P3
M1	1(1)	1(2)	1(1)	1(2)	1(1)	1	1(1)	1(2)	1	1(1)	0
M2	1(2)	1(1)	1(2)	1(1)	1(2)	0	1(2)	1(1)	0	1(2)	1

(a) Machine -part Incidence Matrix

	Example 1		Example 2			Example 3			Example 4		
	P1	P2	P1	P2	P3	P1	P2	P3	P1	P2	P3
M1	4	3	3	2	3	3	4	3	2	3	-
M2	3	4	2	4	3	-	2	3	-	3	4

(b) Processing Time

Table 8 Computational Results for the Two Machine Case

Example No.	Number of Jobs	Makespan	
		Jackson's Algorithm	Exchange-Heuristic
1	2	7	7
2	3	10	10
3	3	10	10
4	3	7	7

The results show that both algorithms yielded the same makespan in all four examples. In the two machine case, it should be more efficient to use Jackson's algorithm than the exchange-heuristic algorithm, since the exchange-heuristic algorithm has a more complicated procedure.

Multiple Machine Case

Scheduling problems for m machines $(m \geq 3)$ with p jobs $(p \geq 3)$ have been proven to be very difficult to solve to completion. The difficulty is not in modeling but in computation, since there is a total of $(p!)^m$ schedules and since each schedule should be examined to select the one that gives the minimum makespan. Therefore, Jackson's algorithm cannot be implemented in the multiple machines.

Heuristic have been developed to handle these problems. They have shown significant promise in solving the job shop scheduling problem. The most widely used heuristic belong to the class of procedures are called dispatching rules.[8] Dispatching rules are simply logical decision rules that enable a decision maker to select the next job for processing at a machine when it becomes available. The most commonly used dispatching rules are listed in Table 9.[9,10]

Table 9 Standard Dispatching Rules

Rule Name	Description
RANDOM	Select a job at random
FCFS	Select a job on a first-come, first-served basis
DDATE	Select a job with the earliest due date
SPT	Select a job with the shortest processing time
LPRT	Select a job with the least remaining processing time
S/OPR	Select a job with the minimum ratio of job slack time to the number of operations remaining
ODD	Select a job with the earliest operational due date
LSTART	Select a job with the earliest late start time
OSLK	Select a job with the least operational slack

In general, some of these rules are considered to be superior to others. The choice of dispatching rule should be determined by other factors, such as the measure of performance, the level of uncertainty, and the job arrival pattern. According to the chosen dispatching rule, sequencing is determined by the schedule sequentially over time instead of all at once.

It is necessary to demonstrate the performance of the exchange-heuristic algorithm. Yang et al. [6,7] have conducted scheduling efficiency evaluation to

(1) compare the improvements made by the exchange-heuristic with the starting solution obtained via zero-one approach.

(2) compare the results obtained versus the lower bound value of the schedule when such a bound is available, and

(3) compare the results obtained with those achieved by the "expert" human scheduler.

Consider the nine machines and 15 parts problem in Table 10, where each machine type has one machine. On the basis of the proposed machine cell formation algorithm[11], the final solution has been derived in Table 11.

Table 10 Initial Machine-part Incidence Matrix for the Nine Machine Case

M/C	Part														
	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
M1	1	0	0	1	1	0	1	0	0	1	0	0	0	0	0
M2	0	1	0	0	0	1	0	1	0	0	0	0	1	0	0
M3	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0
M4	0	0	0	0	0	1	0	1	0	0	1	0	1	0	1
M5	1	0	1	1	0	0	1	0	0	0	0	0	0	0	0
M6	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
M7	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1
M8	1	0	1	1	1	0	1	0	0	0	0	0	0	0	0
M9	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0

Table 11 Final Machine-part Incidence Matrix for the Nine Machine Case

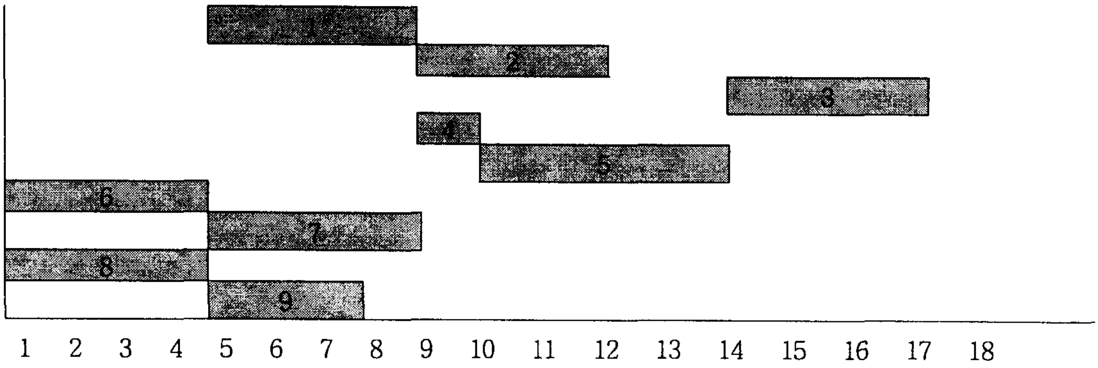
M/C	Part														
	P13	P2	P8	P6	P11	P5	P1	P10	P7	P4	P3	P15	P9	P12	P14
M2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
M4	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0
M1	0	0	0	0	0	1	[illegible]	[illegible]	[illegible]	1	0	0	0	0	0
M8	0	0	0	0	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1	0	0	0	0
M9	0	0	0	0	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0	0	0	0
M5	0	0	0	0	0	0	[illegible]	[illegible]	[illegible]	1	1	0	0	0	0
M3	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0
M7	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1
M6	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1

For example, in manufacturing cell-3, machines M3, M7, and M6 are grouped as a machine cell, and parts P15, P9, P12, and P14 are grouped as a part family. The data used in the algorithm to determine the schedule for manufacturing cell-3 is shown in Table 12.

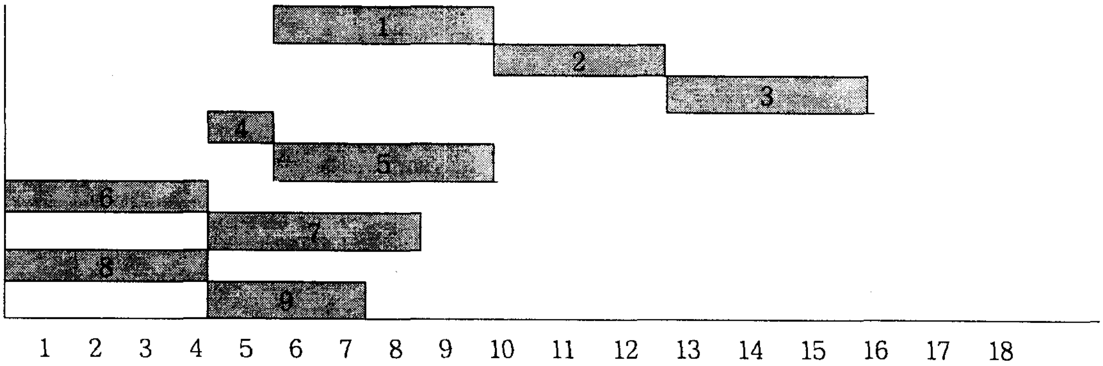
Table 12 Cell Scheduling Matrix for Manufacturing Cell-3

Operation	Machine			Processing Time	Immediate Preprocessors
	M3	M6	M7		
1	1	0	0	4	-
2	0	1	0	3	1
3	0	0	1	3	1,2
4	1	0	0	1	-
5	0	0	1	4	4
6	1	0	0	4	-
7	0	0	1	4	6
8	0	0	1	4	-
9	0	1	0	3	8
Availability	1	1	1		

Before the implementation of the exchange-heuristic algorithm in the cell scheduling problem, the initial schedule should be obtained. The RANDOM rule is used to generate the initial schedule. Figure 2a shows the initial schedule, where the makespan is 16 units. However, when the exchange-heuristic algorithm is used, the makespan is reduced to 15units. Figure 2b summarizes the final schedule in this example.



(a) Initial Schedule for Manufacturing Cell-3



(b) Final Schedule for Manufacturing Cell-3

Figure 2 Initial and Final Schedules for Manufacturing Cell-3

6. Conclusions

The cell scheduling module determines the sequences of parts in cells. To determine the schedule in each cell, the exchange-heuristic algorithm is used. The exchange-heuristic algorithm is a heuristic approach for multiple resource constrainted-project scheduling in job shop scheduling problems with alternative machines. The algorithm is reviewed and the procedure converting the manufacturing cell to cell scheduling matrix is investigated.

In the two machine case, Jackson's algorithm is compared to the exchange-heuristic algorithm. The results show that both algorithms yielded the same makespan in all four examples. In the multiple machine case, Jackson's algorithm cannot be implemented. But the exchange-heuristic provided more improved makespan than the initial schedule.

There are two hierarchical levels in the cell scheduling problem, scheduling within cell and scheduling between cells. The exchange-heuristic algorithm corresponds to the solution of the first level. After the completion of scheduling in this level, the second level should be considered. The determination of the schedule to this level may present another area of potential future research. Furthermore, the survey shows that 75% of firms gave "meeting the due date" the highest priority as a scheduling criterion. Thus, the exchange-heuristic algorithm with a different criterion or with multiple criteria also present areas for further research.

algorithm with a different criterion or with multiple criteria also present areas for further research.

[REFERENCE]

[1] Ham, I., Hitomi, K., and Yoshida, T., Group Technology, Kluwer-Nijhoff Publishing, Boston, 1985.

[2] Radharamanan, R., "A heuristic algorithm for group scheduling", International Industrial Engineering Conference Proceedings, pp.229-236, 1986.

[3] Gupta, J. N. D., "Optimal schedules for special structure flowshops", Naval Research Logistic Quarterly, Vol. 22, pp.255-269, 1975.

[4] Szwarc, W., "Special case of the flow-shop problem", Naval Research Logistic Quarterly, Vol.24, pp.483-492, 1977.

[5] Taylor, J. F., and Ham, I., "The use of microcomputers for group scheduling", Manufacturing Engineering Transactions, pp.483-491, 1981.

[6] Yang, T., Ignizio, J. P., and Deal, D. E., "An exchange heuristic for generalized job shop scheduling", Engineering Optimization, Vol. 15, pp.83-96, 1989a.

[7] Yang, T., Ignizio, J. P., and Song, J. S., "An exchange heuristic for project scheduling with limited resources", Engineering Optimization, Vol. 14, pp.189-205, 1989b.

[8] Rembold, U., Nnaji, B. O., and Storr, O., Computer Integrated Manufacturing and Engineering, Addison-Wesley, Massachusetts, 1993.

[9] Askin, R. G., and Standridge, C. R., Modeling and Analysis of Manufacturing Systems, John Wiley & Sons, New York, 1993.

[10] Elsayed E. A., and Boucher T. O., Analysis and Control of Production Systems, Prentice-Hall, New Jersey, 1994.

[11] 임춘우, 이노성, "퍼지집합에 기초한 셀 생산방식에서의 머신-셀 구성에 관한 연구“, 제어 · 자동화 · 시스템공학 논문지, 3권, 2호, pp.300-305, 1997. 6.