Journal of Systems Engineering and Electronics
Vol. 32, No. 2, April 2021, pp.261 – 271

Solving flexible job shop scheduling problem by a multi-swarm collaborative genetic algorithm

WANG Cuiyu, LI Yang, and LI Xinyu*

School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China

Abstract: The flexible job shop scheduling problem (FJSP), which is NP-hard, widely exists in many manufacturing industries. It is very hard to be solved. A multi-swarm collaborative genetic algorithm (MSCGA) based on the collaborative optimization algorithm is proposed for the FJSP. Multi-population structure is used to independently evolve two sub-problems of the FJSP in the MSCGA. Good operators are adopted and designed to ensure this algorithm to achieve a good performance. Some famous FJSP benchmarks are chosen to evaluate the effectiveness of the MSCGA. The adaptability and superiority of the proposed method are demonstrated by comparing with other reported algorithms.

Keywords: flexible job shop scheduling problem (FJSP), collaborative genetic algorithm, co-evolutionary algorithm.

DOI: 10.23919/JSEE.2021.000023

1. Introduction

Scheduling is an important aspect of planning in current manufacturing systems; production scheduling allocates relative production resources to tasks and order to meet all constraints and optimization goals [1−3]. Therefore, the scheduling technology can improve manufacturing efficiency by reducing scheduling conflicts, and reducing process time [4]. In modern manufacturing system, intelligent systems and machines are used to improve production efficiency [5]. The intelligent systems and machines are widely used in modern manufacturing systems so that the same machine can perform multiple operations. [6]. In this case, the flexible job shop scheduling problem (FJSP) has attracted increasing attention because it considers machine flexibility [7].

The FJSP is a typical scheduling problem in modern manufacturing systems [8,9]. It is also an NP-hard problem [10]. In 1990, Brucker & Schile [11] first presented this problem and some methods to solve it. The methods to solve the FJSP primarily include the genetic algorithm (GA) [12,13], the iterated greedy algorithm [14], the tabu search (TS) [15], the variable neighbourhood search algorithm (VNS) [16], the particle swarm optimization (PSO) algorithm [17] and some hybrid algorithms [18,19]. The results of research on the FJSP are also applied to several practical cases [20−22].

The FJSP can be decomposed into two different sub-problems: machine selection (MS) problem and operation sorting (OS) problem. At present, unified coding is usually used to solve the problem. However, due to the different characteristics of these two sub-problems, unified coding will make the solution space more complex, which is not conducive to the solution of the algorithm. Therefore, to solve the FJSP, a multi-swarm collaborative genetic algorithm (MSCGA) based on the collaborative optimization algorithm is proposed [23−25]. The collaborative evolutionary algorithm simulates the evolutionary process of interaction between two species in nature. [26]. While a species is evolving, it interacts with and adapts to other species. This situation is similar to that of the two sub-problems in the FJSP. In this algorithm, the coevolution of different populations is used to solve two sub-problems. The experimental results demonstrate that the proposed MSCGA algorithm has obvious advantages.

The main innovation of this paper is to combine the characteristics of the FJSP, combining the advantages of the evolutionary algorithm and collaborative optimization to improve the GA. This outcome indicates that the proposed approach is an effective method to use in FJSP research.

The remainder of this paper is organized as follows. Section 2 provides details on the literature review. Section 3 describes the problem formulation. Section 4 pre-

Manuscript received November 23, 2020.
*Corresponding author.
This work was supported by the National Key R&D Program of China (2018AAA0101700), and the Program for HUST Academic Frontier Youth Team (2017QYTD04).

sents the MSCGA-based approach for solving the FJSP. Section 5 reports the experimental results. Section 6 discusses the conclusions and future studies.

2. Literature review

After the FJSP was first presented in 1990, many solving methods have been published. At present, there are two categories to solve permutation flow shop scheduling problem (PFSP): the exact method and the approximation method.

Brucker & Schile [11] presented this problem firstly and proposed a polynomial graphical algorithm. Gomes et al. [27] presented a new integer linear programming model and used commercial software to solve it. However, it is difficult to obtain a satisfactory solution in a short time when an accurate algorithm is used to solve large-scale problems [12]. Therefore, approaches used for solving the FJSP focus on approximation methods [7,10].

Kacem et al. [28] used a localization approach to address the MS sub-problem and applied a GA to solve the OS sub-problem. Ho et al. [29] proposed a learnable genetic architecture (LEGA). Yin et al. [30] proposed a multi-objective genetic algorithm to solve the FJSP. Homayouni et al. [13] proposed an operation-based multi-start biased random key genetic algorithm (BRKGA) coupled with greedy heuristics for the FJSP with transportation. Shi et al. [31] proposed a multi-population genetic algorithm (GA) with an Erdos & Renyi (ER) network for solving the FJSP. Chen et al. [32] presented a self-learning genetic algorithm for FJSP. Ding et al. [17] proposed an improved PSO algorithm. By improving the encoding/decoding scheme, a good solution was obtained. Ricardo and Arturo [19] proposed a Pareto approach for the FJSP with process plan flexibility. Lu et al. [33] presented a new multi-objective discrete virus optimization algorithm. Li et al. [12] improved the Jaya algorithm to solve the FJSP with transportation and setup times. Alejandro et al. [34] showed a biomimicry hybrid bacterial foraging optimization algorithm for the FJSP. Ding and Gu [17] improved the PSO algorithm for the FJSP by considering a new encoding and decoding method.

There are also several hybrid algorithms for the FJSP. Zribi et al. [35] proposed a hierarchical method for the FJSP. Gao et al. [36] proposed a hybrid genetic for solving the FJSP. The variable neighborhood descent algorithm is also used to speed up the search process of the GA algorithm. Li et al. [18] proposed a hybrid GA and TS algorithm for the FJSP and achieved good results. Wu et al. [37] presented a hybrid pigeon-inspired optimization and simulated annealing algorithm for the FJSP.

In this paper, the MSCGA is developed for the FJSP. Few papers have used the co-evolutionary algorithm to solve this problem [38]. Only Rou et al. [39], Xing et al. [40] and Shi et al. [31] have designed three co-evolutionary algorithms for solving the FJSP. Therefore, it is compared to these three algorithms. The proposed method obtains better results than these algorithms.

3. Problem model

There exists a set of n jobs $J = \{J_1, J_2, J_3, \cdots, J_n\}$ and a set of m machines $M = \{M_1, M_2, M_3, \cdots, M_m\}$. Each job J_i consists of a sequence of operations $\{O_{i1}, O_{i2}, O_{i3}, \cdots, O_{in_i}\}$ where n_i is the number of operations in J_i. Each operation $O_{ij}(i = 1, 2, \cdots, n; j = 1, 2, \cdots, n_i)$ must be processed by one machine in a set of given machines $M_{ij} \subseteq M$ [18].

The scheduling goal of this paper is to minimize the maximal completion time, also known as the makespan. It contains two sub-problems, making the FJSP become more complex and challenging [29].

The hypotheses considered in this paper are summarized as follows [7]:

(i) All machines and jobs are available at time zero.

(ii) No precedence constraints among operations of different jobs.

(iii) Each job cannot be processed by two machines at the same time and each machine can only perform one operation at the same time.

(iv) The transfer time between machines is assumed to be negligible.

4. The proposed MSCGA for solving FJSP

4.1 Optimization strategy of MSCGA

Because of the collaborative optimization strategy, the proposed MSCGA explores the solution space of the MS and OS separately and collaboratively. This optimization process is performed until the termination criterion is obtained. In this method, each job contains one MS swarm to determine the selected machines for each operation. There is one OS swarm to optimize the scheduling of all jobs. Under this optimization strategy, each swarm can choose its own algorithms. This helps ensure the most suitable algorithm for each swarm on the basis of its own features. In this study, the GA is used for each MS and OS swarm. Therefore, the overall process of the MSCGA proposed in this paper is as follows [26]:

Step 1 Parameters setup.

Step 2 Initialization. Generate N+1 swarms, where there are N MS swarms and one OS swarm, and select one individual from every N+1 swarm to form one FJSP solution.

Step 3 Evaluation. Calculate the fitness value of each individual in the swarm and combine it with selected individuals from other swarms.

Step 4 Terminate criteria satisfied? If yes, stop and output the results, otherwise go to Step 5.

Step 5 Collaborative evolution. Generate new individuals for each swarm according to the genetic operators. Go back to Step 3.

4.2 Encoding and decoding

The encoding method in [18] is adopted in this paper. Because there are two types of swarms in this method, the two sub-problems require different encoding methods.

The encoding method of an operating system cluster is composed of job numbers. Assuming there are n jobs, the length of the OS string is equal to $\sum_{i=1}^{n} On_i$. The initial OS population is generated according to the encoding principle.

Each MS swarm represents the MS of the corresponding job. The chromosome length is equal to the number of operations in the relative job.

An FJSP solution is made up by one OS individual and N MS individuals selected from the MS swarms one by one. [18]. The procedure of decoding is as follows:

Step 1 Generate the machine of each operation based on the MS string.

Step 2 Determine the set of operations for every machine.

Step 3 Determine the set of machines for every job.

Step 4 Calculate the allowable starting time for every operation.

Step 5 Check the idle time of the machine, and obtain the idle areas. Then, check these areas in turn.

Step 6 Calculate the completion time of every operation.

Step 7 Generate the sets of time for every operation of every job.

4.3 Initial population and fitness evaluation

The operation-based encoding method is employed in this paper. Therefore, any chromosome can be decoded into a feasible solution. The optimization goal of this paper is to minimize the maximum completion time.

4.4 Genetic operators

Good and effective genetic operators are important for the proposed algorithm, as they can effectively handle the problem. There are three genetic operators in a GA: selection, crossover and mutation. This paper uses two different coding methods: OS swarms and MS swarms. In addition, these swarms have their own genetic operators. This operation is described in the following sub-sections.

4.4.1 Selection

In this paper, the OS swarms and MS swarms have the same selection operator. In addition, this paper uses two selection operators. The first one is the elitist selection scheme where the good fitness in parents is transferred to offspring. The other selection operator is the random selection operator. In this case, the algorithm will randomly select individuals for crossover and mutation operations.

4.4.2 Crossover

The OS swarm adopts precedence operation crossover (POX). With POX, the offspring effectively inherits good characteristics from the parents. The process for POX is shown as follows (P stands for the parent and O for the offspring):

Step 1 Job set $J = \{J_1, J_2, J_3, \cdots, J_n\}$ is randomly divided into two groups Jobset1 and Jobset2.

Step 2 Move the job in P1 that belongs to Jobset1 to the position O1; move the job belonging to Jobset2 in P2 to the position O2.

Step 3 Put the remaining jobs in P2 into O1, and the remaining jobs in P1 into O2.

An example is shown in Fig. 1. This example represents four jobs, and Jobset1 = {1, 4}.

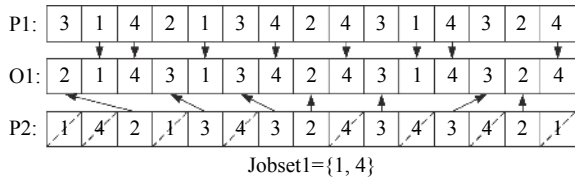

Fig. 1 An example of POX for the OS swarm

In this paper, two-point crossover is used for MS swarm crossover operation. This crossover operation randomly chooses two positions at first. Then, this algorithm generates two child strings by swapping all elements between the positions of the two parent strings. This crossover ensures the generation of the feasible offspring when the selected parents are feasible.

4.4.3 Mutation

In this paper, two mutation operators are adopted for the OS swarm. In MSCGA, one mutation operator is selected randomly (50%) to mutate the individual in the OS swarm.

The first one is the swapping mutation. The process for swapping mutation is shown as follows.

Step 1 Select two positions in P.

Step 2 Swap the elements in the selected positions to generate O.

The second mutation operator is based on the neighborhood mutation method.

Step 1 Select three jobs in the parent, and generate all neighborhood chromosomes.

Step 2 Select the best chromosome as the current chromosome in the neighborhood chromosome.

For the MS swarm, a mutation operator is designed as follows:

Step 1 Select r positions in the parent.

Step 2 Change the value of each location to another machine in the corresponding collection.

4.5 Terminating criteria

If the current iteration number reaches the maximum algebra, the algorithm stops.

4.6 Framework of the MSCGA

The basic procedure of this algorithm is described as follows [26]:

Step 1 Set the parameters of the MSCGA.

Step 2 Initialization. Generate N+1 swarms, where there are N MS swarms and one OS swarm, and select one individual from every N+1 swarm to form one FJSP solution.

Step 3 Each individual in each population is evaluated, where individuals are combined with all randomly selected partners from other populations. When Gen = 1, the optimal f_{best} is recorded.

Step 4 Is the terminating criteria satisfied? (Gen $\leqslant$ max Gen?)

If yes, go to Step 6; if it is not, go to Step 5.

Step 5 Collaborative evolution. Generate the new swarms; q is the serial number of each swarm (q = 1, 2, 3,$\cdots$, i,$\cdots$, N, N+1), and initialize q = 1.

Step 5.1 The operator is used to generate a new population, which is set as the qth group.

Step 5.2 Is $q \leqslant N+1$?

If yes, go to Step 5.3.

If it is not, set Gen = Gen+1 and go back to Step 3;

Step 5.3 Set $q = q$+1 and go to Step 5.1;

Step 6 Output the best fitness f_{best} with the solution.

The flow chart is shown in Fig. 2.

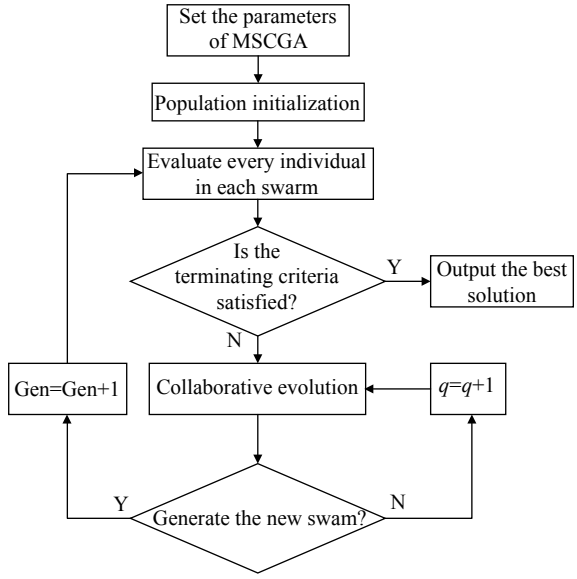

Fig. 2 Framework of MSCGA

5. Experimental results

This paper selects three well-known benchmarks to evaluate the performance of the proposed MSCGA. These benchmarks include 48 open problems for the FJSP. The first instance is the MK data with 10 problems from Brandimarte [41]. The second consists of data of 18 problems from Peres et al. [42]. The third instance is the Fattahi data with 20 problems from Fattahi et al. [43]. Comparisons between the proposed algorithm and other algorithms including three co-evolutionary algorithms are also provided.

The algorithm parameters in this paper are set as follows: the size of the population of the OS swarm Popsize_1= 300; the population size of the MS swarm Popsize_2= 300; the maximum generations maxGen = 200; the crossover probabilistic of OS swarm p_{c1}=0.8; the crossover probability of MS swarm p_{c2}=0.8; the mutation probability of the OS swarm p_{m1}=0.2; and the mutation probability of the MS swarm p_{m2}=0.2. The proposed MSCGA is coded in C++ and implemented on a computer with a 2.3 GHz Core (TM) i7 CPU with 16 GB RAM memory.

5.1 Results and comparison with the MK data

The MK data with 10 problems are among the most well-known benchmarks for the FJSP. Many researchers have used this benchmark to evaluate algorithms. Table 1 shows the results of the proposed algorithm. LB is the lower bound, and UB is the upper bound. MSCGA is the proposed algorithm. LEGA2008 [12], VNSGA2019 [39], HBFOA2020 [35], PSO2020 [17], SLGA2020 [32], and two collaborative evolutionary algorithms, CCGA2010 [39] and MPICA2011 [40] are selected for comparison.

Table 1 shows that the proposed MSCGA obtains the

nine best results among these algorithms. This indicates that the proposed approach achieves good results for the MK data. Figs. 3−5 illustrate the Gantt charts of problems MK02, MK09 and MK10 respectively.

Table 1 Experimental results of MK data and comparison with other methods

Problem	(LB, UB)	LEGA2008	VNSGA2019	HBFOA2020	PSO2020	SLGA2020	CCGA2010	MPICA2011	MSCGA
MK01	(36, 42)	40	40	40	40	40	41	39	40
MK02	(24, 32)	29	26	26	29	27	27	29	26
MK03	(204, 211)	—	204	204	204	204	204	204	204
MK04	(48, 81)	81	60	60	66	60	62	65	60
MK05	(168, 186)	186	173	172	175	172	173	173	173
MK06	(33, 86)	86	58	57	77	69	64	67	57
MK07	(133, 157)	157	144	139	145	144	140	144	139
MK08	(523, 523)	523	523	523	523	523	523	523	523
MK09	(299, 369)	369	307	307	320	320	328	311	307
MK10	(165, 296)	296	198	205	239	254	225	229	198

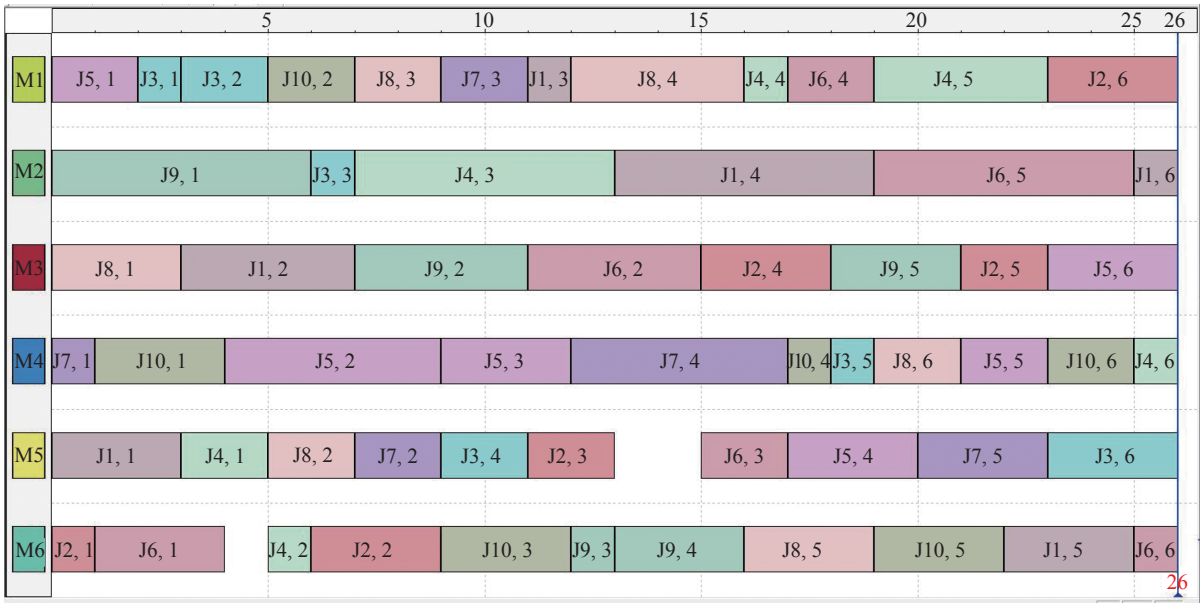

Fig. 3 Gantt chart of problem MK02 (Makespan=26)

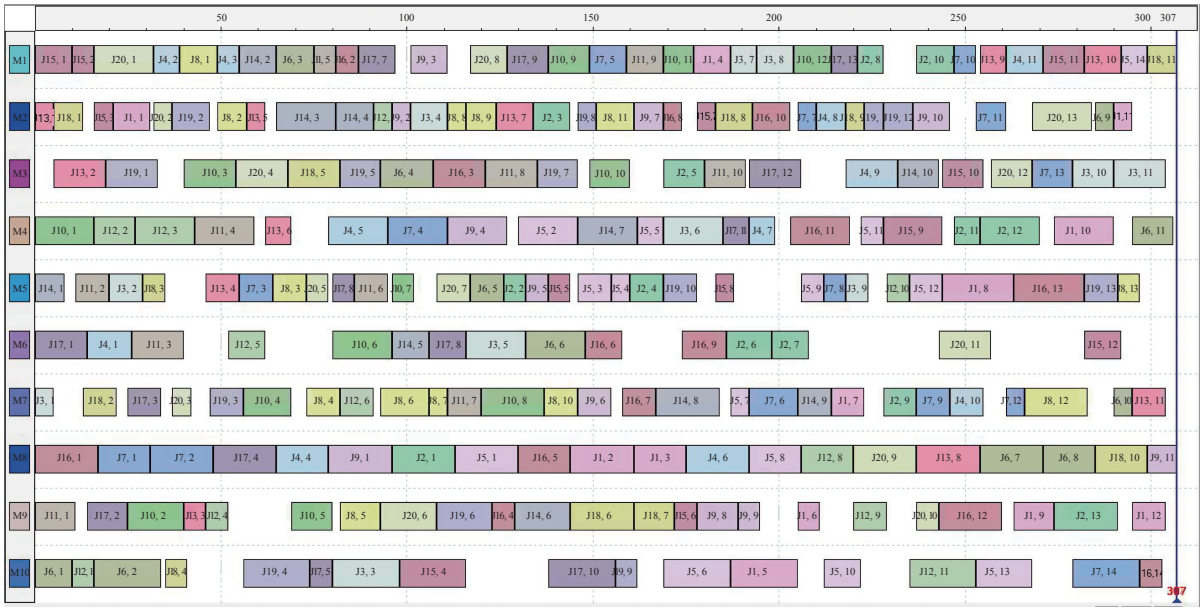

Fig. 4 Gantt chart of problem MK09 (Makespan=307)

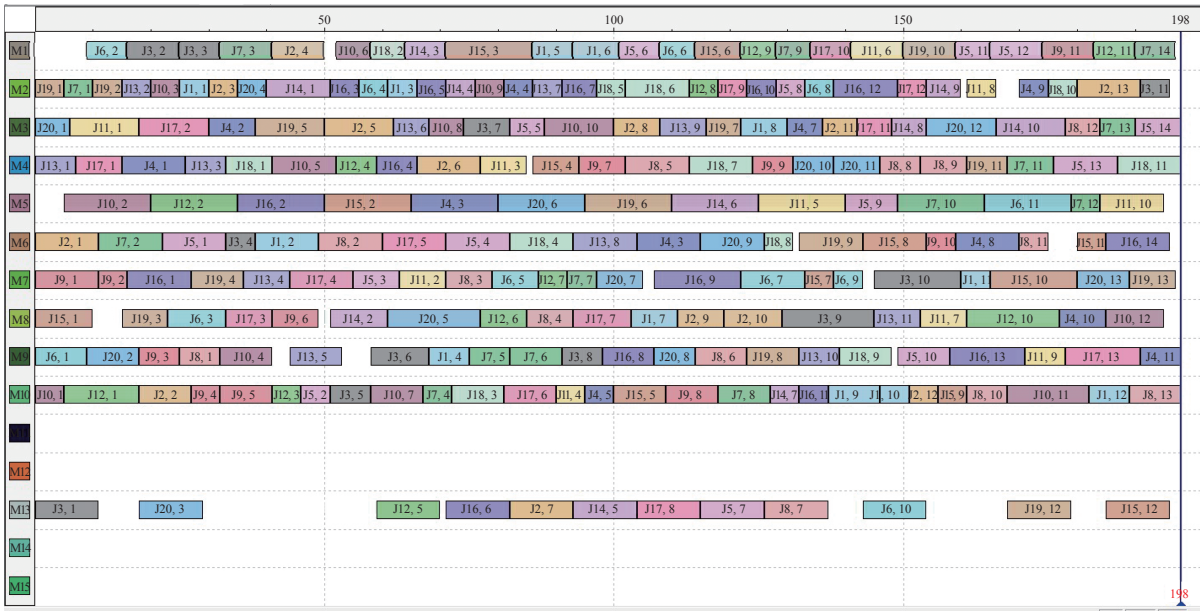

Fig. 5 Gantt chart of problem MK10 (Makespan=198)

5.2 Results and comparison with the Peres data

The Peres data with 18 problems were adopted from Peres et al. [42]. Table 2 shows the experimental results and comparisons with the other algorithms. The MSCGA is the proposed algorithm. Integrated approach tabu search (IATS), TS, general PSO (GPSO) and discrepancy search (DS) represent the reported algorithms from Peres et al. [42], Mastrolilli et al. [15], Gao et al. [44] and Hmida et al. [45] respectively. The proposed MSCGA obtains the 13 best results among these five algorithms. The experimental results show that the number of the best results obtained by MSCGA is greater than the number obtained by the other algorithms. This means that the proposed approach can obtain increasingly better results than the other approaches. Figs. 6−8 illustrate the Gantt charts of problems 01a, 07a and 11a respectively.

Table 2 Experimental results of Peres data and comparisons with other methods

Problem	(LB, UB)	IATS	TS	GPSO	DS	MSCGA
01a	(2 505, 2 530)	2 530	2 518	2 539	2 518	2 515
02a	(2 228, 2 244)	2 244	2 231	2 244	2 231	2 231
03a	(2 228, 2 235)	2 235	2 229	2 232	2 229	2 229
04a	(2 503, 2 565)	2 565	2 503	2 523	2 503	2 506
05a	(2 189, 2 229)	2 229	2 216	2 234	2 216	2 216
06a	(2 162, 2 216)	2 216	2 203	2 218	2 196	2 197
07a	(2 180, 2 408)	2 408	2 283	2 361	2 283	2 279
08a	(2 061, 2 093)	2 093	2 069	2 086	2 069	2 069
09a	(2 061, 2 074)	2 074	2 066	2 073	2 066	2 066
10a	(2 198, 2 362)	2 362	2 291	2 362	2 291	2 287
11a	(2 010, 2 078)	2 078	2 063	2 083	2 063	2 060
12a	(1 969, 2 047)	2 047	2 034	2 050	2 031	2 033
13a	(2 161, 2 302)	2 302	2 260	2 342	2 257	2 248
14a	(2 161, 2 183)	2 183	2 167	2 174	2 167	2 167
15a	(2 161, 2 171)	2 171	2 167	2 173	2 165	2 165
16a	(2 148, 2 301)	2 301	2 255	2 324	2 256	2 255
17a	(2 088, 2 168)	2 169	2 141	2 162	2 140	2 142
18a	(2 055, 2 139)	2 139	2 137	2 157	2 127	2 132

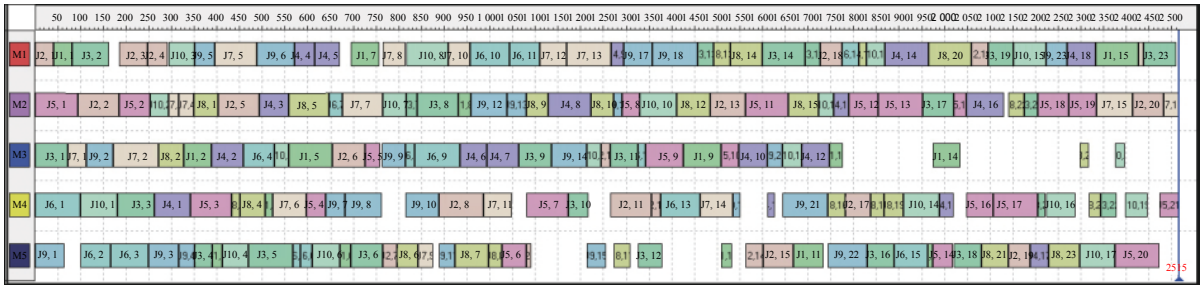

Fig. 6 Gantt chart of problem 01a (Makespan=2 515)

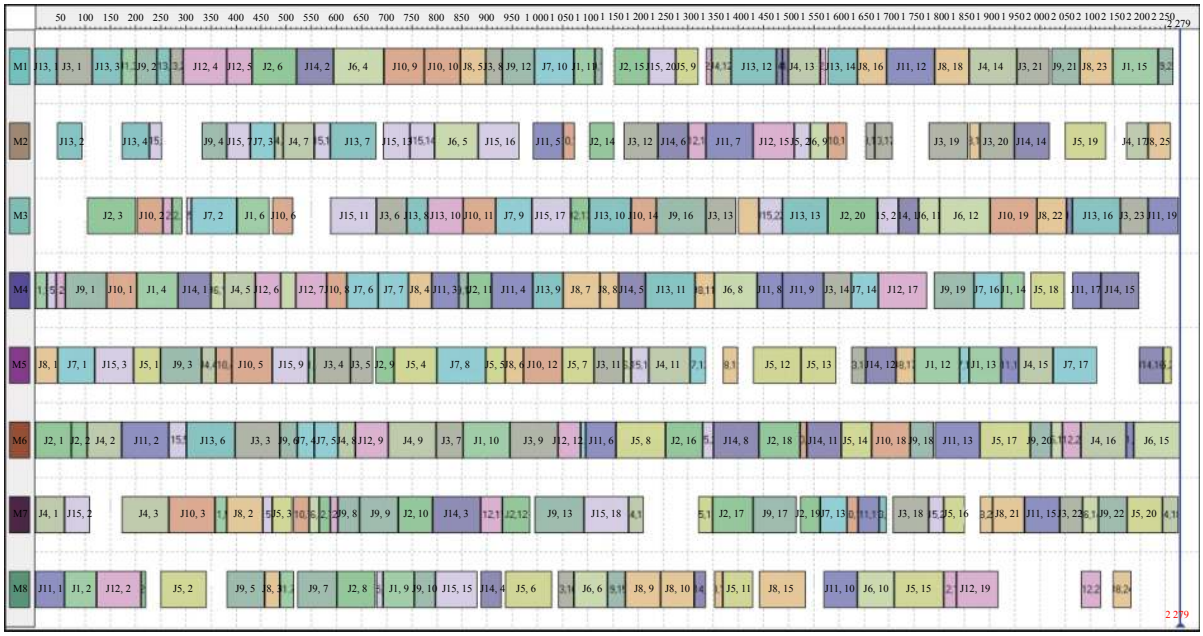

Fig. 7 Gantt chart of problem 07a (Makespan=2 279)

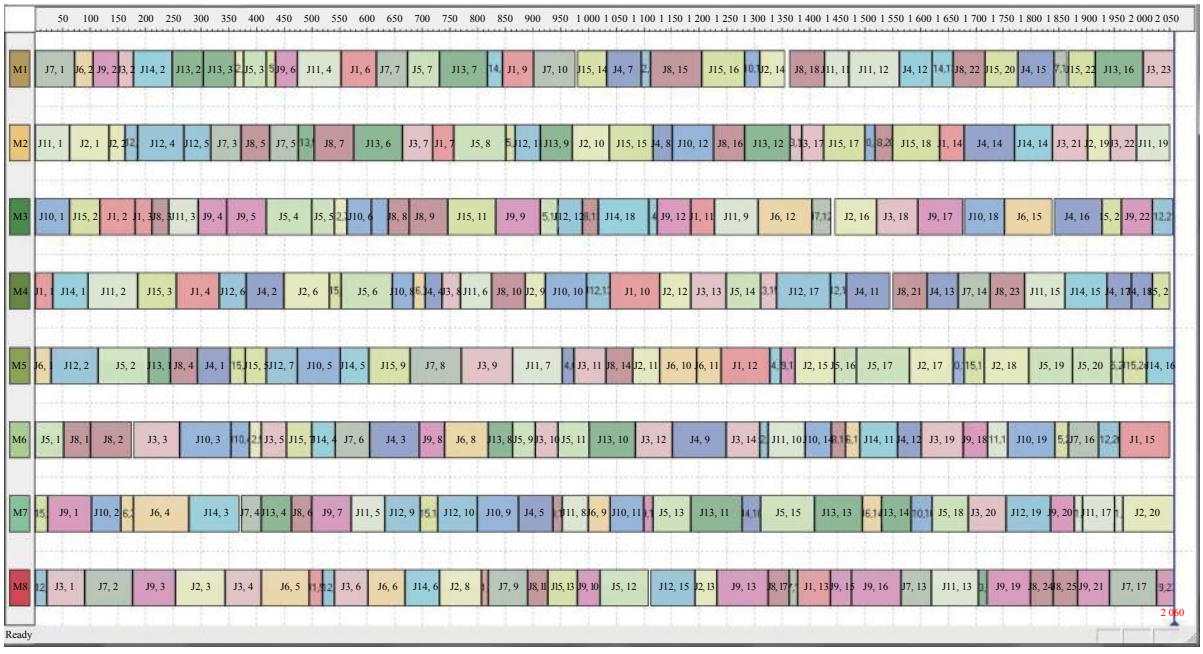

Fig. 8 Gantt chart of problem 11a (Makespan=2 060)

5.3 Results and comparison with Fattahi data

The Fattahi data with 20 problems is adopted from Fattahi et al. [43]. Table 3 shows the experimental results and comparison with the other algorithms. The MSCGA represents the proposed algorithm. The results of the artificial immune algorithm (AIA) and the hybrid harmony search (HHS) are adopted from Yuan et al. [46]. The results of the second mathematical evaluation model (M2), mixed integer linear programming (MILP), modified iterated greedy (MIG) and multi-population genetic algorithm-ER (MPGA-ER) are adopted from Demir et al. [47], Birgin et al. [48], Aqel et al. [14] and Shi et al. [31] respectively. The proposed MSCGA obtains the 18 best results. The experimental results show that the number of the best results obtained by MSCGA is greater than the other algorithms except HHS, which also obtains the same number of results as the proposed algorithm. Fig. 9 and Fig. 10 illustrate the Gantt charts of problems MFJS09 and MFJS10 respectively.

Table 3 Experimental results of Fattahi data and comparison with other methods

Problem	AIA	HHS	M2	MILP	MIG	MPGA-ER	MSCGA
SFJS01	66	66	66	66	66	66	66
SFJS02	107	107	107	107	107	107	107
SFJS03	221	221	221	221	221	221	221
SFJS04	355	355	355	355	355	355	355
SFJS05	119	119	119	119	119	119	119
SFJS06	320	320	320	320	320	320	320
SFJS07	397	397	397	397	397	397	397
SFJS08	253	253	253	253	253	253	253
SFJS09	210	210	210	210	210	210	210
SFJS10	516	516	516	516	516	516	516
MFJS01	468	468	468	468	462	468	468
MFJS02	448	446	446	446	446	446	446
MFJS03	468	466	466	466	450	466	466
MFJS04	554	554	564	554	554	554	554
MFJS05	527	514	514	514	514	514	514
MFJS06	635	634	634	634	634	634	634
MFJS07	879	879	928	879	881	879	879
MFJS08	884	884	/	/	889	884	884
MFJS09	1 088	1 055	/	/	1 059	/	1 055
MFJS10	1 267	1 196	/	/	1 214	/	1 196

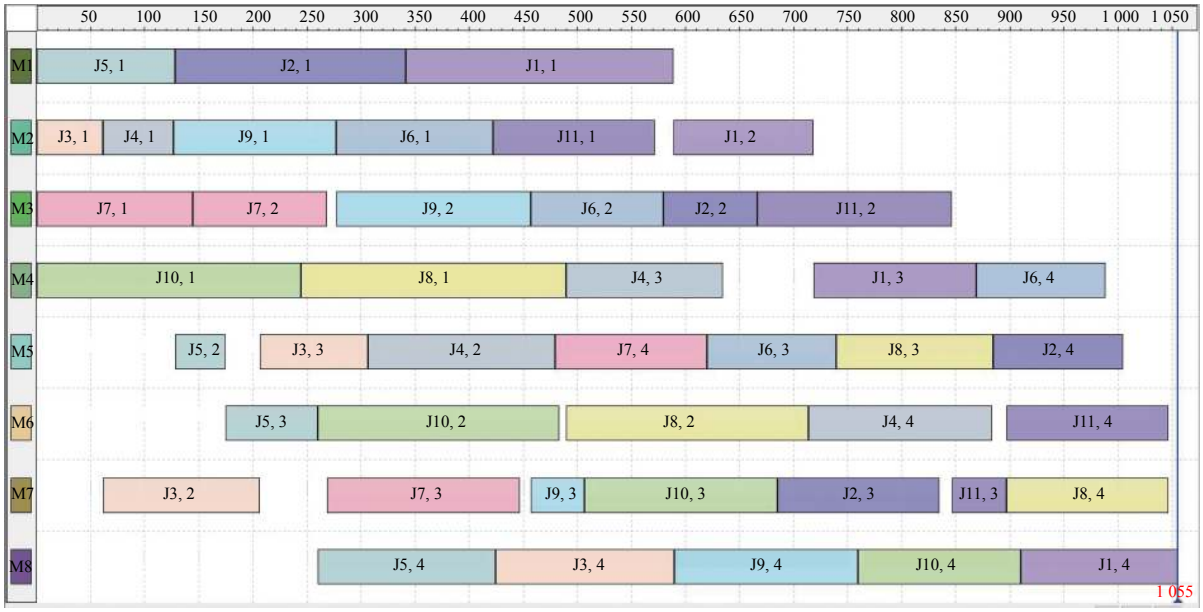

Fig. 9 Gantt chart of problem MFJS09 (Makespan=1 055)

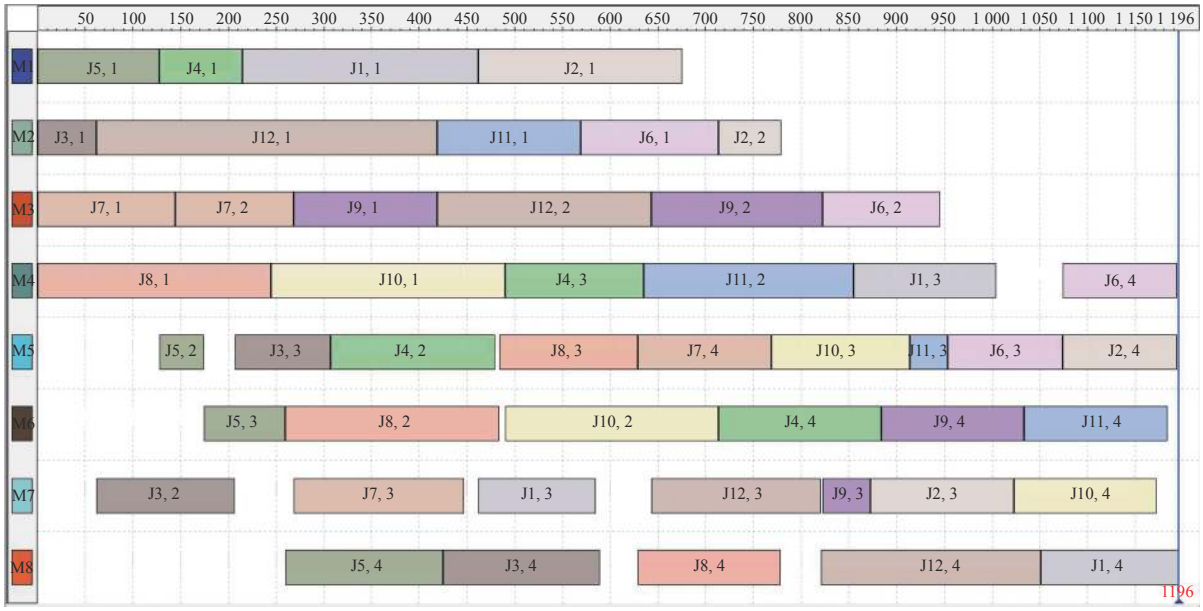

Fig. 10 Gantt chart of problem MFJS10 (Makespan=1 196)

5.4 Analysis of experimental results

Table 4 gives the overall experimental results. The statistical data demonstrates that the accuracy of the solution of the proposed algorithm is obviously better than that of other algorithms. The superiority of the number of optimal solutions shows that the algorithm has a strong ability of searching for optimal solutions. At the same time, the average error reflects the stability of the algorithm. To sum up, the proposed MSCGA has certain advantages in the FJSP.

Table 4 Overall experimental results

Benchmark		MK data	Peres data	Fattahi data
Number of optimal solutions	MSCGA	**9**	**13**	**18**
	Optimal comparison algorithm	**9**	12	16
Average error	MSCGA	**0.147 3**	**0.019 1**	—
	Optimal comparison algorithm	0.151 0	0.019 3	—

6. Conclusions and future studies

The FJSP is an important problem in modern manufacturing systems. This paper develops an MSCGA to solve it. Moreover, some famous FJSP benchmarks are used to evaluate its effectiveness. The contributions of this research include the following:

Based on the features of the FJSP, we design all the parts of the MSCGA method. The MSCGA has been successfully used to solve the FJSP. This outcome indicates that the MSCGA is an effective method to use in the FJSP research.

The MSCGA combines the advantages of evolutionary algorithms and collaborative optimization. It provides a new way to solve problems that contain several sub-problems.

Future work will expand on the implementation and scope of the algorithm. First, to obtain more new solutions, additional effective algorithms can be combined with collaborative optimization to solve the FJSP. Second, we can extend this new way to solve the multi-objective FJSP or other scheduling problems in the manufacturing field.

References

[1] AKYOL D E, BAYHAN G M. A review on evolution of production scheduling with neural networks. Computers & Industrial Engineering, 2007, 53(1): 95–122.

[2] YUE F, SONG S J, JIA P, et al. Robust single machine scheduling problem with uncertain job due dates for industrial mass production. Journal of Systems Engineering and Electronics, 2020, 31(2): 350–358.

[3] WANG P, REINELT G, TAN Y J. Self-adaptive large neighborhood search algorithm for parallel machine scheduling problems. Journal of Systems Engineering and Electronics, 2012, 23(2): 208–215.

[4] XU Y, WANG L. Differential evolution algorithm for hybrid

flow shop scheduling problems. Journal of Systems Engineering and Electronics, 2011, 22(5): 794–798.
[5] JIANG E D, WANG L, PENG Z P. Solving energy-efficient distributed job shop scheduling via multi-objective evolutionary algorithm with decomposition. Swarm and Evolutionary Computation, 2020, 58: 100745.
[6] AGARWAL A K, KUMAR D R R. Job shop and flexible job shop scheduling problems: scheduling operations. International Journal of Innovations in Engineering Research and Technology, 2020, 7(5): 334–350.
[7] XIE J, GAO L, PENG K K, et al. Review on flexible job shop scheduling. IET Collaborative Intelligent Manufacturing, 2019, 1(3): 67–77.
[8] PENG J G, LIU M Z, ZHANG X, et al. Hybrid heuristic algorithm for multi-objective scheduling problem. Journal of Systems Engineering and Electronics, 2019, 30(2): 327–342.
[9] JIANG E D, WANG L. Multi-objective optimization based on decomposition for flexible job shop scheduling under time-of-use electricity prices. Knowledge-Based Systems, 2020, 204: 106177.
[10] GAO K Z, CAO Z G, ZHANG L, et al. A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems. IEEE/CAA Journal of Automatica Sinica, 2019, 6(4): 904–916.
[11] BRUCKER P, SCHILE R. Job-shop scheduling with multi-purpose machines. Computing, 1990, 45(4): 369–375.
[12] LI J Q, DENG J W, LI C Y, et al. An improved Jaya algorithm for solving the flexible job shop scheduling problem with transportation and setup times. Knowledge-Based Systems, 2020, 200: 106032.
[13] HOMAYOUNI S M, FONTES D B M M, GONCALVES J F. A multistart biased random key genetic algorithm for the flexible job shop scheduling problem with transportation. International Transactions in Operational Research, 2020. DOI: 10.1111/itor.12878.
[14] AQEL G A, LI X Y, GAO L. A modified iterated greedy algorithm for the flexible job shop scheduling problem. Chinese Journal of Mechanical Engineering, 2019, 32(2): 157–167.
[15] MASTROLILLI M, GAMBARDELLA L. Effective neighbourhood functions for the flexible job shop problem. Journal of Scheduling, 2000, 3(1): 3–20.
[16] AMIRI M, ZANDIEH M, YAZDANI M, et al. A variable neighbourhood search algorithm for the flexible job shop scheduling problem. International Journal of Production Research, 2010, 48(19): 5671–5689.
[17] DING H J, GU X S. Improved particle swarm optimization algorithm based novel encoding and decoding schemes for flexible job shop scheduling problem. Computers and Operations Research, 2020, 121: 104951.
[18] LI X Y, GAO L. An effective hybrid genetic algorithm and tabu search for flexible job shop scheduling problem. International Journal of Production Economics, 2016, 174: 93–110.
[19] RICARDO P R, ARTURO H A. A hybrid estimation of distribution algorithm for flexible job-shop scheduling problems with process plan flexibility. Applied Intelligence, 2018, 48: 3707–3734.
[20] MATI Y, LAHLOU C, PERES S D. Modelling and solving a practical flexible job shop scheduling problem with blocking constraints. International Journal of Production Research, 2010, 49(8): 2169–2182.
[21] BURDETT R L, CORRY P, EUSTACE C, et al. A flexible job shop scheduling approach with operators for coal export terminals —a mature approach. Computers and Operations Research, 2020, 115: 104834.
[22] ZHANG C J, ZHOU Y, PENG K K, et al. Dynamic flexible job shop scheduling method based on improved gene expression programming. Measurement & Control, 2020. DOI: 10.1177/0020294020946352.
[23] WANG J J, WANG L. A knowledge-based cooperative algorithm for energy-efficient scheduling of distributed flow-shop. IEEE Trans. on Systems, Man, and Cybernetics: Systems, 2020, 50(5): 1805–1819.
[24] ZHAO F Q, HE X, WANG L. A two-stage cooperative evolutionary algorithm with problem-specific knowledge for energy-efficient scheduling of no-wait flow-shop problem. IEEE Trans. on Cybernetics, 2020. DOI: 10.1109/TCYB.2020.3025662.
[25] PAN Z X, LEI D M, WANG L. A knowledge-based two-population optimization algorithm for distributed energy-efficient parallel machines scheduling. IEEE Trans. on Cybernetics, 2020. DOI: 10.1109/TCYB.2020.3026571.
[26] LI X Y, GAO L, ZHANG L P, et al. An effective multi-swarm collaborative evolutionary algorithm for flexible job shop scheduling problem. Proc. of the 15th International Conference on Computer Supported Cooperative Work in Design, 2011: 281–184.
[27] GOMES M C, POVOA A B P, NOVAIS A Q. Optimal scheduling for flexible job shop operation. International Journal of Production Research, 2005, 43(11): 2323–2353.
[28] KACEM I, HAMMADI S, BORNE P. Approach by localization and multiobjective evolutionary optimization for flexible job shop scheduling problems. IEEE Trans. on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 2002, 32(1): 1–13.
[29] HO N B, TAY J C, LAI E M K. An effective architecture for learning and evolving flexible job shop schedules. European Journal of Operational Research, 2007, 179(2): 316–333.
[30] YIN L J, LI X Y, GAO L, et al. A novel mathematical model and multi-objective method for the low-carbon flexible job shop scheduling problem, considering productivity, energy efficiency and noise reduction. Sustainable Computing: Informatics and Systems, 2017, 13: 15–30.
[31] SHI X Q, LONG W, LI Y N, et al. Multi-population genetic algorithm with ER network for solving flexible job shop scheduling problems. PLoS One, 2020, 15(5): e0233759.
[32] CHEN R H, YANG B, LI S, et al. A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem. Computers & Industrial Engineering, 2020, 149: 106778.
[33] LU C, LI X Y, GAO L, et al. An effective multi-objective discrete virus optimization algorithm for flexible job-shop scheduling problem with controllable processing times. Computers & Industrial Engineering, 2017, 104: 156–174.
[34] ALEJANDRO V S, AHMED A, MOHAMMED F B. Mathe-

matical modeling and a hybridized bacterial foraging optimization algorithm for the flexible job-shop scheduling problem with sequencing flexibility. Journal of Manufacturing Systems, 2020, 54: 74–93.
[35] ZIRBI N, KACEM I, KAMEL A E, et al. Assignment and scheduling in flexible job shops by hierarchical optimization. IEEE Trans. on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 2007, 37(4): 652–661.
[36] GAO J, SUN L Y, GEN M. A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems. Computers and Operations Research, 2008, 35: 2892–2907.
[37] WU X L, SHEN X L, LI C B. The flexible job-shop scheduling problem considering deterioration effect and energy consumption simultaneously. Computers & Industrial Engineering, 2019, 135: 1004–1024.
[38] ZHANG G H, ZHANG L J, SONG X H, et al. A variable neighborhood search based genetic algorithm for flexible job shop scheduling problem. Cluster Computing, 2019, 22: 11561–11572.
[39] ROU L Y, ASMUNI H. A study of cooperative co-evolutionary genetic algorithm for solving flexible job shop scheduling problem. International Journal of Computer and Information Engineering, 2010, 4(12): 1849–1854.
[40] XING L N, CHEN Y W, YANG K W. Multi-population interactive coevolutionary algorithm for flexible job shop scheduling problems. Computational Optimization and Applications, 2011, 48: 139–155.
[41] BRANDIMARTE P. Routing and scheduling in a flexible job shop by taboo search. Annals of Operations Research, 1993, 41(3): 157–183.
[42] PERES S, PAULLI J. An integrated approach for modeling and solving the general multiprocessor job shop scheduling using tabu search. Annals of Operations Research, 1997, 70: 281–306.
[43] FATTAHI P, MEHARABAD M S, JOLAI F. Mathematical modeling and heuristic approaches to flexible job shop scheduling problems. Journal of Intelligent Manufacturing, 2007, 18: 331–342.
[44] GAO L, PENG C Y, ZHOU C, et al. Solving flexible job shop scheduling problem using general particle swarm optimization. Proc. of the 36th Conference on Computers & Industrial Engineering, 2006: 3018–3027.
[45] HMIDA A B, HAOUAI M, HUGUET M J, et al. Discrepancy search for the flexible job shop scheduling problem. Computers and Operations Research, 2010, 37: 2192–2201.
[46] YUAN Y, XU H, YANG J D. A hybrid harmony search algorithm for the flexible job shop scheduling problem. Applied Soft Computing, 2013, 13(7): 3259–3272.
[47] DEMIR Y, ISLEYEN S K. Evaluation of mathematical models for flexible job shop scheduling problems. Applied Mathematical Modelling, 2013, 37(3): 977–988.
[48] BIRGIN E G, FEOFILOFF P, FERNANDES C G, et al. A MILP model for an extended version of the flexible job shop problem. Optimization Letters, 2014, 8: 1417–1431.

Biographies

WANG Cuiyu was born in 1983. She received her M.S. degree in industrial engineering with the Department of Industrial & Manufacturing System Engineering, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan, China, in 2007. She is currently pursuing her Ph.D. degree in industrial engineering with Huazhong University of Science and Technology. Her research interest is scheduling algorithm.
E-mail: 76325434@qq.com

LI Yang was born in 1996. He received his B.S. degree in industrial engineering from the School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan, China, in 2018. He is currently pursuing his Ph.D. degree in industrial engineering, with Huazhong University of Science and Technology. His research interests include scheduling algorithm and optimization algorithm.
E-mail: 281366572@qq.com

LI Xinyu was born in 1985. He is currently a professor with the Department of Industrial & Manufacturing System Engineering, School of Mechanical Science and Engineering, Huazhong University of Science and Technology, Wuhan, China. His research interests include intelligent optimization and scheduling, and intelligent manufacturing system.
E-mail: lixinyu@mail.hust.edu.cn