Journal of the Korean Institute
of Industrial Engineers
Vol. 18, No.1, Jun., 1992

A heuristic m-machine flowshop scheduling method under the total tardiness criterion

Yong Sun Choi*, Seong Soo Lee** and Soung Hie Kim*

Total Tardiness 基準下에서의 m-machine Flowshop Scheduling을 위한 發見的 技法에 관한 硏究

崔 容 銑* · 李 性 秀** · 金 聖 曦*

Abstract

Flowshop scheduling problem is known to be NP-complete. Since the optimization apporach like branch-and-bound is limited by exponentially growing computation time, many heuristic methods have been developed. Total tardiness is one of the criteria that the researchers have recently considered in flowshop scheduling. There, however, are few literatures which studied the general (m machine)-flowshop scheduling under the total tardiness criterion. In this paper, a heuristic scheduling method to minimize total tardiness at the (m machine, n job)-flowshop is presented. A heuristic value function is proposed to be used as a dispatching criterion in initial schedule generation. And the schedule improving procedure, by pairwise interchange of tardy job with the job right ahead of it, is introduced. Illustrative examles and simulated results are presented.

1. Introduction

Much effort has been given to the study of the flowshop scheduling problem. As an optimization approach, various branch and bound methods have been developed([1], [2], [3], [4], [5]). Although the branch and bound method is the best optimizing technique available for the NP-complete scheduling problem, it has a limitation : exponentially growing computation time[6]. Therefore, many heuristic methods have been developed to encounter this

*한국과학기술원 산업공학과

**Anderson Consulting

drawback([6], [7], [8], [9], [10], [11], [12]).

And, as a criterion of evaluating the schedule, most of the studies have dealt with the makespan, and have not considered the due date of each job. But one of the major characteristics of the recent studies in flowshop scheduling is to consider other evaluation criterion than makespan alone([13], [14], [15], [16]). They have considered the due date of each job and used those like maximum lateness, mean tardiness, or total tardiness, as a criterion of evaluating the schedule.

This paper suggests a heuristic (m machine, n job)−static flowshop scheduling method which minimizes the total tardiness. The general features of the static flowhop problem are described in Baker[17]. Studies of the single or two machine flowshop problem have been done by several researchers and are well cited by Sen [16]. There, howeveer, are few literatures which studied the general m-machine flowshop scheduling under the total tardiness criterion. In this paper, we have proposed a value function which can be used as a dispatching criterion in generating the initial shedule. And the schedule improving procedure, by pairwise interchange of the tardy job with the job right ahead of it, is introduced. All procedures have been coded in Pascal for the IBM−PC and simulated for various problems. The results of the simulation in two ways, for the small and large scale problem, show that our method have comparatively good performance. For almost every problem configuration, our method has shown strict dominance over the heuristic II proposed by Gelders and Sambandom[14].

2. Schedule Generation Procedure

Overall procedure of schedule generation is as follows:

1) Generate an inintial schedule using the proposed value function, described in 2.1.1, as a dispatching criterion,

2) Under the total tardiness criterion, improve the schedule by the pairwise interchange of the tardy job in the current schedule with the job right ahead of it.

2.1. Generation of an Initial Schedule by Using the Proposed Value Function as a Dispatching Criterion

A value function is proposed to be used as a dispatching criterion in generating an initial schedule. It is used to determine the relative prominence of each job, which is not scheduled yet, as a candidate job to be appended to the current partial schedule.

2.1.1. Value Function

Notations

P_{ij} : processing time at machine j of job i

d_i : due date of job i

m : number of machines

σ : current partial schedule(initial part of a complete schedule)

σk : temporarily expanded partial schedule with a candidate job k appended to σ

$\{\sigma\}$: jobs included in σ

$\{\bar{\sigma}\}$: jobs not included in σ

$EST_{ij}^{\sigma k}$: earliest starting time at machine j of job i under σk

LFT_{ij} : latest finishing time at machine j of job i

$FT_j^{\sigma k}$: finishing time of σk at machine j

For any given problem(i.e., when m and n is determined, and P_{ij} and D_i for each job is given), EST_{ij}^{ϕ}(i.e. when $\sigma=\phi$), LFT_{ij}, and FT_j^{ϕ} are determined recursively as,

$$EST_{11}^{\phi}=0,\ i=1,\cdots,n$$

$$EST_{ij}^{\phi}=EST_{i,j-1}+P_{ij},\quad i=1,\cdots,n,\ j=2,\cdots,m,$$

$$LFT_{im}=d_i,\ i=1,\cdots,n,$$

$$LFT_{i,j-1}=LFT_{ij}-P_{ij},\quad i=1,\cdots,n,\ j=1,\cdots,m-1,$$

and,

$$FT_j^{\phi}=0,\quad j=1,\cdots,m.$$

And, at any intermediate stage of initial schedule generation procedure(i.e., with each candidate job_k to append to current partial schedule σ), $EST_{ij}^{\sigma k}$ of each job_i which belongs to $\{\bar{\sigma}\}$ and $FT_j^{\sigma k}$ are iteratively modified as,

$$EST_{kj}^{\sigma k}=EST_{kj}^{\sigma},\quad j=1,\cdots,m,$$

$$EST_{i1}^{\sigma k}=EST_{i1}^{\sigma}+P_{k1},\quad i\in\{\bar{\sigma k}\},$$

and,

$$EST_{ij}^{\sigma k}=\max\{EST_{kj}^{\sigma}+P_{kj},\ EST_{i,j-1}^{\sigma k}+P_{i,j-1}\},\quad i\in\{\bar{\sigma k}\},\ j=2,\cdots,m,$$

And,

$$FT_1^{\sigma k}=FT_1^{\sigma}+P_{K1},$$

$$FT_j^{\sigma k}=\max\{FT_{j-1}^{\sigma},\ EST_{kj}^{\sigma}\}+P_{kj},\ j=2,\cdots,m,$$

recursively. Note that LFT_{ij} does not change, regradless of the expansion of σ.

Then as a candidate job to be appended to σ, the relative value of each job_k in $\{\bar{\sigma}\}$, denoted by V_k^{σ}, is determined as follows,

$$V_k^{\sigma}=\sum_{j=1}^{m}\sum_{i\in\{\bar{\sigma k}\}}[\{LFT_{ij}-EST_{ij}^{\sigma k}\}-P_{ij}]-\max\{FT_m^{\sigma k}-d_k,\ 0\}-\sum_{j=1}^{m}\max\{EST_{kj}^{\sigma k}-FT_j^{\sigma},\ 0\},\quad k\in\{\bar{\sigma}\}.$$

Basically, the value function is used to determine the relative measure of prominence of jeach job_k in $\{\bar{\sigma}\}$ to identify the most prominent one among them, as a candidate of next job to be appended to σ. The next job to be appended to σ is selected as the one with maximum value, V_k^{σ}. The proposed value function has been tried to consider the prospective global as well as the local effects relevantly with σ expansion. The time window($LFT_{ij}-EST_{ij}^{\sigma k}$) represents the available processing time at $machine_j$ of job_i, under current temporary partial schedule σk. And the time window subtracted by the corresponding processing time ($\{LFT_{ij}-EST_{ij}^{\sigma k}\}-P_{ij}$) represents the slack time of processing at $machine_j$ of job_i. The absolute value of a negative slack time represents the tardiness of the corresponding operation.

More in detail, the value function is decomposed into the following 3 parts.

*$A=\sum_{j=1}^{m}\sum_{i=\{\bar{\sigma k}\}}[\{LFT_{ij}-EST_{ij}^{\sigma k}\}-P_{ij}]$

The sum of slack times at each $machine_j$ of each job_i in $\{\bar{\sigma k}\}$, under temporary σk. It has been adopted to measure the relative slackness in processing the rest unscheduled jobs under σk.

*$B=\max\{FT_m^{\sigma k}-d_k,\ 0\}$

The own tardiness of job_k under σk.

$$*C=\sum_{j=2}^{m}\max\{EST_{kj}^{\sigma k}-FT_j^{\sigma},\ 0\}$$

The sum of newly induced machine idle time at each machine caused by appending job k after σ(note that there is no idle time at the first machine). When $\sigma=\phi$ (i.e. when to select the first job in sequencing), it is computed as $\sum_{j=2}^{m} EST_{kj}^{\phi}$ since $EST_{kj}^{\phi k}=EST_{kj}^{\phi}$, and $FT_j^{\phi}=0$, for j=1,…, m. $\sum_{j=2}^{m} EST_{kj}^{\phi}$ is the initial machine

idle time caused by placing job k first in sequencing.

The larger value of term A can be considered better to the prospective resultant final schedule, while term B and C are in the opposite way. Therefore the value function has been designed that the value is to be determined by subtracting term B and C from term A. And the next job to be appended to the current partial schedule σ is determined as the one with maximum V_k^{σ}.

As an example, the value computation is illustrated. The example and the inital time windows (i.e., when $\sigma=\phi$) are shown in Table 1 and Table 2. And the modified time windows when each candidate job k in $\{\bar{\sigma}\}=\{1, 2, 3\}$ is selected as an initial job are shown in Table 3 (a), (b), and (c) respectively. To show how each value term is determined, the graphical representation of Table 3(a), i.e. when job 1 is considered as an initial candidate branching job, is shown in Figure 1 with processing times. Each both-directed arrow represents the modified time window at each machine of each job. Each box represens the corresponding processing time. And each interval between each pair of two arrows represents each value term of V_1^{ϕ}, where A_{ij}^{1} and C^1 represent the corresponding value term indices (Note that term B^1 does not appear in this case. By placing job 1 first in the job sequence, it is finished at time 3, before the due date of job 1, time 6. Therefore $B^1=\max\{3-6, 0\}=0$). That is,

$$\begin{aligned}V_1^{\phi}&=(A_{21}^{1}+A_{22}^{1}+A_{31}^{1}+A_{32}^{1})-B^1C^1\\&=[\{(8-1)-5\}+\{(10-6)-2\}\\&\quad+\{(6-1)-3\}+\{(12-4)-6\}]-0-1\\&=(2+2+2+2)-1=7\end{aligned}$$

In the same way, it can be seen that

$$\begin{aligned}V_2^{\phi}&=(A_{11}^{2}+A_{12}^{2}+A_{31}^{2}+A_{32}^{2})-B^2C^2\\&=(-2-3-2-2)-0-5=-14,\end{aligned}$$

and

$$\begin{aligned}V_3^{\phi}&=(A_{11}^{3}+A_{12}^{3}+A_{21}^{3}+A_{22}^{3})-B^3-C^3\\&=(0-5+0-1)-0-3=-9,\end{aligned}$$

from Table 1 and 3. By comparing the computed values, job 1 is selected as the first job in the resultant partial schedule, that is $\{\sigma\}=\{1\}$ and $\{\bar{\sigma}\}=\{2, 3\}$.

〈Table 1〉. Example problem 1

job no.	processing time		due date
	machine 1	machine 2	
1	1	2	6
2	5	2	10
3	3	6	12

〈Table 2〉. Initial time windows

job no.	machine 1		machine 2	
	EST	LFT	EST	LFT
1	0	4	1	6
2	0	8	5	10
3	0	6	3	12

〈Table 3〉. Modified time windows

job no.	machine 1		machine 2	
	EST	LFT	EST	LFT
1	*	*	*	*
2	1	8	6	10
3	1	6	4	12

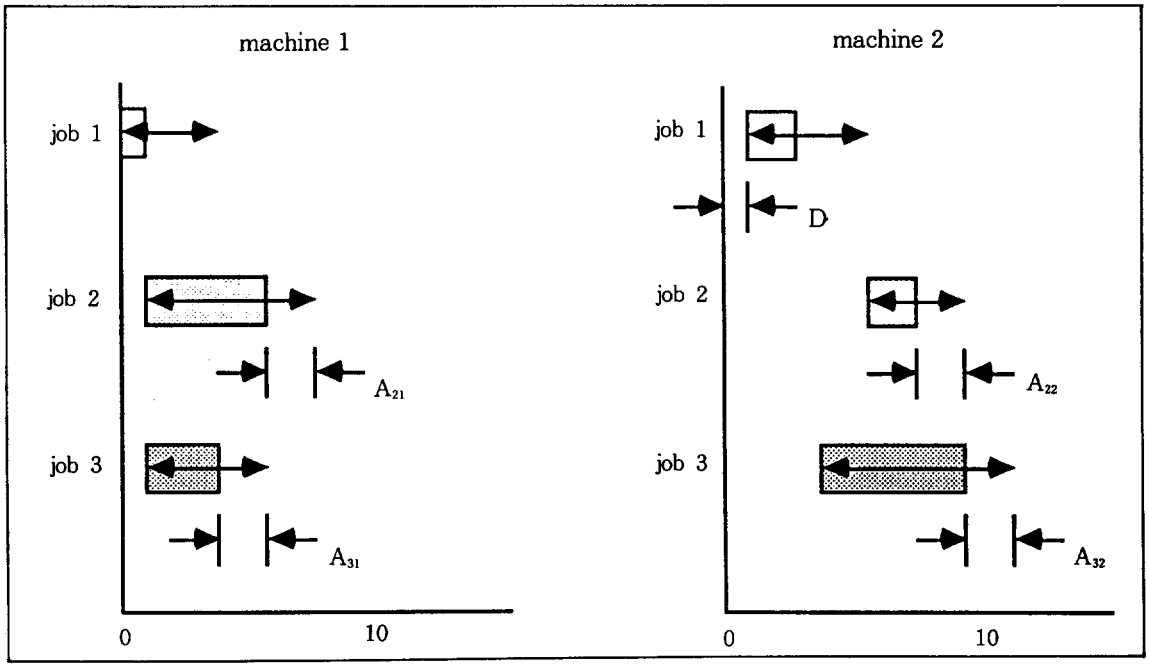
(a) when k=1, i.e., job 1 selected first

job no.	machine 1		machine 2	
	EST	LFT	EST	LFT
1	5	4	7	6
2	*	*	*	*
3	5	6	8	12

(b) when k=2

job no.	machine 1		machine 2	
	EST	LFT	EST	LFT
1	3	4	9	6
2	3	8	9	10
3	*	*	*	*

(c) when k=3



[Fig.1] Modified windows, processing times, and value terms when k=1

The procedure of the initial schedule generation using the proposed value function as a dispatching criterion is described simply as follows:

procedure construct_initial_schedule

1. Set$\{\sigma\}=\phi$. Generate initial time windows of each job at each machine.

repeat

2.1. Compute V_k^σ for each job k in $\{\bar{\sigma}\}$, and find job k*, where $k^* = \max_k [V_k^\sigma \mid k \in \{\bar{\sigma}\}]$

2.2. Set $\sigma = \sigma k^*$, and propagate the time windows of the jobs in $\{\bar{\sigma}\}$

until $\{\bar{\sigma}\} = \phi$

As an example, the initial schedule generation is illustrated. The processing times and the due dates of the example (5 job, 4 machine)-problem are shown in Table 4. And Table 5 shows the jobs which are not in the current partial schedule σ(in column $\{\bar{\sigma}\}$), computed values for each of them(column V_k^σ), the selected job with maximum value(column k*), and the resultant new partial schedule(column σ) are shown at each iteration. As a result, we get the final job sequence 2-1-3-5-4 with total tardiness 86. The minimum total tardiness that can be obtained by whole enumeration is 82(at job sequence 2-1-3-4-5) and the maximum total tardiness is 278(at job sequence 5-4-1-3-2). The initial job sequence obtained by using the proposed value function as a dispatching criterion shows comparatively good performance ratio+((278-86)/(278-82))≅0.98) between the best and the worst job sequence under the total tardiness criterion.

〈Table 4〉. Example problem 2

job no.	processing time				due date
	machine 1	machine 2	machine 3	machine 4	
1	4	19	9	34	73
2	4	2	12	8	33
3	18	8	10	9	56
4	7	10	25	28	92
5	3	8	37	7	71

〈Table 5〉. Computed values and schedule generation

iteration	$\{\bar{\sigma}\}$	V_k^σ					k*	σ
1	{1,2,3,4,5}	-103	126	-222	-234	-147	2	2
2	{1,3,4,5}	-16	*	-132	-136	-87	1	2-1
3	{3,4,5}	*	*	-79	-142	-113	3	2-1-3
4	{4,5}	*	*	*	-123	-109	5	2-1-3-5

2.2. Improvement of the Schedule

The basic idea of the schedule improvement procedure is to interchange the tardy job with the job right ahead of it in the current schedule to reduce the total tardiness. The systemic procedure of the schedule improvement is described as follows:

procedure improve_schedule

1. (Initialization) Set current schedule, denoted by s, as the initial schedule generated by the proposed value function. And set n to the number of jobs
2. Assess tardiness of each job, under s, and sort the tardy jobs in descending order of job tardiness. And set t to the number of tardy jobs
3. **for i=1 to t do**
   3.1. Generate candidate schedule, denoted by **s'**, by interchanging the i_{th} tardy job with the job right ahead of it in s
   3.2. If s' has better performance(i.e. less total tardiness) than s, set s to s' and go to step 2.
4. (stopping criterion of improving procedure)
   4.1. generate(n−1) alternative schedules by pairwise interchanging the i_{th} job, in s(i=2, ···, n) obtained at the end of step 3, and set s' as the one which has the minimum total tardiness among them
   4.2. if s' has better performance than s, set s to s' and go to step 2. Unless stop.

An example of the schedule improving procedure is illustrated.

stage 1)

1. s=2−1−3−5−4, n=5.
2. Table 6 shows the tardiness of each job under s, order of each tardy job in descending order of job tardiness, and total tardiness(TT) of s. t=3.

3.1. s'=2−1−3−4−5 by interchanging the most tardy job 4 with job 5, right ahead of job 4 in s.

3.2. Total tardiness of s'=82. So, improved. Set s=2−1−3−4−5.

〈Table 6〉

s=2-1-3-5-4 TT=86

job	tardiness
1	0
2	0
3	23
4	44
5	19

〈Table 7〉

s=2-1-3-4-5 TT=82

job	tardiness
1	0
2	0
3	23
4	15
5	44

Stage 2)

2. s=2−1−3−4−5. Tardiness of each job under s, job tardiness order of tardy jobs, and total tardiness of s are shown in Table 7. t=3.
3. Candidate schedules(s') and their performances are shown in Table 8. None of these schedules have less total tardiness than s.
4. Alternative schedules generated by the pairwise adjacent job interchanging to s and their performances are shown in Table 9. None of these schedules have less total tardiness than s. Therefore, we get the final job sequence 2−1−3−4−5 with total tardiness 82, which is the minimum total tardiness found by whole enumeration.

〈Table 8〉

s=2−1−3−4−5 TT=82

candidate schedule	total tardiness
2-1-3-5-4	86
2-3-1-4-5	104
2-1-4-3-5	102

〈Table 9〉

s=2−1−3−4−5 TT=82

alternate schedule	total tardiness
1-2-3-4-5	139
2-3-1-4-5	104
2-1-4-3-5	102
2-1-3-5-4	86

2.3. Simulation Results

The overall procedure has been coded in Pascal for the IBM-PC, and has been simulated for various examples. In simulation, we have used random numbers between 1 and 40 for each processing time P_{ij}. The due date of each job, d_i, was given as random from the randomly selected one of the three different tardiness intervals(between 1.1 and 1.3 times of $\sum_{j=1}^{m} P_{ij}$, between 1.3 and 1.5 times, and between 1.1 and 1.5 times). This was to check the effects of the ***inherent tardiness factor*** and ***due date range***[16] on the performance of our method. The simulation has been performed in two ways. The first was for small scale problems. The number of jobs is selected as one of {5, ···, 8}, and the number of machines is selected as one of {4, 6, 8, 10}. For each problem configuration(number of jobs and number of machines), 40 random samples are generated. Moreover, for each sample, whole (number of jobs)−factorial schedules are enumerated, and the maximum and minimum total tardiness of these schedules are checked to be compared with the total tardiness of the schedule generated using our method. We have used the performance measure as PR(Performance Ratio), which is computed for each sample, as

$$PR=\frac{\text{Max. TT}^{+}-\text{TT of generated schedule}}{\text{Max. TT}^{+}-\text{Min. TT}^{++}},$$

where TT represents total tardiness. Table 10 shows the result. In table 10, the range of the PR, the average PR, the standard deviation of the PR, the average computation time, and the number of cases generating the minimum total tardiness sequence found by whole enumeration (entitled as 'no. of opt.') are shown for each problem configuration. By way of comparison, the results of heuristic Ⅱ Proposed by Gelders and Sambandom[14](GS-Ⅱ henceforth) are shown together. There are few literatures which studied the general (m machine)−flowshop scheduling under the total tardiness criterion. Although GS-Ⅱ applies the criterion of the sum of total weighted tardiness and total weighted holding cost, the GS-Ⅱ may also be considered as the total tardiness criterion by letting the weight of the holding cost of each job be euqal to zero and the weight of tardiness of each job be equal. To observe the performance of the initial schedule(i.e. before improving) of each heuristic, two more rows of data are added. Figures 2 and 3 are the graphical representations of table 10, the average PR and the average computation time column, respec-

+, ++ : found by whole enumeration

〈Table 10〉

no. of job	no. of machine	no.of sample	PR range	avg. PR	s.d.	no.of opt.	comp. time
5	4	40	0.807-1.0	0.9745	0.0424	17	3.53
			0.952-1.0	0.9968	0.0088	32	4.88
			0.474-1.0	0.9535	0.0925	15	6.63
			0.605-1.0	0.9763	0.0686	29	7.46
	6	40	0.835-1.0	0.9590	0.0502	17	5.05
			0.952-1.0	0.9968	0.0088	32	8.20
			0.474-1.0	0.9535	0.0925	15	8.68
			0.605-1.0	0.9763	0.0686	29	10.13
	8	40	0.802-1.0	0.9574	0.0535	14	5.90
			0.873-1.0	0.9848	0.0302	28	8.68
			0.668-1.0	0.9390	0.0775	14	13.05
			0.795-1.0	0.9781	0.0400	23	14.85
	10	40	0.743-1.0	0.9641	0.0534	16	7.63
			0.912-1.0	0.9962	0.0151	35	10.46
			0.668-1.0	0.9450	0.0734	14	16.17
			0.668-1.0	0.9753	0.0593	28	18.22
6	4	40	0.796-1.0	0.9529	0.0517	10	4.68
			0.919-1.0	0.9871	0.0216	22	7.58
			0.679-1.0	0.9428	0.0664	10	10.20
			0.810-1.0	0.9707	0.0450	17	10.73
	6	40	0.629-1.0	0.9395	0.0711	11	7.78
			0.894-1.0	0.9848	0.0260	25	11.56
			0.474-1.0	0.9535	0.0925	15	8.68
			0.605-1.0	0.9763	0.0686	29	10.13
	8	40	0.842-1.0	0.9493	0.0435	7	10.07
			0.895-1.0	0.9805	0.0269	18	14.77
			0.793-1.0	0.9548	0.0474	8	20.73
			0.842-1.0	0.9775	0.0347	19	23.31
	10	40	0.800-1.0	0.9580	0.0543	15	12.40
			0.899-1.0	0.9907	0.0250	33	18.82
			0.633-1.0	0.9264	0.0778	5	26.80
			0.709-1.0	0.9607	0.0640	16	29.65

* : 1. our initial schedule 3. GS-Ⅱ initial schedule
2. our final schedule 4. GS-Ⅱ final schedule

〈Table 10〉(continued)

no. of job	no. of machine	no.of sample	PR range	avg. PR	s.d.	no.of opt.	comp. time
7	4	40	0.716-1.0	0.9490	0.0517	4	7.70
			0.862-1.0	0.9800	0.0265	12	12.10
			0.692-1.0	0.9315	0.0706	6	14.68
			0.740-1.0	0.9545	0.0604	14	16.60
	6	40	0.744-1.0	0.9310	0.0726	5	11.42
			0.825-1.0	0.9743	0.0442	19	16.72
			0.620-1.0	0.9194	0.0706	3	22.35
			0.650-1.0	0.9460	0.0712	8	24.57
	8	40	0.810-1.0	0.9430	0.0539	11	14.98
			0.835-1.0	0.9764	0.0349	17	21.80
			0.672-1.0	0.9125	0.0842	5	30.63
			0.696-1.0	0.9399	0.0733	9	33.51
	10	40	0.828-1.0	0.9514	0.0365	4	19.08
			0.913-1.0	0.9848	0.0251	33	28.18
			0.718-1.0	0.9323	0.0637	4	39.33
			0.770-1.0	0.9569	0.0537	10	43.55
8	4	40	0.883-1.0	0.9418	0.0556	2	11.08
			0.906-1.0	0.9821	0.0231	15	17.18
			0.770-1.0	0.9351	0.0560	2	20.90
			0.794-1.0	0.9547	0.0438	6	23.10
	6	40	0.847-1.0	0.9390	0.0499	2	16.75
			0.859-1.0	0.9768	0.0289	11	25.02
			0.758-0.991	0.9050	0.0588	0	31.30
			0.803-1.0	0.9250	0.0574	2	34.65
	8	40	0.806-1.0	0.9390	0.0499	2	22.00
			0.857-1.0	0.9755	0.0308	13	31.60
			0.695-1.0	0.9057	0.0754	2	43.93
			0.700-1.0	0.9341	0.0619	8	47.46
	10	40	0.848-1.0	0.9352	0.0459	3	27.48
			0.869-1.0	0.9737	0.0291	12	40.48
			0.738-1.0	0.8905	0.0736	1	56.80
			0.744-1.0	0.9222	0.0650	3	61.40

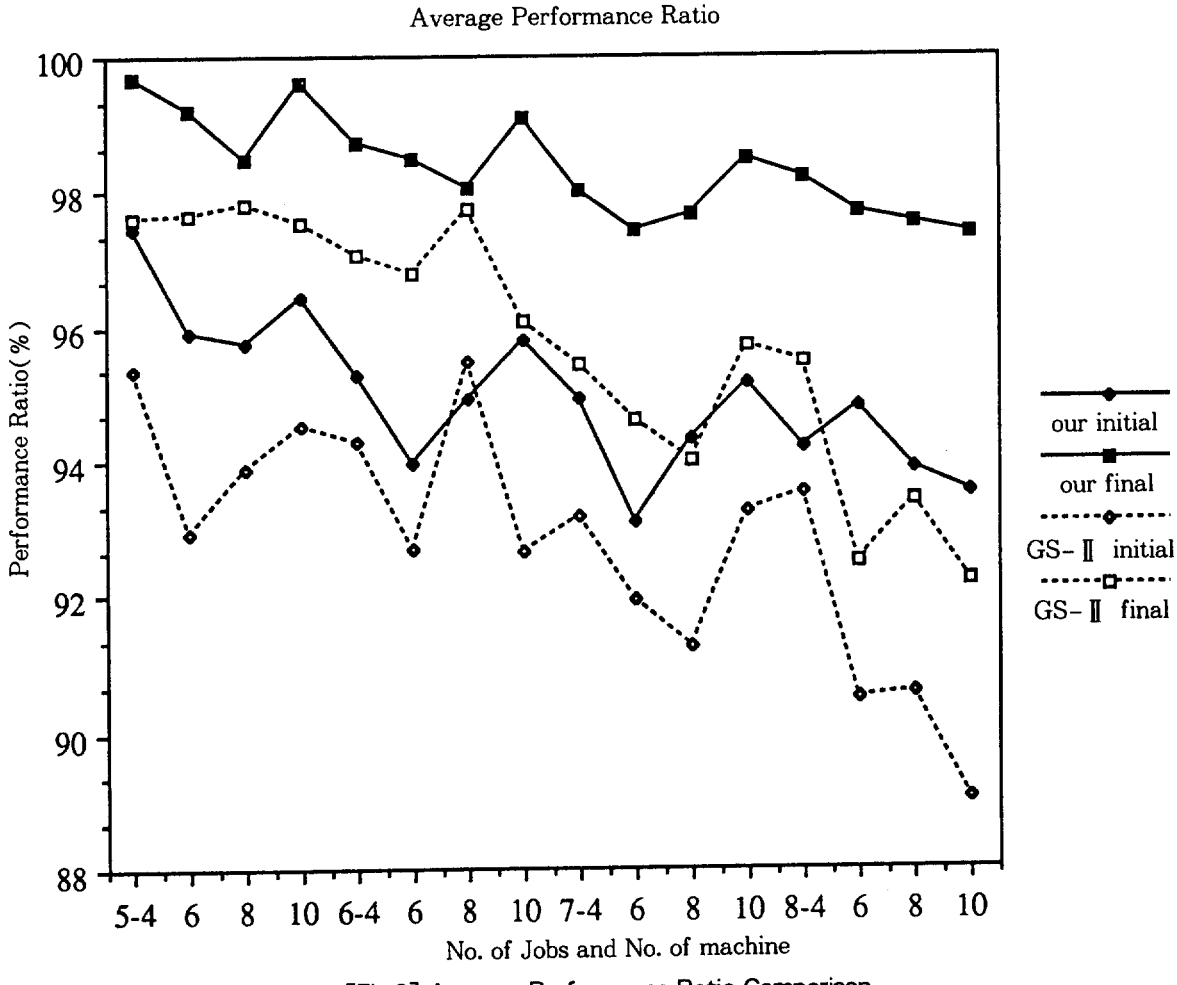
tively. From the table and figures, strict dominance of our method over GS-Ⅱ is easily recognized for both the initial and final schedules at every problem configuration. To summarize, the following observations are made:

1. Average computation time increases as the number of jobs and the number of machines increase in both methods.

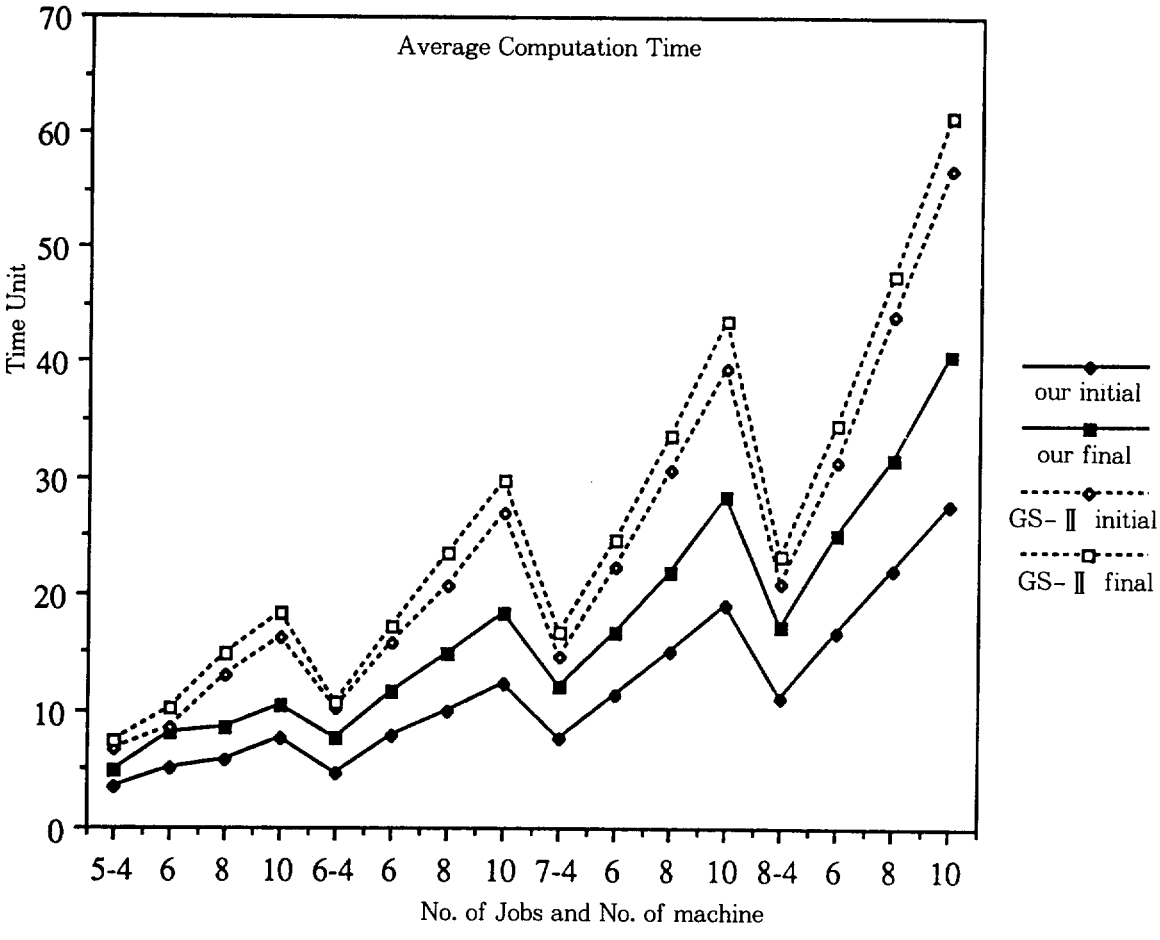
2. In average computation time, our method dominates (i.e. takes less time than) GS-Ⅱ in every problem configuration for both the initial and final schedules.

3. The average PR drops slowly as the number of jobs and the number of machines increase for all four categories of schedules. But the average PR of the schedules in the category of our final schedule drops less slowly (maximum 2% differnece in a small scale problem simulation) than those of the other three categories.

4. For the average PR, our method again dominates (i.e. has better average PR than) GS-Ⅱ in almost every problem configuration for



[Fig.2] Average Performance Ratio Comparison



[Fig. 3] Average Compution Time Comparison

both the initial and final schedules. Especially when the number of jobs is eight, initial schedules show better average PR's than the final schedules by GS-Ⅱ.

5. Similarly, the range of the PR and 'no. of opt.' column show the dominance of our method over GS-Ⅱ.

The second form of simulation is for large scale problems(when the number of jobs is greater than or equal to 10). In this case, we cannot find out the best and worst total tardiness schedules since flowshop scheduling is NP-complete, and we can not use the PR as a performance measure. Therefore, we just compare our method with the GS-Ⅱ. The number of jobs is selected randomly from[10, ···, 20], and the number of machines is selected randomly from[4, ···, 20]. For each randomly selected problem configuration(number of jobs and number of machines), each processing time, P_{ij}, and due date, d_i, are assigned in the mann-

er described above. Three hundred random samples were generated. For each sample, four values, the initial and final total tardiness ratios of our schedules over the GS-Ⅱ's schedules and the initial and final computation time ratios of ours over the GS-Ⅱ's were computed. Table 11 shows the averaged result, average total tardiness ratio (ATTR) and average computation time ratio (ACTR) for both the initial and final schedules. From the table, we can see that our method dominates GS-Ⅱ in both the computation time and total tardiness for both the initial and final schedules, in the large scale problem, as well.

〈Table 11〉

	ATTR(%)	ACTR(%)
initial schedule	91.02	47.12
final schedule	88.89	59.41

3. Conclusion

In this paper, we have suggested a heuristic scheduling method to minimize the total tardiness in the (m machine, n job)-flowshop problem. The method is composed of two steps: 1) the construction of the initial schedule using the proposed value function as a dispatching criterion, and 2) the improvement of the schedule by pairwise interchange of the tardy job with the job right ahead of it. The results of the simulation in two ways, for the small and large scale problems, show that our method performs comparatively well. For almost every problem configuration, our method has shown strict dominance over the heuristic Ⅱ proposed by Gelders and Sambandom.

References

[1] P.F. Bestwick and N.A.J. Hastings, A New Bound for Machine Scheduling. *Operations Research Quarterly* 27, 479-487 (1976).

[2] A.P.G. Brown and Z.A. Lomnicki, Some Applications of the Branch And Bound algorithm to the Machine Scheduling Problem. *Operations Research Quarterly* 17, 173-186 (1966).

[3] Z.A. Lomnicki, A Branch-and-Bound Algorithm for the Exact solution to the Three Machine Scheduling Problem. *Operations Research Quarterly* 16, 89-100(1965).

[4] B.B. McMachon and P.G. Burton, Flow-shop Scheduling with Branch and Bound Method. *Operations Research* 15, 472-481 (1967).

[5] R.D. Smith and R.A. Dudek, A General Algorithm for Solution of the n-job, m-Machine Sequencing Problem of the Flow Shop. *Operations research* 15, 71-82(1967).

[6] J.R. King and A.S. Spachis, Heuristics for Flow-Shop Scheduling. *Int. Jr. Prod. Res.* 18, 345-357(1980).

[7] M.C. Bonney and S.W. Grundy, Solutions to the Constrained Flowshop Sequencing Problem. *Operations Research Quarterly* 27, 869-883(1976).

[8] H.G. Campbell, R.A. Dudek, and M.L. Smith, A heuristic Algorithm for the n job and m Machine Sequencing Problem. *Management Science* 16, B630-B637(1970).

[9] J.N.D. Gupta, Heuristic Algorithms for Multistage Flowshop Scheduling Problem. *AIIE Transactions* 4, 11-18(1972).

[10] D.S. Palmer, Sequencing Jobs through A Multi-Stage Process in the Minimum Total Time-A Quick Method of Obtaining A Near Optimum. *Operations Research Quarterly* 16, 101-107(1965).

[11] M. Wildmer and A. Hertz, A New Heuristic Method for the Flow Shop Sequencing Problem. *European Jr. Opl. Res.* 41, 186-193 (1989).

[12] Y.B. Park, C.D. Pedgen, and E.E. Enscore, A Survey and Evaluation of Static Flowshop Scheduling Heuristics. *Int. Jr. Prod. Res.* 22, 127-141(1984).

[13] T.D. Fry, L. Vincens, K. Macleod, and S. Fernandez, A Heuristic Solution Procedure to Minimize $\overline{T}$ on a Single Machine. *J. Opl Res. Soc.* 40, 293-297(1989).

[14] L.F. Gelders and N. Sambandom, Four Simple Heuristics for Scheduling a Flow-shop. *Int. Jr. Prod. Res.* 16, 221-231(1978).

[15] J. Grabowski, E. Skubalska, and C. Smutnicki, On Flow Shop Scheduling with Release and Due Dates to Minimize Maximum Lateness. *Jr. Opl. Res. Soc.* 34, 615-620 (1983).

[16] T. Sen, P. Dileepan, and J.N.D. Gupta, The Two-machine Flowshop Scheduling Problem with Total Tardiness. *Computers Opns. Res.* 16, 333-340(1989).

[17] K.R. Baker, *Introduction to Sequencing and Scheduling*, John Wiley & Sons Inc., New York(1974).