Journal of the Korean Institute of Industrial Engineers
Vol. 32, No. 4, pp. 373-381, December 2006.

Scheduling Methods for a Hybrid Flowshop with Dynamic Order Arrival

Geun-Cheol Lee†

College of Business Administration, Konkuk University

주문 생산 방식을 따르는 혼합 흐름 공정에서의 일정계획에 관한 연구

이 근 철

건국대학교 경영대학 경영·경영정보학부 경영학전공

This paper considers a scheduling problem for a hybrid flowshop with dynamic order arrival. A hybrid flowshop is an extended form of a flowshop, which has serial stages like a flowshop but there can be more than one machine at each stage. In this paper, we propose a new method for the problem of scheduling with the objective of minimizing mean tardiness of orders which arrive at the shop dynamically. The proposed method is based on the list scheduling approach, however we use a more sophisticated method to prioritize lots unlike dispatching rule-based methods. To evaluate the performance of the proposed method, a simulation model of a hybrid flowshop-type production system is constructed. We implement well-known dispatching rules and the proposed methods in the simulation model. From a series of simulation tests, we show that the proposed methods perform better than other methods.

Keywords: Scheduling, Hybrid Flowshop, Simulation

1. Introduction

In this paper, we consider a scheduling problem of a hybrid flowshop (HFS), which is also known as a flexible flow line or a flowshop with parallel machines. An HFS can be regarded as a traditional flowshop except that each stage can have more than one machine. Adding a machine for a certain operation is a natural method of increasing the capacity of the operation (Gupta *et al*. 1997). For this reason, in real industries, HFSs are more commonly seen than traditional flowshops (Huang and Li 1998). Especially, in electronic manufacturing industries, such as semiconductor and printed circuit board manufacturing companies, HFS-type production lines are often seen and such examples are given in the recent literatures (Quadt 2004 ; Lee *et al*. 2004 ; Alisantoso *et al*. 2003 ; Lee *et al*. 2003 ; Jin *et al*. 2002). In this paper, we consider a typical HFS, which is composed of K serial stages each stage consisting of identical parallel machines. <Figure 1> shows a schematic view of an HFS.

Due to its practical importance, a great deal of research has been performed on the scheduling of HFSs and recent reviews in Wang (2005), Kis and Pesch (2005), and Quadt (2004) show that numerous literatures

This work was supported by the Korea Research Foundation Grant funded by Korea Government (MOEHRD, Basic Research Promotion Fund) (KRF-2003-214-D00309)

† Corresponding author : Geun-Cheol Lee, 1 Hwayang-dong Gwangjin-gu Seoul 143-701, Korea,
Tel : 02-450-4100, Fax : 02-3436-6610, E-mail : gclee@konkuk.ac.kr

Received November 2005 ; revision received July 2006; accepted August 2006

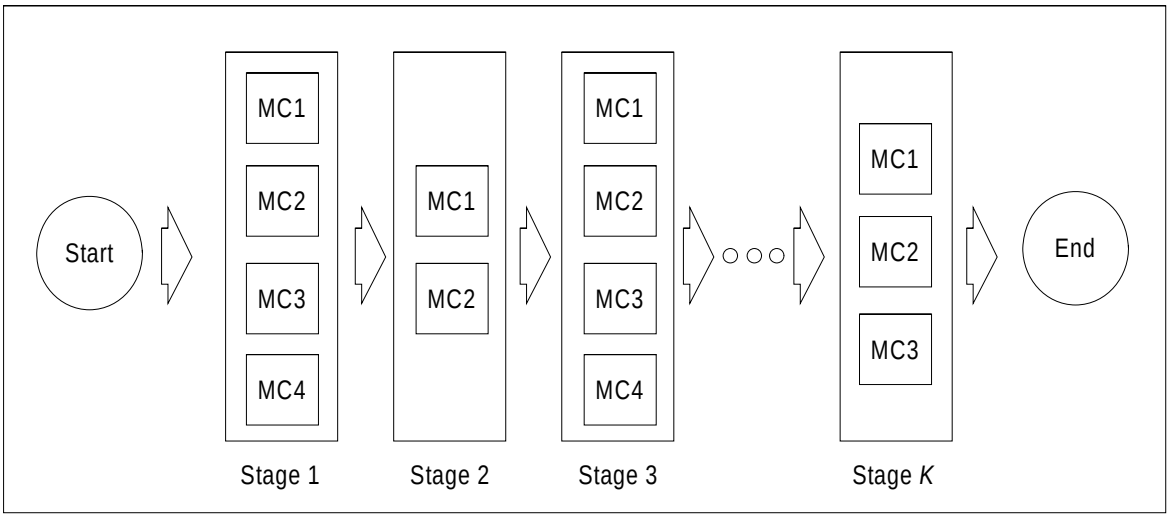

Figure 1. A schematic view of an HFS

related with HFS scheduling problems. Although most of this work was done on two-stage HFS, the number of studies dealing with K-stage ($K \geq 3$) HFS has been increasing in the last decade. However, much of the research is concerned with throughput-related measures, like minimizing makespan or mean flow time (Guinet and Solomon 1996 ; Brah and Loo 1999 ; Moursli and Pochet 2000 ; Azizoglu *et al*. 2001, Cheng *et al*. 2001). Even though delivering products to customers on time is one of the most pursued goals in today's market, there are not many K-stage ($K \geq 3$) HFS scheduling studies concerned with the due date-related measures. Besides, because of its complexity, dispatching rule based methods are mainly used in such studies (Bengü 1994 ; Brah 1996 ; Lee *et al*. 2003).

Along with the due date considerations, occurrence of changeover operations is a major element that makes HFS scheduling problems more complicated. For such reasons, not many researchers have worked on this issue. There is a long blank period between the early study by Kochhar and Morris (1987), in which changeovers between jobs are allowed on their flexible flow line, and the study by Huang and Li (1998). Recently, Kurz and Askin (2003) proposed heuristics methods for a flexible flow line with sequence-dependent changeover times and Lee *et al*. (2003) suggested a dispatching rule-based method for lot sizing in a printed circuit board production line. However, most of these studies do not deal with dynamic order arrival. That is, all jobs to be processed are given at the beginning of the scheduling horizon in most previous studies. Consequently, to the best of our knowledge, there is no study of HFS scheduling problems considering all these characteristics (dynamic arrival, due date commitment, and changeover times), even though covering such characteristics in a formulation could make the problem more practical. In this study, we focus on the HFS scheduling problem with the objective of minimizing mean tardiness of orders, while orders arrive dynamically and changeover operations between different product types of lots occur.

We present two types of heuristic algorithms to address the problem. Both of them are based on the list scheduling approach. One uses traditional dispatching rules and the other uses a new method based on an independent parallel machine scheduling algorithm. In the next section, the considered HFS scheduling problem is described in more detail and then we introduce two approaches of the scheduling for the problem in section 3. For evaluating the scheduling methods used in this study, a simulation model was developed and experiments were performed, which are presented in section 4 with the results. The paper concludes with a summary and extensions of the research.

2. Problem Description

In this study, we assume orders arrive dynamically in the shop. An order is defined by a product type, a due date, and an order size, which is the number of products to be processed in the order. When an order arrives at the shop, the order is generally divided into multiple lots by considering its appropriate size of transfers and processes so that products in the same lot

move together and are processed without interrupt on machines. Every lot released into the shop travels from the first stage to the last stage according to its process flow.

The number of products in one lot is called the lot size of the product type. In this study, the lot sizes of all product types are assumed to be one. In real PCB or semiconductor manufacturing line a lot can be split into several sub-lots at certain stages, lots in this study could be regarded as the smallest sub-lots in practical systems so that they can not be split any longer. The processing time of a lot at each stage can be known from the product type of its order, because the lot size and the processing time of each product type at each stage are pre-determined values. Thus, in the considered scheduling problem, processing of a lot can be regarded as a job and processing of a lot at each stage may be regarded as an operation.

All lots from an order are released into the shop at the same time, but completion times of lots from the order can be different from one another. An order is regarded as completed when the last remaining lot of the order is finished at the last stage of the shop, so we need to control all lots of an order to meet the due date of the order. For managers' point of view, meeting each order's due date would be considered as the most important scheduling factor in terms of customer satisfaction. Therefore, in this paper, we choose minimizing the mean tardiness of orders as the objective of the scheduling problem. Here, the tardiness of an order, o (T_o), is defined as $\max\{0, C_o - D_o\}$, where C_o and D_o are the completion time and the due date of order o, respectively.

One feature that intricates the considered problem is the occurrences of the changeover operations. The changeover operation is needed when a product type of a lot to be processed next is different from the one just processed on the same machine. The changeover time of a lot at each stage can be known from its product type. Further assumptions in this study are: an infinite buffer at each stage; no transportation time between stages; no machine breakdown; and no job split or preemption.

3. Scheduling Methods

In practice, list scheduling approaches with dispatching rules, which can be implemented easily, are commonly used for operations scheduling. In the list scheduling approaches, when a machine becomes available, the lot that has the highest priority among the waiting lots for the machine is selected and loaded on the machine for the next processing. Generally, the priorities of waiting lots are calculated by a pre-specified dispatching rule right before selection of the lot to be processed. Therefore, the priorities can be obtained based on up-to-date information of the system status when a list scheduling approach with a dispatching rule is used. In this paper, we call this kind of list scheduling approach, which uses dispatching rules for prioritizing, the Machine View Prioritizing Method (MVPM), because a lot is selected from the machine's point of view. On the other hand, we suggest an approach, called Order View Prioritizing Method (OVPM), which has a different notion to MVPM. In OVPM, priorities of waiting lots at a certain stage are determined when a new order arrives at the stage, i.e. waiting lots are prioritized at the order's point of view. In next two subsections, we present these two approaches in detail.

3.1 Machine View Prioritizing Method (MVPM)

The Machine View Prioritizing Method (MVPM) is a typical list scheduling method using dispatching rule. In this paper, we contrast this with the Order View Prioritizing Method (OVPM). Two approaches are similar in that a lot is selected according to its priority when a machine becomes available. However, they have a different time point for prioritizing the waiting lots. MVPM prioritizes the lots when a machine becomes available, while OVPM prioritizes them when a new order arrives at a corresponding stage.

The followings are well-known due date-related dispatching rules used for MVPM in this study. Each rule selects a lot that minimizes the quantity shown in parentheses. Here, we call this quantity as the priority value. That is, the higher the priority, the lower the priority value. In the followings, d_i denotes the due date of the order associated with lot i, t is the time at which the dispatching rules are applied, and r_i is the remaining work of lot i.

EDD (Earliest Due Date):
Select a lot with the earliest due date (d_i).

SLACK:
Select a lot with the least slack ($d_i - r_i - t$).

MDD (Modified Due Date):
Select a lot with the minimum modified due date ($max\{d_i, t + r_i\}$).

When obtaining the remaining work of a lot in SLACK and MDD, we actually calculate the remaining work of the order corresponding with the lot, which induces all the lots from the order to have the same priority values. As a result, all lots from one order could be processed consecutively and this compact-processing fashion of an order might give better performance (Choi *et al*. 2005). For more details on dispatching rules, see Brah (1996).

In this study, the rules introduced above are not applied as themselves. Each rule is slightly modified as following such that it can consider occurrences of changeover operations. That is, a big penalty value is added on the priority value of a lot if the lot causes a changeover operation. With this modification, we rename the above rules as SEDD, SSLACK, and SMDD. For example, when we use SEDD for the prioritizing, the priority value of a lot which causes a changeover operation would be '$100 \times t + d_i$' while that of a lot which does not is just 'd_i'. Therefore, a lot causing a changeover operation will have a lower priority in dispatching than lots which do not cause changeover operations. In this fashion, lots with the same product type would flow consecutively in the system, which could eventually reduce the number of changeover occurrences.

In addition to the above three rules, we use the ATCS rule which was proposed by Lee and Pinedo (1997). In this study, the priority function of the ATCS, which was originally devised for parallel machine scheduling with sequence-dependent setup times, is modified as

$$-\frac{1}{r_i} \cdot \exp\left(-\frac{\max(d_i - r_i - t, 0)}{k_1 \bar{r}}\right) \cdot \exp\left(-\frac{s_{ik}}{k_2 \bar{s}}\right),$$

where s_{ik} is the processing time and changeover time of the product type of lot i at stage k and , $\bar{r}$ and $\bar{s}$ are the average remaining work and changeover times, respectively. Two parameters k_1 and k_2, which play important roles in ATCS, are determined as the way Lee and Pinedo introduced in their paper. Like the above three rules, ATCS selects a lot with the minimum value from its priority function.

3.2 Order View Prioritizing Method (OVPM)

In this subsection, we introduce Order View Prioritizing Method (OVPM), the suggested approach for the problem. First, we explain when the lots are prioritized in OVPM and then present how the priorities of the lots are determined.

3.2.1 Time of Prioritizing

In OVPM, like MVPM, a lot is selected to be processed according to its priority value from a machine which becomes available. However, unlike MVPM, priority values of lots at a certain stage are not calculated when a machine becomes available at the stage, but those are calculated when a new order arrives at the stage. An order is regarded as a new order at a certain stage, when one lot of the order first arrives at the stage.

Compared to the frequency of the prioritizing in the MVPM, that in the OVPM is much less, because the number of the prioritizing executions by MVPM is approximately the number of lots released in the shop times the number of stages while for OVPM it is approximately the number of orders arrived at the shop times the number of stages. Although the frequency of the prioritizing in OVPM is much smaller than that in MVPM, which means OVPM uses less up-to-date information than MVPM, this disadvantage of OVPM is covered by the more sophisticated method of prioritizing such as solving an independent parallel machine scheduling problem at the stage, which is presented next.

3.2.2 Method of Prioritizing

To prioritize all lots at a certain stage, we solve a parallel machine scheduling problem (PMSP) at the stage whenever a new order arrives at the stage. In the PMSP, a subset of orders including waiting orders and the newly arrived order at the stage is designated as a target set of orders to be scheduled. Among the lots from the target orders, the lots which have not passed the stage are only included in scheduling. Note that the goal of solving the PMSP is the prioritizing of lots waiting at the considered stage, i.e. deciding priority value of each lot waiting at the stage.

The processing and changeover times of product types in the PMSP are same as those at the stage in the original problem. However, the due date of each order is modified considering the remaining time of the order at the downstream stages. Although, most previous researchers used processing times of operations to obtain remaining time of an order at the downstream stages, i.e., the sum of processing times of operations

of the order at the downstream stages, remaining time of a order is not affected by processing times of its remaining operations but rather by amount of inventory at the downstream stages. In this paper, we use inventory data and production rate of the last stage to calculate the remaining time at the downstream stage. Note that we use the last stage's production rate only, because that is the production rate of the final product. Thus, the remaining time of the order at the downstream stages is calculated as, L_k/μ, where L_k is the number of lots which stay at the downstream stages of stage k and μ is the production rate of the last stage which can be estimated from real time statistics during the simulation, i.e., the number of completed lots divided by current simulation time. Finally, the modified due date of order o at stage k ($\hat{D}_{ok}$) for the PMSP can be obtained by $D_o - L_k/\mu$.

When a lot of new order arrives at a stage, a PMSP is generated. In this study, to solve the scheduling problem, our algorithm generates multiple alternative sequences of the target orders and evaluates the generated sequences, obtaining a schedule for each alternative sequence. After the evaluation, one sequence that gives the best performance, i.e. the minimum mean estimated tardiness of the target orders is selected. As a matter of course, the solution of the PMSP will be the schedule obtained by using the selected sequence of the orders. When generating alternative sequences, we use the concept of the algorithm by Nawaz *et al*. (1983). That is, we check possible insertion positions of the new order in the sequence of the waiting orders, which is given from the past. Thus, there can be N^o+1 alternative sequences if the number of the waiting orders is N^o at the stage. For each alternative sequence, list scheduling is done for the lots to obtain the implied schedule. We compute the mean estimated tardiness of the target orders for each alternative sequence and choose the one which gives the best value. Note that the calculated performance is an estimated value because we use modified due date for an order (o), $\hat{D}_{ok}$. To summarize the procedure of the method of prioritizing at a stage, we use the following notation: V_{ok} and v_{ik} are the priority value of order o and lot i at stage k, respectively.

Procedure (Method of prioritizing at stage k)

Step 0: (Data preparation) Compose the set of the target orders to be scheduled and calculate the modified due date, $\hat{D}_{ok}$, for all target order o. The sequence of the waiting orders is given as σ^*.

Step 1: (Search) Find the best insert position of new order, $p^* = \arg\min_p\{LS(\sigma_p)\}$ and its corresponding sequence σ_{p^*}, where σ_p is the sequence when new order is inserted at p-th position in sequence σ^*, and $LS(\sigma)$ is the function which returns mean estimated tardiness of the target orders when list scheduling is done with sequence σ.

Step 2: (Prioritizing) Set V_{ok} = the position of order o in sequence σ_p and $v_{ik} = V_{ok}$ for all lot i from order o.

As in MVPM, we additionally consider the changeover operation occurrence in OVPM. That is, a big penalty value is added on the priority of a lot which causing a changeover operation. Note that, the list scheduling in the procedure is for prioritizing, not for scheduling. Only the priority values of the lots are decided by the above procedure. One more list scheduling is needed to construct an actual schedule.

4. Simulation Experiment

We developed a simulation model and performed simulation tests to study the results of the introduced scheduling methods.

4.1 Data generation

4.1.1 Pre-specified data

- The number of stages (K) : 15, 30, 60.
- The number of product types : 50 for each shop size.

4.1.2 Randomly generated data

- The number of machines at stage k (m_k) : DU(1, 10), where $DU(L, U)$ is the discrete uniform distributions with range [L, U].
- The processing and changeover times of the product type of lot i at stage k (p_{ik} and s_{ik}) : $DU(1 \cdot m_k, 10 \cdot m_k)$ and $DU(1 \cdot m_k, 10 \cdot m_k)$.
- The interarrival time of orders : exponentially distributed (The arrival rate of orders are depend on the shop configurations).
- The quantity of order o (Q_o) : $DU(1, \overline{Q})$, where $\overline{Q}$ is a pre-specified parameter adjusting the range of the order size.
- The due date of order o (D_o) : $A_o + u \cdot P_o$, where

A_o is the time when order o arrives, u is a pre-specified due date tightness parameter, and P_o is an approximate production lead time of order o. P_o is calculated by $\sum_{k=1}^{K}(\hat{s}_{ok}+\hat{p}_{ok})+(Q_o-1)\cdot \hat{p}_{ob_o}/m_b$, where $\hat{s}_{ok}$ and $\hat{p}_{ok}$ are the processing time and changeover time of the product type of a lot from order o at stage k, respectively, and b_o, which can be estimated by $\{\arg\max_k(\hat{s}_{ok}+Q_o\cdot \hat{p}_{ok}/m_k)\}$, is the index of the bottleneck stage of order o.

4.2 Experiment design

For the experiment, simulation sets were generated as follows.

- Three levels for the order size range (*DU*(1, 20), *DU*(1, 50), and *DU*(1, 100), i.e. $\overline{Q}$ is 20, 50, and 100).
- Three levels for the number of stages (15, 30, and 60).
- Four levels for the due date tightness parameter *u* which is generated from *DU*(5, 7), *DU*(4, 8), *DU*(7, 9), and *DU*(8, 10).
- Five runs for each of the above combinations.
- Totally 180(=3 × 3 × 4 × 5) simulation sets were generated.

In each simulation set, a different shop configuration (the number of machines, and the processing and changeover times of product types at each stage) and a different order arrival rate were used.

For each simulation set, we made a run for the simulation time of 200,000 unit time and the first 50,000 unit time is considered as the warm-up period. If an order is not completed by the end of the simulation time, tardiness of the order is obtained with estimated completion time of the order, $\hat{C}$, which is calculated as $(I+q)/\mu$, where I is the amount of inventory preceding the order, q is the uncompleted quantity of the order, and μ is the production rate of the last stage. In this study, the 4 rules for MVPM and one for OVPM were tested for each problem set. The model was coded with C language and all tests were carried out on a Linux machine with AMD Athlon process of 1.8 GHz clock speed.

4.3 Test results and Analysis

Overall results of the test are given in table 1. The performance of the algorithms was shown with a relative measure, called the relative deviation index (RDI). The RDI is defined as $(S_a-S_b)/(S_w-S_b)$, where S_a is the solution value (mean tardiness) obtained by method a, and S_b and S_w are respectively the best and the worst solution values among those obtained by the methods tested. The method with the smaller RDI values can be regarded as the better method. <Table 1> shows average and standard deviation of RDIs of each method and the number of simulation runs for which each method found the best solution (NBS). Overally, OVPM showed outstanding performance as measured by both RDI and NBS. Although ATCS can make good schedules at one stage (i.e., parallel machines), its superiority seems not to be maintained at hybrid flowshops, especially in dynamic scheduling.

Table 1. The overall results of the test

Scheduling methods		RDI*		NBS**
MVPM	SEDD	0.298	(0.321)	73
	SSLACK	0.324	(0.342)	71
	SMDD	0.130	(0.190)	86
	ATCS	0.773	(0.415)	40
OVPM		0.015	(0.043)	145

* Average and standard deviation (in parenthesis) of relative deviation.

** Number of simulation runs (out of 180) for which each scheduling method found the best solutions.

To see the impact of the factors (i.e. the number of stages, the due date tightness, and the range of order size) on the performance, an analysis of variance (ANOVA) was done on the solution values (i.e. mean tardiness of orders), and results are given in table 2. The results showed that performance of the scheduling methods was significantly affected by the number of stages and the range of order size at the significance level of 0.01.

Since the ANOVA result showed that the performance was affected by the two factors, we summarized the test results for the different values of the factors in tables 3~4. From the tables, it can be seen that dominance of OVPM compared to the other methods. The range of the order size can be a major fact that can affect on the system performance, because the sizes of the generated orders are related with the frequency of the changeover operations in the system. Indeed, table 4 shows a wide discrepancy in results for the different ranges of the order size. Especially, when the order size range is wide, OVPM dominated other methods severely.

Table 2. Analysis of variance

Source of variation	Degree of freedom	Sum of squared error	Mean squared error	F
Scheduling methods	4	304173494	76043373	8.0*
Number of stages	2	545594686	2702797343	28.7*
Due date tightness	3	68281534	22760511	2.4
Order size range	2	7325352774	3662676387	385.9*
Error	888	8429248114	9492397	
Total	899	16672650602		

* Statistically different at the significance level of 0.01.

Table 3. Performance of the scheduling methods for different numbers of stages

Scheduling Methods		RDI†			NBS‡		
		15	30	60	15	30	60
MVPM	SEDD	0.386	0.329	0.180	18	24	31
	SSLACK	0.392	0.375	0.207	19	24	28
	SMDD	0.097	0.148	0.145	26	27	33
	ATCS	0.933	0.740	0.645	4	15	21
OVPM		0.021	0.011	0.012	45	51	49
NSR*		60	60	60	60	60	60

† Average relative deviation index.
‡ Number of simulation runs for which each method found the best solutions.
* Number of simulation runs.

Table 4. Performance of the scheduling methods for different ranges of the order size

Scheduling Methods		RDI			NBS		
		[1,20]	[1,50]	[1,100]	[1,20]	[1,50]	[1,100]
MVPM	SEDD	0.000†	0.335	0.560	58	13	2
	SSLACK	0.000	0.356	0.617	59	11	1
	SMDD	0.000	0.088	0.302	59	24	3
	ATCS	0.383	0.948	0.988	37	3	0
OVPM		0.009	0.029	0.007	48	42	55
NSR		60	60	60	60	60	60

† less than 0.000.

Table 5 shows the computation time of each method in the simulation run. Although the frequency of prioritizing in OVPM is much less than that in MVPM, the method of prioritizing in OVPM takes more time than that in MVPM. However, we could anticipate that the computation time of one execution of the proposed method is not too big, which encourages us to develop more sophisticated method that could consume more computation time in the future study. (Note that, the number of execution of the procedure of the prioritizing method is approximately 'the number of stages times the number of orders arrived' in one simulation run.)

Table 5. Computation time

Number of stages	Average CPU time (s)				
	MVPM				OVPM
	SEDD	SSLACK	SMDD	ATCS	
15	4.4	4.7	5.0	34.5	10.8
30	7.5	8.0	8.6	48.0	15.6
60	16.2	16.8	17.7	66.9	28.5

5. Conclusions

In this paper, we focused on a hybrid flowshop scheduling problem with the objective of minimizing mean tardiness of orders which arrive dynamically and the changeover operations are needed when lots of the different product types are processed consecutively on the same machine. We tested two types of scheduling methods for the hybrid flowshop scheduling problem using the simulation model. One is Machine View Prioritizing Method (MVPM), which is same as the typical list scheduling approach with the dispatching rules. Some dispatching rules are devised for changeover consideration. The other is Order View Prioritizing Method (OVPM) in which more sophisticated method of prioritizing is used than MVPM. To evaluate the performance of the considered scheduling methods, computational experiments were done with a simulation model. Various results of the experiments showed that an OVPM outperformed all MVPMs.

This research can be extended in several ways. First, one may devise a more sophisticate OVPM through expanding scope of considering orders or using different measures when deciding order sequence, which could result a better performance. Although, in this study, we took several practical matters into account such as dynamic arriving orders, changeover times, and due dates of orders, which have been seldom considered in the previous hybrid flowshop scheduling studies, there still remain other practical matters which might affect system performance, such as existence of unrelated parallel machines at stages, machine breakdown and maintenance times, different process batch sizes on some machines, finite buffer between stages and so on. Thus, research which considers those facts is also needed in the near future. Finally, through our research, it can be seen that accurate estimations of remaining times of jobs can be crucial in scheduling those jobs, but it is a very hard problem because it depends on many real-time environmental variables, such as inventory status around the jobs, and the scheduling method used. Better estimation methods of remaining times will guarantee more robust scheduling results. Furthermore, such a method can be directly applied to order promising problems (e.g. Available- To-Promise), which is becoming an essential part of production management, especially in make-to-order industries.

References

Alisantoso, D., Khoo, L. P., and Jiang, P. Y. (2003), An immune algorithm approach to the scheduling of a flexible PCB flow shop. *International Journal of Advanced Manufacturing Technology*, **22**, 819-827.

Azizoglu, M., Cakmak, E. and Kondakci, S. (2001), A flexible flowshop problem with total flow time minimization. *European Journal of Operational Research*, 132, 528-538.

Bengü, G. (1994), A simulation-based scheduler for flexible flowlines. *International Journal of Production Research*, **32**, 321-344.

Brah, S. A. (1996), A comparative analysis of due date based job sequencing rules in a flow shop with multiple processors. *Production Planning and Control*, **7**, 362-373.

Brah, S. A. and Loo, L. L. (1999), Heuristics for scheduling in a flow shop with multiple processors. *European Journal of Operational Research*, **113**, 113-122.

Cheng, J., Karuno, Y., and Kise, H. (2001), A shifting bottleneck approach for a parallel machine flowshop scheduling problem. *Journal of the Operations Research Society of Japan*, **44**, 140-156.

Choi, S-W., Kim, Y-D., and Lee, G-C. (2005), Minimizing total tardiness of orders with reentrant lots in a hybrid flowshop. *International Journal of Production Research*, **43**, 2049-2067.

Guinet, A. G. P. and Solomon, M. M. (1996), Scheduling hybrid flowshops to minimize maximum tardiness or maximum completion time. *International Journal of Production Research*, **34**, 1643-1654.

Gupta, J. N. D., Hariri, A. M. A., and Potts, C. N. (1997), Scheduling a two-stage flow shop with parallel machines at the first stage. *Annals of Operations Research*, **69**, 171-191.

Huang, W. and Li, W. (1998), A two-stage hybrid flowshop with uniform machines and setup times. *Mathematical and Computer Modelling*, 27, 27-45.

Jin, Z. H., Ohno, K., Ito, T., Elmaghraby, S. E. (2002), Scheduling hybrid flowshops in printed circuit board assembly lines. *Production and Operations Management*, **11**, 16-230.

Kis, T. and Pesch, E. (2005), A review of exact solution methods for the on-preemptive multiprocessor flowshop problem. *European Journal of Operational Research*, **164**, 592-608.

Kochhar, S. and Morris, R. J. T. (1987), Heuristic methods for

flexible flow line scheduling, *Journal of Manufacturing Systems*, **6**, 299-314.

Kurz, M. E. and Askin, R. G. (2003), Comparing scheduling rules for flexible flow lines. *International Journal of Production Economics*, **85**, 371-388.

Lee, G-C., Kim, Y-D., Kim, J-G., and Choi, S-H. (2003), A dispatching rule-based approach to production scheduling in a printed circuit board manufacturing system. *Journal of the Operational Research Society*, **54**, 1038-1049.

Lee, G-C., Kim, Y-D., and Choi, S-W. (2004), Bottleneck-focused scheduling for a hybrid flowshop. *International Journal of Production Research*, **42**, 165-181.

Lee, Y. H. and Pinedo, M. (1997), Scheduling jobs on parallel machines with sequence-dependent setup times. *European Journal of Operational Research*, **100**, 464-474.

Moursli, O. and Pochet, Y. (2000), A branch-and-bound algorithm for the hybrid flowshop. *International Journal of Production Economics*, **64**, 113-125.

Nawaz, M., Enscore, E., and Ham, I. (1983), A heuristic algorithm for the m-machine, n-job flowshop sequencing problem. *OMEGA, The International Journal of Management Science*, **11**, 91-95.

Quadt, D. (2004), *Lot-sizing and scheduling for flexible flow lines, Lecture notes in economics and mathematical systems*, Springer, Berlin Germany.

Wang, H. (2005), Flexible flow shop scheduling: optimum, heuristics and artificial intelligence solutions, *Expert Systems*, **22**, 78-85.