Int j simul model 18 (2019) 4, 689-698
ISSN 1726-4529 Original scientific paper

BATCH OPTIMIZATION IN INTEGRATED SCHEDULING OF MACHINING AND ASSEMBLY

Yang, M. S.[#]; Ba, L.[#]; Xu, E. B.; Li, Y.; Gao, X. Q.; Liu, Y. & Li, Y.
School of Mechanical and Precision Instrument Engineering, Xi'an University of Technology, Xi'an 710048, China
E-Mail: yangmingshun@xaut.edu.cn, xautbali@163.com ([#] Corresponding authors)

Abstract
To survive the fierce market competition, many manufacturing enterprises have applied integration measures at all levels of the production process. Against this backdrop, this paper mainly establishes a job-shop scheduling problem (JSP) that aims to minimize the makespan of products through integrated scheduling of machining and assembly under batch production environment. Then, the established problem was modelled considering the influence of different batch number on the makespan. In view of the complexity and discreteness of the established problem, a genetic algorithm (GA) was designed to obtain the optimal scheduling sequence in different batch production situations. The effectiveness of our problem model and algorithm was verified through the analysis on an example with different batch conditions. The research findings help to design a realistic and feasible scheduling plan for manufacturing enterprises.
(Received, processed and accepted by the Chinese Representative Office.)

Key Words: Integration of Machining and Assembling, Equal-Batch Splitting, Genetic Algorithm (GA), Batch Production

1. INTRODUCTION

With the growing demand for personalized products, manufacturing enterprises are engaged in an increasingly fierce competition in the market [1, 2]. To survive the market competition, many manufacturing enterprises have applied integration measures at all levels of the production process, ranging from the integration between planning and scheduling to that between production and transport. Among the various integration measures, the integrated scheduling of machining and assembly greatly improves the production efficiency, shortens the production cycle, and reduces the production cost, compared with the traditional separated scheduling of the two operations.

The job-shop scheduling (JSP) is a hotspot in the research of production scheduling. Let n be the number of jobs and m be the number of machines. Assuming that each job has a unique process route, the basic JSP [3, 4] aims to minimize the makespan by adjusting the machining sequence of jobs on each machine, without violating the process route of any job.

So far, much research has been done on the JSP. For example, Sharma et al. [5] improved the artificial bee colony (ABC) algorithm for basic JSPs, and verified its superiority with multiple standard sets. Piroozfard et al. [6] designed an improved biogeography-based optimization (BBO) algorithm, carried out a comparative analysis on standard examples, and proved that the improved BBO outperformed contrastive algorithms like greedy randomized adaptive search procedure (GRASP), parallel genetic algorithm (PGA) and heuristics genetic algorithm (HGA). Akram et al. [7] developed a fast-simulated annealing (SA) algorithm for basic JSPs, which effectively avoids the local optimum trap. Amirghasemi and Zamani [8] put forward an improved genetic algorithm (GA) to solve the JSP.

With the development of manufacturing technology, many multi-functional computer numerical control (CNC) machine tools have emerged. Such a machine tool supports multiple machining operations, such as turning and milling. Therefore, multiple machines are available for the same machining operation of jobs. The one-to-many relationship between operation

and machine has been introduced to the basic JSP, creating a new branch of the JSP called flexible job-shop scheduling problem (FJSP). Considering energy conservation, Mokhtari and Hasani [1] proposed an improved evolutionary algorithm to solve a multi-objective FJSP. Xiao et al. [2] set up a new hybrid Petri network model based on the common hybrid Petri network and with a combination of differential Petri net and controlled Petri net. Kato et al. [9] set up a model of a multi-objective FJSP, and created a hybrid particle swarm optimization (PSO) algorithm to minimize the makespan, maximum machine load and total load of the problem. Wu and Sun [10] modelled the FJSP for the minimal makespan and energy consumption, and solved the problem with a non-dominated sorting genetic algorithm (NASGA) with heuristic rules. Focusing on the machining time, Lin [11] established a model of the FJSP, and solved the problem with a hyper heuristic algorithm based on backtracking search.

In addition, many new factors have been introduced to the basic JSP. For instance, Kuhpfahl and Bierwirth [12] included the delivery date in the JSP, constructed a model of the problem for the minimal total tardiness penalty, and solved the model with an improved local search algorithm. Chaouch et al. [13] presented an improved ant colony optimization (ACO) algorithm to tackle a distributed JSP. Taking machines and operators as resources, Li et al. [14] modelled a parallel-resource JSP, and proposed an improved GA to solve the problem. Considering delivery time, Yazdani et al. [15] established a JSP for the minimal early/tardiness penalty, and solved the problem model with an improved competition algorithm. Kundakci and Kulak [16] explored a JSP with random insertion, machine failure and variable working hours. Bierwirth and Kuhpfahl [17] built a model of the JSP for the minimal tardiness penalty, and extended the GRASP to solve the model.

There are two main defects of the existing studies on the JSPs. First, the basic JSP only pursues the minimal makespan in the machining process, while the machining scheduling and assembly scheduling in actual production are closely intertwined. The machining scheduling focuses on machine control in the job shop, while the assembly scheduling highlights the resource optimization in the assembly line. Second, the existing JSP models rarely consider the batch factor, which is a common production mode in actual production. To overcome the defects, this paper establishes a JSP for the minimal makespan through integrated scheduling of machining and assembly in batch production environment. Then, the problem was modelled and solve by a GA, aiming to optimize the machining sequence under different batch conditions.

2. PROBLEM DESCRIPTION

In the established problem, each product can be broken down into various parts. These parts need to be machined through multiple operations, and then assembled into components. The components will in turn be assembled into products. During the production, the parts are machined in batches. The production process is constrained by multiple factors, namely, operation sequence and machine resources.

To minimize the makespan through integrated scheduling of machining and assembly, the equal-batch splitting strategy was adopted to determine the batch of each part on the machining line and at each assembly node, and identify the machining sequence of each batch of parts.

For the minimal makespan of the products, the optimal scheduling plan should be obtained by determining the number, size and machining sequence of parts in batch production, considering the preparation time in parts machining.

3. MODEL CONSTRUCTION

The following symbols were defined before setting up a minimum makespan model for integrated scheduling of machining and assembly in batch production environment: makespan (MS); the total batch size of part i (N_i); the batch of type i parts (P_i); the batch number corresponding to sub-batch b of part i (PC_{ib}); the smallest batch of parts (PC_i^{min}); the number of part types (n); the number of assembly nodes (m); the number of machines (p); the number of machining operations for part i (q_i); the makespan of machining operation j of sub-batch b of part i on machine k (EM_{ibjk}); the makespan of sub-batch b of assembly node j (EA_{jb}); the Boolean variable about whether machining operation j of sub-batch b of part i is implemented on machine k (O_{ibjk}) (if yes, $O_{ibjk} = 1$; otherwise, $O_{ibjk} = 0$); the unit machining time of part i on machine k ($TM_{(i,k)}$); the machining time of sub-batch b of part i on machine k (TM_{ibk}); the completion time of the assembly of sub-batch b on assembly node j (TA_{jb}); the preparation time for machining operation j of sub-batch b of part i on machine k (BF_{ibjk}); the type of part i on machine k (SN_{ki}); the start time of sub-batch b of part i on machine k (SM_{ibk}); the start time of sub-batch b of part i on assembly node j (SA_{jb}); the machining sequence of sub-batch a and sub-batch b of part i on machine k (SF_{iabk}); the Boolean variable about the machining sequence of sub-batch b and sub-batch j of part i on machine h and machine k (X_{ibnk}); the Boolean variable about the machining sequence of sub-batch b and sub-batch j of part i on machine k (Y_{ijk}); the Boolean variable about whether part i is processed on assembly node j (RA_j).

Considering the influence of the number of batches of the part being machines/assembled on the scheduling result, the minimum makespan model for integrated scheduling of machining and assembly in batch production environment can be established as follows:

Objective function:

$$\text{Min}(MS) = \text{Min}(\max(PC_{jb} \times TA_{jb})) \tag{1}$$

Constraints:

(1) The relationship between the total number of parts and the sub-batch of parts:

$$\sum_{b=1}^{p_i} PC_{ib} = N_i \tag{2}$$

(2) The makespan of machining operation j in sub-batch b of part i on equipment k:

$$EM_{ibjk} = \sum_{j=1}^{q_i} O_{ibjk} \times TM_{ibjk} \tag{3}$$

$$TM_{ibjk} = PC_{ib} \times T_{M(i,k)} \tag{4}$$

(3) Machining sequence of parts in different batches on the same machine:

$$EM_{ibjk} - \sum_{j=1}^{q_i} O_{ibjk} \times TM_{ibjk} - BF_{ibjk} + M(1 - SF_{iabk}) \geq EM_{iajk} \tag{5}$$

It is assumed that sub-batch a and sub-batch b of part i on equipment k are machined one after the other, i.e. $SN_{ki} = i_a$, $SN_{k(i+1)} = i_b$ and $BF_{ibjk} = 0$.

(4) Machining sequence of the same part in the same batch on different machines:

$$EM_{ibk} - \sum_{j=1}^{q_i} O_{ibjk} \times TM_{ibjk} + M \times (1 - X_{ibhk}) \geq EM_{ibh} \tag{6}$$

(5) The same machining operation of different batches of parts cannot be processed on multiple machines at the same time:

$$\sum_{k=1}^{p} O_{ibjk} = 1, \forall i, j \tag{7}$$

(6) The multiple machining operations of a batch of parts cannot be processed on the same machine at the same time:

$$\sum_{j=1}^{q_i} O_{ibjk} = 1, \forall i, k \tag{8}$$

(7) Assembly node *j* cannot be started before the required parts are completely machined:

$$PCA_j = (1 - RA_j) \times PC_i^{\min} \tag{9}$$

$$PC_i^{\min} = \min\{PC_i^1, PC_i^2, PC_i^3 ... PC_i^n\} \tag{10}$$

(8) Equal-batch splitting constraint on batch size and the total number of products:

$$PC_{ib} = \text{fix}(\frac{N_i}{p_i}), \quad b = 1, 2, \cdots, p_i - 1 \tag{11}$$

$$PC_{ib} = \text{fix}(\frac{N_i}{p_i}) + \text{mod}(\frac{N_i}{p_i}), \quad b = p_i \tag{12}$$

4. ALGORITHM DESIGN

This paper decides to design a GA to solve the complexity and discreteness of the established problem. The key steps of the algorithm design are introduced in this section.

4.1 Coding

The coding method was designed based on the type and batch of parts. Each chromosome represents the batch type, the number of sub-batches and the scheduling sequence of the sub-batches of each part. In each chromosome, a gene is expressed by two digits. For example, "31" means the first sub-batch of the third type of part. Besides, the sequence of numbers in a chromosome stands for the machining sequence and the machining operation of each sub-batch of the part. As shown in Table I, the chromosome {21 11 31 22 21 12 13 32 31 31 32 21 12 11 22 12 32 22} describes the machining of three types of parts. Each type of part is divided into two batches of machining. Each part needs to receive three machining operations. The first 21 represents the first machining operation of the first sub-batch of the second type of part, and the second 21 represents the second machining operation of the first sub-batch of the second type of part. The rest can be deduced by analogy.

Table I: An example chromosome.

Machining sequence of sub-batch	21-1	11-1	31-1	22-1	21-2	12-1	13-1	32-1	31-1
Gene	21	11	31	22	21	13	32	31	31
Machining sequence of sub-batch	31-2	32-2	21-3	12-2	11-2	22-2	12-3	11-3	22-3
Gene	32	21	12	11	22	22	12	11	22

4.2 Crossover

To prevent infeasible solutions, a POX crossover operator [18] was employed to randomly generate a part type number in two parent chromosomes. Once the positions of all batches of part are determined, these positions were directly swapped in one child chromosome. Then, the other child chromosome with other genes was padded (Fig. 1).

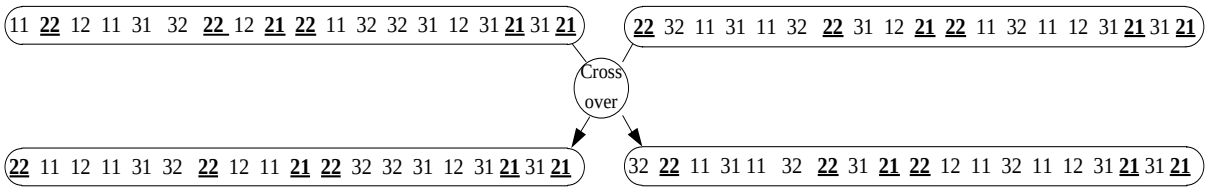

Figure 1: Crossover operator of batch scheduling.

4.3 Verification of feasible solution

In our problem, there are sequential constraints between assembly batches and machining batches of parts. To eliminate infeasible solutions, it is necessary to verify the feasibility of each solution in the following steps:

Step 1. Traverse all positions of the chromosome to find the final product and batch to be assembled.

Step 2. Judge whether the part number is in the last position of the chromosome. If not, the part number is swapped with that of the part at the last position; if yes, go to Step 3.

Step 3. Find the type and batch number of the part in the second to last position and record the current position of the part number, denoted as P.

Step 4. Determine whether the parts required for the assembly position have been completed before P. If not, go to Step 5; if yes, terminate the judgement process.

Step 5. Adjust the positions of all parts that do not meet sequential constraints until all subparts are completed before assembly.

4.4 Parameter definition

The parameters of the GA as defined as follows: population size (*N*); coding length of chromosome (*Lchrom*); crossover probability (*Crossover*); mutation probability (*Mutr*); number of iterations (*Generation*). The workflow of the proposed GA is illustrated in Fig. 2.

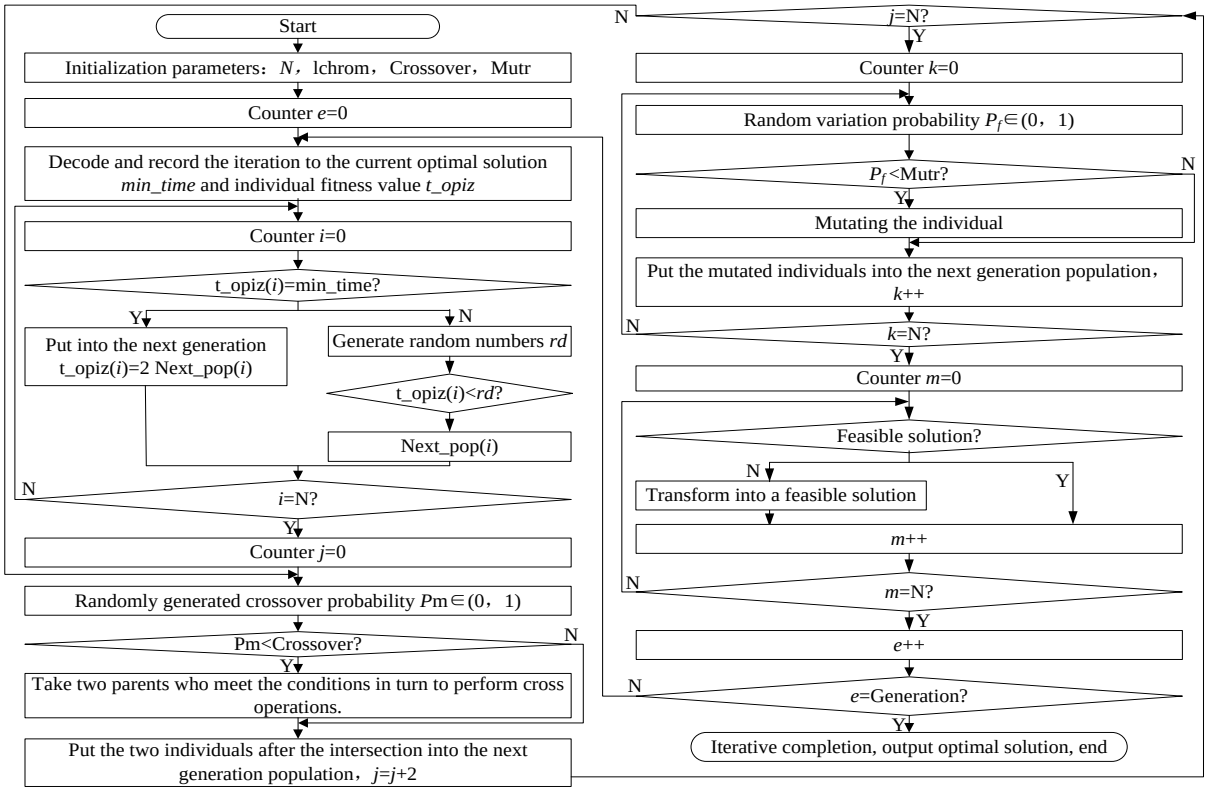

Figure 2: The workflow of the GA.

5. EXAMPLE VERIFICATION AND RESULTS ANALYSIS

This section aims to verify the effectiveness of the proposed algorithm in solving the established problem with real-world examples. The structure of the example product is shown in Fig. 3, which needs to be produced in 12 batches. The relationship between parts, components and machines are displayed in Table II. It can be seen that there is a total of five parts, J1-J5, each of which needs go through five machining operations. Besides, there are three components: P6-P8, five machines: M1-M5, and four assembly nodes: M6-M9. The assembly cannot start before the corresponding parts have been machined. Assembly and machining of different parts can take place concurrently.

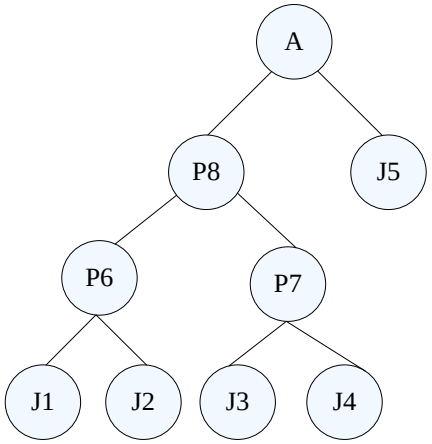

Figure 3: The product structure.

Table II: The relationship between parts, components and machines.

Parts	**Processing and assembly machine tools**								
	M1	M2	M3	M4	M5	M6	M7	M8	M9
job1	3	6	4	7	6	0	0	0	0
job2	10	8	5	4	10	0	0	0	0
job3	9	5	5	4	7	0	0	0	0
job4	5	3	9	3	5	0	0	0	0
job5	10	3	5	3	4	0	0	0	0
part6	0	0	0	0	0	20	0	0	0
part7	0	0	0	0	0	0	25	0	0
part8	0	0	0	0	0	0	0	30	0
A	0	0	0	0	0	0	0	0	35

Table III: The machine for each product or part.

Parts	**Processing procedure**				
	1	2	3	4	5
job1	M3	M1	M2	M4	M5
job2	M2	M3	M5	M1	M4
job3	M3	M4	M1	M2	M5
job4	M2	M1	M3	M4	M5
job5	M3	M2	M5	M1	M4
Components and products	**Assembly procedure**				
	1	2	3	4	5
part6	M6	M6	M6	M6	M6
part7	M7	M7	M7	M7	M7
part8	M8	M8	M8	M8	M8
A	M9	M9	M9	M9	M9

5.1 Hypotheses

(1) Each part that has been machined is transported directly to the assembly node for assemblage.
(2) Each machine can only process one batch of parts at a time.
(3) The parts must be processed according to the machining sequence. No machining operation can be skipped.
(4) The products can wait between machining operations. The machines are idle if no product arrives.
(5) The machines never break down.

5.2 Results analysis

The GA parameters were configured as: population size, 35; number of iterations, 500; crossover probability, 0.9; mutation probability, 0.1. The product was divided into equal batches, and the GA was programmed on the Matlab. The optimal solution is given in Fig. 4. The start time of machining, corresponding machine and makespan of each component and assembly in the optimal schedule are shown in Fig. 5.

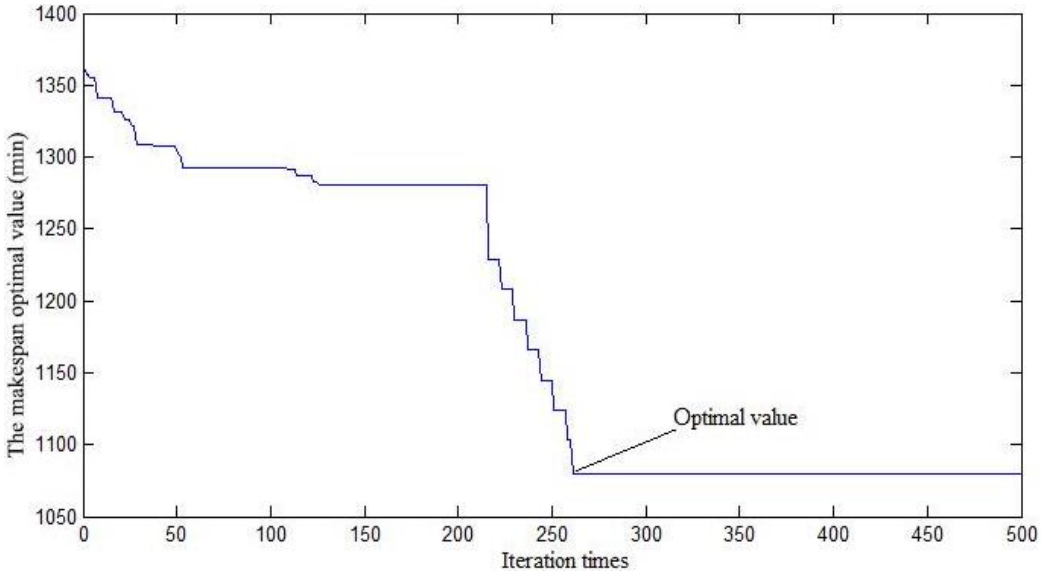

Figure 4: The optimal makespan curve of two-batch condition.

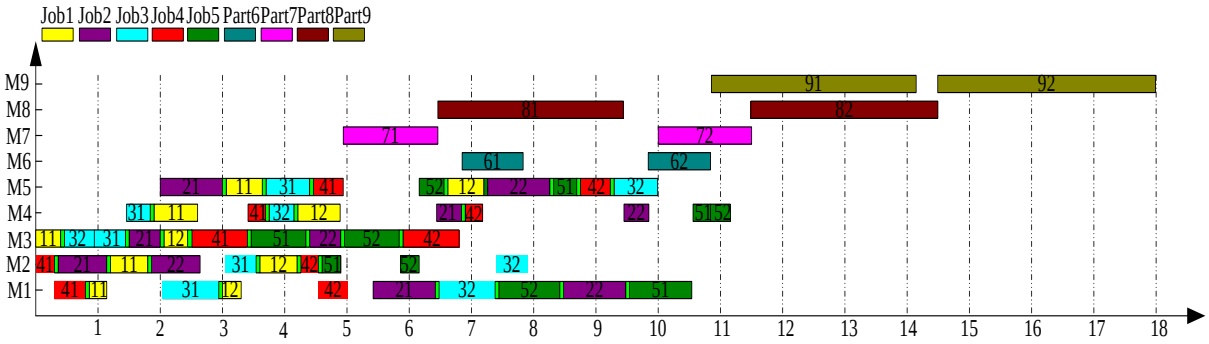

Figure 5: The Gantt chart of the scheduling of two equal batches (h).

From the above results, it can be seen that, under two equal batches, the optimal makespan was 1,080 min, i.e. 18 h, and the optimal solution was {11 41 21 32 41 11 31 11 21 31 11 22 21 12 21 22 31 12 41 51 31 12 42 51 22 52 42 41 32 12 11 31 41 71 12 52 42 21 61 32 42 32 52 12 22 51 22 62 52 42 32 72 51 52 81 82 91 92}.

Without changing the algorithm parameters, the three-batch and one-batch conditions were simulated. The results are as shown in Table IV and Figs. 6 to 9.

Table IV: The optimal solutions of various batch conditions.

Batch strategy	**Optimal makespan** (min)
Equal quantity /3 batches	1,159
Equal quantity /2 batches	1,080
One-batch	1,602

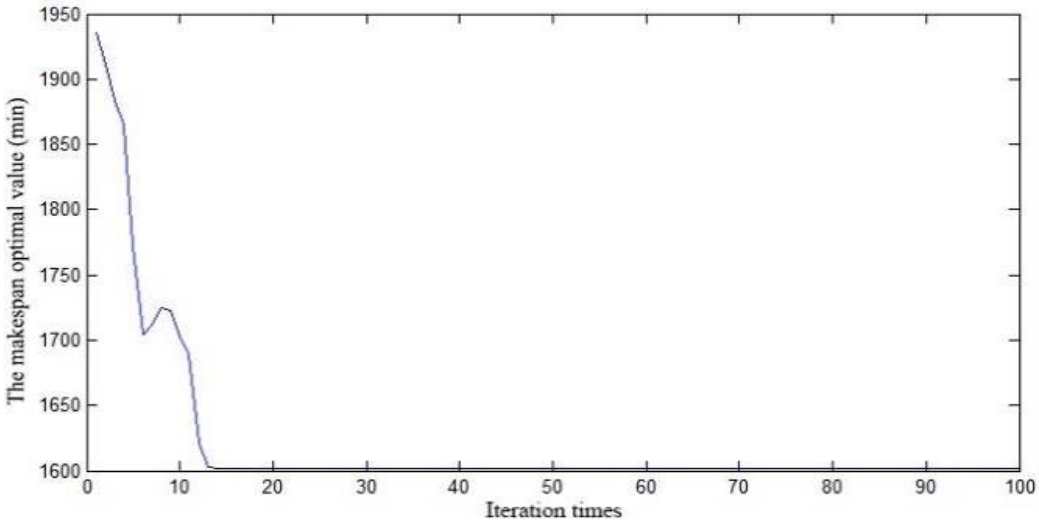

Figure 6: The optimal makespan curve of one-batch condition (optimal value: 1,602 min).

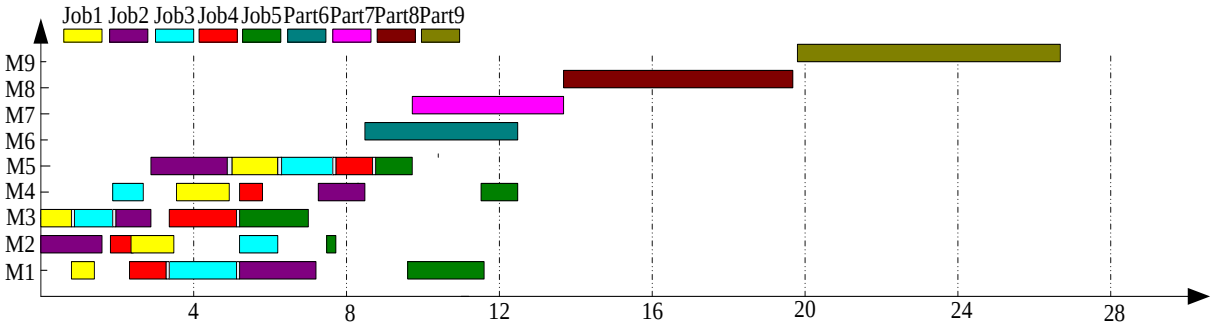

Figure 7: The Gantt chart of the scheduling of one-batch condition.

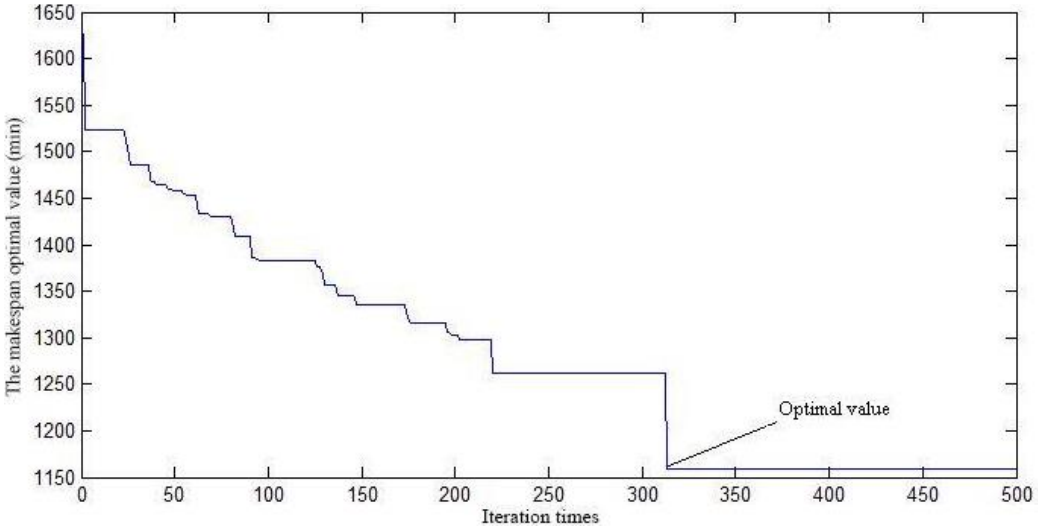

Figure 8: The optimal makespan curve of equal batch conditions.

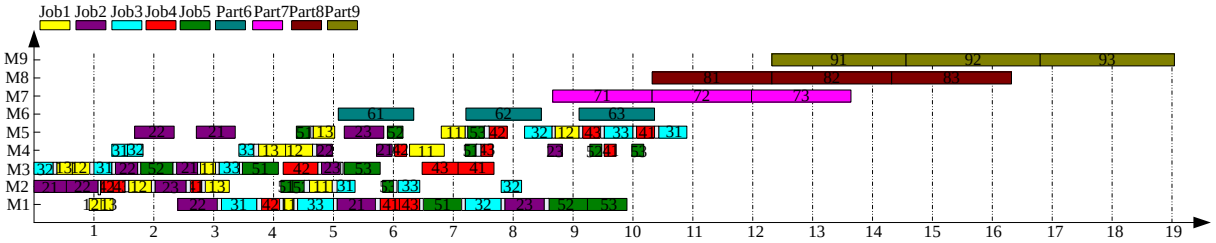

Figure 9: The Gantt chart of the scheduling of equal batch conditions.

According to the simulation results, the author compared the one-batch condition with equal batch conditions.

(1) The optimal makespan of the no batch condition was inferior than the equal batch conditions. If the products are not processed in batches, some machines will work continuously for a long time. A machined part cannot enter the assembly line until the entire batch is machined. Under equal batch conditions, each machine works for a shorter time, and different parts of the same product are processed at the same time. In this way, the makespan is effectively shortened.

(2) The optimal makespan of two-batch condition is better than that of three-batch condition. With the increase in the number of batches, there is a dramatic increase in the number of feasible solutions. In this case, the GA may fall into the local optimum trap.

6. CONCLUSIONS

Considering the integration between machining and assembly, this paper establishes a JSP that aims to minimize the makespan through integrated scheduling of the two operations in batch production environment. Next, a GA was designed in detail to overcome the complexity and discreteness of the established problem. The effectiveness of our model and algorithm was verified through example analysis. The results show that equal-batch scheduling brought shorter makespan than one-batch scheduling. The future research will improve the GA for our JSP, aiming to create a scheduling plan with better quality.

ACKNOWLEDGEMENTS

This research is supported by the National Natural Science Foundation of China (Grant No: 61402361, 60903124); project supported by the scientific research project of Shaanxi Provincial Department of Education (Grant No: 14JK1521); Shaanxi province science and technology research and development project (Grant No: 2012KJXX-34); Xi'an University of Technology Initial Foundation for the PhDs (Grant No: 102-451117013).

REFERENCES

[1] Mokhtari, H.; Hasani, A. (2017). An energy-efficient multi-objective optimization for flexible job-shop scheduling problem, *Computers & Chemical Engineering*, Vol. 104, 339-352, doi:10.1016/j.compchemeng.2017.05.004

[2] Xiao, N.; Ni, C. D.; Guo, S. J. (2017). Modelling and simulation for production logistics system in industrial enterprises based on hybrid network, *International Journal of Simulation Modelling*, Vol. 16, No. 1, 157-166, doi:10.2507/IJSIMM16(1)CO3

[3] Shahrabi, J.; Adibib, M. A.; Mahootchi, M. (2017). A reinforcement learning approach to parameter estimation in dynamic job shop scheduling, *Computers & Industrial Engineering*, Vol. 110, 75-82, doi:10.1016/j.cie.2017.05.026

[4] Chen, Q.; Deng, L. F.; Wang, H. M. (2018). Optimization of multi-task job-shop scheduling based on uncertainty theory algorithm, *International Journal of Simulation Modelling*, Vol. 17, No. 3, 543-552, doi:10.2507/IJSIMM17(3)CO14

[5] Sharma, N.; Sharma, H.; Sharma, A. (2018). Beer froth artificial bee colony algorithm for job-shop scheduling problem, *Applied Soft Computing*, Vol. 68, 507-524, doi:10.1016/j.asoc.2018.04.001

[6] Piroozfard, H.; Kuan, Y. W.; Asl, A. D. (2017). An improved biogeography-based optimization for achieving optimal job shop scheduling solutions, *Procedia Computer Science*, Vol. 115, 30-38, doi:10.1016/j.procs.2017.09.073

[7] Akram, K.; Kamal, K.; Zeb, A. (2016). Fast simulated annealing hybridized with quenching for solving job shop scheduling problem, *Applied Soft Computing*, Vol. 49, 510-523, doi:10.1016/j.asoc.2016.08.037

[8] Amirghasemi, M.; Zamani, R. (2015). An effective asexual genetic algorithm for solving the job shop scheduling problem, *Computers & Industrial Engineering*, Vol. 83, 123-138, doi:10.1016/j.cie.2015.02.011

[9] Kato, E. R. R.; de Aguiar Aranha, G. D.; Tsunaki, R. H. (2018). A new approach to solve the flexible job shop problem based on a hybrid particle swarm optimization and random-restart hill climbing, *Computers & Industrial Engineering*, Vol. 125, 178-189, doi:10.1016/j.cie.2018.08.022

[10] Wu, X.; Sun, Y. (2018). A green scheduling algorithm for flexible job shop with energy-saving measures, *Journal of Cleaner Production*, Vol. 172, 3249-3264, doi:10.1016/j.jclepro.2017.10.342

[11] Lin, J. (2019). Backtracking search based hyper-heuristic for the flexible job-shop scheduling problem with fuzzy processing time, *Engineering Applications of Artificial Intelligence*, Vol. 77, 186-196, doi:10.1016/j.engappai.2018.10.008

[12] Kuhpfahl, J.; Bierwirth, C. (2016). A study on local search neighborhoods for the job shop scheduling problem with total weighted tardiness objective, *Computers & Operations Research*, Vol. 66, 44-57, doi:10.1016/j.cor.2015.07.011

[13] Chaouch, I.; Driss, O. B.; Ghedira, K. (2017). A modified ant colony optimization algorithm for the distributed job shop scheduling problem, *Procedia Computer Science*, Vol. 112, 296-305, doi:10.1016/j.procs.2017.08.267

[14] Li, J.; Huang, Y.; Niu, X. (2016). A branch population genetic algorithm for dual-resource constrained job shop scheduling problem, *Computers & Industrial Engineering*, Vol. 102, 113-131, doi:10.1016/j.cie.2016.10.012

[15] Yazdani, M.; Aleti, A.; Khalili, S. M.; Jolai, F. (2017). Optimizing the sum of maximum earliness and tardiness of the job shop scheduling problem, *Computers & Industrial Engineering*, Vol. 107, 12-24, doi:10.1016/j.cie.2017.02.019

[16] Kundakci, N.; Kulak, O. (2016). Hybrid genetic algorithms for minimizing makespan in dynamic job shop scheduling problem, *Computers & Industrial Engineering*, Vol. 96, 31-51, doi:10.1016/j.cie.2016.03.011

[17] Bierwirth, C.; Kuhpfahl, J. (2017). Extended GRASP for the job shop scheduling problem with total weighted tardiness objective, *European Journal of Operational Research*, Vol. 261, No. 3, 835-848, doi:10.1016/j.ejor.2017.03.030

[18] Pezzella, F.; Morganti, G.; Ciaschetti, G. (2008). A genetic algorithm for the flexible job-shop scheduling problem, *Computers & Operations Research*, Vol. 35, No. 10, 3202-3212, doi:10.1016/j.cor.2007.02.014