An Improved Genetic Algorithm for Solving the Mixed-Flow Job-Shop Scheduling Problem with Combined Processing Constraints

ZHU Haihua[1*], *ZHANG Yi*[1], *SUN Hongwei*[2], *LIAO Liangchuang*[2], *TANG Dunbing*[1]

1. College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, P.R. China; 2. Jiangsu Automation Research Institute, Lianyungang 222006, P.R. China

(Received 28 April 2021; revised 15 May 2021; accepted 9 June 2021)

Abstract: The flexible job-shop scheduling problem (FJSP) with combined processing constraints is a common scheduling problem in mixed-flow production lines. However, traditional methods for classic FJSP cannot be directly applied. Targeting this problem, the process state model of a mixed-flow production line is analyzed. On this basis, a mathematical model of a mixed-flow job-shop scheduling problem with combined processing constraints is established based on the traditional FJSP. Then, an improved genetic algorithm with multi-segment encoding, crossover, and mutation is proposed for the mixed-flow production line problem. Finally, the proposed algorithm is applied to the production workshop of missile structural components at an aerospace institute to verify its feasibility and effectiveness.

Key words: mixed-flow production; flexible job-shop scheduling problem (FJSP); genetic algorithm; encoding

CLC number: TH165 **Document code:** A **Article ID:** 1005-1120(2021)03-0415-12

0 Introduction

With the increasingly diversified demands, the production mode of manufacturing enterprises has continuously evolved into multi-variety and small-batch production. More and more manufacturing enterprises are applying mixed-line production lines in flexible job-shops. However, the flexibility of mixed-flow shops increases the difficulty of scheduling, thus traditional methods for the classic flexible job-shop scheduling problem (FJSP) cannot be directly adapted to those mixed-flow job-shops.

FJSP is a fundamental problem in manufacturing. It is a generalization of the job-shop scheduling problem (JSP)[1] that removes the limitation of the unique machine specified in each operation and concerns the processing flexibility[2]. In the past few decades, experts and scholars have conducted many studies on FJSPs and developed various solution methodologies[3]. Most researchers assumed that a machine cannot process more than one operation at a certain time[4] and only needs to meet the conventional routing constraints. However, in many cases, in order to ensure the accuracy of assembly, several jobs must be processed simultaneously on the same machine, i.e., combined processing. Combined processing is a processing technology that clamps two or more parts in accordance with the assembly relationship by using the same reference and processes the related operations in one single chucking. This technology can enssure the accuracy of assembly, reduce the difficulty of processing and improve the working efficiency. It has been widely used in the production of high precision components, like the manufacturing of various molds and shell parts. The scheduling problem in these job-shops is exactly a FJSP with combined processing constraints. In this

*Corresponding author, E-mail address: h.zhu@nuaa.edu.cn.

How to cite this article: ZHU Haihua, ZHANG Yi, SUN Hongwei, et al. An improved genetic algorithm for solving the mixed-flow job-shop scheduling problem with combined processing constraints[J]. Transactions of Nanjing University of Aeronautics and Astronautics, 2021, 38(3): 415-426.

http://dx.doi.org/10.16356/j.1005-1120.2021.03.006

problem, some machines can process multiple operations of different jobs at the same time. Moreover, in addition to meeting the conventional routing constraints, it is also necessary to meet the combined processing constraints between different jobs. Therefore, it is of great significance and practical value to establish the model of FJSPs with combined processing constraints.

Although FJSPs with combined processing constraints widely exist in real-world job-shops, there is few literature about this problem. Some have studied the hybrid flow shop scheduling problem. Ribas et al.[5] presented an extensive review on hybrid flow shop scheduling problems. Engin et al.[6] proposed an effective genetic algorithm to minimize the makespan time. They adopted a new mutation operator and the best values of the control parameters to solve the hybrid flow shop scheduling with the multiprocessor task problem. Seidgar et al.[7] presented a new bi-objective mixed integer programming model for the two-stage assembly flow shop scheduling problem with preventive maintenance activities, and employed the non-dominated ranking genetic algorithm to find the pareto-optimal front for large sized problems. Pan et al.[8] presented a discrete artificial bee colony algorithm with an efficient initialization scheme and a self-adaptive strategy for generating neighboring food sources to solve the lot-streaming flow shop scheduling problem. Ye et al.[9] proposed an effective shuffled frog-leaping algorithm to solve the hybrid flow-shop scheduling problem with identical parallel machines. Dios et al.[10] proposed a set of heuristics that captures some special features of the missing operations to address the hybrid flow shop scheduling problem for makespan minimization. Compared with FJSPs, the hybrid flow shop scheduling problem lacks flexibility.

At present, genetic algorithms (GAs) are the most commonly used methods to solve FJSPs. Kacem[11] proposed a generic algorithm to apply advanced genetic manipulations to solve a FJSP. Pezzella et al.[12] presented a genetic algorithm for a FJSP by integrating different strategies for a generating the initial population, selecting the individuals for reproduction and reproducing new individuals. Kacem et al.[13] put forward a generic algorithm to build an ideal assignment model to solve a FJSP. Gao et al.[14] developed a hybrid GA with advanced crossover and mutation operators for solving the multi-objective FJSP. Ishikawa et al.[15] proposed the hierarchical multi-space competitive distributed GA to find an optimal solution for a FJSP with a low computational cost. Zhang et al.[16] proposed an improved non-dominated sorting genetic algorithm (NSGA-Ⅱ) with an extended operation-based encoding and an active scheduling decoding mechanism to solve the FJSP.

There are some other heuristic algorithms to solve FJSPs. Gao et al.[17] proposed an improved artificial bee colony algorithm to solve the FJSP with fuzzy processing time. Others use hybrid algorithms to solve FJSPs. Li et al.[18] proposed an effective hybrid algorithm to hybridize the GA and tabu search for the FJSP with the objective of minimizing the makespan. Rey et al.[19] proposed a novel meta-heuristics algorithm by combining GA with particle swam optimization (PSO) to find adequate job release times to meet specific due dates. Xia et al.[20] developed an easily implemented hybrid approach for the multi-objective FJSP, which combined a PSO algorithm and simulated annealing (SA). Dong et al.[21] built a related disjunctive graph model and proposed a hybrid GA -ant colony optimization to solve a FJSP more effectively. These above methods are very efficient in solving classical FJSPs, but cannot be applied to mixed-line job-shops directly.

For mixed-flow assembly lines, Xiong[22] researched the scheduling problem of assemblies in JSP. Lu et al.[23] employed an improved NSGA-Ⅱ multi-target heritage algorithm to optimize the U-shaped mixed-flow assembly line. None of the above methods can provide a good solution for mixed-flow shop scheduling with combined machining constraints. Wang et al.[24] presented the optimization of a mixed production line based on the logic intelligent reasoning method and GA. But this method can only solve the ranking length of the mixed-flow production line.

In order to solve the FJSP with combined processing constraints for mixed-flow job-shops, the process state model of the mixed-flow production line is analyzed. On this basis, a mathematical model of the FJSP with combined processing constraints is established. Then, an improved GA with multi-segment encoding, crossover, and mutation is proposed to deal with the mixed-flow production line scheduling problem. Finally, the proposed algorithm is applied to the missile structure production workshop of an aerospace research institute, and the feasibility and effectiveness of the method are verified.

The rest of the paper is organized as follows: In Section 1 the mixed-flow JSP with combined processing constraint is defined in detail and then the mathematical model and a multi-part GA is explained in Section 2 and Section 3. Simulation experiments are presented in Section 4 and the conclusions are drawn in Section 5.

1 Mixed-Flow Job-Shop Scheduling Problem with Combined Processing Constraints

The traditional FJSP is described as follows: A manufacturing system is equipped with several machines to process several workpieces. Each workpiece has several operations with a sequential constraint relationship. The operation numbers of these workpieces are different with each other and each operation can be processed on one or more machines. The goal of scheduling is to select an appropriate machine and an appropriate processing sequence for each operation to improve the whole system's performance.

However, in the process of mixed-flow production line, there are not only components but also assemblies combined by two or more components. Therefore, different from the manufacturing system only with components, there are also combined operations such as assembly operation and combined processing operation. This section analyzes the model of process state in the mixed-flow production line, as shown in Fig.1. Fig.1(a) depicts the flow of assembly operation, where J_1 and J_2 represent two components belonging to the same assembly; and O_{ij} represents the jth operation of the ith component. These two components first carry out their operations separately, and then conduct their assembly operation. O_{12} & O_{23} represents the assembly operation, where J_1 and J_2 are processed at the same time. Fig.1(b) displays the flow of combination processing operation, where J_3 and J_4 represent two components belonging to the same assembly. These two components first carry out their operations separately, and then exert the next operation of J_3 and J_4 that needs to be processed on the same machine at the same time. Moreover, O_{33} and O_{43} can start processing only after O_{32} and O_{43} are finished.

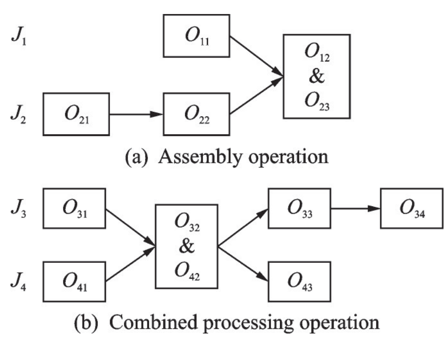

Fig.1 Process state model

In order to research the scheduling problem of the mixed-flow production line in the flexible job-shop, we expand the traditional FJSP into a FJSP for the mixed-flow production line. It is described as follows.

A number of machines are arranged in a manufacturing system to produce several assemblies. Each assembly contains two or more components. Each component contains several operations with sequential constraints (including assembly operation, combined processing operation). The operation numbers of these components are different and each operation can be processed on one or more machines. The goal of scheduling is to select the appropriate processing unit for each operation, and select the appropriate processing sequence for the operations assigned to each processing unit, so as to optimize the whole system's performance.

In this paper, the maximum due time is taken as the optimization goal, and the following assumptions are made.

(1) There is no processing priority between assemblies.

(2) There is no preparation time for each operation.

(3) The transportation time between the machines is set to a fixed value.

(4) Each machine can only work on one component or one assembly at the same time.

2 Mathematical Model of Mixed-Flow Job-Shop Scheduling Problem

For the convenience of illustrating the mathematical model, the variables to be used are defined as follows.

N_C: The number of assemblies;

N_{Ci}: The number of components of the ith assembly;

N_{Cij}: The number of operations in the jth component of the ith assembly;

N_M: The number of machines;

G_i: The ith assembly;

J_{ij}: The jth component of the ith assembly;

O_{ijk}: The kth operation in the jth component of the ith assembly;

M_m: The mth machine;

C_i: The completion time of the ith assembly;

C_{ij}: The completion time of the jth component of the ith assembly;

t_{sijk}: The start processing time of the kth operation in the jth component of the ith assembly;

t_{oijk}: The end processing time of the kth operation in the jth component of the ith assembly;

m_{smn}: The start processing time of the nth operation in the mth machine;

m_{omn}: The end processing time of the nth operation in the mth machine;

t_{ijkm}: The processing time of the kth operation in the jth component of the ith assembly;

t_t: The transporting time.

$$\alpha_{ijk}=\begin{cases}1 & O_{ijk} \text{ is a combined operation}\\ 0 & \text{Otherwise}\end{cases}$$

$$\operatorname{cov}(O_{ijk},O_{ij'k'})=\begin{cases}1 & \text{Combined process}\\ 0 & \text{Otherwise}\end{cases}$$

$$X_{ijkm}=\begin{cases}1 & O_{ijk} \text{ is processed in the } m\text{th machine}\\ 0 & \text{Otherwise}\end{cases}$$

The constraints are as follows

$$\sum_{m=1}^{M} X_{ijkm}=1 \tag{1}$$

$$t_{sij(k+1)}\geqslant\begin{cases}t_{oijk}+t_t & \alpha_{ijk}=0\\ \max\left(t_{oijk},\max\limits_{\operatorname{cov}(O_{ijk},O_{ij'k'})=1}(t_{oij'k'})\right)+t_t & \alpha_{ijk}=1\end{cases} \tag{2}$$

$$m_{sm(n+1)}-m_{omn}\geqslant 0 \tag{3}$$

$$t_{oijk}-t_{sijk}=\sum_{m=1}^{M}(X_{ijkm}\cdot t_{ijkm}) \tag{4}$$

Among these equations, Eq.(1) implies that each operation in each component of each assembly can only be processed by one machine. Eq.(2) shows that if the current operation is a separate operation, this operation will start only after its previous operation ends, and if the current operation is a combined operation, its starting time needs to consider all the related components of the same assembly including completion time and shipping time. Eq.(3) indicates that each machine can only start the next operation after the current operation ends. Eq.(4) illustrates that the processing time is determined by the corresponding processing machine.

This paper considers the max completion time as the decision variable, written as C and defined as

$$C=\max_{1\leqslant i\leqslant N} C_i=\max_{1\leqslant i\leqslant N}\left(\max_{1\leqslant j\leqslant N_i} C_{ij}\right) \tag{5}$$

The optimization target is minimizing the max completion time, defined as

$$\min C \tag{6}$$

3 A Multi-part GA

GA is an algorithm of computational intelligence that simulates the process of natural selection and survival of the fittest, which is able to obtain the feasible solution of the problem. It has a good global search capability and is widely used in the field of production scheduling.

GA is widely used in solving the traditional FJSP. In the process of GA encoding in the traditional FJSP, the chromosome is divided into two parts: Process chromosome and machine chromosome. However, in the FJSP for the mixed-flow production line proposed in this paper, the usual encoding way cannot express all the production information and should be improved.

To propose the multi-part GA, this paper modifies the encoding, cross, mutation operations of the usual GA, and the other steps remain the same. The chromosome is divided into three parts: Assembly process chromosome, component process chromosome and machine process chromosome. The cross and mutation are accordingly modified.

3.1 Flow of GA

First, several terms are introduced as follows.

Individual: A solution to the problem;

Population: A set of solutions to the problem;

Fitness: An indicator for judging how good a solution is;

Chromosome: A series of numbers containing information that represent individuals;

Gene: A digit in a chromosome that contains the basic information of an individual;

Encoding: The process of translating an chromosome to an individual;

Decoding: The inverse process of encoding;

Selection: An operation of the GA that simulates the natural selection;

Cross: An operation of the GA that simulates natural reproduction;

Mutation: An operation of the GA that simulates natural variation.

The flow of the GA is shown in Fig.2.

Step 1 Analyze the problems to be solved and summarize the characteristics of the problems.

Step 2 According to the characteristics of practical problems, the coding and decoding schemes are designed to associate the mathematical model of the problem with the GA.

Step 3 Set the population size and select a random initialization method to initialize the chromosome population.

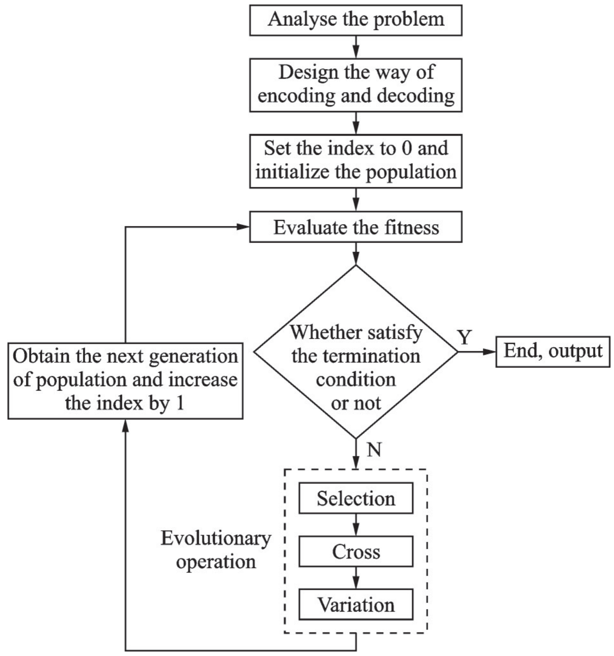

Fig.2 Flow of GA

Step 4 Decode the chromosomes in the population and evaluate the fitness value of chromosomes.

Step 5 Judge whether the algorithm meets the termination condition according to the population fitness value. If not, go to Step 6; Otherwise, end the algorithm and output the optimal solution.

Step 6 Perform the selection step on the population, so that the chromosomes with high fitness survives to the next generation, and the chromosomes with low fitness is eliminated. Commonly used selection methods are binary tournament, roulette, etc.

Step 7 Select some chromosomes according to the crossover probability and perform crossover operation according to the set rules.

Step 8 According to the mutation probability and the set rules, select some chromosomes for mutation operation, and set the mutation probability generally by the idea of simulated annealing.

Step 9 Generate the next generation population, and go to Step 5.

3.2 Multi-segment encoding

The encoding way divides the chromosome into three parts.

(1) Assembly process chromosome

In this part, each gene represents an assembly index. The count of each assembly index indicates the total operation numbers of the assembly. The order number that each gene appears in the corresponding assembly index sequence means the order number of an assembly operation. This part is used to determine the order of processing in the level of assembly, and its length equals to the total number of all operations.

(2) Component process chromosome

In this part, each assembly has a corresponding assembly sub-chromosome. In each assembly sub-chromosome, each gene represents a component index. The count of each component index indicates the total operation numbers of the component. The order number that each gene appears in the corresponding component index sequence means the order number of a component operation. This part is used to determine the order of processing in the level of component, and its length equals to the total number of all operations.

(3) Machine chromosome

In this part, each gene represents a machine index. The index of the machine selected for each processing is in the set of optional machines. This part is used to determine the machine of each processing, and the length also equals to the total number of all operations.

Therefore, in this encoding way, the length of each chromosome is three times of the total number of operations.

Taking the problem with two assemblies and four machines as an example. Table 1 contains all of the assembly information, and the symbol "—" indicates that the corresponding process cannot be processed on the corresponding machine. Among these operations, O_{112} and O_{123} is a pair of combination process, and O_{212} and O_{222} is another pair of combination process. In Table 1, the processing information of each pair of combination process is identical.

Table 1 Processing information of two assemblages and four machines

Assembly	Component	Operation	Processing time/h			
			M_1	M_2	M_3	M_4
G_1	J_{11}	O_{111}	—	—	3	2
		$O_{112}(O_{123})$	4	3	—	—
	J_{12}	O_{121}	—	3	4	2
		O_{122}	2	4	—	3
		$O_{123}(O_{112})$	4	3	—	—
G_2	J_{21}	O_{211}	2	—	—	3
		$O_{212}(O_{222})$	4	—	—	2
		O_{213}	—	2	4	—
		O_{214}	—	3	4	
	J_{22}	O_{221}	3	—	2	—
		$O_{222}(O_{212})$	4	—	—	2
		O_{223}	—	3	—	2

The encoding of the chromosome is shown in Fig.3. The assembly process chromosome [1 2 1 2 2 2 1 2 2 1] and the part process chromosome [2 1 2 3 | 1 2 3 1 2 1] together determine the order of processing in the system, that is, $O_{121} \rightarrow O_{211} \rightarrow O_{111} \rightarrow O_{221} \rightarrow O_{212}$ & $O_{222} \rightarrow O_{213} \rightarrow O_{122} \rightarrow O_{223} \rightarrow O_{214} \rightarrow O_{112}$ & O_{123}. The machine chromosome [3 2 4 1 | 1 2 2 3 2 1] represents that operation O_{111} is processed on M_3, operation O_{121} is processed on M_2, operation O_{112} is processed on M_4, O_{112} & O_{123} is processed on M_1, and so on.

This encoding way is able to represent all the feasible solutions of the FJSP for the mixed-flow production line without generating any illegal solution, and it facilitates the subsequent cross and mutation.

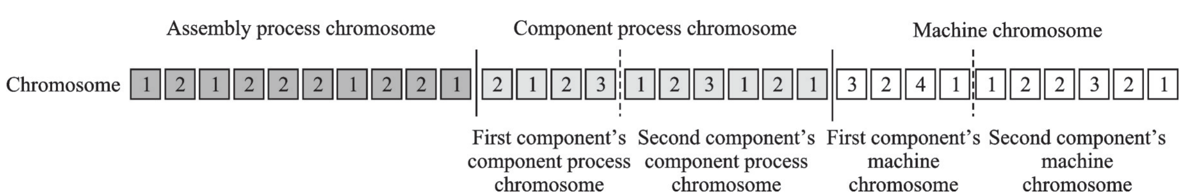

Fig.3 Diagram of chromosome

3.3 Population initialization and multi-segment cross

In order to increase the diversity of the initial population, this paper initializes the population in a random way, and selects the next generation in a binary tournament way.

Since the chromosome is divided into three parts and they are different from each other, each part needs to cross separately. Firstly, the coefficient of cross k is defined as the ratio of the number of the genes learned by the individual to the total number of the genes in the chromosome. Then, according to the probability of cross, two chromosomes P1 and P2 are randomly selected. Fig.4 shows the selected two chromosomes.

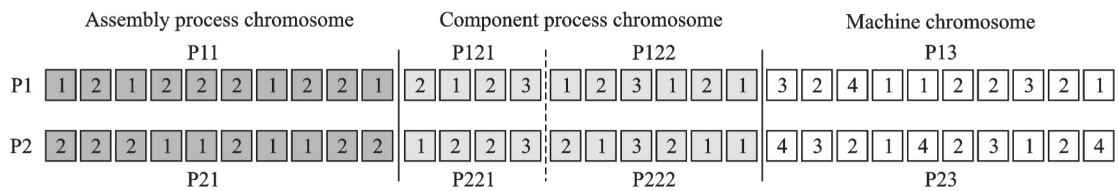

Fig.4 Diagram of the chosen chromosomes

3.3.1 Cross of assembly process chromosomes

The assembly process chromosome of the two parental chromosomes are P11 and P21. As shown in Fig.5, firstly, the length of P11 and the coefficient of cross are multiplied to get the length of cross L_{11}. The process number of P21 and the coefficient of cross are multiplied to get the length of cross L_{21}. In this example, $L_{11}= 2$, $L_{21}=2$. Then, two genes are selected from the genes of G_1 at location [1 3 7 10] in P11. Two genes are selected from the genes of G_2 at location [2 4 5 6 8 9] in P11. In Fig.5, genes at location [3,7] and genes at location [5,8] are chosen and the genes to cross of P11 are at location [3 5 7 8]. In the same way, the genes to cross of P21 are at location [2 3 5 7]. Finally, the genes are exchanged to cross of P11 and P21 and obtain the child chromosomes S11 and S21 are obtained.

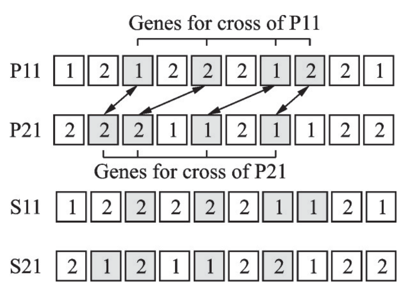

Fig.5 Cross procedure between general operation parts

3.3.2 Cross of component process chromosomes

The component process chromosomes of the two parental chromosomes are P121, P122, P221, P222. The way of cross is basically the same as that of the assembly process chromosomes, but there are still two differences:

(1) The genes that represent assembly processes do not need to cross;

(2) The cross of component process chromosomes needs to be segmented.

The cross of P121 and P221 are taken as examples, as shown in Fig.6. In this example, the index of assembly process is 3 and it will not participate in cross.

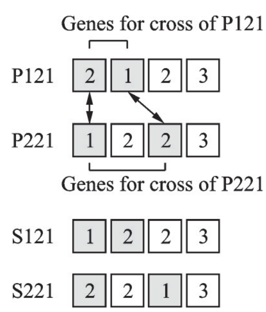

Fig.6 Cross procedure between combined operation parts

3.3.3 Cross of machine chromosomes

Due to the difference of constraints, the cross of the machine chromosomes is different with that of the process chromosomes. As shown in Fig.7, firstly, the length of P13 or P23 and the coefficient of cross are multiplied to get the length of cross. In this example, $L_3=4$. Then, four genes are selected from the genes of the machine chromosomes. In Fig.7, genes at location [2 5 7 8] of P13 and P23 are chosen to cross. Finally, exchange the genes are exchanged to cross of P13 and P23 and obtain the child chromosomes S13 and S23.

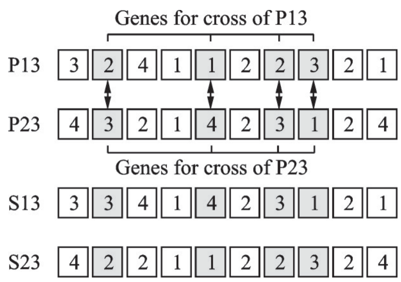

Fig.7 Cross procedure between machine parts

3.4 Mutation

The role of mutation is to avoid getting the algorithm into local optimum. In this paper, the way of mutation is designed using the idea of simulated annealing. That is, the probability of mutation is relatively high at the beginning, but it gradually goes down along with the increase of the generation.

The concreate way of mutation is as follows. For assembly process chromosomes and component process chromosomes, it selects two genes randomly and exchanges them, as shown in Figs.8, 9. For machine chromosomes, it randomly selects a process and randomly re-selects the corresponding machine from the optional machine set, as shown in Fig.10. Hill function is selected to describe the probability of mutation

$$p(i \rightarrow j)=\begin{cases}1 & F(i)\geqslant F(j)\\ T & F(i)<F(j)\end{cases} \tag{7}$$

$$T=A\times\frac{B^2}{B^2+t^2} \tag{8}$$

where i represents the state before mutation; j the state after mutation; $F(i)$ the fitness at state I; $F(j)$ the fitness at state j. In Eq.(8), A and B are the parameters of the algorithm, and t means the generation.

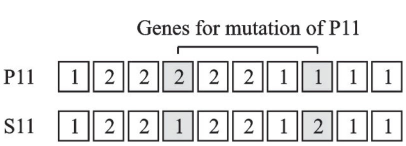

Fig.8 Mutation procedure between assembly parts

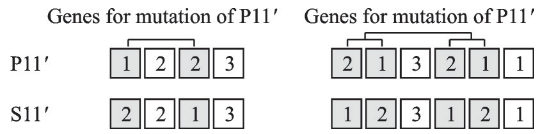

Fig.9 Mutation procedure between operation parts

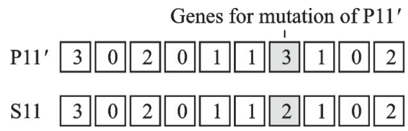

Fig.10 Mutation procedure between machine parts

4 Case Study

The aerospace institute in Shanghai is a scientific research institute that undertakes the research and production of aircraft. The production workshop of missile structural components under its jurisdiction is responsible for the manufacture of various structural parts. The products of workshop are complex components, and the processing process is completed in a flexible operation mode. Therefore, the processing of production can be described as a FJSP for a mixed-flow production line.

There are four workshops in this factory, in which the ordinary lathing machines and the ordinary milling machines are arranged in workshop 1, the computer numerical control (CNC) lathing machines are arranged in workshop 2, the numerical control milling machines are arranged in workshop 3, and the fitters are arranged in workshop 4. The machine information in the case is shown in Table 2.

There are five assemblies in this case. Their

Table 2 Information of machines

No. of machine	Type of machine	Workshop
M_1	Ordinary lathing	1
M_2	Ordinary lathing	1
M_3	Ordinary milling	1
M_4	Ordinary milling	1
M_5	CNC lathing	2
M_6	CNC lathing	2
M_7	CNC milling	3
M_8	CNC milling	3
M_9	Fitter	4
M_{10}	Fitter	4

processing information is shown in Table 3 and their processing flows are shown in Fig.11. Here, the transporting time is considered as a constant because it is much smaller than the processing time.

The size of the generation is 100. The max iteration is 100. The probability of cross is 0.8. The coefficient of cross is 0.4. The parameter of simulated annealing A is 0.6 and that of B is 10.

The learning curve is shown in Fig.12. It can be seen that the oscillation of the learning curve is very large at first, and then gradually decreases. The mutation operator in the form of simulated annealing plays an important role in this process. Finally, the algorithm converges at the 40th generation.

Table 3 Data in the case

Assembly	Component	Operation	Type	Processing time/h									
				M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
G_1	J_{11}	O_{111}	Ordinary lathing	10	8	—	—	—	—	—	—	—	—
		O_{112}	Fitter	—	—	—	—	—	—	—	—	5	6
		$O_{113}(O_{124})$	Assembling	—	—	—	—	—	—	—	—	4	6
	J_{12}	O_{121}	Ordinary lathing	8	8	—	—	—	—	—	—	—	—
		O_{122}	CNC lathing	—	—	—	—	4	5	—	—	—	—
		O_{123}	CNC milling	—	—	—	—	—	—	5	3	—	—
		$O_{124}(O_{113})$	Assembling	—	—	—	—	—	—	—	—	4	6
G_2	J_{21}	O_{211}	Ordinary lathing	12	9	—	—	—	—	—	—	—	—
		$O_{212}(O_{222})$	CNC lathing	—	—	—	—	8	6	—	—	—	—
		O_{213}	CNC milling	—	—	—	—	—	—	3	6	—	—
		O_{214}	Fitter	—	—	—	—	—	—	—	—	8	6
	J_{22}	O_{221}	Ordinary milling	—	—	5	8	—	—	—	—	—	—
		$O_{222}(O_{212})$	CNC lathing	—	—	—	—	8	6	—	—	—	—
		O_{223}	Fitter	—	—	—	—	—	—	—	—	7	9
		O_{224}	CNC milling	—	—	—	—	—	—	4	3	—	—
G_3	J_{31}	O_{311}	Ordinary milling	—	—	6	8	—	—	—	—	—	—
		O_{312}	Ordinary lathing	9	5	—	—	—	—	—	—	—	—
		O_{313}	CNC lathing	—	—	—	—	2	3	—	—	—	—
		$O_{314}(O_{323})$	Assembling	—	—	—	—	—	—	—	—	7	5
	J_{32}	O_{321}	Ordinary milling	—	—	6	4	—	—	—	—	—	—
		O_{322}	CNC lathing	—	—	—	—	10	12	—	—	—	—
		$O_{323}(O_{314})$	Assembling	—	—	—	—	—	—	—	—	7	5
G_4	J_{41}	O_{411}	Ordinary milling	—	—	10	8	—	—	—	—	—	—
		O_{412}	Ordinary lathing	4	6	—	—	—	—	—	—	—	—
		$O_{413}(O_{422})$	CNC milling	—	—	—	—	—	—	6	8	—	—
		O_{414}	CNC lathing	—	—	—	—	3	7	—	—	—	—
		$O_{415}(O_{424})$	Assembling	—	—	—	—	—	—	—	—	6	6
	J_{42}	O_{421}	Ordinary milling	—	—	13	11	—	—	—	—	—	—
		$O_{422}(O_{413})$	CNC milling	—	—	—	—	—	—	6	8	—	—
		O_{423}	Fitter	—	—	—	—	—	—	—	—	3	5
		$O_{424}(O_{415})$	Assembling	—	—	—	—	—	—	—	—	6	6
G_5	J_{51}	O_{511}	Ordinary lathing	9	6	—	—	—	—	—	—	—	—
		O_{512}	Ordinary milling			4	5	—	—	—	—	—	—
		O_{513}	CNC milling	—	—	—	—	—	—	8	7	—	—
		$O_{514}(O_{524})$	Assembling	—	—	—	—	—	—	—	—	4	9
	J_{52}	O_{521}	Ordinary lathing	—	—	3	8	—	—	—	—	—	—
		O_{522}	Ordinary lathing	—	—	—	—	9	4	—	—	—	—
		O_{523}	CNC milling	—	—	—	—	—	—	3	5	—	—
		$O_{524}(O_{514})$	Assembling	—	—	—	—	—	—	—	—	4	9

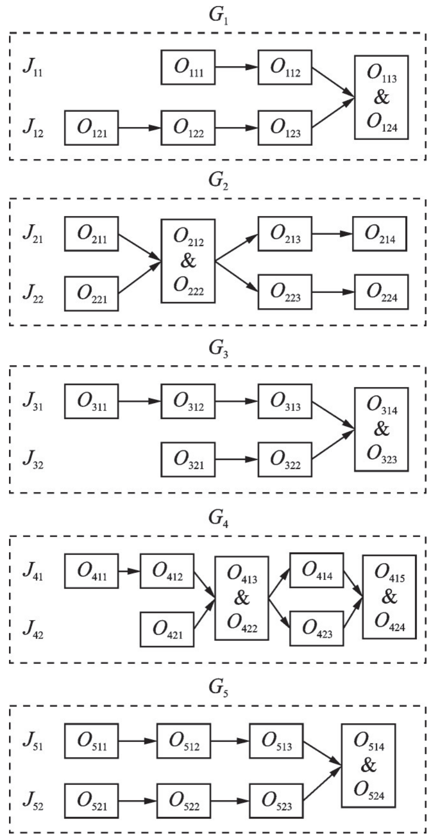

Fig.11 Diagram of the processing flows

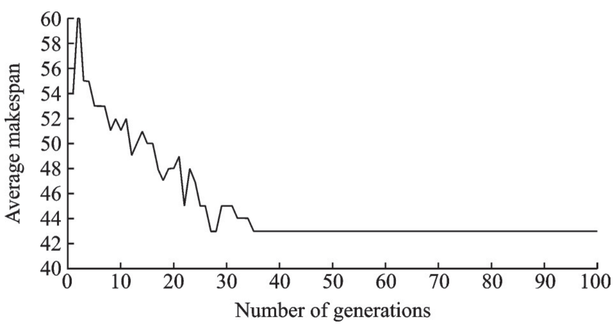

Fig.12 Change of makespan during evolution

The optimal scheduling solution to the FJSP for the mixed-flow production line of the missile structural component obtained by the proposed algorithm is shown in Fig.13 where the vertical axis is the number of machines, and the horizontal is the time line. O_{212} & O_{222} represent combined processing operation. O_{413} & O_{422}, O_{314} & O_{323}, O_{415} & O_{424}, O_{113} & O_{124}, O_{514} & O_{524} represent the assembly operations. The completion time of the task is 42. It can be seen that the result satisfies the constraints of assembly and combined processing. So the proposed algorithm is feasible.

In some studies, dispatching rules are usually

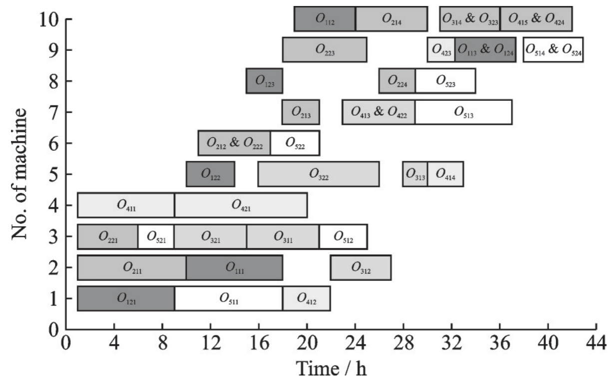

Fig.13 Gatt of the solution

applied to guide workshop scheduling. It generally consists of two stages: Machine selection and buffer job sequencing. It needs to arrange proper machine and buffer job sequencing. The machine selection rules are shortest queue (SQ) and shortest processing time (SPT). The buffer job sequencing rules are: First in first out (FIFO) and shortest job first (SJF). The final combined dispatching rules include SPT + FIFO and SQ + SJF. Compared with the above dispatching rules, the results in Table 4 show that the scheduling solution obtained by the improved GA has shorter completion time.

Table 4 Comparison between various methods

Method	Makespan
SPT+FIFO	126
SQ+SJF	129
GA	42

5 Conclusions

This paper studies the flexible job-shop scheduling for the mixed-flow production line. Firstly, the process state model of the mixed-flow production line is analyzed. Secondly, a mathematical model of FJSP for the mixed-flow production line is established. Thirdly, for the above model, a multipart GA with multi-segment encoding is designed. Finally, the proposed algorithm is applied to the production workshop of missile structural components at an aerospace institute to verify its feasibility and effectiveness.

Future research directions include designing more efficient initialization mechanism for the proposed multi-part GA, and considering more optimization targets for FJSPs in mixed-flow production lines.

References

[1] PEZZELLA F, CIASCHETTI G, CIASCHETTI G. A genetic algorithm for the flexible job-shop scheduling problem[J]. Comput Oper Res, 2008, 35(10): 3202-3212.

[2] DONG R, HE W. Hybrid genetic algorithm-ant colony optimization for FJSP solution[J]. Comput Integra Manuf Sys, 2012, 18(11): 2492-2501.

[3] CINAR D, TOPCU Y, OLIVEIRA J A. A taxonomy for the flexible job shop scheduling problem[C]// Proceedings of Conference on Optimization Control and Applications in the Information Age. Macedonia, Greece:[s.n.], 2014: 15-20.

[4] XIANG W, LEE H P. Ant colony intelligence in multi-agent dynamic manufacturing scheduling[J]. Eng Appl Artif Intell, 2008, 21(1): 73-85.

[5] RIBAS I, LEISTEN R, FRAMIÑAN J M. Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective[J]. Computers & Operations Research, 2010,37(8): 1439-1454.

[6] ENGIN O, CERAN G, YILMAZ M K. An efficient genetic algorithm for hybrid flow shop scheduling with multiprocessor task problems[J]. Applied Soft Computing, 2011,11(3): 3056-3065.

[7] SEIDGAR H, ZANDIEH M, MAHDAVI I. Bi-objective optimization for integrating production and preventive maintenance scheduling in two-stage assembly flow shop problem[J]. Journal of the Chinese Institute of Industrial Engineers, 2016,33(6): 404-425.

[8] PAN Q K, TASGETIREN M F, SUGANTHAN P N, et al. A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem[J]. Computer Engineering & Applications, 2011, 181 (12): 2455-2468.

[9] YE X, LING W, WANG S, et al. An effective shuffled frog-leaping algorithm for solving the hybrid flow-shop scheduling problem with identical parallel machines[J]. Engineering Optimization, 2013, 45(12): 1409-1430.

[10] DIOS M, FERNANDEZ-VIAGAS V, FRAMINAN J M. Efficient heuristics for the hybrid flow shop scheduling problem with missing operations[J]. Computers & Industrial Engineering, 2018,115: 88-99.

[11] KACEM I. Genetic algorithm for the flexible job-shop scheduling problem[J]. Computers & Operations Research, 2008, 35(10): 3202-3212.

[12] PEZZELLA F, MORGANTI G, CIASCHETTI G. A genetic algorithm for the flexible job-shop scheduling problem[J]. Computers & Operations Research, 2008,35(10): 3202-3212.

[13] KACEM I, HAMMADI S, BORNE P. Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems[J]. IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews), 2002, 32 (1): 1-13.

[14] GAO J, SUN L, GEN M. A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems[J]. Computers & Operations Research, 2008,35(9): 2892-2907.

[15] ISHIKAWA S, KUBOTA R, HORIO K. Effective hierarchical optimization by a hierarchical multi-space competitive genetic algorithm for the flexible job-shop scheduling problem[J]. Expert Systems With Applications,2015,42(24): 9434-9440.

[16] ZHANG C Y, DONG X, WANG X Q, et al. Improved NSGA-Ⅱ for the multi-objective flexible job-shop scheduling problem[J]. Journal of Mechanical Engineering, 2010,46(11): 156-164.

[17] GAO K Z, SUGANTHAN P N, PAN Q K, et al. An improved artificial bee colony algorithm for flexible job-shop scheduling problem with fuzzy processing time[J]. Expert Systems with Applications an International Journal, 2016,65(C): 52-67.

[18] LI X, LIANG G. An effective hybrid genetic algorithm and tabu search for flexible job-shop scheduling problem[J]. International Journal of Production Economics, 2016,174: 93-110.

[19] REY G Z, BEKRAR A, TRENTESAUX D, et al. Solving the flexible job-shop just-in-time scheduling problem with quadratic earliness and tardiness costs [J]. The International Journal of Advanced Manufacturing Technology, 2015,81(9): 1871-1891.

[20] XIA W, WU Z. An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems[J]. Computers & Industrial Engineering, 2005,48(2): 409-425.

[21] DONG R, HE W. Hybrid genetic algorithm-ant colony optimization for FJSP solution[J]. Computer Integrated Manufacturing Systems, 2012,18(11): 2492-

2501.

[22] XIONG Hegen. Research on dynamic job-shop scheduling for die & mould corporations[D]. Wuhan: Huazhong University of Science and Technology, 2005. (in Chinese)

[23] LU Jiansha, CHEN Zhongliang, LI Xiulin, et al. U-shaped mixed-flow assembly line collating sequence model and its resolution[J]. Light Industry Machinery, 2008, 26(6): 87-90. (in Chinese)

[24] WANG Xianhong, SHI Guoquan. Applications of a new logic intelligent inference method in scheduling optimization of mixed production lines[J]. China Mechanical Engineering, 2015, 173: 83-112. (in Chinese)

Acknowledgements This work was supported by the National Key Research and Development Program of China (No.2020YFB1710500), the National Natural Science Foundation of China (No.51805253), and the Fundamental Research Funds for the Central Universities (No. NP2020304).

Author Dr. **ZHU Haihua** received his Ph.D. degree in mechanical engineering and automation from Nanjing University of Science and Technology, Nanjing, China, in 2013. From 2009 to 2011, he was a research scholar with the University of Greenwich, London, UK. He is currently an associate professor with the College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, China. His research interest includes smart manufacturing systems, digital twin, and Internet of things.

Author contributions Dr. **ZHU Haihua** designed the study, complied the models, interpreted the results and wrote the manuscript. Mr. **ZHANG Yi** contributed to data and model components for the mixed-flow job-shop scheduling model. Mr. **SUN Hongwei** contributed to the discussion and background of the study. Mr. **LIAO Liangchuang** contributed to data analysis of scheduling solutions. Prof. **TANG Dunbing** conducted the analysis. All authors commented on the manuscript draft and approved the final manuscript.

Competing interests The authors declare no competing interests.

(Production Editor: ZHANG Bei)

一种基于改进遗传算法的组合加工约束混流车间调度方法

朱海华[1]，张　毅[1]，孙宏伟[2]，廖良闯[2]，唐敦兵[1]

(1.南京航空航天大学机电学院,南京 210016,中国; 2.江苏自动化研究所,连云港 222006,中国)

摘要:具有组合加工约束的柔性作业车间调度问题是混流生产线中常见的任务排产问题。然而,传统车间调度方法均未将组合加工约束考虑进调度模型中,无法满足混线生产模式的现实情况。针对这一问题,分析了混流生产线的工艺状态模型。在此基础上,基于传统柔性作业车间调度问题,建立了具有组合加工约束的混线车间调度问题的数学模型。然后,针对组合加工约束,提出了一种改进的多段编码、交叉、变异的遗传算法。最后,将该算法应用于某航空航天研究所导弹结构件生产车间,验证了该方法的可行性和有效性。

关键词:混线生产;柔性作业车间调度问题;遗传算法;编码