Mathematical Biosciences and Engineering, 16(3): 1334–1347.
DOI: 10.3934/mbe.2019065
Received: 20 November 2018
Accepted: 21 January 2019
Published: 20 February 2019

http://www.aimspress.com/journal/MBE

*Research article*

# Solving flexible job shop scheduling problems with transportation time based on improved genetic algorithm

**Guohui Zhang*, Jinghe Sun, Xing Liu, Guodong Wang and Yangyang Yang**

School of Management Engineering, Zhengzhou University of Aeronautics, Zhengzhou 450015, China

*** Correspondence:** Email: zgh09@zua.edu.cn.

**Abstract:** In the practical production, after the completion of a job on a machine, it may be transported between the different machines. And, the transportation time may affect product quality in certain industries, such as steelmaking. However, the transportation times are commonly neglected in the literature. In this paper, the transportation time and processing time are taken as the independent time into the flexible job shop scheduling problem. The mathematical model of the flexible job shop scheduling problem with transportation time is established to minimize the maximum completion time. The FJSP problem is NP-hard. Then, an improved genetic algorithm is used to solve the problem. In the decoding process, an operation left shift insertion method according to the problem characteristics is proposed to decode the chromosomes in order to get the active scheduling solutions. The actual instance is solved by the proposed algorithm used the Matlab software. The computational results show that the proposed mathematical model and algorithm are valid and feasible, which could effectively guide the actual production practice.

**Keywords:** flexible job shop scheduling problem; transportation time; genetic algorithm; active scheduling

## 1. Introduction

Production scheduling problems are very important in manufacturing systems. It could make effective production planning to improve the production efficiency. The job shop scheduling problem (JSP) is one of the most popular scheduling problems existing in practice. This problem consists in assigning a set of operations on a set of machines such as each operation must be processed on one

machine. However, as the flexibility of processing in the shop continues to increase, each machine could process multiple operations. That is, each operation could be processed by many different machines. Hence, this problem named flexible job shop scheduling problem (FJSP). FJSP is an extension of the classic JSP. In the FJSP, there are two sub problems: machine selection and operation sequencing. The FJSP is known to be strongly NP-hard. There are many research on swarm intelligence optimization algorithms for the flexible job shop scheduling problems, such as genetic algorithm (GA) [1–5], particle swarm optimization (PSO) [6,7], ant colony algorithm (ACO) [8], shuffled frog-leaping algorithm [9], teaching learning based optimization algorithm (TLBO) [10], virus optimization algorithm [11], and differential evolution [12].

This paper studies the FJSP with transportation time for the following reasons. In most of the existing literatures on FJSP, each job could be processed immediately on the next machine after it is completed on the previous machine. In other words, the transportation times between the machines are often neglected. However, the transportation times are possible to affect the quality of the product in some specific production fields. And, transportation times have a direct impact on the production cycle of the product in the actual production process. Hence, in this paper, the transportation time is considered in the mathematic model, and to be solved.

At present, there are some research literatures considering the transportation times. Hurinkab [13] applied a novel simulated annealing to consider scheduling hybrid flowshop problems to minimize both total completion time and total tardiness. They integrated two realistic and practical assumptions which are sequence-dependent setup and transportation times into the problem. Naderi et al. [14] incorporated the sequence-dependent transportation times with a single-transporter system into the flow-shop. They proposed an adaptation of the simulated annealing algorithm to solve the problem. Boudhar and Haned [15] studied scheduling preemptive jobs on the identical parallel machines to minimize the total completion time. Several heuristic algorithms were presented and a high-quality lower-bound one was obtained considering the transportation of an interrupted job from a machine to another machine. Naderi et al. [16] considered the transportation times in the permutation flow-shops. They proposed six different models for the problem. Rossi and Andrea [17] proposed a swarm intelligence approach based on the disjunctive graph model to to schedule a manufacturing system with resource flexibility and separable setup times. Karimi et al. [18] proposed an adaptation of the imperialist competitive algorithm hybridized by a simulated annealing based local search to solve the problem. Overall, more literature about the transportation time focus on flow shop scheduling problem, but few on flexible job shop scheduling problem.

In this paper, we develop a modified GA to solve the FJSP with transportation time. The mathematical model is established to minimize the maximum completion time. The instance from the actual production is solved through the proposed algorithm. The computational results show that the proposed mathematical model and algorithm are valid and feasible.

The remainder of this paper is organized as follows. The problem description and mathematical model are presented in Section 2. The improved genetic algorithm is described in Section 3. The computational results are discussed in Section 4. Finally, Section 5 is the conclusion and addressed several promising research directions.

## 2. Problem description and mathematics modeling

### *2.1. Problem description*

Flexible job shop scheduling problem (FJSP) could be described as: $n$ jobs should be processed on $m$ machines. Each job may have a different number of operations. Each operation could be processed by any machine out of the set of alternative machine set. Each operation is completed and the job is transported to the next machine. If the two continuous operations are not processed on the same machine, the transportation times between the different machines is necessary to considered. FJSP is an NP-hard combinatorial optimization problem which could not be solved in a polynomial time. The optimization objective is to get the minimum makespan with the processing time and transportation time.

In this paper, further discussion on this problem is based on the following assumptions:

(1) The same job can only be processed by one machine at the same time, and the job cannot be interrupted once it starts processing.
(2) Each job could be processed more than once on the same machine.
(3) Operations belonging to different jobs could be processed in parallel.
(4) All the machine can be used at zero time.
(5) All jobs can be processed at zero time.
(6) The processing route of each job is fixed and known, that is, each operation will be sent to the next processing machine immediately after completion of processing, and the operation can be processed at this time.
(7) The processing time will be different due to the difference of the selected processing machines, and the processing time is known.
(8) The transportation time may be different between the different machines.

### *2.2. Mathematical model*

The mathematical model is described as: The job set $J = \{J_1, J_2, J_3,... J_g,... J_n\}$, $J_g$ is the $g$ job ($g$ = 1, 2, 3,... $n$). Machine set $M = \{M_1, M_2, M_3,... M_i,... M_m\}$, $M_i$ is the $i$ machine ($i$ = 1, 2, 3,... $m$). $O_{jh}$ denotes the $h$ operation of the job $j$, and defines the $O_{j(h\text{-}1)}$ as the previous operation of the $O_{jh}$, $O_{j'h'}$ represents the previous operation of the machine where $O_{jh}$ is processed. $F_{jh}$ expresses the processing completion time of the job $j$. $T_{ijh}$ indicates the time required for the $h$ operation of job $j$ on the machine $i$. $S_{ijh}$ is the processing start time of the operation $h$ of job $j$ on machine $i$. $C_{ijh}$ is the end time of the operation $h$ of the job $j$ on the machine $i$. $TransTime_{ie}$ is the transportation time of the job between the machine $M_i$ and the machine $M_e$. $C_j$ is the completion time of the job $j$. $C_{max}$ represents the maximum completion time in the completion time of all jobs. Considering the minimization of makespan, its objective functions and constraints are as follows:

$$C_{\max} = \min(max_{1\le j\le n}(C_j)) \tag{1}$$

$$C_{ijh} = S_{ijh}+T_{ijh} \tag{2}$$

$$C_{ijh}-C_{ij'h'} \ge T_{ijh} \tag{3}$$

$$\begin{cases} C_{ej(h-1)} + TransTime_{ie} & C_{ij'h'} < C_{ej(h-1)} + TransTime_{ie} \\ C_{ij'h'} & C_{ij'h'} > C_{ej(h-1)} + TransTime_{ie} \end{cases} \tag{4}$$

Eq 1 represents the expression of the total objective function, that is, the maximum completion time minimization; Eq 2 indicates that the completion time of the operation is equal to the sum of the operation starting time and the operation time; Eq 3 represents the resource constraints; Eq 4 indicates that the starting time of the machine in the process (the opening time of the gap) is less than that of the process. The transportation time of the last operation is constrained by the transportation time. Otherwise, the processing procedure is constrained by the resources of the current processing machine. Therefore, if the connected operation of the same job is processed on the same machine, it is only constrained by machine resources. For the adjacent two operations in the same job, the transportation time is taken into account, instead of the process sequence constraint in the traditional machining model.

For the convenience of understanding, Table 1 gives an example of a partial flexible job shop scheduling problem. The "-" in Table 1 indicates that the horizontal corresponding operation cannot be processed on a vertical corresponding machine. Table 2 gives the transportation time between the different machines.

**Table 1.** An example of partial flexible job shop scheduling problem.

| Jobs | Operations | Processing time | | | | |
|---|---|---|---|---|---|---|
| | | $M_1$ | $M_2$ | $M_3$ | $M_4$ | $M_5$ |
| $J_1$ | $O_{11}$ | 4 | 3 | - | 5 | 2 |
| | $O_{12}$ | 4 | - | 3 | - | 5 |
| $J_2$ | $O_{21}$ | - | 5 | - | 4 | - |
| | $O_{22}$ | 3 | 2 | 6 | - | - |
| | $O_{23}$ | - | 7 | 5 | 5 | 4 |
| $J_3$ | $O_{31}$ | 3 | 2 | 4 | 3 | - |
| | $O_{32}$ | 5 | - | 3 | 4 | - |

**Table 2.** The transportation time between different machines.

| Machines | Transportation time | | | | |
|---|---|---|---|---|---|
| | $M_1$ | $M_2$ | $M_3$ | $M_4$ | $M_5$ |
| $M_1$ | 0 | 2 | 3 | 2 | 4 |
| $M_2$ | 2 | 0 | 3 | 4 | 3 |
| $M_3$ | 3 | 3 | 0 | 5 | 2 |
| $M_4$ | 2 | 4 | 5 | 0 | 1 |
| $M_5$ | 4 | 3 | 2 | 1 | 0 |

Figures 1, 2 represent Gantt charts without and with the transportation time, respectively according to the Tables 1 and 2. In Figure 2, the yellow box denotes the transportation time between the two different machines. $T_{11}$ indicates the transportation time of the operation $O_{11}$ from $M_5$ to $M_1$ after the operation $O_{11}$ is completed on $M_5$. Obviously, the two figures are different. The makespan of

the problem without transportation time is much smaller than that of the problem with transportation time. However, the problem with transportation time is closer to the actual production.

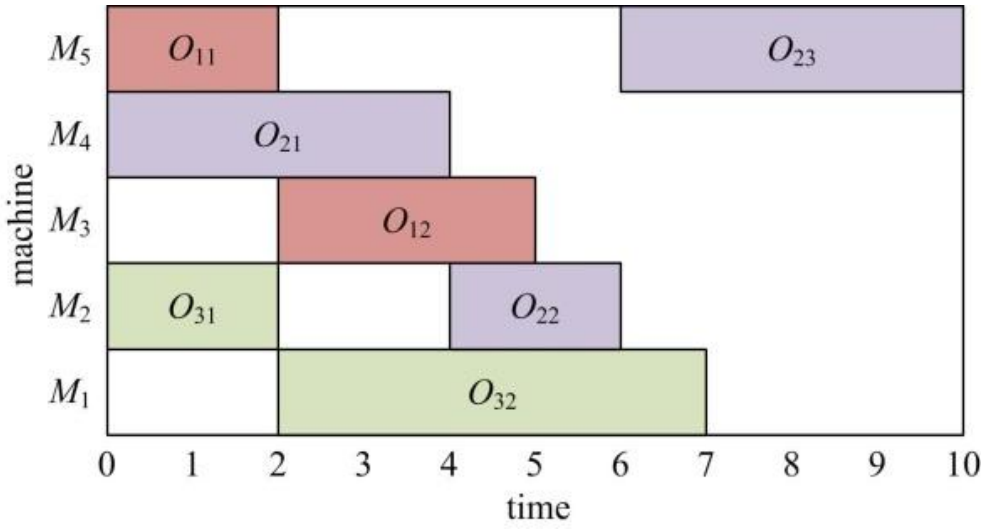


**Figure 1.** The Gantt chart without transportation time.

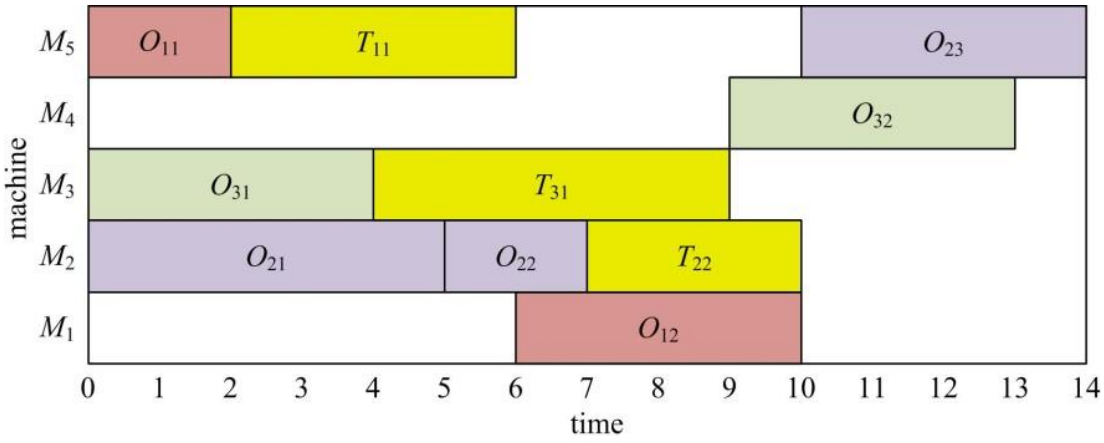


**Figure 2.** The Gantt chart with transportation time.

## 3. The proposed improved genetic algorithm

### *3.1. Encoding and decoding*

When the genetic algorithm is used to solve the problem, encoding and decoding are the first problems to be solved in the genetic algorithm. The problem of scheduling jobs in FJSP could be decomposed into two sub-problems: the routing sub-problem, which is assigning each operation to a machine selected out of a set of capable machines and the scheduling sub-problem, which consists of sequencing the assigned operations on all machines in order to obtain a feasible schedule minimizing the predefined objective function. In this paper, we adopted the method of the encoding method in [1] to encode two sub problems onto a chromosome, that is, a feasible solution of FJSP.

(1) Machine selection part: this part of the chromosome length is equal to the sum number of all operations. Each gene value is presented in integer, from left to right in sequence according to the process sequence of processing jobs. Each integer indicates the sequence number of the current operation of processing the job in the alternative machine set. Taking Table 1 as an example, as shown in the left half of the Figure 3, the gene string 3-2-2-1-4-3-2 indicates that the process $O_{11}$ is processed on the fourth machine in the alternative machines set, that is, the selected machine is $M_5$. Operation $O_{21}$ is processed on the second machine in the alternative machines set, that is, the selected machine is $M_4$, and so on.

(2) Operation sequencing part: The length of this part of the chromosome is equal to the total operations. Each gene is coded by the job number. The number of job shows the number of the work sequencing. As shown in the right half part of Figure 1, the gene string is 3-1-1-2-3-2-2, and the corresponding processing procedure is $O_{31}$-$O_{11}$-$O_{12}$-$O_{21}$-$O_{32}$-$O_{22}$-$O_{23}$.

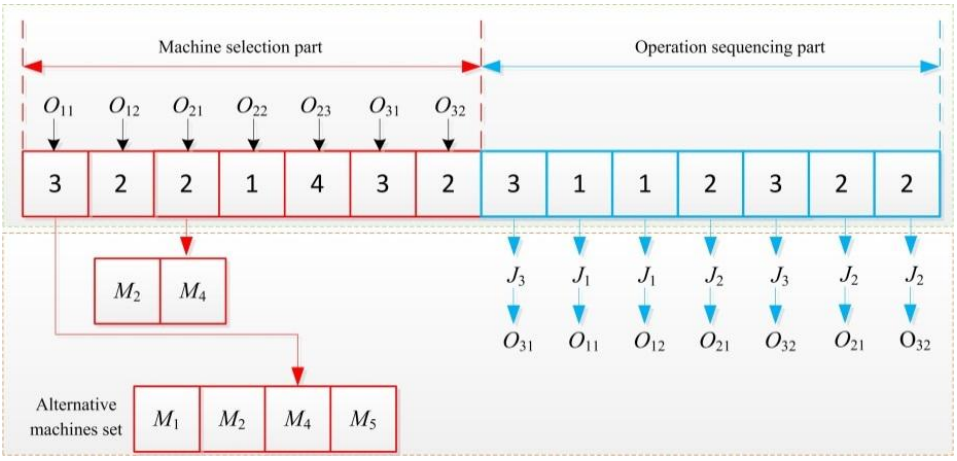


**Figure 3.** Chromosome encoding.

When decoding chromosomes, there are three kinds of scheduling, that is, semi-active scheduling, activity scheduling, and non-delayed scheduling. It has proved that the optimal scheduling values for regular performance measure exist in the active scheduling set. That is, in active scheduling, some processing procedures can be found to make it earlier. Because chromosomes contain two parts, namely machine selection sub problem and operation sequencing sub problem. First, machine selection is decode: Read the machine selection part from left to right, and convert it to machine sequence matrix $J_m$ and time sequence matrix $T$. $J_{m(j,\,h)}$ represents the machine number of the $h$ operation of the $j$ job, $J_m(j,...)$ represents the arrangement of all machine numbers processed in priority according to the working procedure of the job, and $T_{(j,\,h)}$ represents the $h$ processing time of the $j$ job. $J_{m(j,\,h)}$ is a one-to-one correspondence with $T_{(j,\,h)}$. It is shown that it is decoded in formula 5 and formula 6.

$$J_m = \begin{bmatrix} 4 & 3 & \\ 4 & 1 & 5 \\ 3 & 3 & \end{bmatrix} \tag{5}$$

$$T = \begin{bmatrix} 5 & 3 & \\ 4 & 3 & 4 \\ 4 & 3 & \end{bmatrix} \tag{6}$$

Secondly, the operation sequencing is decoded. The chromosome part of the operation is read in turn from left to right. The machine matrix and time matrix are decoded according to the machine selection part. And the processing machine and processing time corresponding to the processing procedure of each operation are obtained in turn. In order to ensure the generation of active scheduling after chromosome decoding, the operation left shift insertion method is applied to operation sequencing. The left shift insertion method is executed as follows:

(1) If the job $O_{jh}$ is the first process in machine $M_i$, it can be processed directly from the processing time of the previous operation $O_{j(h\text{-}1)}$ plus the end of the job transportation time.

(2) If operation $O_{jh}$ is the first processing procedure of the job $J$, then the machining directly starts from the zero time of the machine $M_i$. Otherwise, look up all the interval idle time [$TS_i$, $TE_i$]. $TS_i$ indicates the start time of the idle time on the machine $M_i$, and the $TE_i$ indicates the end time of the idle time on the machine $M_i$.

(3) Taking into account the transportation time, according to the formula 7, the earliest processing start time $t_a$ of the operation $O_{jh}$ is obtained, which can satisfy the order constraint of the job processing procedure.

$$t_a = \max\{F_{j(h\text{-}1)} + TransTime_{ie}, TS_i\} \tag{7}$$

(4) According to the Eq 8, determine whether the interval idle time can satisfy the insertion condition. If the satisfaction is inserted into the current idle time section, as shown in Figure 4; otherwise, the machine $M_i$ is processed on the machine in accordance with the time $t_b$ of Eq 9, in which the $TM_i$ represents the end time of the last machining process of the current machine $M_i$, as shown in Figure 5.

$$t_a + T_{ijh} \leq TE_i \tag{8}$$

$$t_b = \max\{F_{j(h\text{-}1)} + TransTime_{ie}, TM_i\} \tag{9}$$

According to the method presented above, the chromosome of the operation sequencing part is decoded until the end of the chromosome.

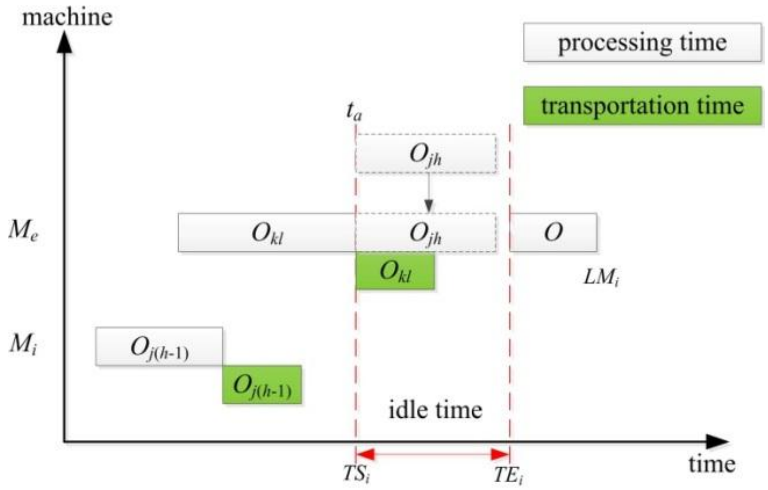


**Figure 4.** $t_a + T_{ijh} \leq TE_i$.

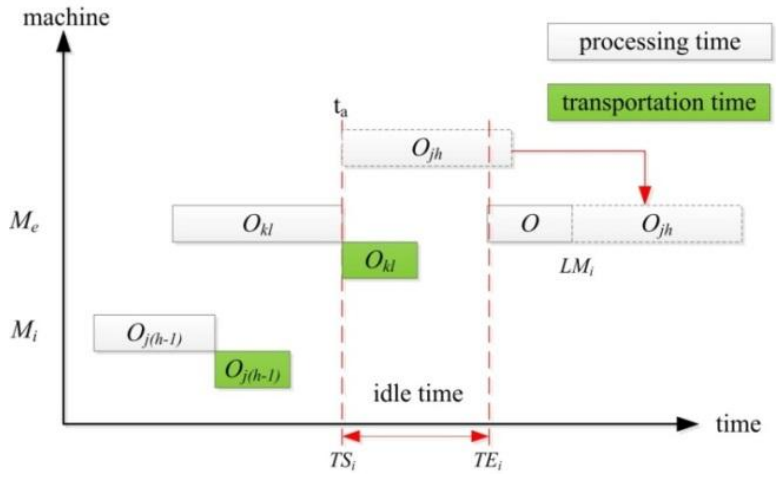


**Figure 5.** $t_a + T_{ijh} > TE_i$.

### *3.2. Initialization method*

The quality of the initial solution directly affects the solution quality and convergence speed of the genetic algorithm. Because FJSP not only solves machine selection sub-problem, but also solves the operation sequencing sub-problem. In this paper, the machine selection part uses integer random initialization method according to the characteristics of FJSP. That is, the value on each gene position of the machine selection part chromosome is randomly generated from the alternative machine set. The detailed description could be found in [2].

The operation sequencing part of each chromosome is also generated by random method.

### *3.3. Crossover operator*

The purpose of crossover is to exchange the information between the parents and retain the excellent information in the parent generation to produce new individuals. In this way, it could effectively reduce the probability of producing the infeasible solutions and search for the new generation. Since each chromosome consists of two parts, different methods are designed for crossover operator.

(1) Machine selection part: In order to ensure that the solution is still feasible solution after crossover operation, the multi-point crossover operation is adopted. That is, multiple crossover points are randomly selected, and two parents are used to exchange gene blocks. Figure 6 gives an example.

(2) Operation sequencing part: This part is based on operation encoding, and it is easy to produce infeasible solutions by traditional crossover operation. The improved method is to randomly divide the jobs into two groups, and better retain good operation sequence information from the parent individual. The detailed description could be found in [2].

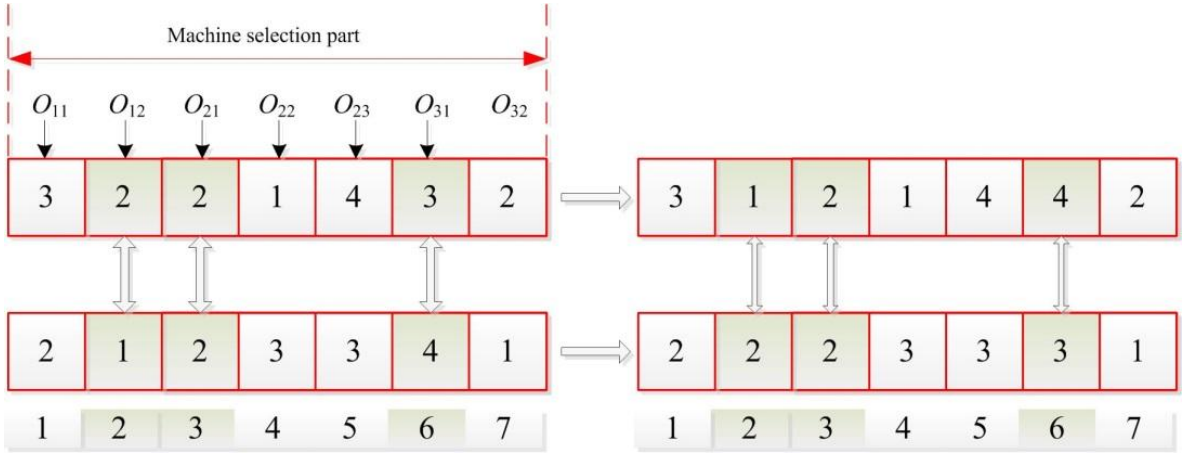


**Figure 6.** The machine selection part crossover.

### *3.4. Mutation operator*

Mutation operation is to increase the diversity of the population by changing the gene of each chromosome to produce a new individual. To some extent, improve the local search ability of genetic algorithm. Two different mutation operators are adopted for the two parts of the chromosome.

(1) Machine selection part: A gene location is randomly selected from the chromosome, and then a machine is randomly selected from the corresponding alternative machine set to replace the machine at the current gene location.

(2) Operation sequencing part: Interchange method is used. Two genes are randomly selected. Then, the values at the two gene positions are exchanged with each other.

### *3.5. Selection operator*

Through crossover operation and mutation operation, there may be bad solutions in the population. The selection operation is adopted to keep good individuals alive with greater probability. And the number of the population remains unchanged, so as to improve the computational efficiency and accelerate the global convergence. The more commonly used selection operators are rank-based selection, roulette wheel selection, seed selection and tournament selection. The tournament selection method has been proved that it has better or fairly convergence and computational complexity compared with other selection operators. In our proposed algorithm, the tournament selection is adopted. The fitness of a number of individuals was selected from the population to be compared, and the individuals with high fitness were selected and placed in the cross pool to fill the crossover pool.

## 4. Computational experiments and results

According to the above improved genetic algorithm, we use Matlab7.0 software. The instance from the actual production is executed on a P4 CPU (1.9 GHz) and 4G memory with Window 7 PC operating system. The main parameters of the genetic algorithm are as follows: population size 40, crossover probability 0.8, mutation probability 0.6, and maximum number of iterations is 200.

Table 3 shows the FJSP instance of 9 jobs on 5 machines after an actual instance is simplified. The "-" in Table 3 indicates that the operation of the job cannot be processed on the corresponding machine.

**Table 3.** An instance of FJSP.

| Job | Operation | Processing time | | | | |
|---|---|---|---|---|---|---|
| | | $M_1$ | $M_2$ | $M_3$ | $M_4$ | $M_5$ |
| $J_1$ | $O_{11}$ | - | 6 | 3 | 5 | - |
| | $O_{12}$ | 6 | - | 7 | - | 10 |
| $J_2$ | $O_{21}$ | - | 7 | - | 6 | - |
| | $O_{22}$ | 9 | 5 | 4 | - | - |
| | $O_{23}$ | - | 8 | 6 | 9 | 7 |
| $J_3$ | $O_{31}$ | 11 | - | 9 | 9 | 6 |
| | $O_{32}$ | - | 7 | 6 | 8 | - |
| | $O_{33}$ | 5 | - | 6 | - | 5 |
| $J_4$ | $O_{41}$ | 5 | 6 | 6 | 9 | - |
| | $O_{42}$ | 8 | - | 11 | 10 | - |
| | $O_{43}$ | 10 | 13 | - | 9 | 7 |
| $J_5$ | $O_{51}$ | - | 7 | 8 | 7 | - |
| | $O_{52}$ | 9 | - | 6 | - | 6 |
| $J_6$ | $O_{61}$ | 9 | - | 8 | 10 | 11 |
| | $O_{62}$ | 6 | - | 6 | - | 6 |
| | $O_{63}$ | 10 | 11 | 7 | 5 | - |
| $J_7$ | $O_{71}$ | - | 7 | - | 10 | 8 |
| | $O_{72}$ | 6 | - | 9 | 3 | 10 |
| $J_8$ | $O_{81}$ | 7 | 8 | 8 | - | - |
| | $O_{82}$ | 7 | - | 10 | 9 | 8 |
| | $O_{83}$ | - | 7 | 5 | - | 5 |

Table 4 represents the transportation time between the different machines based on the example in Table 3. In Table 4, the number of rows represents the machine corresponding to the operation, and the number of columns represents the processing machine corresponding to the next operation of the job.

**Table 4.** Transportation time between different machines.

| *Machine* | $M_1$ | $M_2$ | $M_3$ | $M_4$ | $M_5$ |
|---|---|---|---|---|---|
| $M_1$ | 0 | 1 | 1.5 | 2.5 | 1.5 |
| $M_2$ | 1 | 0 | 1 | 1.5 | 2 |
| $M_3$ | 1.5 | 1 | 0 | 2 | 2.1 |
| $M_4$ | 2.5 | 1.5 | 2 | 0 | 1.7 |
| $M_5$ | 1.5 | 2 | 2.1 | 1.7 | 0 |

In the actual production process, the transportation time between the different machines is existence. Hence, in this paper, the transportation time is added to the scheduling problem. As shown

in Figure 7, the optimal objective with transportation time is 32. And the Figure 8 is the Gantt chart of the FJSP without transportation time. The transportation time should be considered in the actual production for the entire production scheduling. The transportation time has a large impact on the final completion time. The pink box in Figure 7 represents the transportation time of each operation of each job. The other colors in Figure 7 represent the processing time of each operation on the corresponding machine. For example, the pink box of 701 on machine $M_5$ indicates that the transportation time of the $O_{71}$. That means that the first operation of the job $J_7$ is processed on machine $M_5$ and transported to the machine $M_4$. The transportation time needs 1.7 unit time.

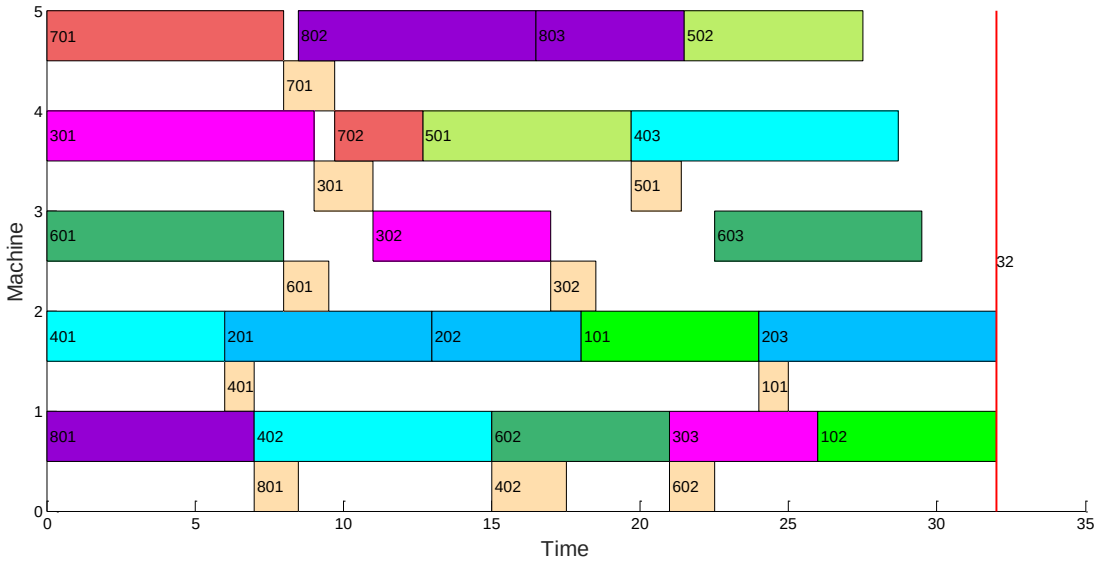


**Figure 7.** The Gantt chart with transportation time.

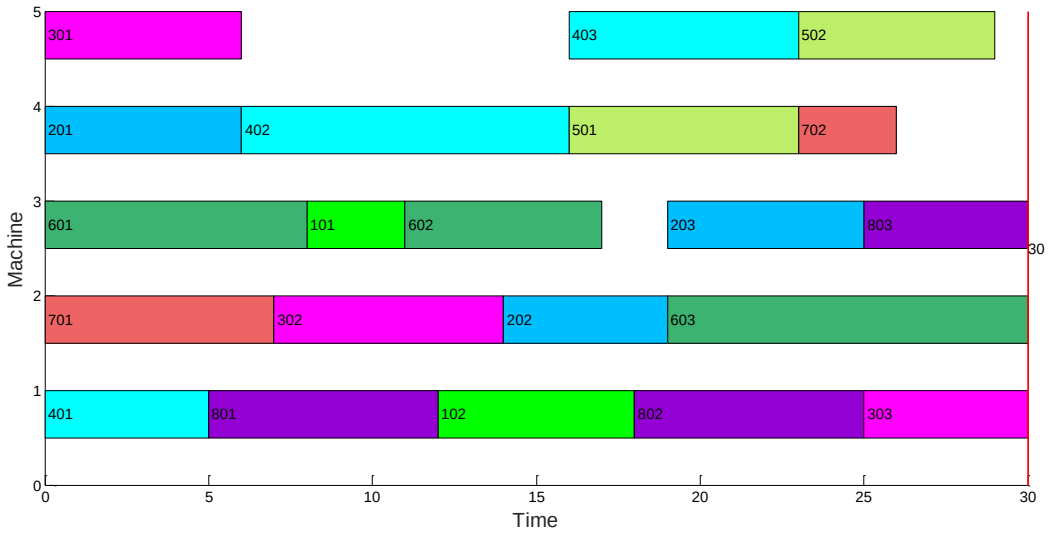


**Figure 8.** The Gantt chart without transportation time.

As shown in Figure 9, the improved genetic algorithm is used to solve the FJSP convergence

curve with transportation time. The dotted line in Figure 9 represents the mean change curve of each generation of objective values, and the real line represents the change curve of the optimal solution for each generation. The convergence curve of the optimal value is a downward trend. However, the mean value does not clearly reflect the downward trend. This shows that the algorithm needs further improvement in future research.

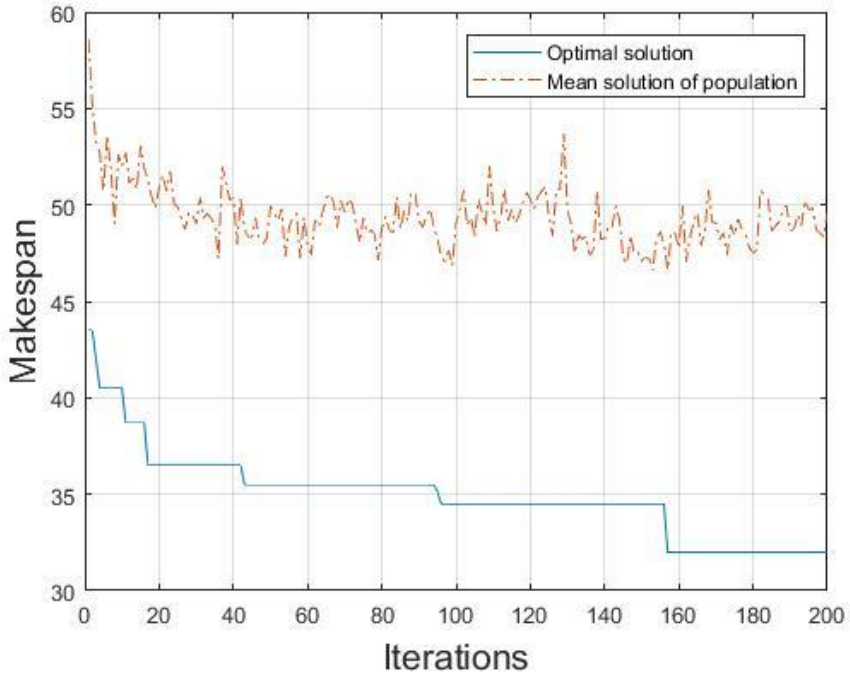


**Figure 9.** The convergence curve.

## 5. Conclusion and future research

In this research, the flexible job shop scheduling problem considering transportation time is solved. The mathematical model considering transportation time is established. An improved genetic algorithm based on the traditional genetic algorithm is proposed. In our proposed algorithm, the operation left shift insertion method is applied to decode the operation sequencing part. The improved genetic algorithm is realized by Matlab, and the performance of the computational results show that the proposed algorithm is feasible and effective. In the future research, the more effective optimization algorithm is designed according to characteristics of the flexible job shop scheduling problem with transportation time.

## Acknowledgements

This paper presents work funded by the National Natural Science Foundation of China (No. 51705472), Excellent Youth Foundation of Science & Technology Innovation of Henan Province (No.184100510001), Key scientific research projects of Henan Province (No. 182102210457), and Humanities and Social Sciences of Ministry of Education Planning Fund (No. 18YJAZH125).

## Conflict of interest

The authors declared that they have no conflicts of interest to this work.

## References


1. X. Y. Li and L. Gao, An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem, *Int. J. Prod. Econ.*, **174** (2016), 93–110.
2. X. Y. Li, C. Lu, L. Gao, et al., An effective multi-objective algorithm for energy efficient scheduling in a real-life welding shop, *IEEE T. Ind. Inf.*, **14** (2018), 5400–5409.
3. X. Y. Li, L. Gao, Q. K. Pan, et al., An effective hybrid genetic algorithm and variable neighborhood search for integrated process planning and scheduling in a packaging machine workshop, *IEEE T. Syst. Man. Cy. A.*, (2018), DOI: 10.1109/TSMC.2018.2881686.
4. G. H. Zhang, L. J. Zhang, X. H. Song, et al., A variable neighborhood search based genetic algorithm for flexible job shop scheduling problem, *Clust. Comput.*, (2018), 1–12.
5. G. H. Zhang, L. Gao and Y. Shi, An effective genetic algorithm for the flexible job-shop scheduling problem, *Expert. Syst. Appl.*, **38** (2011), 3563–3573.
6. G. H. Zhang, X. Y. Shao, P. G. Li, et al., An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem, *Comput. Ind. Eng.*, **56** (2009), 1309–1318.
7. J. Zhang, J. Jie, W. L. Wang, et al., A hybrid particle swarm optimisation for multi-objective flexible job-shop scheduling problem with dual-resources constrained, *Int. J. Comput. Sci. Math.*, **8** (2018), 526.
8. J. Wu, G. D. Wu, J. J. Wang, et al., Flexible job-shop scheduling problem based on hybrid ACO algorithm, *Int. J. Simul. Model.*, **16** (2017), 497–505.
9. D. M. Lei, Y. L. Zheng and X. P. Guo, A shuffled frog-leaping algorithm for flexible job shop scheduling with the consideration of energy consumption, *Int. J. Prod. Res.*, **55** (2017), 3126–3140.
10. Y. Xu, L. Wang, S. Y. Wang, et al., An effective teaching–learning-based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time, *Neurocomputing*, **148** (2015), 260–268.
11. C. Lu, X. Y. Li, L. Gao, et al., An effective multi-objective discrete virus optimization algorithm for flexible job-shop scheduling problem with controllable processing times, *Comput. Ind. Eng.*, **104** (2017), 156–174.
12. Y. Z. Zhou, W. C. Yi, L. Gao, et al., Adaptive differential evolution with sorting crossover rate for continuous optimization problems, *IEEE T. Cybern.*, **47** (2017), 2742–2753.
13. J. Hurink and S. Knust, Tabu search algorithms for job-shop problems with a single transport robot, *Eur. J. Oper. Res.*, **162** (2005), 99–111.
14. B. Naderi, M. Zandieh, A. K. G. Balagh, et al., An improved simulated annealing for hybrid flowshops with sequence-dependent setup and transportation times to minimize total completion time and total tardiness, *Expert. Syst. Appl.*, **36** (2009), 9625–9633.
15. M. Boudhar and A. Haned, Preemptive scheduling in the presence of transportation times, *Comput. Oper. Res.*, **36** (2009), 2387–2393.

16. B. Naderi, A. A. Javid and F. Jolai, Permutation flowshops with transportation times: Mathematical models and solution methods, *Int. J. Adv. Manuf. Tech.,* **46** (2010), 631–647.
17. A. Rossi, Flexible job shop scheduling with sequence-dependent setup and transportation times by ant colony with reinforced pheromone relationships, *Int. J. Prod. Econ.,* **153** (2014), 253–267.
18. S. Karimi, Z. Ardalan and B. Naderi, et al., Scheduling flexible job-shops with transportation times: Mathematical models and a hybrid imperialist competitive algorithm, *Appl. Math. Model.,* **41** (2017), 667–682.


AIMS Press