ISSN 2616-5775 Vol. 5, Issue 11: 1-9, DOI: 10.25236/AJCIS.2022.051101

Application of Improved Genetic Algorithm in FJSP

Yujie Shen

Shanghai Maritime University, Shanghai, 200120, China
shenyujie_1028@126.com

Abstract: *In this paper, the traditional genetic algorithm is improved, through MSOS coding, combined with IPOX and MPX adaptive crossover operators, the fitness function is improved, the adaptive crossover mutation probability function is introduced, and the simulation experiments are carried out on two standard instances of MK01 and* 8×8. *The solution of the* 8×8 *standard instance with the traditional genetic algorithm verifies the effectiveness and superiority of the improved genetic algorithm proposed in this paper.*

Keywords: *Flexible job-shop Scheduling, Genetic Algorithm, Algorithm improving, MSOS*

1. Introduction

With the development of the manufacturing industry and the intensification of competition, traditional job shop scheduling can hardly satisfy industrial production needs in the new epoch. Flexible job-shop scheduling (FJSP), as an extension of the traditional job shop scheduling problem (JSP), is gradually becoming the mainstream shop scheduling problem in the current industrial development. Therefore, it is of practical significance to deepening this paper's discussion and research on FJSP.

FJSP has been proved to be an NP-Hard problem[0], but with the expansion of the scale of FJSP, the traditional algorithm can hardly tackle FJSP of great scale, therefore more intelligent algorithms are needed to complete the vacancy in solving FJSP. Among all heuristic algorithms, the Genetic algorithm has fast convergence and good robustness, which shows great performance in tackling scheduling problems[0]. However, traditional genetic algorithm is prone to problems such as premature maturity and unrepresentative population selection, so many scholars made some improvements to it. Driss et al[0] Driss et al. proposed a new coding form and different crossover mutation strategies for the FJSP problem and verified its efficiency through an example. Li et al[0]raised a hybrid algorithm combining a genetic algorithm and tabu search algorithm, which combines efficient coding intersection with Neighbor forms and improves the local search ability of the genetic algorithm. Li Shanghan et al.[0] proposed a hierarchical hybrid hyperheuristic genetic algorithm to solve the fuzzy FJSP problem, combined the adaptive genetic algorithm and the variable neighbourhood search with the introduction of a simulated annealing mechanism, and verified the effectiveness of the algorithm by simulation.

2. FJSP mathematical model

As an extension of classic JSP, the problem can be described as, for a set $J = \{J_1, J_2, \cdots, J_n\}$ composed of n mutually independent jobs, there is $\forall J_i \in J$, which needs to go through a series of the processing is completed in sequence according to the fixed processing sequence. The set $M = \{M_1, M_2, \cdots, M_m\}$ represents the available processing machines, and O_{ij} represents that job j is processed on the machine i. If O_{ij} can only be processed on some machines in M, it is a partially flexible job shop scheduling problem; if O_{ij} can be processed on any machine in M, it is a fully flexible job shop scheduling problem. Therefore, FJSP can be described with a triplet[6] as formula (1).

$$F_m||C_{max} \tag{1}$$

3. Traditional genetic algorithm and its shortcomings

3.1. Traditional genetic algorithm

The Genetic Algorithm (GA) is a method to search for the optimal solution by imitating the natural

ISSN 2616-5775 Vol. 5, Issue 11: 1-9, DOI: 10.25236/AJCIS.2022.051101

evolution process. The basic operators of GA include selection operator, crossover operator and mutation operator. The step of the calculation process is as follows:

Firstly, encoding individuals, initializing the population and calculating each individual's fitness value, then the next generation of the population is selected by the roulette method. Secondly, the new-generation population is formed through the crossover operation between individuals and the individual's own mutation operation. It iterates continuously in this way until the termination condition is reached. The algorithm flow chart is shown in Figure 1: .

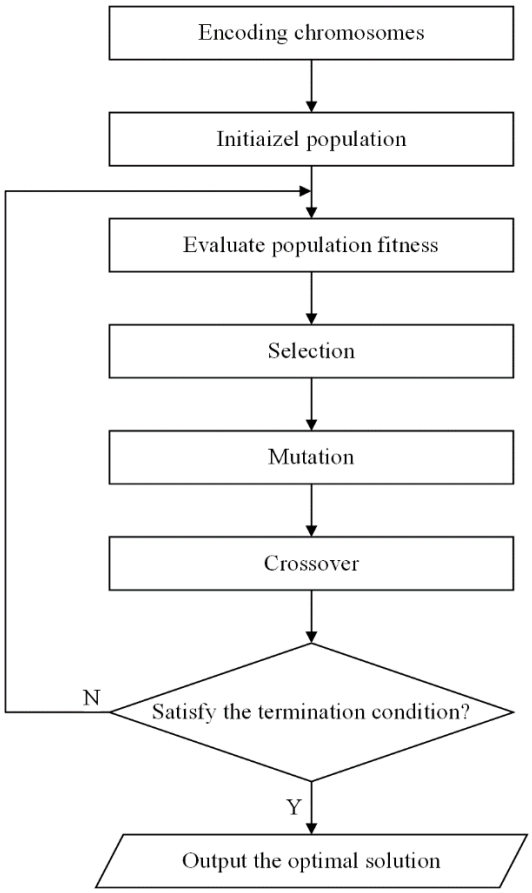

Figure 1: Traditional GA flow chart

3.2. Traditional genetic algorithm's shortcomings

Due to GA its own traits, the traditional genetic algorithm will have some defects worthy of attention and optimization[0]:

(1) Convergence in advance when a satisfactory solution or optimal solution is not obtained, namely that "premature";

(2) Chromosomal templates with good performance are destroyed during evolution, negatively affecting evolutionary efficiency;

(3) Inability to adapt to evolving populations.

In view of the above defects, the author hopes to propose some optimization methods to improve the performance of traditional genetic algorithms.

4. Improvements for traditional GA

4.1. Chromosome encoding and decoding

Encoding is based on the segmented encoding of machine selection (MS) and Operations Sequencing (OS), and the length of each part is T_0.[0] If there are n different jobs being processed in m machines,

ISSN 2616-5775 Vol. 5, Issue 11: 1-9, DOI: 10.25236/AJCIS.2022.051101

then $T_0 = n \times m$, the number of gene coding points on the chromosome is $2T_0 = 2 \times n \times m$ n, therefore. The encoding method of MSOS is shown in Fig.

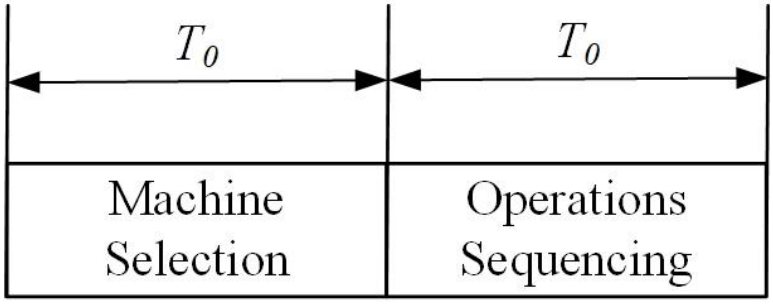

Figure 2: MSOS encoding description

Decoding is the inverse of encoding. For instance, if there is an encoding of a chromosome as (1,2,1,2,1,3,2,3,3,1,3,4,1,4,5,3,2,2), then the decoding of which is shown as Fig

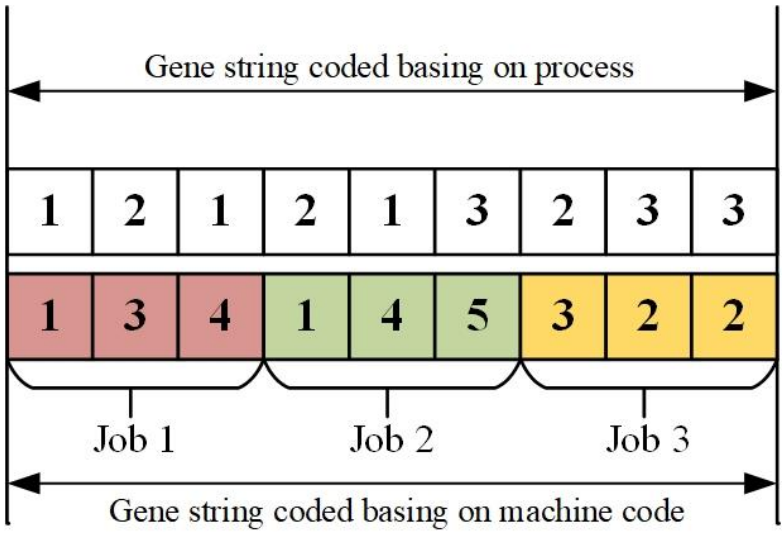

Figure 3: Coding instance

4.2. Population initialization

The quality of the population initialization affects the quality of the final solution.

In this paper, the population is initialized according to the global initialization method proposed by Zhang Guohui et al.[0] Since this initialization method is mature, most of them can be satisfactorily solved by this initialization method.

4.3. Fitness function transformation

The selection operation aims to pick out the better individuals in the population, retain its excellent genes, and make the population evolve in a better direction in the iterative process. Usually, the individual is eliminated during evolution according to the fitness of the individuals in the population. In general, the greater the fitness, the more likely the chromosome is to be retained. Since the optimization objective in this paper is C_{max}, that is to minimize the maximum completion time, the smaller the maximum completion time, the better the genes of the population.

In order to solve the possibility of slow convergence in the later stage of evolution due to the use of the fitness function in the early stage of evolution to select excellent individuals, the selected excellent individuals are not representative, and the convergence is slow. Therefore, an adaptation function that can dynamically change the fitness value variation scale according to the evolution process of the population is designed. According to the optimization objective, the fitness function can be obtained as shown in Equation (2), and the reciprocal of the maximum completion time is shown in Equation (3)[0]:

ISSN 2616-5775 Vol. 5, Issue 11: 1-9, DOI: 10.25236/AJCIS.2022.051101

$$f(i) = \frac{h(i) - h_{min}}{h_{max} - h_{min}} \tag{2}$$

$$h(i) = \frac{1}{C_{max}} \tag{3}$$

In the Equations, i is any chromosome in the initial population; h_{min} represents the minimum value of $h(i)$; h_{max} represents the maximum value of $h(i)$. When the individual similarity between populations is high, the value of $h_{max} - h_{min}$ becomes smaller, so that the overall fitness $f(i)$ becomes larger accordingly, effectively avoiding the slow convergence in the later stage of evolution.

After determining the fitness of each chromosome, the roulette method is used to select the chromosomes in the initial population.

4.4. Crossover and mutation

4.4.1. Crossover

The crossover is to randomly exchange some genes between two individuals in the population to generate a new combination of genes, hoping to combine beneficial genes together. The crossover of the two parts of the gene string in the chromosome is carried out separately, and the first part is based on the process of coding the gene string. The IPOX crossover operator is used for the crossover, and the second part uses the multipoint crossover MPX method for the crossover based on the machine assignment of coding gene strings.

4.4.2. IPOX crossover operator

The IPOX crossover operation is improved on the basis of the POX crossover operation. It only crosses the processing sequence of the process in the parent chromosome and retains the machine assigned by the process in the job to the offspring. The specific operation process of IPOX is as follows:

Step 1 Randomly divide all jobs into sets J_1 and J_2;

Step 2 Duplicate the jobs of P_1 contained in J_1 to C_1, and duplicate the jobs of P_2 contained in J_2 to C_2;

Step 3 Duplicate the jobs of P_1 contained in J_1 to C_2, and duplicate the jobs of P_2 contained in J_2 to C_1, and retain their sequences.

As Figure shows, P_1 and P_2 are the two parent chromosomes of the scheduling problem, and the crossover between them produces offspring chromosomes C_1 and C_2.

J_1={1,3}, J_2={2}

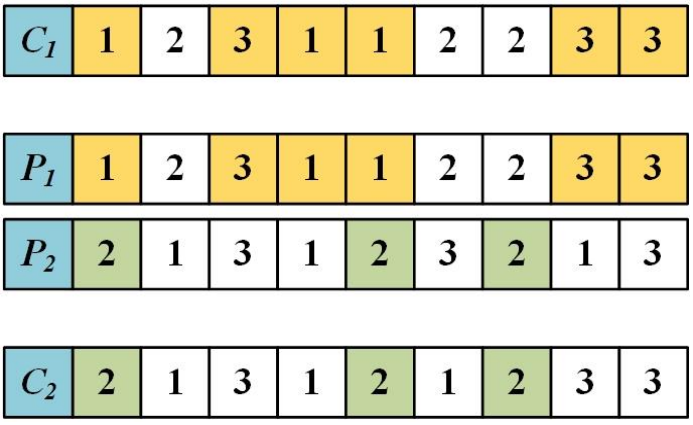

Figure 4: IPOX interchanging process

4.4.3. MPX crossover operator

For the machine selected by the process in the parent chromosome of MPX crossover, the processing order of the process is retained to the offspring. The process of multi-point crossover operation is as

ISSN 2616-5775 Vol. 5, Issue 11: 1-9, DOI: 10.25236/AJCIS.2022.051101

follows:

Step 1 Randomly generate an integer consisting of integers 0-1 equal to the length of the chromosome set $Rand[0,1]$;

Step 2 Select the processes that have the same position between those in set A and 1 in Rand [0,1] in B, then exchange the machines assigned to them. The processing sequences of other machines in P_1 and P_2 are retained to the progeny, resulting in progeny chromosomes C_1 and C_2. Since it is the progeny obtained by cross-exchange of processing machines in the same process, the resulting progeny chromosome must be a feasible solution.

As Fig shows, P_1 and P_2 are the two parent chromosomes of the scheduling problem, and the crossover between them produces offspring chromosomes C_1 and C_2.

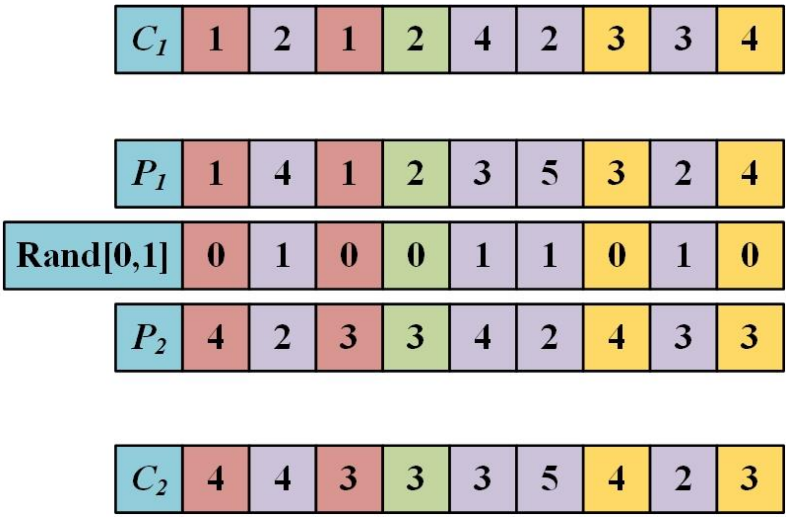

Figure 5: MPX interchanging process

4.4.4. Mutation

The purpose of the mutation operation is to improve the local search ability of the algorithm and maintain the diversity of the population while preventing the phenomenon of premature maturity. This paper designs the mutation algorithm according to OS and MS as follows:

(1) Mutation based on OS

Insertion mutation is performed on this part of the gene, that is, a gene is randomly selected from the chromosome and inserted into a random position.

(2) Mutation based on MS

Since each process can be completed by multiple machines, two processes are randomly selected, and then a machine is selected from the set of machines that perform these two processes (using a proportional selection strategy, the short processing time is preferred), and the selected machine number is put into the corresponding gene string based on the machine allocation encoding so that the solution obtained can be guaranteed to be a feasible solution.

4.5. The probability of crossover and mutation

Since the probability of occurrence of crossover in traditional genetic algorithms is fixed, it is generally given by empirical values. When the given value is small, the search range will be reduced, which is not conducive to finding a better solution; when the given value is large, it is easy to cause the already excellent chromosomal genes to deteriorate after crossover, which is not conducive to the evolution of chromosomes in the direction of superiority; in addition, the problem of premature maturity is easy to occur in the experiment. This paper adopts the method of adaptively changing the probability of crossover and mutation proposed in the literature to improve the population. The ability to evolve in the early stage of evolution and reduce the risk of falling into a local optimum to avoid premature problems. The adaptive crossover probability function is shown in Equation (4), the adaptive mutation probability function ^{0}is shown in Equation (5):

ISSN 2616-5775 Vol. 5, Issue 11: 1-9, DOI: 10.25236/AJCIS.2022.051101

$$p_c = \begin{cases} k_1 - \dfrac{k_1(g' - g_{avg})}{g_{max} - g_{avg}}, & g' > g_{avg} \\ k_2 \quad , & g' \le g_{avg} \end{cases} \tag{4}$$

$$p_m = \begin{cases} k_3 - \dfrac{k_3(g - g_{avg})}{g_{max} - g_{avg}}, & g' > g_{avg} \\ k_4 \quad , & g' \le g_{avg} \end{cases} \tag{5}$$

In the Equations, g_{max} represents the maximum fitness of all chromosomes in the current population; g_{avg} represents the average fitness of all chromosomes in the current population; g is the fitness degree of the individuals selected for mutation in the population.; g' is the value with the larger fitness among the two crossed chromosomes; $k_1, k_2, \mathrm{k}_3, \mathrm{k}_4 \in (0,1)$.

4.6. The overall flow of the algorithm

By redesigning the adaptive crossover probability and mutation probability, introducing the IPOX and MPX crossover algorithms, and optimizing the fitness function, the traditional genetic algorithm is improved. The overall flow of the algorithm is shown in Figure

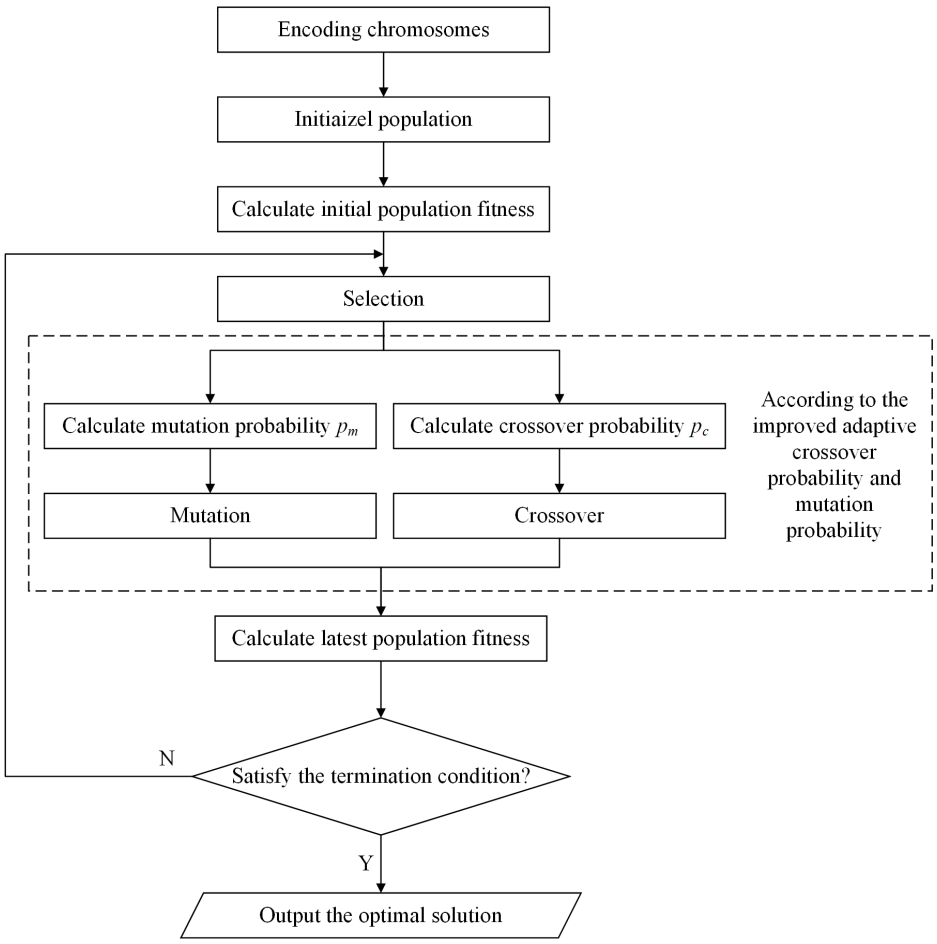

Figure 6: Improved GA flow chart

5. Experiments and Results

In order to verify the performance of the improved genetic algorithm (IGA), two standard instances of MK01 and 8×8 are used for experimental verification in this paper. The experiment environment is Python 3.10 Windows 11, and PyCharm 2022.1.1 is used to do the programming practice. The solution to the 8×8 test set is shown in Figure and Figure , The solution to the MK01 test set is shown in Fig and Figure .

ISSN 2616-5775 Vol. 5, Issue 11: 1-9, DOI: 10.25236/AJCIS.2022.051101

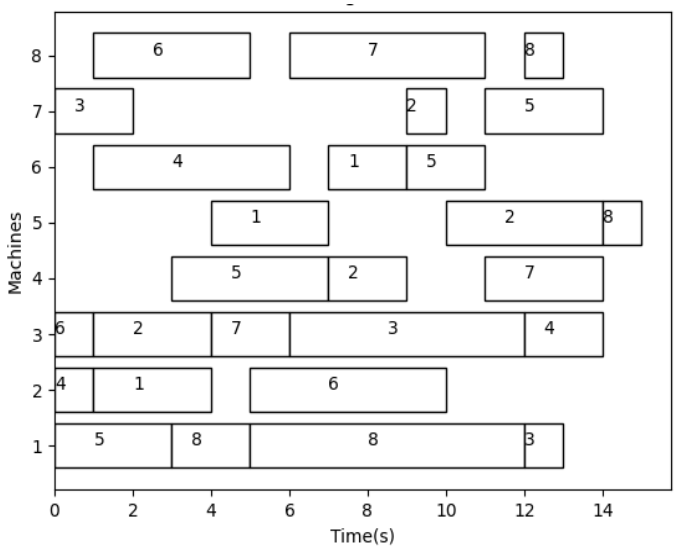

Figure 7: Instance 8×8 scheduling Gantt chart

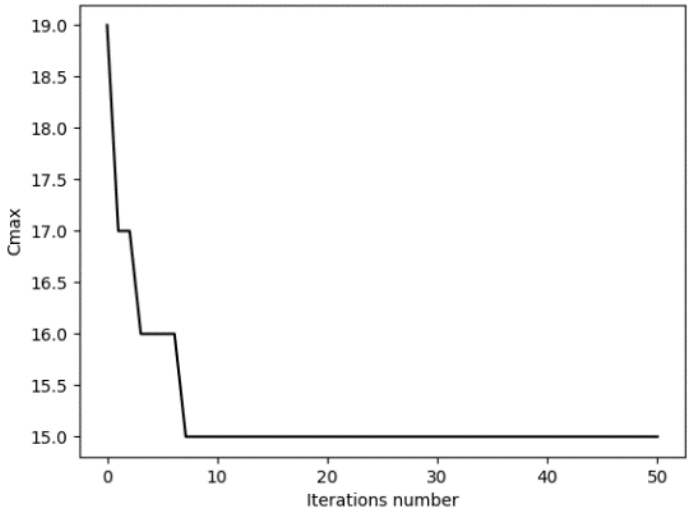

Figure 8: Improved GA iterations in makespan for the 8×8 instance

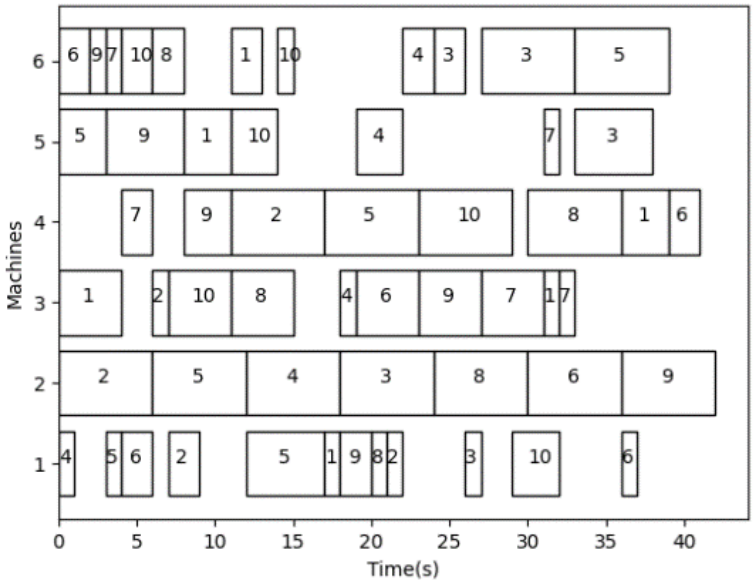

Figure 9: Instance Mk01 scheduling Gantt chart

ISSN 2616-5775 Vol. 5, Issue 11: 1-9, DOI: 10.25236/AJCIS.2022.051101

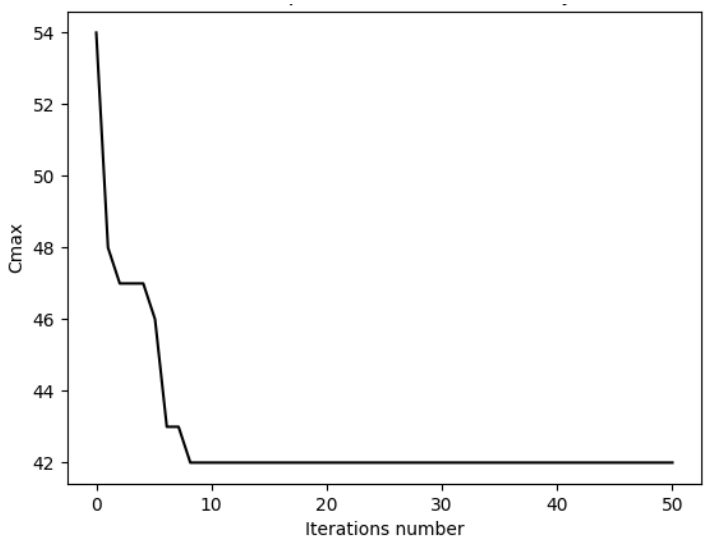

Figure 10: Improved GA iterations in makespan for the instance Mk01

In order to verify the superiority of the improved genetic algorithm, this paper uses the different solutions to the 8×8 standard instance of IGA and GA to compare with the traditional genetic algorithm, and the iterative results of the traditional genetic algorithm are shown in Figure

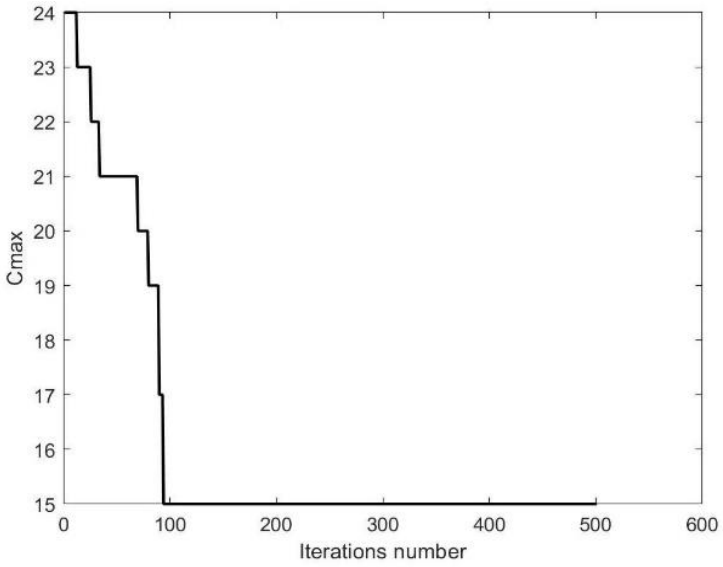

Figure 11: Traditional GA iterations in makespan for instance 8×8

By comparison, it can be found that the IGA only needs less than 10 iterations to complete the iteration, while the traditional GA needs more than 90 iterations to complete the iteration. Therefore, it can be considered that the improved GA is effective, fully reflecting its rationality and superiority.

6. Conclusion

In this paper, the existing improved genetic algorithm is studied and combined, and a new description of the fitness function is given. Through the MSOS encoding and mutation mechanism, combined with the IPOX and MPX crossover operators, the traditional genetic algorithm is improved. The simulation experiments are carried out on two standard instances MK01 and 8×8, and the traditional genetic algorithm for the 8×8 standard instance is conducted for comparison, therefore the improvement of the genetic algorithm is effective, so it also verifies the rationality and superiority of the IGA.

The next research direction mainly lies in the practical application of the IGA and the proposal of genetic evolution operators including crossover operator, mutation operator, selection operator, etc. that can adapt to more environments.

ISSN 2616-5775 Vol. 5, Issue 11: 1-9, DOI: 10.25236/AJCIS.2022.051101

References

[1] Xu Hua, Cheng Bing. Hybrid Genetic Bat Algorithm for Single-Objective Flexible Job Shop Scheduling Problem [J]. Journal of Chinese Computer Systems, 2018, 39(5): 1010-1015.
[2] Chaudhry I A, Khan A A. A research survey: review of flexible job shop scheduling techniques [J]. International Transactions in Operational research, 2016, 23(3): 551-591.
[3] Driss I, Mouss K N, Laggoun A. A new genetic algorithm for flexible job-shop scheduling problems [J]. Journal of Mechanical Science and Technology, 2015, 29(3): 1273-1281.
[4] Li X Y, Gao L. An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem [J]. International Journal of Production Economics, 2016, 174: 93-110.
[5] Li Shanghan, Hu Rong, Qian Bin etc. A Hyperheuristic Genetic Algorithm for Fuzzy Flexible Job Shop Scheduling [J]. Control Theory & Applications, 2020, 37(2): 316-330.
[6] Michael L. Pinedo. Scheduling [M]. New York, NY, USA: Springer Science+Business Media, LLC, 2016: 13-19.
[7] Li Qing, Wei Guagncun, Gao Lan, Qiu Guohua, Xiao Xinguang. An Improved Genetic Algorithm for Solving TSP Problems [J]. Software Guide, 2020, 19(03): 116-119.
[8] Gao Liang, Zhang Guohui, Wang Xiaojuan. Flexible Job Shop Scheduling Intelligent Algorithm and Its Application [M]. Wuhan: Huazhong University of Science and Technology Press, 2012.
[9] Chen Jinguang, Ma Lingye, Ma Lili. An Improved Genetic Algorithm for Solving Job Shop Scheduling Problem [J]. Computer Systems & Applications, 2021, 30(05): 190-195.
[10] Zhang Chaoyong, Liu Qiong, Qiu Haobo, Shao Xinyu. Research on Flexible job-shop Scheduling Problem Considering Processing Cost and Time [J]. Mechanical Science and Technology for Aerospace Engineering, 2009, 28(08): 1005-1011.
[11] Zheng Xianpeng, Wang Lei. Improved Genetic Algorithm for Job Shop Scheduling Problem [J]. Journal of Hebei University of Science and Technology, 2019, 40(06): 496-502.