Bulletin of the JSME

Scheduling jobs with limited waiting time property on flexible flowshop

--Manuscript Draft--

Manuscript Number:	
Full Title:	Scheduling jobs with limited waiting time property on flexible flowshop
Article Type:	Research Paper (JSME Journals)
Section/Category:	Journal of Advanced Mechanical Design, Systems, and Manufacturing (Manufacturing System)
Manuscript Classifications:	Product Planning (製品企画); Production System (生産システム一般); Operation Planning (作業設計); Production Management (生産管理); Production Planning (生産計画)
Corresponding Author:	BongJoo Jeong, Ph.D. Kongju National University 공주시, 충남 — Chungcheongnam-do KOREA, REPUBLIC OF
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	Kongju National University
Corresponding Author's Secondary Institution:	
First Author:	Sang-Oh SHIM
First Author Secondary Information:	
Order of Authors:	Sang-Oh SHIM
	BongJoo Jeong, Ph.D.
	June-Yong BANG
	JeongMin PARK
Order of Authors Secondary Information:	
Abstract:	In this paper, we consider a 2-stage hybrid flowshop scheduling problem with identical parallel processors in each stage. It is assumed that a queue(Q)-time of each job, which is a waiting time to be processed in the current stage, should be less than a predetermined time due to quality issue of the product. To minimize makespan of jobs, we developed several methodologies to solve the scheduling problem. The results of the simulation experiments of the suggested ones are also presented and show better performance than the existing ones.
Additional Information:	
Question	**Response**
1.Are you a member of the JSME ?	No
1-2.Are you a member of The Heat Transfer Society of Japan ? as follow-up to "1.Are you a member of the JSME ?"	No
2.Which category do you choose?	Design, Machine Element & Tribology, Information & Intelligent Technology, Manufacturing, and Systems
3.Do you submit a special issue ?	No

Scheduling jobs with limited waiting time property on flexible flowshop

Sang-Oh SHIM*, BongJoo JEONG **, June-Yong BANG***, JeongMin PARK*
*Department of Business Administration, Hanbat National University,
34158, Dongseo-daero 125, Yuseong-gu, Daejeon, Republic of Korea
** Department of Business Administration, Kongju National University,
32588, Gongjudaehak-ro 56, Gongju-si, Chungcheongnam-do, Republic of Korea
Email: jbj@kongju.ac.kr
*** Department of Industrial and Management Engineering, Sungkyul University,
14097, Sungkyul University-ro 53, Manan-gu, Anyang-si, Gyeonggi-do, 14097, Republic of Korea

Abstract
In this paper, we consider a 2-stage hybrid flowshop scheduling problem with identical parallel processors in each stage. It is assumed that a queue(Q)-time of each job, which is a waiting time to be processed in the current stage, should be less than a predetermined time due to quality issue of the product. To minimize makespan of jobs, we developed several methodologies to solve the scheduling problem. The results of the simulation experiments of the suggested ones are also presented and show better performance than the existing ones.

Keywords : Scheduling, Hybrid flowshop, Limited waiting time, Time window

1. Introduction

Scheduling problems on flexible(hybrid) flowshop can be found easily in many manufacturing environments. Usually, in *k*-stage hybrid flowshop scheduling problems, there are *n* independent jobs consisting of *k* operations and they should be processed on a machine in each stage. In *j*-th stage (for $j = 1,\ldots k$), there are $m_j(\geq 1)$ parallel machines which are identical, uniform or unrelated. In this research, we consider 2-stage hybrid flowshop scheduling problems with queue(Q)-time sensitive constraints in which the operation of each job on second stage should be started within a predetermined time after finishing the previous (1^{st} stage) operation of the job.

There are lots of this kind of a problem in many manufacturing systems, especially semiconductor manufacturing fabrications. Examples include diffusion workstation within semiconductor fabrication. In the most diffusion workstation of 300mm (12″) sized-wafer fabrication, there are 2 or 3 stages (diffusion chamber, cleaning and etching) with identical (or non-identical) parallel machines and wafers are processed on a machine of each stage. Also, as a semiconductor technology has been developed, the distance between circuits on the surface of semiconductor wafer has become narrower. Consequently, it is very important to control particles which are critical for the quality of wafers since the sizes of these particles are greater than the distance between circuits. Even though the particles in the air of semiconductor manufacturing fabrications are strictly controlled, they sometimes cling to the wafers.

The one of the methods to avoid that particles cling to the surfaces of wafers is that the next process starts in a limited time after the current process is done. The more products wait for the next operation, the more particles cling to them. Therefore, the product which completes the current process should be processed for the next operation as soon as possible. Especially, since in the diffusion process, particles can cling more easily than other workstations, many manufacturing managers usually determine the time-duration and they want wafers to be processed in that duration after the current process is done. This time-duration is determined empirically by operators or determined by using statistical method and is called as limited waiting time or queue(Q)-time. In this study, the 2-stage hybrid flowshop

scheduling problem with the property of the limited waiting time between stages is addressed to minimize makespan of jobs.

As surveyed by Linn and Zhang (1999), Ribas et al. (2010), Tosun et al. (2020), there are lots of studies on hybrid flowshop scheduling problems, including Brah and Hunsucker (1991), Rajendran and Chaudhuri (1992), Gupta et al. (1997), Gupta (1998), Soewandi (1998), Moursli and Pochet (2000), Azizoglu et al. (2001), Lee and Kim (2003), Dugardin et al. (2010), Gomez-Gasquet et al. (2012), Eskandari and Hosseinzadeh (2014), Fernandez-Viagas et al. (2018) and Han et al. (2019). For the 2-stage hybrid flowshop scheduling problem with arbitrary waiting times to minimize makespan, branch and bound (B&B) algorithms are suggested by Yang and Chern (1995) and Bonquard and Lente (2006), developing dominance properties, lower bound schemes and heuristics for obtaining upper bounds. Besides, several optimal solution properties and a B&B algorithm where $k \geq 2$ are suggested by Fondervelle et al. (2006) and Bouquard et al. (2006) present some properties by using Max-Plus algebra. Finally, a heuristic algorithm for a 2-stage hybrid flowshop (in which there are one batch processor in stage 1 and one single processor in stage 2) scheduling problem with waiting times is devised by Su (2003). Since then, several studies on the flowshop with waiting time have been studied. (Ruiz and Vázquez-Rodríguez, 2010, Behnamian and Fatemi Ghomi, 2011, Ying and Lin, 2018, Öztop et al., 2019, Lin et al., 2021)

A lot of research has been done on hybrid flowshop, while relatively few have been done on the scheduling problems with waiting time constraints in the field of scheduling problems of flowshop. A special case assuming that waiting times of all jobs in each stage are infinite, is the well-known 2-machine flowshop scheduling problem, which can be solved optimally by the Johnson's algorithm (1954) to minimize makespan of jobs. Also, if all the waiting times should be zero, this problem can be reduced to the no-wait two-machine flowshop scheduling one and optimal solution can be obtained in polynomial time (Gilmore and Gomory, 1964). On the other hand, Yang and Chern (1995) proved that the problem with arbitrary waiting times is NP-hard. Recently, An et al. (2016) developed a branch-and-bound algorithm for a two-machine flowshop problem with a limited waiting time and sequence-dependent setup times. Lee (2020) and Jeong et al. (2021) developed meta-heuristic algorithms for two-machine flowshop scheduling problem with limited waiting time constraints. However, considered problem in this study is not handled yet to the best of our knowledge.

In this paper, we suggest several heuristics for 2-stage hybrid flowshop (consisting of identical parallel machines in each stage) scheduling problems with the limited waiting time constraints to minimize makespan. This research is organized as follows. In the next section, this problem considered here is described clearly and formulated as a mathematical programming. Then, we develop some optimal solution properties in this study. To evaluate performance of the suggested algorithms, computational experiments on randomly generated problems are conducted in the result section and results are reported. Finally, we conclude the research and suggest further research.

2. Problem description

In the problem under the consideration, there are n independent jobs to be processed through 2 stages which consist of M1 and M2 identical parallel machines, respectively. In this problem, it is assumed that (i) at time zero, all of the jobs are available; (ii) each processing time is given; (iii) no setup time is required; (iv) only one job can be processed in each machine at a time, and only one machine can process a job simultaneously; and (v) preemption is not allowed. After finishing the first operation, the second operation should be started in a predetermined time, sometimes called limited waiting time or Q(queue) time. This time period of the different jobs might be different and we let w_i be waiting time of job i.

To describe the problem more clearly, we use the following notation.

p_{ik} processing time of job i in stage k ($i = 1, \dots, n$ and $k = 1, 2$)

w_i (predetermined) limited waiting time (Q-time) of job i

C_{jtk} completion time of t-th positioned job on a j-th machine in stage k

C_{max} maximum completion time of all jobs; i.e. makespan

S_{jt2} start time of the job (which is completed in the t-th position on j-th machine at stage 1) at stage 2

Z_{ijkt} = 0 if job i is in the t-th position on j-th machine at k-th stage (t=1,…n), otherwise 0.

With the notation, a mathematical formulation is given as follows.

Objective function:

$$\text{Min } Z = C_{\max} \tag{1}$$

Constraints:

$$\sum_{t=1}^{n}\sum_{j=1}^{M_k} Z_{ijkt} = 1 \qquad \forall i = 1,\ldots,n;\ \forall k = 1,2 \tag{2}$$

$$\sum_{t=1}^{n}\sum_{i=1}^{n} Z_{ijkt} \le n \qquad \forall j = 1,\ldots,M_k;\ \forall k = 1,2 \tag{3}$$

$$\sum_{i=1}^{n} Z_{ijkt} \le 1 \qquad \forall j = 1,\ldots,M_k;\ \forall k = 1,2;\forall t = 1,\ldots,n \tag{4}$$

$$C_{jtk} = C_{jt-1k} + \sum_{t=1}^{n}\sum_{i=1}^{n}\sum_{j=1}^{M_k} p_{ik} Z_{ijkt} \qquad \forall k = 1,2 \tag{5}$$

$$C_{jtk} \ge C_{jt-1k},\ C_{jtk} \ge 0 \qquad \forall j = 1,\ldots,M_k;\ \forall k = 1,2;\forall t = 2,\ldots,n \tag{6}$$

$$S_{jt2} \ge C_{jt1},\ S_{jt2} - C_{jt1} \le \sum_{i=1}^{n} w_i Z_{ij1t} \qquad \forall j = 1,\ldots,M_1;\ \forall t = 1,\ldots,n \tag{7}$$

$$C_{\max} \ge C_{jt2} \qquad \forall j = 1,\ldots,M_2;\ \forall t = 1,\ldots,n \tag{8}$$

The objective function (1) shows that the objective of the problem is minimizing makespan and constraint (2) implies that each job must be scheduled once. Constraint (3) ensures that the number of jobs to be scheduled in a machine should be less than the number of total jobs and constraint (4) states that only one job should be scheduled at the schedule horizon. The definition of the completion time is represented in constraint (5) and constraint (6) indicates that processing of each job should be started after the completion of its previous job. The limited waiting time constraints are shown in constraint (7) and makespan is defined in constraint (8).

3. Solution approach

3.1. Dominance Properties

Although the mathematical programming is provided, it is hard to obtain optimal solutions by increasing variables and constraints in a reasonable time as the number of jobs and machines are increased. Therefore, heuristic algorithms are considered here to response dynamic scheduling requirements quickly using some dominance properties.

Permutation schedules are the one of the most common methods to solve the flowshop scheduling problem, in which same order of jobs in each stage is considered. In hybrid flowshop scheduling problems, permutation schedules imply that jobs chosen in order of permutations of jobs are assigned to the machine with the earliest start time at each stage. Optimal solutions from this manner are not optimal for the problem. Therefore, in this study, non-permutation schedules are considered.

Here, we do not consider an idle time inserted intentionally, called as intentional idle time, since it increases makespan. Of course, an inevitable idle time may be occurred in two cases; 1) when an operation in the second stage cannot start immediately as soon as an operation in the first stage is completed, since previous operations in second stage are not done, 2) when the associated first operation should be shifted, since it is impossible for an operation in second stage to start in the limited waiting time.

Following properties are the ones that are suggested in this paper. First, using the results of Azizoglu and Kirca (1998), in which shows the property of maximum completion time on each machine in the parallel machine scheduling problem with the objective of minimizing total regular costs, we suggest property 1.

Property 1. If the number of machines in stage 1 is greater than or equal to that of machines in stage 2 and $\max_i p_{i1} \le \min_i p_{i2}$, then there exists an optimal schedule in which the waiting times of all jobs in stage 2 are zero and the completion time of each machine in stage 2 cannot exceed $\max_i p_{i1} + \{\sum p_{i2} + (M_2-1)\max_i p_{i2}\}/M_2$.

Proof. The second stage is bottleneck since $M_2 \le M_1$ and $\max_i p_{i1} \le \min_i p_{i2}$. That is, whenever the first operation of

any job is completed on a machine, this job can be processed on a machine in stage 2 without any intentional idle time (the idle time between jobs in stage 1 can be occurred since $M_2 \le M_1$ and $\max_i p_{i1} \le \min_i p_{i2}$). Consequently, there is no idle time between two successive jobs in stage 2 in any optimal schedules and there are only M_1 idle times to wait for starting the second operations of which the first operations are completed. According to the results of Azizoglu and Kirca (1998), $\{\sum p_{i2} + (M_2-1) \max_i p_{i2}\}/ M_2$ can be upper bound of total processing times of jobs in stage 2. Also, there can be idle time in stage 2 as much as $\max_i p_{i1}$.This completes the proof. □

Similar to property 1, a property is developed for the case when the stage 1 becomes bottleneck. With these two properties, we can obtain an initial upper bound of makespan under each condition.

Property 2. If the number of machines in stage 2 is greater than or equal to that of machines in stage 1 and $\max_i p_{i2} \le \min_i p_{i1}$, then there exists an optimal schedule in which the waiting times of all jobs in stage 2 are zero and the completion time of each machine in stage 2 cannot exceed $\max_i p_{i2} + \{\sum p_{i1} + (M_1-1) \max_i p_{i1}\}/ M_1$.

Proof. In this case, the second stage is bottleneck since $M_1 \le M_2$ and $\max_i p_{i2} \le \min_i p_{i1}$. That is, whenever the first operation of any job is completed on a machine, this job can be processed on a machine in stage 2 without any intentional idle time (the idle time between jobs in stage 2 can be occurred since $M_1 \le M_2$ and $\max_i p_{i2} \le \min_i p_{i1}$). Consequently, there is no idle time between two successive jobs in stage 1 in any optimal schedule. In similar, $\{\sum p_{i1} + (M_1-1) \max_i p_{i1}\}/ M_1$ can be upper bound of total processing times of jobs in stage 1 and there can be the completion time of the last scheduled job with $\max_i p_{i2}$ in stage 2.□

By using the suggested properties, heuristic algorithms are developed as following. Let $C_{[i]k}$ be the completion time of the i-th completed job in stage k (if more than 2 jobs are completed at the same time on different machines, then the job on the machine with the lowest index has the smallest [i]). Then, each $C_{[i]k}$ can be defined as the following equations.

$$C_{[i]1} = \max \{C_{[i-1]1} + p_{[i]1}, C_{[i-1]2} - w_{[i]}\} \quad \text{for all } i = 1,\ldots,n;$$
$$C_{[i]2} = \max \{C_{[i]1}, C_{[i-1]2}\} + p_{[i]2} \quad \text{for all } i = 1,\ldots,n;$$
$$C_{[0]1} = C_{[0]2} = 0. \qquad (9)$$

3.2. Heuristic Algorithms

In this paper, three heuristic algorithms, called H1, H2 and H3, are suggested. The first heuristic algorithm, H1, is one by using Johnson's rule (1954). It is assumed that this problem is 2- machine flowshop scheduling problem, and the sequence of jobs is obtained. Then, by using list scheduling method and considering the limited waiting time, the schedule is constructed one by one. The second one, H2, is the algorithm by using the algorithm devised by Gilmore and Gomory (1964). Suppose that the second operations should start as soon as the first operation is completed, and this problem is considered as conventional 2-machine flowshop scheduling problem. Then, this problem, F_2/no-wait/C_{max}, can be solved optimally. After obtaining a sequence, construct a list schedule using the concept of backward scheduling method suggested by Kim (1995). Notice that a schedule is obtained by scheduling the second operation first in order of reversed sequence for a reversed time frame. That is, the start time and completion time of jobs in the original problem are the completion and start time in the reversed one, respectively. After obtaining a reversed schedule, this time frame is adjusted to the original one (It is easy to understand by imaging a mirror). The last method, H3, is similar to H1 except using a sequence obtained by modified largest processing time rule for a list scheduling. The followings are detailed procedure of each heuristic.

H1

Step 1. Obtain a sequence by using Johnson's rule assuming that there is only one machine in each stage, that is, consider this problem as conventional 2-machine flowshop scheduling problem. Let U be the set of the obtained sequence.

Step 2. Select the first (leftmost) job in the set U. Say job x.

Step 3. Construct a schedule in stage 1 by using list scheduling method, in which a job is assigned to the machine with the earliest start time (ties are broken by selecting a machine with the lowest index). Update the start time $S_{[x]1}$ and completion time $C_{[x]1}$ of the job x.

Step 4. Construct a schedule in stage 2 by selecting a machine with the minimum completion time, of which start time is greater than $C_{[x]1}$. Update the start time $S_{[x]2}$ and completion time $C_{[x]2}$. If the second operation of a job cannot start in the limited waiting time, i.e. $S_{[x]2}-C_{[x]1} \geq w_x$, the completion time of the first operation associated with the job is adjusted to start in the limited waiting time, new $C_{[x]1} = S_{[x]2}-w_x$. And update other variable $S_{[x]1}$.

Step 5. If all jobs are scheduled then stop, otherwise, eliminate job x in the set U and go to step 2.

H2

Step 1. Obtain a sequence by using algorithm suggested by Gilmore and Gomory (1964) assuming that there is only one machine in each stage and waiting time in the second stage is not allowed, that is, consider this problem as conventional 2-machine flowshop scheduling problem with no-wait constraints (all w_i's are zero). Let U be the set of the obtained sequence.

Step 2. In the reversed time frame problem, let $S'_{[i]k}$ and $C'_{[i]k}$ be a start time and a completion time of a job in stage k, respectively. Notice that If k=1 in this notation, it implies that k=2 in the original problem.

Step 3. Select the last (rightmost) job in the set U'. Say job x.

Step 4. Construct a schedule in stage 1 (stage 2 in the original problem) by using list scheduling method, in which a job is assigned to the machine with the earliest time, (ties are broken by selecting a machine with the lowest index). Update the start time $S'_{[x]1}$ and completion time $C'_{[x]1}$ of the job x.

Step 5. Construct a schedule in stage 2 (stage 1 in the original problem) by selecting a machine with the smallest completion time, of which start time is greater than $C'_{[x]1}$. Update the start time $S'_{[x]2}$ and completion time $C'_{[x]2}$. If the second operation of a job cannot start immediately, i.e. $S'_{[x]2}-C'_{[x]1} \geq 0$, the completion time of the first operation associated with the job is adjusted to the start time of the second operation, i.e. $C'_{[x]1} = S'_{[x]2}$. And update another variable $S'_{[x]1}$.

Step 6. If all jobs are scheduled then stop and compute $C'_{\max}$, otherwise, eliminate job x in the set U and go to step 3.

Step 7. Restore the reversed problem to the original one. That is, without any shift of jobs, each start time and completion time of job is re-calculated reversely. In this manner, the last scheduled job in the reversed problem is the one scheduled first in the original problem, in which the start time and completion time in stage 1 are 0 and $p_{[1]2}$, respectively. In similar way, reschedule other jobs and update original variables.

Step 8. In the original problem, sort the completion times of jobs in stage 1. Let set U be the set of the obtained order of jobs.

Step 9. Select a leftmost job, i.e. a job with the smallest C_{i1}, in the set U.

Step10. If the completion time can be improved by shifting this job on the same machine, reschedule this job as well as succeeding jobs in each stage and update variables. Otherwise, eliminate this job in the set U and repeat step 9 and 10 until set U is none.

H3

Step 1. Sort jobs in non-increasing order of processing times of jobs in stage 1. Let U be the set of the obtained order.

Step 2. Select the first (leftmost) job in the set U. Say job x.

Step 3. Construct a schedule in stage 1 by using list scheduling method, in which a job is assigned to the machine with the earliest start time (ties are broken by selecting a machine with the lowest index). Update the start time $S_{[x]1}$ and completion time $C_{[x]1}$ of the job x.

Step 4. Construct a schedule in stage 2 by selecting a machine with the minimum completion time, of which start time is greater than $C_{[x]1}$. Update the start time $S_{[x]2}$ and completion time $C_{[x]2}$. If the second operation of a job cannot start in the limited waiting time, i.e. $S_{[x]2}-C_{[x]1} \geq w_x$, the completion time of the first operation associated with the job is adjusted to start in the limited waiting time, new $C_{[x]1} = S_{[x]2}-w_x$. And update other variables $S_{[x]1}$.

Step 5. If all jobs are scheduled then stop, otherwise, eliminate job x in the set U and go to step 2.

4. Computational experiments

To see performances among the suggested heuristics, computational experiments are performed on randomly generated problems. All heuristics are coded using commercial software ezDFS, which is used in manufacturing execution system (MES) of real manufacturing environments and in which users can make methodologies and generate test data sets easily, using graphical user interface (GUI) tool. The number of jobs is set to 20, 40, 60, 80, and 100 respectively, and the number of machines in each stage is set to 5, 10 and 15, respectively. We generate the processing times and waiting times of jobs using a discrete uniform distribution between [1, 50] and [1, 100], respectively. In this experiment, we generate 450 problem instances for the tests by 10 runs for each of the above combinations.

As performance measures, we use relative deviation index (RDI) and the number of the best solutions (NBS). The RDI of each heuristic algorithm H is obtained by computing $(X-B)/(W-B)$, where X, B and W are makespan obtained by heuristic algorithm H, the best makespan among all heuristic algorithms and the worst makespan among all heuristic algorithms for a problem, respectively. The RDI is between 0 and 1 and shows how each heuristic algorithm performs well compared to others. If the RDI of heuristic algorithm H for a problem is the smallest, then it can be said that algorithm H shows a better performance than others for the problem. Also, the NBS can be represented as the number of best cases, in which each algorithm shows the minimum makespan.

Table 1 shows overall results (average and standard deviation of RDIs of each heuristic, the number of the best solutions and average and deviation of CPU times). Overall, it can be seen that H3 shows a better performance than other methods as measured by RDI and NBS. However, it seems that there is no significant difference between CPU times to get solutions.

Table 1 The overall results of the test

Heuristics	RDI*		NBS**	CPU time	
H1	0.58	(0.42)	57	0.12	(0.36)
H2	0.53	(0.67)	105	0.08	(0.51)
H3	0.33	(0.59)	363	0.14	(0.75)

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

** Number of simulation runs (out of 450) for which each heuristic found the best solutions

To show the effects of various factors (i.e. the number of jobs and machines in each stage) on the relative performance of the algorithms, an analysis of variance (ANOVA) was done for each heuristic method on the solution values (i.e. makespan of jobs). Results are shown in table 2. The ANOVA table shows that performance of the heuristics was significantly affected by the used heuristic at the significance level of 0.01. This implies that H3 outperforms others as described earlier. Also, the relative performance of the algorithms was not affected by the other factors such as the number of jobs and the number of machines in each stage. This means certain algorithms consistently work better than other algorithms regardless of characteristics of the problems.

Table 2 Analysis of variance for the performance of the each heuristics

Source of variation	Degree of freedom	Sum of squared errors	Mean squared errors	F
Heuristics	2	209	104.5	13.8†
Number of jobs	4	65	16.3	2.1
NM1*	2	58	29.0	3.8
NM2**	2	63	31.5	4.2
NM1 × NM2	4	93	23.3	3.1
Error	1335	10096	7.6	
Total	1349	10584		

* Number of machines in stage 1

** Number of machines in stage 2

† Statistically different at the significant level of 0.01

In addition to the above tests, another series of tests were performed to see the absolute performance of the outperformed algorithm, H3, by comparing solutions of the algorithm with optimal solutions and lower bounds. Since no efficient optimal solution algorithm exists for the problem, optimal solutions were obtained with mixed integer programming for the small sized problem. In this case, the number of machines in each stage is set to 2 and 3 and the number of jobs is set to 5 and 6. Other data were generated in the same way as described earlier. Results of the test are summarized in table 3, which shows the percentage gaps between solutions of the heuristic algorithm and optimal solutions (lower bounds) and the computation times required to get optimal solution (lower bound). Since a very short computation time is needed (less than 0.05 second) to obtain a solution, it is not shown. From the table, it might be said that the suggested algorithm, H3, found very good solutions and found optimal solutions.

Table 3 Percentage gap between solutions of the algorithm and optimal solutions

NM1	NM2	Number of jobs	Percentage gaps*			CPU time**	
2	2	5	0.70**	(1.55)	8	121.2	(549.1)
	3	6	2.26	(4.18)	7	369.8	(1032.6)
3	2	5	3.31	(4.81)	6	898.7	(804.5)
	3	6	1.33	(2.98)	8	1542.1	(2408.1)

* Average and standard deviation (in parenthesis) of the percentage gaps of heuristic solutions from the optimal solutions and the number of problems for which the algorithm found optimal solutions out of 10 test problems.
**Average and standard deviation (in parenthesis) of the CPU time required for finding an optimal solution.

For larger sized problem, the solutions were compared with lower bound which is based on the branch and bound algorithm suggested by Brah and Hunsucker (1991). In this lower bound, we assume that waiting time of each job is infinite, i.e. $w_i=\infty$, that is, it is not necessary for jobs to be processed in its waiting time after the completion in stage 1. The branch and bound algorithm suggested by Brah and Hunsucker (1991) is the one that consider k-stage hybrid flowshop scheduling problem with identical parallel machines in every stage to minimize makespan of jobs. For obtaining lower bounds, the number of machines in each stage is set to 4 and 5 and the number of jobs is set to 8 and 10. Also, other data were generated in the same way as described earlier. Also, it can be seen that there is no significant difference between lower bounds and solutions by heuristic and solutions from the suggested heuristic algorithm may be considered relatively good.

Table 4 Percentage gap between solutions of the algorithm and lower bounds

NM1	NM2	Number of jobs	Percentage gaps*			CPU time**	
4	4	8	7.26	(5.00)	2	10.3	(5.4)
	5	10	7.67	(7.08)	3	36.1	(40.7)
5	4	8	9.51	(10.46)	3	49.6	(69.3)
	5	10	10.19	(8.05)	2	91.5	(95.4)

* Average and standard deviation (in parenthesis) of the percentage gaps of heuristic solutions from the lower bounds and the number of problems for which solutions of the algorithm are equal to lower bounds of 10 test problems.
**Average and standard deviation (in parenthesis) of the CPU time required for finding lower bounds.

5. Conclusion

In this paper, we focused 2-stage hybrid flowshop with identical parallel processors in each stage with the objective of minimizing makespan of jobs. In each stage, there are identical parallel machines and there are waiting times for jobs to be processed in the second stage. The waiting time of each job cannot be greater than a predetermined time due to its deterioration. We tested 3 heuristic algorithms which are based on Johnson's rule (1954), Gilmore and Gomory's algorithm (1964) and the modified largest processing time rule, respectively. The suggested algorithms show good performances and among those algorithms, H3 outperformed others and gave very good solutions which

are nearly optimal in a very short computation time. The suggested algorithm in this paper may be very useful for real manufacturing system since it is motivated by real scheduling problems and extended in several ways. First, one may devise more sophisticate one based on the suggested algorithm for more than 2 stage hybrid problem or for the objective of different measures. Also, it is necessary to find some optimal solution properties and develop more efficient algorithms using them.

Acknowledgement

This research was supported by the research fund of Hanbat National University in 2018

Reference

An, Y. -J., Kim, Y. -D., Choi, S. -W., Minimizing makespan in a two-machine flowshop with a limited waiting time constraint and sequence-dependent setup times, Computers & Operations Research, Vol. 71, (2016), pp. 127-136.

Azizoglu, M., Cakmak, E. and Kondakci, S., A flexible flowshop problem with total flow time minimization, European Journal of Operational Research, Vol. 132, No. 3, (2001), pp. 528–538.

Behnamian, J. Fatemi Ghomi, S.M.T., Hybrid flowshop scheduling with machine and resource-dependent processing times, Applied Mathematical Modelling, Vol. 35, No. 3, (2011), pp. 1107-1123.

Bouquard, J. -L. and Lente, C., Two-machine flow shop scheduling problems with minimal and maximal delays, 4OR (A Quarterly Journal of Operations Research), Vol. 4, No. 1, (2006), pp.15-28.

Bouquard, J. -L. and Lente, C. and Billaut, J. -C., Application of an optimization problem in Max-Plus algebra to scheduling problems, Discrete Applied Mathematics, Vol. 154, No. 15, (2006), pp. 2064-2079.

Brah, S. A. and Hunsucker, J. L., Branch and bound algorithm for the flow shop with multiple processors, European Journal of Operational Research, Vol. 51, No. 1, (1991), pp. 88–99.

Dugardin, F., Yalaoui, F. and Amodeo, L., New multi-objective method to solve reentrant hybrid flow shop scheduling problem, European Journal of Operational Research, Vol. 203, No. 1, (2010), pp.22–31.

Eskandari, H. and Hosseinzadeh, A., A variable neighbourhood search for hybrid flow-shop scheduling problem with rework and set-up times, Journal of The Operational Research Society, Vol. 65, No. 8, (2014), pp.1221–1231.

Fernandez-Viagas, V., Molina-Pariente, J.M. and Framinan, J.M., New efficient constructive heuristics for the hybrid flowshop to minimise makespan: a computational evaluation of heuristics, Expert Systems with Applications, Vol. 114, (2018), pp.345–356.

Fondrevelle, J., Oulamara, A. and Portmann, M. -C., Permutation flowshop scheduling problems with maximal and minimal time lags, Computers & Operations Research, Vol. 33, No.6, (2006), pp. 1540-1556.

Gilmore, P. -C. and Gomory, R. -E., Sequencing a one state-variable machine: a solvable case of the traveling salesman problem, Operations Research, Vol. 12, No. 5, (1964), pp. 655-679.

Gomez-Gasquet, P., Andres, C. and Lario, F.C., An agent-based genetic algorithm for hybrid flowshop with sequence dependent setup times to minimize makespan, Expert Systems with Applications, Vol. 39, No. 9, (2012), pp.8095–8107.

Gupta, J. N. D., Two-Stage hybrid flowshop scheduling problem, Journal of the Operational Research Society, Vol. 39, No. 4, (1998), pp. 359–364.

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

Han, W., Guo, F. and Su, X., A reinforcement learning method for a hybrid flow-shop scheduling problem, Algorithms, Vol. 12, No. 11, (2019), pp.1–15.

Jeong BJ. Han, J. -H. and Lee, J. -Y., Metaheuristics for a flow shop scheduling problem with urgent jobs and limited waiting times, Algorithms, Vol. 14, No. 11, (2021), pp. 323.

Johnson, S. -M., Optimal two- and three-stage production schedules with setup times included, Naval Research Logistics Quarterly, Vol. 1, No. 1, (1954), pp. 61-68.

Lee, G. -C. and Kim, Y. -D., A branch-and-bound algorithm for a two-stage hybrid flowshop scheduling problem minimizing total tardiness, International Journal of Production Research, Vol. 42, No. 22, (2004), pp. 4731 – 4743

Lee, J. -Y., A genetic algorithm for a two-machine flowshop with a limited waiting time constraint and sequence-dependent setup times, Mathematical Problems in Engineering, Vol, 2020, Article ID 8833645, pp. 13.

Linn, R. and Zhang, W., Hybrid flow shop scheduling: a survey, Computers and Industrial Engineering, Vol. 37, No. 1-2, (1999), pp. 57–61.

Lin, S. -W., Cheng, C. -Y., Pourhejazy, P., Ying, K. -C. and Lee, C. -H., New benchmark algorithm for hybrid flowshop scheduling with identical machines, Expert Systems with Applications, Vol. 183, No. 30, (2021), 115422.

Moursli, O. and Pochet, Y., A branch-and-bound algorithm for the hybrid flowshop, International Journal of Production Economics, Vol. 64, No. 1-3, (2000), pp. 113–125.

Öztop, H., Tasgetiren, M. F., Eliiyi, D. T. and Pan, Q. -K., Metaheuristic algorithms for the hybrid flowshop scheduling

problem, Computers & Operations Research, Vol. 111, (2019), pp. 177-196.

Rajendran, C. and Chaudhuri, D., A multi-stage parallel-processor flowshop problem with minimum flowtime, European Journal of Operational Research, Vol. 57, No. 1, (1992). pp. 111–122.

Ribas, I., Leisten and R., Framiñan, J.M., Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective, Computers & Operations Research, Vol. 37, No. 8, (2010), pp. 1439-1454.

Ruiz, R., Vázquez-Rodríguez, J.A., The hybrid flow shop scheduling problem, European Journal of Operational Research, Vol. 205, No. 1, (2010), pp. 1-18.

Soewandi, H., Sequencing jobs on two- and three-stage hybrid flowshop to minimize makespan, Dissertation for the Degree of Ph. D in the North Carolina State University, (1998).

Su, L. -H., A hybrid two-stage flowshop with limited waiting time constraints, Computers and Industrial Engineering, Vol. 44, No. 3, (2003), pp. 409-424.

Tosun, Ö., Marichelvam, M.K. and Tosun, N., A literature review on hybrid flow shop scheduling, International Journal of Advanced Operations Management, Vol. 12, No. 2, (2020), pp. 156-194.

Yang, D. -L. and Chern, M. -S., A two-machine flowshop scheduling problem with limited waiting time constraints, Computers and Industrial Engineering, Vol. 28, No. 1, (1995), pp. 63-70.

Ying, K. -C. and Lin, S. -W., Minimizing makespan for the distributed hybrid flowshop scheduling problem with multiprocessor tasks, Expert Systems with Applications, Vol. 92, (2018), pp. 132-141.