Using the LPT and the Palmer Approaches to Solve Group Flexible Flow-shop Problems

Tzung-Pei Hong,[†] *Pei-Ying Huang*[††]*, and Gwoboa Horng*[†††]*,*

[†]National University of Kaohsiung, Kaohsiung, Taiwan
[††]National Taiwan University, Taipei, Taiwan
[†††]National Chung-Hsing University, Taichung, Taiwan

Summary
In simple flow shop problems, each machine operation center includes just one machine. If at least one machine center includes more than one machine, the scheduling problem becomes a flexible flow-shop problem. Recently, group scheduling has also been proposed and discussed. In the group scheduling, each job belongs to a specific group and all the jobs are processed group by group. In this paper, we propose a heuristic algorithm to solve group flexible flow-shop problems with more than two machine centers, which have the same number of machines. It first determines the job sequencing in each group by combining both the LPT and the Palmer approaches to solve the flexible flow-shop problems of more than two machine centers. It then determines group sequence by the Palmer approach. Experiments are also made to compare the performance of the proposed algorithm.

Key words:
Group Scheduling, Flexible Flow Shop, Dynamic Programming, Palmer Algorithm.

1. Introduction

Scheduling is an important process widely used in manufacturing, production, management, computer science, and so on. Appropriate scheduling not only reduces manufacturing costs but also reduces possibilities for violating due dates. Finding good schedules for given sets of jobs can thus help factory supervisors effectively control job flows and provide solutions for job sequencing. In simple flow-shop problems, each machine center has just one machine. If at least one machine center has more than one machine, the problem is called a flexible flow-shop problem [2]. Flexible flow shops are thus generalization of simple flow shops. Scheduling jobs in flexible flow shops is considered an NP-complete problem [1][4][5]. Recently, group scheduling has also been proposed and discussed. For group scheduling, each job belongs to a specific group and all the jobs are processed group by group [7][8][10].

The LPT and the Palmer algorithms are two simple approaches commonly used in scheduling. The LPT algorithm is used for scheduling a set of independent tasks with arbitrary execution time on an arbitrary number of processors. The Palmer algorithm is used to schedule jobs in a simple flow-shop with more than two machines to achieve a nearly minimum completion time. Both the LPT and the Palmer algorithms have the advantage of low computational time complexity. In this paper, we thus propose an algorithm based on LPT and Palmer to solve group flexible flow-shop problems with more than two machine centers. We assume all machine centers have the same number of machines. The proposed one is a heuristic algorithm. It determines the job sequencing in each group by combining both the LPT [3] and the Palmer approaches to solve flexible flow-shop problems of more than two machine centers. It then determines the group sequencing by the Palmer approach. Experimental results show that the proposed algorithm only got a little larger makespans than the optimal one, which is based on the dynamic programming technique. A trade-off can thus be achieved between accuracy and time complexity.

The remainder of this paper is organized as follows. Related scheduling algorithms are reviewed in Section 2. The assumptions and notation used in this paper are described in Section 3. The proposed algorithm for heuristically scheduling on a group flexible flow shop with more than two machine centers is proposed in Section 4. An example to illustrate the proposed heuristic scheduling algorithm is given in Section 5. Experiments for comparing the makespans and execution times of the proposed algorithm with the optimal one are described in Section 6. Finally, conclusions are given in Section 7.

2. Review of Related Scheduling Algorithms

As mentioned above, flexible flow-shop problems are NP-complete. The group flexible flow-shop problems are also NP-complete. No algorithms except exhaustive search can find optimal solutions. In the paper, we thus propose a heuristic algorithm based on the LPT, the Palmer, and Sriskandarajah and Sethi's approaches to solve the group flexible flow-shop problems of more than two machine

Manuscript received March 25, 2006.
Manuscript revised March 30 , 2006.

centers. These related scheduling algorithms are first introduced as follows.

2.1 Review of the LPT Scheduling Algorithm

The discovery of scheduling algorithms for a set of independent tasks with arbitrary execution time and an arbitrary number of processors is a classic sequencing problem of wide interest and application. Among the scheduling algorithms proposed, the LPT (Longest-Processing-Time-first) scheduling algorithm is the simplest one and is widely used in many real-world situations.

Given a set of n independent tasks (T_1 to T_n), each with arbitrary execution time (t_1 to t_n), and a set of m homogeneous processors or machines (P_1 to P_m), the LPT scheduling algorithm assigns the task with the longest execution time (among those not yet assigned) to a free processor whenever this processor becomes free. For cases when there is a tie, an arbitrary tie-breaking rule can be assumed. The algorithm is described as follows.

The LPT Scheduling Algorithm:

```
Input: A set of $n$ tasks, each with arbitrary processing time, and a set of $m$ processors.
Output: A schedule and the final finishing time of all the tasks.
Step 1: Sort the tasks in a descending order according to the processing time.
Step 2: Initialize the current finishing time of each processor to zero.
Step 3: Assign the first task in the task list to the processor with the minimum finishing time.
Step 4: Set the new finishing time of the processor = the old finishing time of the processor + the execution time of the task.
Step 5: Remove the task from the task list.
Step 6: Repeat Steps 3 to 5 until the task list is empty.
Step 7: Among the finishing time of the processors, choose the longest as the final finishing time.
```

The finishing time by the LPT scheduling algorithm is in general not minimal. The computational time spent by the LPT scheduling algorithm is however much lower than that by an optimal scheduling algorithm.

2.2 Review of the Palmer Scheduling Algorithm

The Palmer algorithm [6] was proposed to schedule job sequencing for a flow shop with more than two machines. Given a set of n independent jobs, each having m (m>2) tasks (T_{11} , T_{21}, … , T_{m1}, T_{12}, T_{22}, …, $T_{(m-1)n}$, T_{mn}) that must be executed in the same sequence on m machines (P_1, P_2, …, P_m), the Palmer scheduling algorithm seeks a nearly minimum completion time of the last job. This algorithm is stated as follows.

The Palmer Scheduling Algorithm:

```
Input: A set of $n$ jobs, each having $m$ ($m > 2$) tasks executed respectively on each of $m$ machines.
Output: A schedule with a nearly minimum completion time of the last job.
Step 1: Find the value $\pi_j$ for each job $J_j$ as follows:
        $\pi_j = \sum_{i=1}^{\lceil m/2 \rceil} -(m-2i+1)t_{ij} + (m-2i+1)t_{(m+1-i)j}$,
        where $t_{ij}$ represents the execution time of the $i$-th task $T_{ij}$ in job $J_j$.
Step 2: Sort the jobs in descending order of $\pi_j$'s; if two or more jobs have the same value of $\pi_j$, sort them in an arbitrary order.
Step 3: Schedule the jobs on the machines in the sorted order.
```

After Step 3, scheduling is finished and a completion time has been found.

2.3 Review of the Sriskandarajah and Sethi's Scheduling Algorithm

Sriskandarajah and Sethi proposed a heuristic algorithm [9] for solving the flexible flow-shop problem of two machine centers and the completion time of the derived schedules was close to the optimum. Sriskandarajah and Sethi decomposed the problem into the following three sub-problems and solved each heuristically:

Part 1: Form the machine groups, each of which contains a machine from each center.
Part 2: Use the LPT method to assign jobs to each machine group (flow shop).
Part 3: Deal with job sequencing and timing using the Johnson algorithm.

In this paper, we will extend the above three approaches to solve the group flexible flow-shop problems of more than two machine centers.

3. Assumptions and Notation

Assumptions and notation used in this paper are described in this section.

Assumptions:

- Jobs are not preemptive.
- Each job has m ($m > 2$) tasks with processing times, executed respectively on each of m machine centers.

- All the machine centers have the same number of homogeneous machines.

Notation:

l:	*The number of groups.*
n:	*The number of jobs in a certain group.*
m:	*The number of tasks in each job.*
p:	*The number of machines in each machine center.*
F_i:	*The i-th allocated machine group (flow shop), i = 1 to p.*
F_{ji}:	*The j-th machine of the flowshop F_i, j = 1 to m.*
f_i:	*The completion time of the i-th flowshop.*
f_{ji}:	*The completion time of the j-th machine in the i-th flowshop.*
t_{ijk}:	*The execution time for i-task of j-job in the k-th group.*
tt_{ik}:	*The total execution time of the i-th job in the k-th group.*
mc_{jk}:	*The processing time at the j-th machine center for the k-th group.*
ff:	*The final completion time of the whole schedule.*

4. A heuristic algorithm for group flexible flow-shop scheduling with more than two machine centers

The proposed group flexible flow-shop algorithm first determines the job sequencing in each group by combining both LPT and Palmer approaches to solve flexible flow-shop problems of more than two machine centers. It then determines the group sequencing by the Palmer algorithm. The proposed algorithm is stated below.

The proposed heuristic group flexible flow-shop algorithm:

Input: l groups, each of which has a set of jobs, each having m ($m > 2$) tasks, to be executed respectively on each of m machine centers with p homogenous machines.

Output: A schedule with a completion time.

Level 1: Determining job sequence in each group

Step 1: Set variable k to one, where k represents the number of the current group to be processed.

Step 2: Repeat Steps 3 to 15 until $k > l$.

Part 1: Forming the machine groups.

Step 3: Form p machine groups, each of which contains one machine from each machine center. Each machine group can be thought of as a simple flow shop $F_1, F_2, \ldots, F_p$.

Step 4: Initialize the completion time $f_1, f_2, \ldots, f_p$ of each flow shop $F_1, F_2, \ldots, F_p$ to zero.

Part 2: Assigning jobs to machine groups.

Step 5: For each job J_{jk}, find its total execution time $tt_{jk} = t_{1jk} + t_{2jk} + \ldots + t_{mjk}$ (j = 1 to n, k = 1 to l).

Step 6: Sort the jobs in descending order of processing time tt_{jk}; if any two jobs have the same tt_{jk} values, sort them in an arbitrary order.

Step 7: Find the flow shop F_i with the minimum processing time f_i among all the flow shops; if two flowshops have the same minimum f_i value, choose one arbitrarily.

Step 8: Assign the first job J_{jk} in the sorted list to the chosen flow shop F_i which has the minimum completion time f_i among all p flow shops.

Step 9: Add the total time tt_{jk} of job J_{jk} to the needed total time of the chosen flow shop, F_i; that is:

$$f_i = f_i + tt_{jk}.$$

Step 10: Remove job J_{jk} from the job list.

Step 11: Repeat Steps 7 to 10 until the job list is empty.

After Step 11, jobs are clustered into p groups and are allocated to the p machine flow shops.

Part 3: Dealing with job sequencing in each flow shop

Step 12: For each flow shop F_i, set the initial completion time of the machines f_{ji} (j = 1 to m, i =1 to p) to zero.

Step 13: Find the completion time of each flow shop f_i by the Palmer algorithm stated in Section 2.

Step 14: Find the final completion time $ff = \max_{i=1}^{p}(f_i)$ among the completion time of all the flow shops and save the corresponding job sequence.

Step 15: Set $k = k + 1$.

After Step 15, the individual job sequence for each group has been found.

Level 2: Determining group sequence in the whole schedule

Step 16: Set the processing time mc_{jk} needed for the n jobs in group k on machine center j (j = 1 to m, k = 1 to l) as:

$$mc_{jk} = \max_{i=1}^{p}(f_{ijk}) - \min_{i=1}^{p}(c_{(j-1)ik}),$$

where f_{jik} is the completion time in each flow-shop i at machine center j for group k and $c_{(j-1)ik}$ is the completion time of the first job in each flow-shop i at machine center j-1 for group k.

Step 17: Find the group sequence by the Palmer algorithm stated in Section 2.

Step 18: Schedule the groups based on the above group sequence and with the job sequence of each flow-shop in each group to find the final completion time.

After Step 18, scheduling is finished and a total completion time has been found.

5. An Example for the Proposed Heuristic Algorithm

Assume there are three groups and each of them has five jobs, J_{1i} to J_{5i} (i = 1 to 3). Also assume each job has three tasks to be scheduled via three operations. Each operation is executed by a machine at the corresponding machine center. Each machine center includes two homogeneous machines. Assume the execution times of these jobs are listed in Table 1. The algorithm proceeds as follows.

Table 1: Processing times for the three groups of jobs

	G_1					G_2					G_3				
	J_{11}	J_{21}	J_{31}	J_{41}	J_{51}	J_{12}	J_{22}	J_{32}	J_{42}	J_{52}	J_{13}	J_{23}	J_{33}	J_{43}	J_{53}
t_{1j}	9	4	7	6	9	6	4	7	2	5	8	2	8	5	9
t_{2j}	7	8	3	2	9	3	2	6	7	3	7	5	2	7	3
t_{3j}	8	7	3	4	6	3	2	5	9	4	3	2	6	3	3

The steps in level 1 of the proposed algorithm determine the job sequence in each of the three groups. They are decomposed into three parts. Part 1 first forms two machine groups, F_1, F_2, each of which is thought of as a three-machine flow-shop. Part 2 then, for each job group, assigns the jobs to the machine groups. Results for this example are shown in Table 2.

Table 2. The jobs allocated to each flow shop for each job group

	G_1	G_2	G_3
$Flowshop_i$	*Jobs allocated*		
F_1	J_{51}, J_{31}, J_{41}	J_{42}, J_{52}	J_{33}, J_{43}, J_{23}
F_2	J_{11}, J_{21}	J_{32}, J_{12}, J_{22}	J_{13}, J_{53}

Part 3 then deals with job sequencing in each flow shop in each group. The results are shown in Table 3.

Table 3. The job sequence in each flow shop in each group

		G_1	G_2	G_3
Job sequence	F_1	J_{41}, J_{51}, J_{31}	J_{42}, J_{52}	J_{23}, J_{43}, J_{33}
	F_2	J_{21}, J_{11}	J_{32}, J_{22}, J_{12}	J_{13}, J_{53}

The steps in level 2 are then executed to determine the group sequence in the whole schedule. The processing time for each group of jobs at each machine center is first calculated and shown in Table 4.

In Table 4, the processing time for processing the first tasks of all the jobs in Group 1 at machine center 1 is 22, for processing the second tasks at machine center 2 is 23, and for processing the third tasks at machine center 3 is 25. Similarly, the processing time evaluated for Group 2 is 17, 18, and 14, respectively, and for Group 3 is 17, 18, and 16, respectively. The Palmer procedure is then used to schedule the three groups according to the processing time at each machine center. The obtained group sequence for this example is G_1, G_3, G_2. All the groups of jobs are then scheduled according to the above group sequence together with its job sequence in each flow shop. The final completion time is 62.

Table 4. The processing time of each group of jobs at each machine center

Machine Center	G_1	G_2	G_3
	Processing Time		
Machine Center 1	22	17	17
Machine Center 2	23	18	18
Machine Center 3	25	14	16

6. Experiments

This section reports on experiments made to show the performance of the proposed scheduling algorithms. They were implemented by Visual C++ at an Intel Pentium 4 CPU 2.40GHz. Two parameters are considered, the group number l and the job number of each group n. In the first case, the group number l is fixed at 3 and the job number of each group varies from 3 to 7. In the second case, the group number l varies from 3 to 8 and the job number n of each group is fixed at 6. Each job has three tasks and each machine center has two homogeneous machines. The execution time of each task was randomly generated in the range of 5 to 50. Each set of problems was executed for 20 tests and the makespans and computation times were measured. The optimal approach did not work for more than three groups with seven jobs for the first case and for more than eight groups with six jobs for the second case in our environments due to the large amount of computation time.

The optimal approach considered all possible permutations and combinations and used pruning techniques to increase its efficiency. The makespans obtained in this way were optimal. As for the first case, the group number is 3. The average makespans for problems of three to seven jobs in each group by the proposed method and the optimal one are shown in Figure 1.

The deviation rates of the proposed heuristic algorithm over the optimal algorithm for different numbers of jobs in each group when the group number l is 3 are shown in Table 5.

When the group number l is 3, the average CPU times for problems of three to seven jobs in each group are shown in Figure 2. The algorithm for optimal solutions cannot run over three groups of seven jobs in each due to the high time complexity.

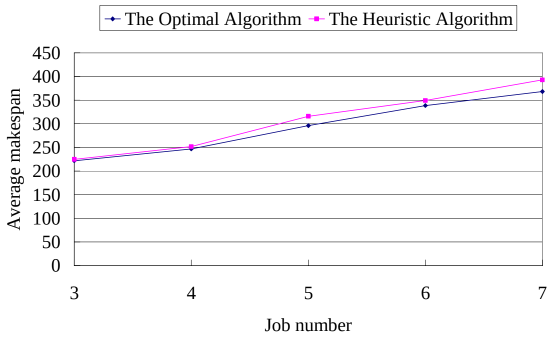

Fig. 1 Average makespans for the group number l = 3 with n = 3 to 7.

Table 5. The deviation rates for different numbers of jobs when the group number is 3

Group number l = 3	
	The Heuristic Algorithm
Job number	*Deviation rate* (%)
3	1.38
4	2.07
5	6.65
6	3.06
7	6.65

When the group number l is 3, the average CPU times for problems of three to seven jobs in each group are shown in Figure 2. The algorithm for optimal solutions cannot run over three groups of seven jobs in each due to the high time complexity.

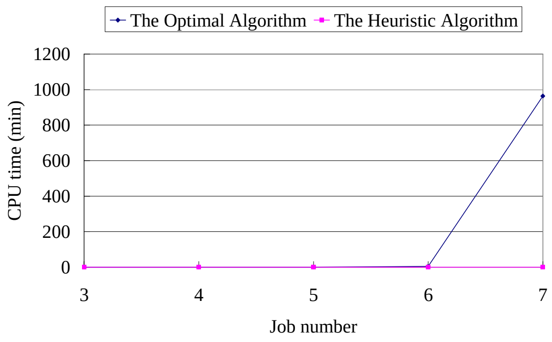

Fig. 2 The average CPU times for different numbers of jobs with l = 3.

Next, in the second set of experiments, the job number n of each group is 6. The average makespans for problems of three to eight groups by the proposed method and the optimal one are shown in Figure 3.

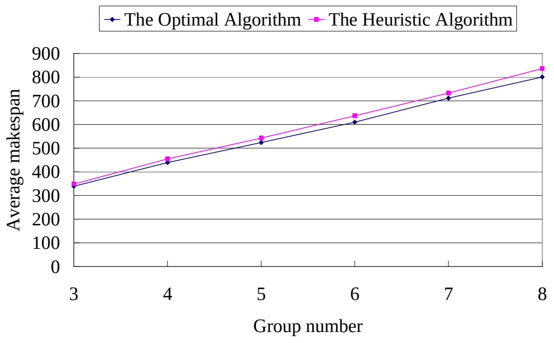

Fig. 3 The average makespans for the job number n = 6 with l = 3 to 8.

The deviation rates of the proposed heuristic algorithm over the optimal algorithm for different numbers of groups with n = 6 are shown in Table 6.

Table 6. The deviation rates for different numbers of groups with n = 6

Job number of each group n = 6	
	The Heuristic Algorithm
Group number	*Deviation rate* (%)
3	3.05760709
4	3.669097539
5	3.590184283
6	4.408751946
7	3.115549617
8	4.459434139

When the job number n of each group is 6, the average CPU times for problems of three to eight groups are shown in Figure 4. The optimal algorithm cannot run over eight groups in this case due to its high time complexity.

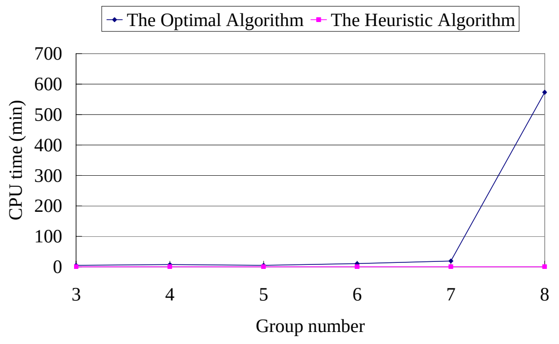

Fig. 4 Average CPU times for different numbers of groups with n = 6.

From the above figures and tables, it is easily seen that the proposed algorithm got only a little larger makespans than the optimal one did. The computational time needed by the optimal algorithm was, however, much larger than that needed by the proposed approach, especially when the job number is large. Actually, since the group flexible flow-shop problem is an NP-hard problem, the optimal approach can work only for a small number of jobs. The proposed approach can solve this problem.

Furthermore, experiments for large job numbers and group numbers were also made to show the performance of the heuristic algorithm. Experiments were made respectively for *n* from 1000 to 9000 with the group number *l* being 10, *n* from 1000 to 9000 with *l* being 100, *n* being 10 with *l* from 1000 to 9000, and *n* being 100 with *l* from 1000 to 9000. The average CPU times for the above cases are shown respectively in Figure 5 to 8, all being solved within a minute. Hence, the proposed approach is feasible even for a large number of jobs.

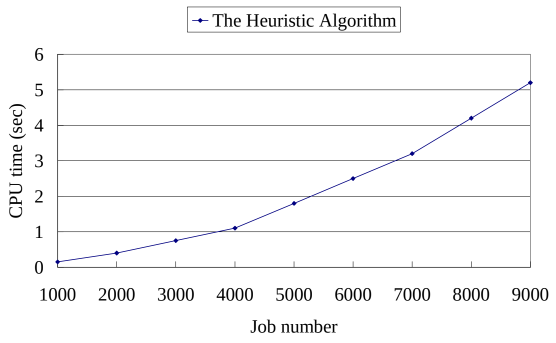

Fig. 5 The average CPU times for *l* = 10 and *n* = 1000 to 9000.

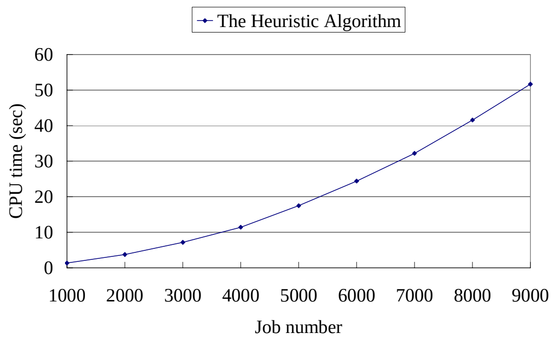

Fig. 6 The average CPU times for *l* = 100 and *n* = 1000 to 9000.

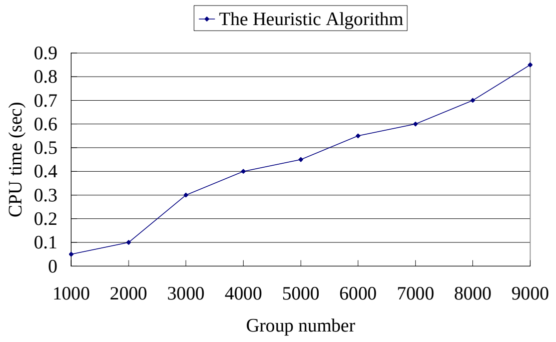

Fig. 7 The average CPU times for *n* = 10 and *l* = 1000 to 9000.

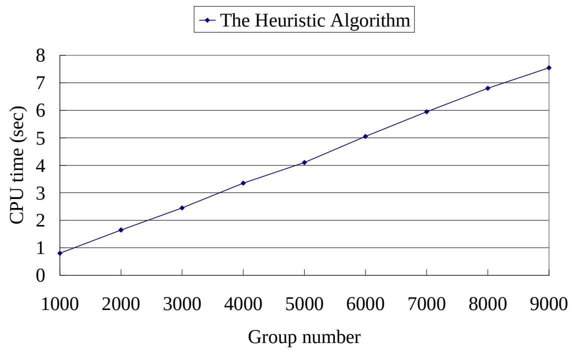

Fig. 8 The average CPU times for *n* = 100 and *l* = 1000 to 9000.

7. Conclusion

Appropriate scheduling cannot only reduce manufacturing costs but also reduce the possibility of violating due dates. Finding good schedules for given sets of jobs can thus help factory supervisors control job flows and provide for good job sequencing.

Scheduling jobs in the group flexible flow shops is an NP-complete problem. In this paper, we propose one algorithm to solve group flexible flow-shop problems with more than two machine centers, which have the same number of machines. The proposed one is a heuristic algorithm. It first determines the job sequencing in each group by combining both LPT and Palmer approaches to solve flexible flow-shop problems of more than two machine centers. It then determines group sequencing in the entire schedule by the Palmer algorithm. It is compared with the optimal one, which entirely uses the dynamic programming technique. The optimal algorithm works only when the job number is small. Experimental results show that the proposed algorithm can save much computational time than the optimal one although the obtained makespans by the former may be a little larger than the latter. A trade-off can thus be achieved between accuracy and time complexity. In the future, we will consider other task constraints, such as setup times, due dates, and priorities.

Acknowledgments

This research was supported by the National Science Council of the Republic of China under contract NSC 94-2213-E-390-005.

References

[1] S. C. Chung and D. Y. Liao, "Scheduling flexible flow shops with no setup effects," ***The 1992 IEEE International. Conference on Robotics and Automation***, pp. 1179-1184, 1992.

[2] R. A. Dudek, S. S. Panwalkar and M. L. Smith, "The lessons of flowshop scheduling research," ***Operations Research***, Vol. 40, pp. 7-13, 1992.

[3] T. P. Hong, C. M Huang and K. M. Yu, "LPT scheduling for fuzzy tasks," ***Fuzzy Sets and Systems,*** Vol. 97, pp. 277-286, 1998.

[4] R. Logendran and N. Nudtasomboon, "Minimizing the makespan of a group scheduling problem: a new heuristic," ***International Journal of Production Economics,*** Vol. 22, pp. 217-230, 1991.

[5] T. E. Morton and D. W. Pentico, Heuristic Scheduling Systems with Applications to Production Systems and Project Management, John Wiley & Sons Inc., New York, 1993.

[6] D. S. Palmer, "Sequencing Jobs Through a Multi-Stage Process in the Minimum Total Time- A Quick Method of Obtaining a Near Optimum," ***Operational Research Quarterly***, Vol. 16, pp. 101-107, 1965.

[7] V. A. Petrov, "Flow line group production planning," ***Business Publications***, London, 1966.

[8] J. Schaller, "A new lower bound for the flow shop group scheduling problem," ***Computers and Industrial Engineering***, Vol. 41, No. 2, pp. 151-161, 2001.

[9] C. Sriskandarajah and S. P. Sethi, "Scheduling algorithms for flexible flow shops: worst and average case performance," ***European Journal of Operational Research***, Vol. 43, pp. 143-160, 1989.

[10] D. L. Yang, M. S. Chern, "Two-machine flowshop group scheduling problem," ***Computers & Operations Research***, Vol. 27, No. 10, pp. 975-985, 2000.