Send Orders for Reprints to reprints@benthamscience.ae

The Open Automation and Control Systems Journal, 2014, *6*, 593-600

Study on Hybrid Flow Shop Scheduling Problem with Blocking Based on GASA

Ze Tao[1,*], Xiaoxia Liu[2] and Pengfei Zeng[1]

[1] *School of Mechanical Engineering, Shenyang Ligong University, Shenyang, 110159, China*

[2] *School of Electrical &Mechanical Engineering, Henan University of Technology, Henan, China*

Abstract: A new hybrid flow shop scheduling problem with blocking is studied. The problem is characterized by multi-stage unrelated parallel machines and no intermediate buffer. Firstly, the math model of the hybrid scheduling problem is constructed. And then the computation method of production period is given. Due to there is no intermediate buffer, and there are unrelated parallel machines in multi-stages, the production period is obtained through pushing from the operational time of each machine. Function objective of the proposed method is to minimize the production period. Finally the scheduling results are obtained based on genetic algorithm and simulated annealing algorithm(GASA). Scheduling results based on two cases and comparisons with some existing algorithms show that the method proposed in this paper is effective and feasible.

Keywords: Hybrid flow shop scheduling problem, blocking, unrelated parallel machines.

1. INTRODUCTION

Hybrid flow shop scheduling problem(HFSP) is extended based on classical flow shop scheduling, in which the parallel machines are considered. Due to its application in petroleum, chemical engineering, metallurgy, and rail transportation, HFSP has been widely studied by many researchers.

It is assumed that there is an infinite buffer between the neighborhood machines in the traditional HFSP, but in practical production process, the buffer is limited or nonexistent. If there is no intermediate buffer, when job n_i is finished on machine m_j, then transported to the machine m_{j+1}, but if the machine m_{j+1} is not free, the next operation of job n_i will not be processed until the machine m_{j+1} is free. These problems are blocking flow shop scheduling problems. For example, under normal conditions, the buffer volume is limited by the workshop space. In rail transportation, it is very strict to railway segments, and each railway segment can be occupied at most one freight train. The freight train is blocked on the railway segment if the former neighborhood railway segment is occupied until the former segment is released. A flow shop scheduling problem with intermediate buffers located between two consecutive machines was studied in [1], and a hybrid discrete differential evolution algorithm is proposed to minimize the make-span. A blocking flow shop scheduling problem with *n* jobs and *m* machines is studied in [2-7]. A hybrid particle swarm optimization algorithm(HPSO) was proposed and improved iterated greedy algorithm was applied to get the initial optimized solution, while PSO algorithm was used for global optimization in [2]. In [7], it deals with a scheduling problem of a real-world production process in the metal-working industry. The production process can be described as an offline stochastic flexible flow-shop problem with limited buffers. In [8], the blocking flow shop scheduling problem with the total completion time criterion is resolved based on one branch and bound algorithm. In [9], three algorithms are applied for resolving the flow shop scheduling with blocking to minimize the total flow time.

In this paper, the blocking hybrid flow shop scheduling problem with multi-stages unrelated parallel machines (BHFSP-UPM) is investigated based on genetic algorithm and simulated annealing algorithm. The objective is to plot a processing routing for each job, and to find a permutation of jobs that minimizes the maximum completion time.

2. DESCRIPTION OF BHFSP-UPM

BHFSP-UPM can be defined as follows. *n* jobs are to be scheduled on *p* stages, and each job has the same *p* operations. Each operation is specified by the required machines and the processing time. Each stage involves m_j machines, and the job processing time is unrelated on different machines of the same stage. There is no intermediate buffer during the processing, and the job will not stay on the processing machine until there is free machine on the next stage. Following assumptions are considered for BHFSP-UPM:

1874-4443/14 2014 Bentham Open

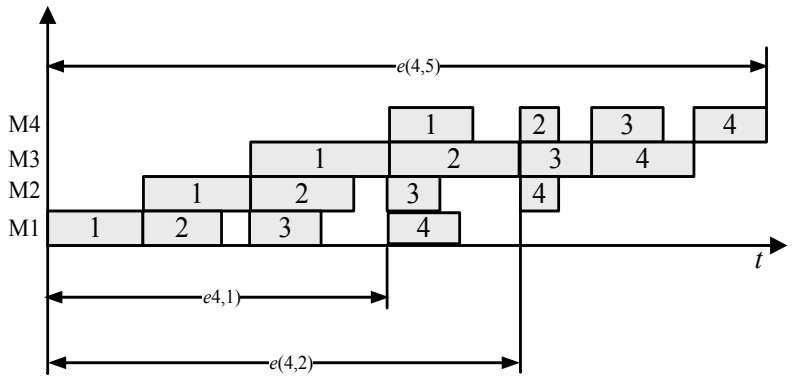

Fig. (1). Computation of $e(\pi(j),k)$ of BHFSP.

- Jobs are available at zero time;
- Job processing cannot be interrupted;
- Jobs are available for processing at a stage immediately after processing completion at the previous stage;
- At least one stage exists parallel machines;
- Parallel machines on the same stage are unrelated in capability and processing rate;
- Each machine can process only one job at the same time;
- Each job must be processed on each stage, but the processing machine is randomly;
- The processing time of each job operation is prespecified;
- A job cannot be processed on more than one machine at the same time;
- Jobs will stay on the machine after processing if the next stage there is no free machine.

3. MATHEMATICAL MODEL OF BHFSP-UPM

The scheduling objective is:

$$\min Z \tag{1}$$

Constrained conditions:

1. the last operation of job *i*

$$T_{ijk} \le Z \tag{2}$$

2. the non-last operation of job *i*

$$T_{ijk} - T_{i(j-1)g} \ge t_{ijk} \quad \forall i,j,g, \quad j \ne 1 \tag{3}$$

3. Operation of job *i* in the processing routing *j* and operation of job *p* in the processing routing *q* both need processing on the machine *m*

$$T_{ijk} - T_{qsk} \ge t_{ijk} \tag{4}$$

$$T_{qsk} - T_{ijk} \ge t_{qsk} \tag{5}$$

4. It is necessary to ensure that only one parallel machine is selected on each stage for all jobs

$$\sum_k X_{ijk} = 1 \tag{6}$$

5. Any operation of job *i* on the machine *k*

$$T_{ijk} > 0 \tag{7}$$

where, Z is the total completing time of the scheduling, and $Z = \max\{C_1, C_1, \cdots, C_n\}$; C_i is the completion time of job i. i represents a job, $i = 1,2,\cdots n$; m_j is the parallel machine number of each stage, $j = 1,2,\cdots p$, p is the total number of stage; t_{ijk} and T_{ijk} represents processing time and completion time of job *i* for its *jth* operation on machine *k*. Equation (2) is a natural constraint to ensure that last operations should be completed before make-span; (3) is to ensure operation *j-1* on machine g precedes the next operation *j* on machine *k* for the same job, (4)and (5) are to ensure that each machine can process at most one job at a time; (4) denotes that operation *s* of job *q* precedes operation *j* of job *i* on the machine *k*; (5) denotes the opposite processing sequence; (6) denotes that on each stage, each job can be processed on at most one machine at any time; (7) is to ensure the completing time of any operation of job *i*.

4. COMPUTATION OF PRODUCTION PERIOD

4.1. Computation Formulas

Due to there are unrelated parallel machines, the computation of production period for BHFSP-UPM is different from BHFSP. The computation formulas of jobs departure time for BHFSP are given in [10, 11]. The comparison graphs between BHFSP and BHFSP-UPM are given in Fig. (**1**) and Fig. (**2**). The production period of BHFSP-UPM is obviously shorter than the period of BHFSP because of the parallel machines. The reasons can be found in Fig. (**1**) and Fig. (**2**). In Fig. (**2**), there are two parallel machines on each stage.

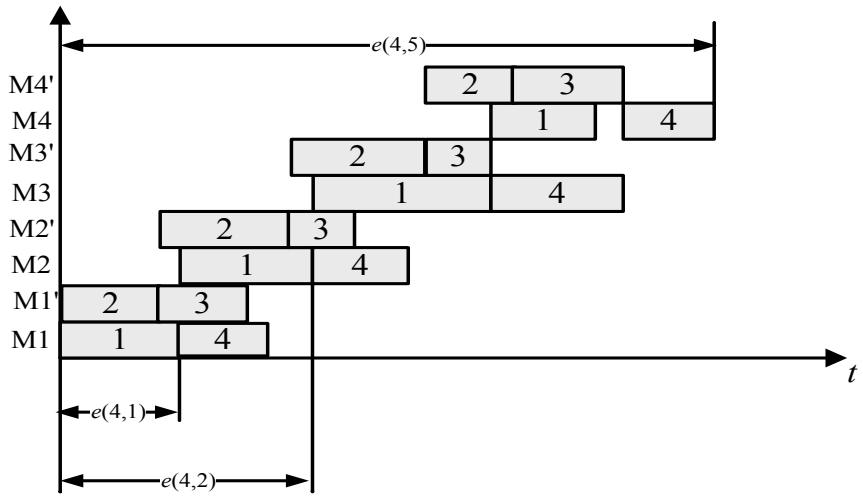

Fig. (2). Computation of $e(\pi(j),k)$ of BHFSP-UPM.

Let a job permutation $\pi=\{\pi(1),\pi(2),\cdots,\pi(n)\}$ represent the schedule of jobs to be processed, and $e(\pi(i),j,m(j))$, i=1,...,n, denotes the starting time of the operation $o(\pi(i),j,m(j))$, j=1,…,p , $m(j)$=1,…, m_j, $o(\pi(i),j,m(j))$ denotes job $\pi(i)$ processed on stage j and the machine $m(j)$, $t(\pi(i),j,m(j))$ is the processing time of $o(\pi(i),j,m(j))$.

The computation formulas of production period for BHFSP:

$$e(\pi(1),1)=0 \tag{8}$$

$$e(\pi(1),k)=e(\pi(1),k-1)+t(\pi(1),k-1)\ \ k=2,...,m. \tag{9}$$

$$e(\pi(i),1)=e(\pi(i-1),2)\ \ j=2,...,n. \tag{10}$$

$$e(\pi(i),k)=\max\{e(\pi(i),k-1)+t(\pi(i),k-1),e(\pi(i-1),k+1)\}i=2,\ldots,n,\ k=2,\ldots,m. \tag{11}$$

$$e(\pi(i),m+1)=e(\pi(i),m)+t(\pi(i),m)\ \ j=2,...,n \tag{12}$$

Where $e(\pi(i),1)$, i=1,…,n, denotes the starting time of job $\pi(i)$ on the first machine. The production period:

$$Z=e(\pi(i),m)+t(\pi(i),m) \tag{13}$$

The computation formulas of production period for BHFSP-UPM are given as follows:

$$e(\pi(1),1,m(1))=0 \tag{14}$$

$$e(\pi(1),j,m(j))=\min(e(\pi(1),j-1,m(j-1))+t(\pi(1),j-1,m(j-1)))$$

$$j=2,\ldots,p,. \tag{15}$$

$$e(\pi(i),j,m(j))=\max\{e(\pi(i),j-1,m(j-1))+t(\pi(i),j-1,m(j-1)),\ \min(e(\pi(s),j+1,m(j+1)))\}$$

s=1,…,i-1, and before processing job $\pi(i)$, job $\pi(s)$ is the last job on each machine.

Z=max($e(\pi(i),p,m(p))+t(\pi(i),p,m(p))$)

4.2. Computation Steps of Production Period

The computation of period for BHFSP-UPM is elaborated in more detail:

Step 1 $e(\pi(1),1,m(1))=0$;

Step 2 Set j=1

Step 3 Set m(j)=1,

$M=t(\pi(1),j,m(j))$.

Step 3.1 m(j)= m(j)+1, if m(j)> m_j go to step 4, otherwise go to step 3.2.

Step 3.2 if $M>t(\pi(1),j,m(1))$, then

$M=t(\pi(1),j,m(j))$, go to step 3.1.

Step 4 $t(\pi(1),j,m(j))=e(\pi(1),j,m(j))+M$, output m(j).

Step 5 update j=j+1, if $j>p$ then go to step 6, otherwise go to step 3.

Step 6 Set j=1, i=2, $e(\pi(i),0,m(0))=0$, $e(\pi(0),j+1,m(j+1))=0$.

Step 7 Set m(j)=1, $N=e(\pi(s),j+1,m(j+1))$.

Step 7.1 m(j)= m(j)+1, if m(j)> m_j then go to step 8, otherwise go to step 7.2.

Step 7.2 if $N>e(\pi(s),j+1,m(j+1))$, then

$N=e(\pi(s),j+1,m(j+1))$, go to step 7.1.

Step 8 $e(\pi(i),j,m(j))=\max\{T(\pi(i),j-1,m(j-1)),N\}$, output m(j).

Step 9 Update j=j+1, if $j>p$ then go to step 10, otherwise go to step 7.

Step 10 Update i=i+1, if $j > n$ then go to step 11, otherwise go to step 7.

Step 11 Stop and output the final sequence, and Z= max($e(\pi(i), p, m(p)) + t(\pi(i), p, m(p))$).

Where $T(\pi(1), j, m(j))$ is the finishing time of operation $o(\pi(1), j, m(j))$.

5. GASA

In theory, GA and SA are both based on probability distributions. However, SA reaches an almost zero probability-jumping phenomenon through endowing a key time variety in searching, and avoiding converging to a local optimum and attaining total optimization. GA realizes optimization through population generic operation based on survival of the fittest. GASA hybrid algorithm is good for searching for an optimum process, enhancing whole and local search ability and efficiency.

The details of GASA are as follows:

- Permutation representation represents a solution of a problem as chromosome.
- Create initial population and fitness function.
- Crossover operation: Before the operation, divide population into K sub-populations (4 sub-populations in simulation): select the optimum individual in each sub-population and crossover with other individuals until producing K new populations, then select the best individual. Adopt MPPX, MGOX, MGPMX1, and MGPMX2 [12] in simulation, which can make the population have obvious diversity.
- Selection. Select optimum chromosome between offspring chromosome produced by crossover in different sub-population and parent.
- Mutation: INV mutation is used to produce small perturbations on chromosomes.
- Metropolis sample process. The stochastic rule, in which the probabilistic decision is made to prevent the optimization process from sticking at a possible local minima, is the main idea behind simulated annealing. The probability of accepting a new individual under the circumstances: min {1, exp(- Δ /t)}>random[0,1] (here t denotes the temperature, Δ denotes difference of aim value between new and old).
- Enhancing memory ability: In order to avoid losing the current optimum solution in the search process, save the current optimum solution through adding memory.
- Cooling scheme: Exponential cooling, $t_k = \lambda t_{k-1}$, and λ =0.9 in simulation.
- Criterion of temperature changing and algorithm termination: In order to adapt dynamic variety of algorithm performance, pay attention to optimum and time performance, adopt two criteria of temperature changing and algorithm termination. In the optimizing process, if the optimum value remains constant through 30 eras then start cooling; if the optimum value remains constant through cooling 30 iterations then stop searching, the optimum value is the solution.

6. CASE STUDY AND ANALYSIS

In order to test the validity and feasibility of GASA and the method proposed in this paper, two cases in [11] are applied.

6.1. Case 1

There are three stages, in each stage, the parallel machines are 3, 2, 4, respectively. The detailed information of jobs and machines are given in Table **1**. Firstly, the GASA algorithm is tested and compared to GA [13], SFLA [14], and EDA [11]. The scheduling results are obtained based on GASA with minimizing the production period as function objective. The algorithm parameters are that population size is 100, crossover rate is 0.8, mutation rate is 0.01. Run the simulation 10 times, and the comparison results given in Table **2**.

In [11, 13, 14], the results are corresponding to the hybrid flow shop scheduling problem with intermediate buffer and unrelated parallel machines. So the results for above problem are given in Table **2**. From Table **2**, it shows that not only GASA has better solutions than conventional GA and SFLA, but also the stability of GASA is much better than that of GA, the optimal solution can be found the production period Z=23. The Gantt graph of one of the optimal results is given in Fig. (**3**). Fig. (**4**) is Gantt graph of one scheduling result of BHFSP-UPM.

6.2. Case 2

In order to further test the validity of GASA algorithm, a large scale hybrid scheduling problem is applied. There are four stages, in each stage, the parallel machines are 3, 3, 2, 2, respectively. The detailed information of jobs and machines are given in Table **3**.

Run the simulation 10 times and compared with algorithm EDA [11], GA [13], SFLA [14]. In [13], it only gave one result, 10 results are given based on other algorithms. The comparisons are given in table 4. From Table **4**, it shows that not only GASA has better solutions than conventional GA and SFLA, but also the stability of GASA is much better than that of GA, the optimal solution can be found Z=299. Compared with EDA [11], the results performance of GASA is bit poor than that of EDA. The Gantt graph of optimal result is given in Fig. (**5**).

Table 1. Processing information of jobs.

Job	Stage 1			Stage 2		Stage 3			
	M1	M2	M3	M4	M5	M6	M7	M8	M9
1	2	2	3	4	5	2	3	2	3
2	4	5	4	3	4	3	4	5	4
3	6	5	4	4	2	3	4	2	5
4	4	3	4	6	5	3	6	5	8
5	4	5	3	3	1	3	4	6	5
6	6	5	4	2	3	4	3	9	5
7	5	2	4	4	6	3	4	3	5
8	3	5	4	7	5	3	3	6	4
9	2	5	4	1	2	7	8	6	5
10	3	6	4	3	4	4	8	6	7
11	5	2	4	3	5	6	7	6	5
12	6	5	4	5	4	4	4	7	5

Table 2. Scheduling results and comparisons.

No.	1	2	3	4	5	6	7	8	9	10
GA	30	27	26	27	29	27	26	27	26	28
SFLA	24	24	24	24	24	24	24	24	24	24
EDA	23	24	23	23	23	23	24	24	23	24
GASA	23	23	23	24	24	23	24	24	23	23

Table 3. Processing information of jobs.

Job	Stage 1			Stage 2			Stage 3		Stage 4	
	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
1	45	48	50	35	35	30	30	35	25	26
2	45	50	45	35	36	35	35	34	25	30
3	50	45	46	35	36	36	31	34	30	31
4	50	48	48	34	38	35	32	33	27	31

Table 3. contd…

Job	Stage 1			Stage 2			Stage 3		Stage 4	
	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10
5	45	46	48	30	35	50	34	32	28	31
6	45	45	45	30	35	50	33	32	30	26
7	47	50	47	31	30	35	35	31	29	25
8	50	45	48	32	30	34	34	30	24	27
9	48	46	46	33	34	30	34	30	25	25
10	45	47	47	33	33	30	35	34	32	26
11	46	50	45	34	30	50	30	35	31	25
12	48	50	47	35	31	35	32	30	25	30

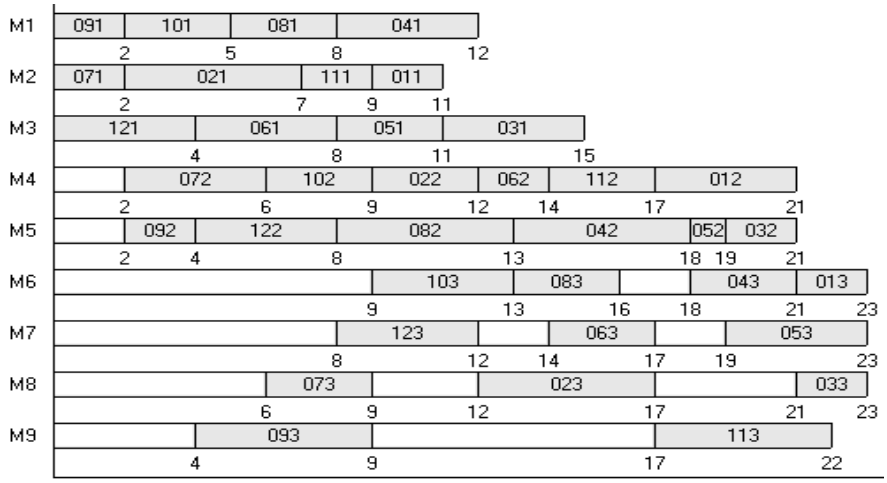

Fig. (3). Gantt graph with intermediate buffer.

Table 4. Scheduling results and comparisons.

No.	1	2	3	4	5	6	7	8	9	10
GA	347									
SFLA	297	313	297	297	313	310	313	311	316	306
EDA	297	297	297	297	298	297	297	298	298	298
GASA	299	299	299	299	301	299	299	301	299	299

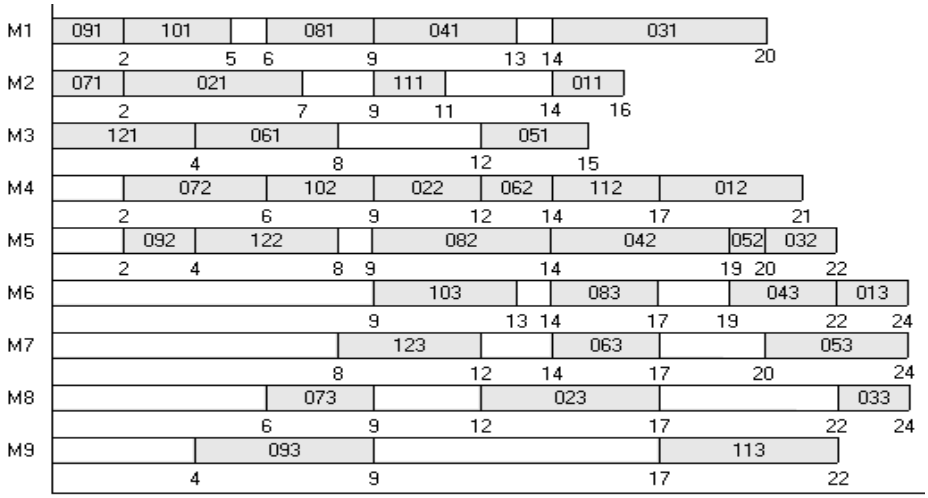

Fig. (4). Gantt graph of BHFSP-UPM.

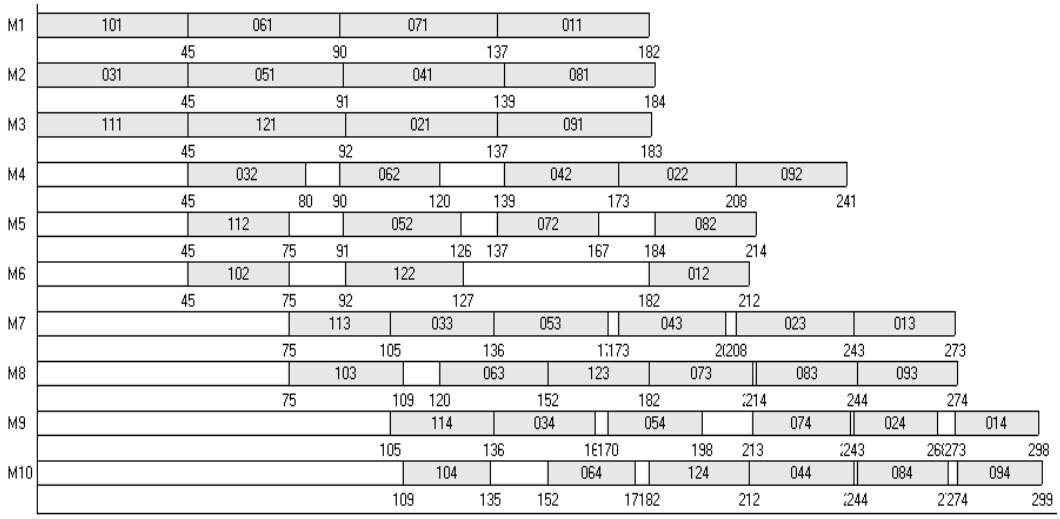

Fig. (5). Gantt graph with intermediate buffer.

The validity and feasibility of GASA has been verified through case 1 and case 2. And then the hybrid flow shop scheduling with no intermediate buffer, and there are unrelated parallel machines is tested based on the math model and the computation method of period proposed in this paper. The case is the same with case one. The comparison Gantt graphs between hybrid flow shop scheduling with intermediate buffer and BHFSP-UPM are given in Fig. (**3**) and Fig. (**4**), respectively. The optimal result of BHFSP-UPM is 24. Due to there is no intermediate buffer, the free time of machines is longer than that results with buffer in Fig. (**3**). In Fig. (**3**), the free time of machine 1, 2, and 3 are all 0, but in Fig. (**4**), the free time of corresponding machine 1, 2, and 3 are 2, 5, and 4, respectively. The starting time of the first operation of job 8 in machine 1 is 5 in Fig. (**3**), but in Fig. (**4**) the starting time is 6. Because the former job 10 is not on the machine 1 until its next operation begins to process on time 6. Compared to the scheduling in Fig. (**3**), there is no intermediate buffer in Fig. (**4**). The first processing machine of job 3 adjusted from machine 3 in Fig. (**3**) to machine 1 in Fig. (**4**), the operational time of machine 3 is 19, although the finishing time of operation 051 is 15. But the operational time of machine 1 is 14, although the processing time of job 3 is 6 on machine 1, it is a bit longer than the processing time on machine 3, the finishing time is 20. In machine 3, the finishing time of job 3 is 23.

From case 1 and case 2, the GASA algorithm and the computation method of scheduling method is feasible for BHFSP-UPM. Compared with other existing algorithms, the results performance obtained with the method proposed in this paper is higher.

CONCLUSION

In this paper, a blocking hybrid flow shop scheduling problem with multi-stage unrelated parallel machines(BHFSP-UPM) is studied. The math model is constructed considered that there is no intermediate. The processing time in parallel machines is different according to the conditions of jobs and machines. The evolution of processing sequence decision can be obtained using a GASA with production period as the criterion based on the computation method of production period. When these factors are

considered during scheduling, generally, a more effective schedule can be obtained. In order to test the performance of the method, scheduling results based on two cases and comparisons with some existing algorithms show that the method proposed in this paper is effective and feasible.

CONFLICT OF INTEREST

The author confirms that this article content has no conflict of interest.

ACKNOWLEDGEMENTS

Project supported by the education department of Liaoning Province (L2012067), and program for Liaoning Excellent Talents in University (No.LJQ2012019).

REFERENCES

[1] Q. K. Pan, L. Wang, L. Gao, and W. D. Li, "An effective hybrid discrete differential evolution algorithm", *Information Sciences*, vol. 181, pp. 668-685, 2011.

[2] Q. L. Zhang, and Y. S. Chen, "Effective hybrid particle swarm optimization algorithm for the blocking flow shop scheduling problem", *Information and Control*, vol. 42, no. 2, pp. 252-257, 2013.

[3] J. J. Liang, Q. K. Pan, T. J. Chen, and L. Wang, "Solving the blocking flow shop scheduling problem by a dynamic multi-swarm particle swarm optimizer", The *International Journal of Advanced Manufacturing Technology*, vol. 55, pp. 755-762, 2011.

[4] W. Trabelsi, C. Sauvey, and N. Sauer, "Heuristics and metaheuristics for mixed blocking constraints flow shop scheduling problems", *Computers & Operations Research*, vol. 39, pp. 2520-2527, 2012.

[5] A. Elmi, S. Topaloglu, "A scheduling problem in blocking hybrid flow shop robotic cells with multiple robots", *Computers & Operations Research*, vol.40, pp. 2543-2555, 2013.

[6] L. Wang, Q. K. Pan, P. N. Suganthan, W. H. Wang, and Y. M. Wang, "A novel hybrid discrete differential evolution algorithm for blocking flow shop scheduling problems", *Computers & Operations Research*, vol. 37, pp. 509-520, 2010.

[7] Q. K. Pan, L.Wang, and L. Gao, "A chaotic harmony search algorithm for the flow shop scheduling problem with buffers limited buffers", *Applied Soft Computing*, vol. 11, pp. 5270-5280, 2011.

[8] C. Almeder, R. F. Hartl, "A metaheuristic optimization approach for a real-world stochastic flexible flow shop problem with limited buffer", *International Journal of Production Economics,* vol. 145, pp. 88-95, 2013.

[9] G. Moslehi, and D. Khorasanian, "Optimizing blocking flow shop scheduling problem with total completion time criterion", *Computers & Operations Research*, vol. 40, pp. 1874-1883, 2013.

[10] L. Wang, Q. K. Pan, M. FatihTasgetiren, "Minimizing the total flow time in a flow shop with blocking by using hybrid harmony search algorithms", *Expert Systems with Applications*, vol. 37, pp. 7929-7936, 2010.

[11] S. Y. Wang, L. Wang, Y. Xu, and G. Zhou, "An estimation of distribution algorithm forsolving hybrid flow-shop scheduling problem", *Acta Automatic Sinica*, vol. 38, no. 3, pp. 437-443, 2012.

[12] J. Byung, R Hyung., and S. Hyun, "A hybrid genetic algorithm for the job shop scheduling problems", *Computers & Industrial Engineering*, vol.45, pp. 597-613,2003.

[13] H.R. Zhou, W. S. Tang, and Y. H. Wei, "Optimize flexible flow-shop scheduling using genetic algorithm", *Compute Engineering and Applications*, vol. 45, no. 30, pp. 224-226, 2009.

[14] Y. Xu, L. Wang, G. Zhou, and S. Y. Wang, "An effective shuffled frog leaping algorithm for solving hybrid flow-shop scheduling problem", In: *Proceedings of the International Conference on Intelligent Computing*, 2011, pp. 560-567.

Received: November 22, 2014 Revised: January 06, 2015 Accepted: January 21, 2015