ICIC Express Letters
Volume **15**, Number **7**, July **2021**

 ISSN 1881-803X
pp. **763–768**

MINIMIZING MAXIMUM COMPLETION TIME IN A NO-WAIT TWO-STAGE FLEXIBLE FLOW SHOP

Wankyu Ryu[1] and Suk-Hun Yoon[2,*]

[1]Department of Industrial Engineering
Seoul National University
1 Gwanak-ro, Gwanak-gu, Seoul 08826, Korea
wankyu84@hotmail.com

[2]Department of Industrial and Information Systems Engineering
Soongsil University
369 Sangdo-ro, Dongjak-gu, Seoul 06978, Korea
*Corresponding author: yoon@ssu.ac.kr

Received November 2020; accepted January 2021

Abstract. *In this paper, we consider a no-wait two-stage flexible flow shop scheduling problem where a single machine is in the first stage and two identical machines in the second stage. The objective is the minimization of maximum completion time. A mixed integer linear programming formulation is presented to obtain an optimal solution for small size problems. A hybrid genetic algorithm (HGA) is proposed for large size problems. The HGA increases the exploration capabilities of a basic genetic algorithm and reduces the premature convergence. Extensive computational tests on randomly generated problems are conducted to evaluate the performance of the HGA.*
Keywords: Scheduling, Flexible flow shop, Maximum completion time, No-wait, Mixed integer linear programming, Genetic algorithms

1. **Introduction.** Scheduling is a decision-making process that deals with the allocation of machines (resources) to jobs (tasks) over time to optimize one or more objectives. A flexible flow shop consists of multiple stages in series with a number of machines in parallel at each stage. A job has to be processed at each stage only on one of the machines [1].

In no-wait production settings, once the processing of a job begins, subsequent processing must be carried out without delays and waiting before or during any operation [2]. The no-wait flow shop is frequently encountered in some process industries such as chemical processing, food processing, pharmaceutical, and steel industries [3].

The no-wait flow shop scheduling problem with minimum makespan is strongly NP-hard when the number of machines is more than two [4]. The computational effort required to solve an instance of problems grows significantly fast as the number of jobs increases. Thus, if the number of jobs is large, it is necessary to consider the heuristic and metaheuristic approaches [5].

AitZai et al. [6] proposed the branch and bound algorithm and particular swarm optimization for the no-wait job shop scheduling problem to minimize the makespan. Allahverdi et al. [7] proposed the block simulated annealing algorithm utilizing block insertion and block exchange operators for the no-wait flow shop scheduling problem with separate setup times. The objective was to minimize total tardiness with an upper bound on makespan. Ding et al. [8] addressed the no-wait flow shop scheduling problem with the makespan criterion. They presented the tabu-mechanism improved iterated greedy algorithm by utilizing the tabu search strategy. Zhao et al. [9] developed the hybrid biogeography-based optimization with variable neighborhood search for the no-wait flow

DOI: 10.24507/icicel.15.07.763

shop scheduling problem to minimize the makespan. They analyzed the global convergence performance of their algorithm with the Markov model. Koulamas and Kyparisis [10] considered the no-wait flow shop scheduling problem with rejection. They presented a third-order polynomial-time dynamic programming algorithm to minimize the sum of total completion time and total rejection cost with ordered jobs.

Gheisariha et al. [11] addressed the flexible flow shop scheduling problems with sequence-based setup time, transportation time, and probable rework to minimize both maximum completion time and mean tardiness. They proposed an enhanced multi-objective harmony search algorithm and a Gaussian mutation utilizing response surface methodology. Ernst et al. [12] considered a two-stage flexible flow shop where first and second stage machines formed disjoint pairs with a various buffer size. They developed a polynomial-time algorithm for the case of equal size buffers and presented two integer linear programs. Peng et al. [13] studied a two-stage flexible flow shop scheduling to minimize the maximum completion time. For the problem with a machine at the first stage and two machines at the second machine, they presented a 2.25-approximation algorithm in $O(n \log n)$. Choi and Lee [14] considered a two-stage flexible flow shop problem with one machine at stage 1 and multiple identical machines at stage 2 to minimize the makespan where the processing times of each job at both stages were identical. They described some optimality conditions and showed the problem to be NP-hard.

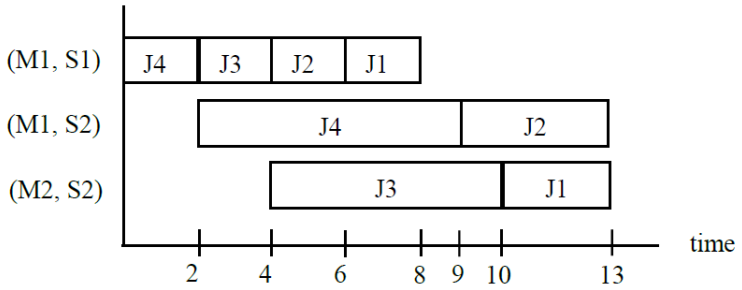
Asefi et al. [15] addressed the no-wait k-stage flexible flow shop scheduling problem to minimize makespan and mean tardiness. They presented the novel hybrid advanced meta-heuristic algorithm. Abdollahpour and Rezaian [16] studied the no-wait flexible flow shop scheduling problem with capacitated machines and mixed make-to-order and make-to-stock production management policy restrictions to the sum of tardiness cost, weighted earliness cost, weighted rejection cost and weighted incomplete cost. They developed the cloudy-based simulated annealing and artificial immune system.

This paper presents a methodology based on a genetic algorithm (GA) for a no-wait two-stage flexible flow shop scheduling problem, in which there is one machine at stage 1 and two identical machines in parallel at stage 2. Once the processing of a job is completed at stage 1, the job has to be processed on any machine at stage 2 without having to wait. The objective is to minimize maximum completion time (makespan). The no-wait process is illustrated by the four-job flexible flow shop where job processing times at stage 1 are all 2 time units and 3, 4, 6 and 7 time units at stage 2 shown in Figure 1. The job sequence is 4-3-2-1 generated by the longest processing time rule at stage 2. If job waiting is allowed, the job completion time will be 13 time units (schedule 1). As Figure 1(b) shows, when job waiting is not allowed, the completion time will be 14 time units (schedule 2).

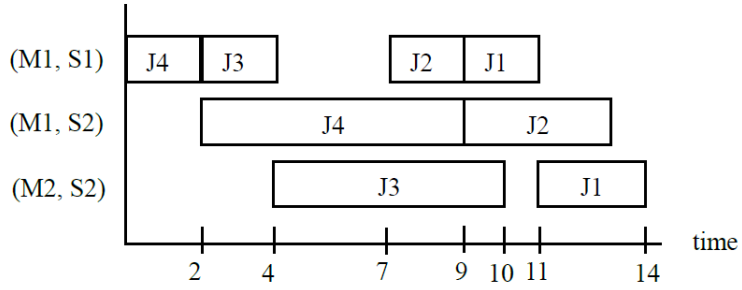
The rest of this paper is organized as follows. In Section 2, the notations and assumptions are defined and a mixed integer linear programming (MILP) model for the problem is provided. Given a job sequence, the MILP can be solved to obtain an optimal solution in a reasonable time. In Section 3, a hybrid genetic algorithm (HGA) is developed to find the best solution for large size problems. In Section 4, the results of extensive computational experiments are provided. The performance of the HGA is compared with that of the GA. Finally, summary and conclusions are provided in Section 5.

2. **Notations and Problem Definition.** The following notations will be used through the paper:

n number of jobs
p_{ij} processing time of job j at stage i
C_{ij} completion time of job j at stage i
$x_j = \begin{cases} 1 & \text{if job } j \text{ is allocated to machine 1 at stage 2} \\ 0 & \text{otherwise} \end{cases}$



(a) Schedule without no-wait (schedule 1)



(b) Schedule with no-wait (schedule 2)

FIGURE 1. Two schedules for a four-job flexible flow shop

$$y_{jl} = \begin{cases} 1 & \text{if job } j \text{ precedes job } l \text{ at stage 1} \\ 0 & \text{otherwise} \end{cases}$$

$C_{\max}$ maximum completion time

M big number

We assume that all jobs are available at time zero. Then the problem can be formulated by a mixed integer linear programming as follows:

$$\begin{aligned}
&\text{minimize } C_{\max} \\
&\text{s.t. } C_{2j} - C_{1j} = p_{2j}, && j = 1, \ldots, n && (1)\\
&\quad C_{1j} + My_{jl} - C_{1l} \geq p_{1j}, && j, l = 1, \ldots, n && (2)\\
&\quad C_{2j} + M(2 + y_{jl} - x_j - x_l) - C_{2l} \geq p_{2j}, && j, l = 1, \ldots, n && (3)\\
&\quad y_{jl} + y_{lj} = 1, && j, l = 1, \ldots, n && (4)\\
&\quad C_{\max} - C_{2j} \geq 0, && j = 1, \ldots, n && (5)\\
&\quad C_{1j} \geq p_{1j}, && j = 1, \ldots, n && (6)\\
&\quad x_j = 0 \text{ or } 1, && j = 1, \ldots, n && (7)\\
&\quad y_{jl} = 0 \text{ or } 1, && j, l = 1, \ldots, n && (8)
\end{aligned}$$

Constraint set (1) ensures that each job is processed without waiting between stages. Constraint set (2) establishes that two jobs cannot be processed simultaneously at stage 1. Constraint set (3) assures that two jobs cannot be processed simultaneously on the same machine at stage 2. Constraint set (4) implies that only permutation schedules can be considered for an optimal schedule. Constraint set (5) defines the maximum completion time. Constraint set (6) states that each job is available at time zero. Constraint sets (7) and (8) insure 0-1 values of variables x_j and y_{jl}.

3. **Hybrid Genetic Algorithm.** Genetic algorithm imitates the evolution of living organisms in the nature and attempts to find the optimum for some complex problems in accordance with the principle of survival of the fittest [17]. The GA operates on the population by applying three main operators to creating the next population from the

current population (a collection of chromosomes): crossover, mutation, and selection at each step (generation). Each chromosome consists of discrete units called genes. In the crossover, the genetic information of two parents is combined to explore the design space and the value of every gene is changed with mutation probability to produce two offspring (children). Meanwhile, individuals (solutions, candidate or chromosome) are chosen by a selection operator for the reproduction. The obtained solution is evaluated by the fitness which measures the performance of the chosen individuals compared to the other whole population. This process continues until a predefined stopping condition is fulfilled [18-20].

The HGA adopts three basic operators of GAs and combines a pairwise interchange (PI) to enhance the exploration capabilities of the GA by restraining the premature convergence. To represent a solution as a chromosome, binary coding for x_j and y_{jl} is used. For example, consider a three-job 2-stage flexible flow shop problem. If the job sequence at stage 1 is 1-2-3 and jobs 1 and 2 are assigned to machine 1 at stage 2, the solution can be represented by $(x_1, x_2, x_3, y_{12}, y_{13}, y_{23}) = (1, 1, 0, 1, 0, 0)$.

An initial population is chosen randomly by assigning 0 or 1 to x_j and y_{jl}. The makespan of each chromosome is used as its fitness value. By letting binary variables (x_j and y_{jl}) be 0 or 1, the MILP of the problem reduces to the LP and thus, the makespan can be obtained in polynomial time. The ratio of the fitness value of each chromosome to the total fitness value is the probability of the chromosome to be selected in the selection process.

Unlike many scheduling problems, the problem is represented by binary coding. The stochastic remainder selection procedure without replacement [21], the single-point crossover and the ordinary mutation operator are used for the selection operator, crossover operator and mutation operator, respectively. In GA, if the fitness values of low fit individuals are too low compared to those of high fit individuals, the process of the GA tends to reach a local optimum too early. Hence, to avoid this premature convergence of the GA, HGA applies a non-adjacent pairwise interchange method to the least fit individual and increases the competent individuals compared to the dominant individuals. Figure 2 shows the process of the HGA.

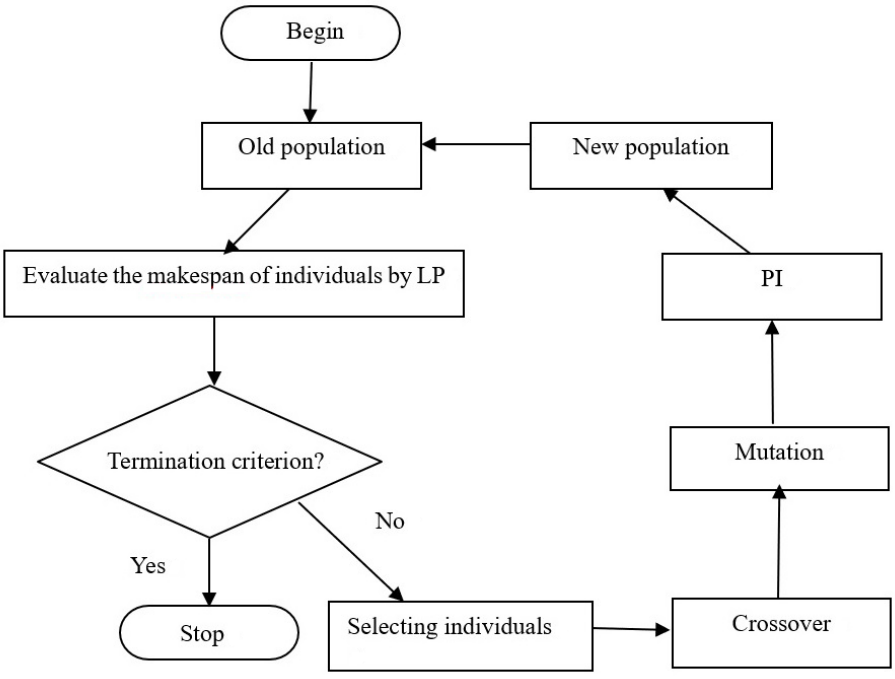


FIGURE 2. HGA cycles

4. **Computational Experiments.** The MILP and HGA were coded in Visual C++ with CPLEX solver and implemented on Unix platform. Test problems were generated randomly using the pseudo-random numbers. Processing times were generated according to the integer uniform distributions provided in $[1, 100]$. The size of test problems is defined by the number of jobs (5, 10, 15 and 20). Experiments were composed of two parts: the preliminary test and main test. In the preliminary test, 10 problems of different job sizes were solved to find the best parameter set of the HGA. The parameter set includes population size (N_p), number of generations (N_g) and mutation rate (p_m). The computational results of the preliminary test are shown in Table 1. The best result was obtained with population size of 40, 40 generations and mutation rate of 0.05.

Table 1. Preliminary test results

N_p	30						40					
N_g	20		30		40		20		30		40	
p_m	0.01	0.05	0.01	0.05	0.01	0.05	0.01	0.05	0.01	0.05	0.01	0.05
5	255	210	299	227	220	210	298	295	227	248	210	217
10	561	537	591	534	537	514	597	561	518	577	554	507
15	792	763	814	788	745	774	781	794	823	779	786	734
20	1,103	948	1,079	1,116	1,046	987	1,085	1,063	1,125	954	993	978
Total	2,711	2,458	2,783	2,665	2,548	2,485	2,761	2,713	2,693	2,558	2,543	**2,436**

Five test problems of different job sizes were solved by the HGA with the best parameter set found in the preliminary test. For small size problems (5 and 10 jobs), the results of HGA were compared with the optimal solutions obtained by the MILP. HGA achieved optimal solutions for all 10 small size problems.

For large size problems, solving the MILP exactly is not recommendable since large amount of CPU time and memory are required. The GA solved the large size problems (15 and 20 jobs). The results of the GA and HGA are shown in Table 2. The HGA provides 7.42% better solutions than the GA on the average, which implies that the HGA helps to avoid the premature convergence.

Table 2. Results for medium and large size no-wait flexible flow shop problems

No. of	GA	HGA	%Dev
Jobs	z_g	z_h	$(z_g - z_h/z_g) \times 100$
15	781	738	5.51
20	1,039	942	9.34

5. **Conclusions.** This paper addressed the problem of minimizing the maximum completion time in a no-wait 2-stage flexible flow shop with one machine at stage 1 and two identical machines in parallel at stage 2. We represented the problem as the mixed integer linear programming formulation to obtain an optimal solution for the small size problems. For the large size problems, the MILP is not appropriate to be solved since the problem is NP-hard in the strong sense. Hence, we developed the HGA which lessened the premature convergence and enhanced the search power. Extensive computational experiments were conducted to evaluate the performance of HGA.

Modern manufacturing and service industries adopting automation and just-in-time production systems require the progressive reduction of inventories and improvement of flexibility. This requirement can be realized by utilizing the concept of no-wait and a series of duplicate resources (hybrid flow shop). Hence, the problem of this paper can be applied to various modern production systems. A variety of hybrid metaheuristic algorithms

equipped with a delicate tuning scheme need to be developed particularly for large size and complex problems. Also, finding effective cuts is essential for the MILP applicable to various large size problems and fast computing.

REFERENCES

[1] M. L. Pinedo, *Scheduling: Theory, Algorithms, and Systems*, 5th Edition, Springer, New York, 2016.

[2] K. R. Baker and D. Trietsch, *Principles of Sequencing and Scheduling*, 2nd Edition, Wiley, New York, 2019.

[3] H. Ye, W. Li and B. R. Nault, Trade-off balancing between maximum and total completion times for no-wait flow shop production, *International Journal of Production Research*, vol.58, no.11, pp.3235-3251, 2020.

[4] H. Rock, The three-machine no-wait flow shop is NP-complete, *Journal of the ACM*, vol.31, no.2, pp.336-345, 1984.

[5] M. Gendreau and J.-Y. Potvin (eds.), *Handbook of Metaheuristics*, 3rd Edition, Springer, New York, 2018.

[6] A. AitZai, B. Benmedjdoub and M. Boudhar, Branch-and-bound and PSO algorithms for no-wait job shop scheduling, *Journal of Intelligent Manufacturing*, vol.27, pp.679-688, 2016.

[7] A. Allahverdi, H. Aydilek and A. Aydilek, No-wait flowshop scheduling problem with separate setup times to minimize total tardiness subject to makespan, *Applied Mathematics and Computation*, vol.365, 2020.

[8] J.-Y. Ding, S. Song, J. N. D. Gupta, R. Zhang, R. Chiong and C. Wu, An improved iterated greedy algorithm with a tabu-based reconstruction strategy for the no-wait flowshop scheduling problem, *Applied Soft Computing*, vol.30, pp.604-613, 2015.

[9] F. Zhao, S. Qina, Y. Zhang, W. Ma, C. Zhang and H. Song, A hybrid biogeography-based optimization with variable neighborhood search mechanism for no-wait flow shop scheduling problem, *Expert Systems with Applications*, vol.126, pp.321-339, 2019.

[10] C. Koulamas and G. J. Kyparisis, The no-wait flow shop with rejection, *International Journal of Production Research*, vol.59, no.6, pp.1852-1859, 2021.

[11] E. Gheisariha, M. Tavana, F. Jolai and M. Rabiee, A simulation-optimization model for solving flexible flow shop scheduling problems with rework and transportation, *Mathematics and Computers in Simulation*, vol.180, pp.152-178, 2021.

[12] A. Ernst, J. Fung, G. Singh and Y. Zinder, Flexible flow shop with dedicated buffers, *Discrete Applied Mathematics*, vol.261, pp.148-163, 2019.

[13] A. Peng, L. Liu and W. Lin, Improved approximation algorithms for two-stage flexible flow shop scheduling, *Journal of Combinatorial Optimization*, vol.41, pp.28-42, 2021.

[14] B.-C. Choi and K. Lee, Two-stage proportionate flexible flow shop to minimize the makespan, *Journal of Combinatorial Optimization*, vol.25, pp.123-134, 2013.

[15] H. Asefi, F. Jolai, M. Rabiee and M. E. T. Araghi, A hybrid NSGA-II and VNS for solving a bi-objective no-wait flexible flowshop scheduling problem, *International Journal of Advanced Manufacturing Technology*, vol.75, pp.1017-1033, 2014.

[16] S. Abdollahpour and J. Rezaian, Two new meta-heuristics for no-wait flexible flow shop scheduling problem with capacitated machines, mixed make-to-order and make-to-stock policy, *Soft Computing*, vol.21, pp.3147-3165, 2017.

[17] J. Luan, Z. Yao, F. Zhao and X. Song, A novel method to solve supplier selection problem: Hybrid algorithm of genetic algorithm and ant colony optimization, *Mathematics and Computers in Simulation*, vol.156, pp.294-309, 2019.

[18] R. Bendaoud, H. Amiry, M. Benhmida, B. Zohal, S. Yadir, S. Bounouar, C. Hajjaj, E. Baghaz and M. El Aydi, New method for extracting physical parameters of PV generators combining an implemented genetic algorithm and the simulated annealing algorithm, *Solar Energy*, vol.194, pp.239-247, 2019.

[19] M. Elhoseny, A. Tharwat and A. E. Hassanien, Bezier curve based path planning in a dynamic field using modified genetic algorithm, *Journal of Computational Science*, vol.25, pp.339-350, 2018.

[20] K. M. Hamdia, X. Zhuang and T. Rabczuk, An efficient optimization approach for designing machine learning models based on genetic algorithm, *Neural Computing and Applications*, vol.33, pp.1923-1933, 2021.

[21] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison Wesley, Reading, 1989.