Journal of Computers Vol. 31 No. 5, 2020, pp. 224-235
doi:10.3966/199115992020103105017

A Hybrid Artificial Bee Colony Algorithm for Multi-objective Flexible Job-Shop Scheduling Problem

Guan-Jun Meng, Xin-Hua Chen[*], Da-Chun Yang, Wei Zhang

School of Mechanical Engineering, Hefei University of Technology, Hefei 230009, Anhui, China
gj1977@163.com, cxhsj0320@163.com, 2575388449@qq.com, 876717948@qq.com

Received 22 May 2019; Revised 9 September 2019; Accepted 21 November 2019

Abstract. Flexible job shop scheduling (FJSP) is an extension of job shop scheduling problem. The traditional optimization algorithm can get better results when solving the single objective FJSP, but it is often inefficient and difficult to obtain the optimal solution in the face of multi-objective FJSP. Using artificial bee colony (ABC) algorithm with fast convergence and good global search capability and tabu search (TS) algorithm with strong local search ability, this paper designs a hybrid improved artificial bee colony (TS-ABC) algorithm for solving multi-objective FJSP. In order to evaluate the solution better, a fitness function based on Pareto rank and crowding distance is proposed. In the updating of the solution, this paper adopts multiple search mechanisms, which uses improved precedence operation crossover (IPOX) and Multi-point Preservative Crossover (MPX) in the employed bee phase and performs the exchange operation and mutation operation in the onlooker bee phase. The method can improve the performance of the search well. The combination of tabu search and improved artificial bee colony algorithm effectively improves the probability of obtaining the optimal solution. In the end, a series of experiments are carried out to solve the problem of the multi-objective FJSP by TS-ABC algorithm.

Keywords: artificial bee colony algorithm, flexible job-shop scheduling problem, multi-objective Optimization, tabu search

1 Introduction

The job shop scheduling problem (JSP) is actually a problem of processing resource allocation. It is necessary to optimally allocate tasks under the premise of order constraints. Reasonably arrange production resources and processing time to obtain optimal cost or efficiency [1]. Flexible job shop scheduling problem (FJSP) is an extension of JSP. In JSP, each operation only needs to select a production equipment. In FJSP, an operation has more than one production equipment that can be selected. We need to choose the most appropriate one. In actual selection, both the machine assignment vector and the operations assignment vector need to be considered. FJSP increases the flexibility of scheduling and is more in line with the actual situation of production. It is a kind of scheduling problem that enterprises urgently need to solve. Therefore, this paper will discuss the problem of solving multi-objective flexible job shop scheduling.

Since the FJSP reduces machine constraints and expands the search range of feasible solutions, it is a more complicated Non-deterministic Polynomial (NP) problem [2]. In recent years, due to the development of related disciplines and optimization techniques, many new optimization algorithms have been proposed in the field of FJSP. Currently, common methods for solving FJSP include particle swarm optimization (PSO) [3], genetic algorithm (GA) [4-5], simulated annealing (SA) [6] and tabu search (TS) algorithm [7-9]. From the existing research, the intelligent optimization algorithm can effectively solve the multi-objective FJSP. The task objectives are mostly minimum completion time, maximum machine workload, and minimum workload. However, some optimization algorithms also have the disadvantages

[*] Corresponding Author

of slow convergence and easy to fall into local optimum. In order to improve the performance of the algorithm, researchers have added different improvement strategies to the basic algorithms. such as variable neighborhood search strategy, niche strategy, and hybrid algorithm strategy [10-11]. This paper proposes a multi-objective FJSP based on hybrid artificial bee colony algorithm. The performance criterias are the minimum completion time, the maximum machine workload and the total workload.

Artificial bee colony algorithm (ABC) is an optimization method based on bee self-organizing model and group intelligence. The algorithm is widely used in many optimization problems because of its simple mechanism, strong robustness and few parameters. In terms of scheduling problems, some scholars have begun to try the application of ABC algorithm. For example, Jia and Guo [12] analyzed the resource-constrained project scheduling problem with the separated ABC and PSO hybrid algorithms. In recent years, ABC algorithm has been successfully applied to FJSP. Li, Pan and Gao [13] proposed a TS-ABC algorithm based on Pareto to solve multi-objective FJSP. Thammano and Phu-Ang [14] designed a local search TS-ABC algorithm for solving FJSP. Although the ABC algorithm is effective in solving the scheduling problem, it still has the defects of premature convergence and easy to fall into the local optimum. Therefore, it is necessary to continue to optimize the algorithm.

In the previous research, when the ABC algorithm was used to solve FJSP, the optimal solution in the algorithm would replace the bad solution, and the bad solution would not be saved. This made the algorithm easy to fall into local optimum and difficult to jump out. Combining with the advantages of the ABC algorithm, this paper proposes a new TS-ABC algorithm for FJSP. The TS is applied to the ABC algorithm. TS algorithm is an intelligent heuristic algorithm. The algorithm has a flexible memory function and the search process can accept the bad solution, which can jump out of local optimal solution. Using TS optimizes the current 30~50 percent best solutions which generate by ABC algorithm. The strategy would make individuals break their own which has reached the optimal solution and avoid the premature convergence. TS algorithm has strong local search ability. Combining with TS, the ABC algorithm will be fully utilized and the solution is more competitive. This paper proves that the TS-ABC algorithm's solution speed and solution quality are improved by experiments.

2 Mathematical Model of FJSP

2.1 Description of the Problem

For the convenience of discussion, the following mathematical symbols are introduced, as shown in Table 1.

Table 1. Mathematics symbol

Mathematics Symbol	Significance
m	Total number of machines
n	Total number of workpieces
i, h	The serial number of the workpiece
j, l	The serial number of the procedure
k	The serial number of the Machine
o_{ij}	The j-th procedure of the i-th workpiece
p_{ijk}	The operating time of the o_{ij} uses the j-th machine
x_{ijk}	If the o_{ij} selects the k-th machine, the x_{ijk} result is equal to 1, otherwise it is equal to 0
y_{ijkhl}	If the o_{ij} is processed before o_{hl} , the y_{ijkhl} result is equal to 1, otherwise it is equal to 0
s_{ij}	The j-th procedure start time of the i-th workpiece
c_{ij}	The j-th procedure completion time of the i-th workpiece
$c_{\max}$	Maximum completion time
w_{aw}	Total workload
w_{wl}	Maximum machine workload

In an industrial environment, FJSP can be described as follows: there are n jobs $\boldsymbol{J}=\{J_1,J_2,...J_n\}$ and m machines $\boldsymbol{M}=\{M_{1,}M_{2,...}M_m\}$ that produce these jobs. Each job is composed of several operations (n_i). each operation can choose several candidate machines, whereas production time of different machine is different. Therefore, the purpose of scheduling is to determine the optimal processing sequence and the reasonable selection of candidate machines under the conditions of machine capacity constraints and job constraints. FJSP needs to meet the following conditions:

(1) all machines are idle at first and can be used normally,

(2) an equipment can produce only one workpiece at the same time,

(3) if the job starts to produce, it cannot be interrupted,

(4) there are no precedence constraints between different jobs,

(5) Processing tasks on the same machine can't start until another task is completed.

2.2 Mathematical Description of FJSP

The goal of FJSP optimization is to assign the appropriate machine to the process, arrange the processing sequence of the processes on each machine and calculate the start and end times, ultimately optimizing one or some of the system's performance. The constraints imposed on FJSP can be mathematically described as follows:

$$s_{ij}+x_{ijk}\times p_{ijk}\le c_{ij}\,, \tag{1}$$

$$c_{ij\le}c_{i(j+1)}\,, \tag{2}$$

$$c_{ij}\le c_{\max}\,, \tag{3}$$

$$s_{ij}+p_{ijk}\le s_{hl}+L(1-y_{ijkhl}), \tag{4}$$

$$c_{ij}\le s_{i(j+1)}+L(1-y_{ijkl(j+1)}), \tag{5}$$

$$\sum_{k=1}^{ijm}x_{ijk}=1, \tag{6}$$

$$s_{ij}\ge 0, c_{ij}\ge 0. \tag{7}$$

Where L is a positive number that is large enough, $i=1,2,3,\cdots,n$; $h=1,2,3,\cdots,n$; $j=1,2,3,\cdots,m$.

Equation (1) (2) represents the order constraint of each workpiece's procedure; Equation (3) represents the workpiece's completion time constraint. The completion time of each workpiece cannot exceed the total completion time; Equation (4) (5) means that only one procedure can be processed in the same machine at the same time; Equation (6) represents machine constraints. The same procedure can only be processed by one machine at the same time; Equation (7) represents that each parameter variable must be a positive number.

In this paper, we need to optimize the objective function:

Maximum completion time function:

$$C_{\max}=\max(C_i\,|\,i=1,2,...n), \tag{8}$$

Total workload function:

$$W_{aw}=\sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{ni}(x_{ijk}p_{ijk}), \tag{9}$$

Maximum machine workload function:

$$W_{wl} = \max(\sum_{i=1}^{n}\sum_{j=1}^{ni} x_{ijk} p_{ijk}). \tag{10}$$

3 Artificial Bee Colony Algorithm

Artificial bee colony algorithm is a population-based meta-heuristic method proposed by Karaboga. Artificial bee colony algorithm is inspired by the foraging behavior of bees, mainly composed of three kinds of bees, namely, the employed bees, onlooker bees and scout bees. The task of employed bees is to leave the hive and find the food source. And then the onlooker bees will evaluate the food sources which searched by employed bees, choose to follow an employed bee according to the quality of food source. If the food source doesn't change under the specified number of times, the bee would be transformed into the scout bee to randomly search for new food source. The location of the food source represents a feasible solution to the optimization problem. The number of nectar of food represent the fitness value of the corresponding food source. In the ABC algorithm, the food source is composed of a n-dimensional vector, evaluated by fitness function. Standard ABC algorithm steps are as follows:

3.1 Initialization

In initialization phase, there are some parameters need to define: the number of iterations of algorithm, the food sources (SN), the number of employed bees, onlooker bees and scout bees, the number of food sources doesn't change and need to abandon (*limits*). Let $\boldsymbol{X}_i = \{X_{i1,}X_{i2,}X_{i3\ldots}X_{in}\}$ represent the *i*th food source and generate randomly by formula 11.

$$x_{ij} = LB_j + (UB_j - LB_j) \times r,\ j{=}1,2,\cdots n{,}i{=}1,2,\cdots SN. \tag{11}$$

r is a random number, the range is [0, 1]. UB_j and LB_j are the upper and lower bounds of search space respectively. The food sources are evaluated by fitness function and assigned to the employed bees.

3.2 Employed Bee Phase

After obtaining the randomly generated food source, the employed bee produces a neighborhood solution $\boldsymbol{X}_{new}$ near the food source, which is generated by formula 12:

$$x_{new,j} = x_{ij} + (x_{ij} - x_{kj}) \times r'. \tag{12}$$

r' is a random number, the range is [0,1]. $k \in \{1,2\ldots SN\} \cap k \neq i.j \in \{1,2,\ldots,n\}$ is a random number. $\boldsymbol{X}_{new}$ will be compared with $\boldsymbol{X}_i$, if the fitness value of $\boldsymbol{X}_{new}$ is better than or equal to the fitness value of $\boldsymbol{X}_i$, $\boldsymbol{X}_{new}$ will replace the $\boldsymbol{X}_i$ as a new food source and the number of times *limits* will become 0, or remain unchanged and the number of *limits* will be increased by 1.

3.3 Onlooker Bee Phase

After the employed bees get the food back to the hive, the onlooker bees will evaluate the food sources by formula 13, and then select an employed bee to follow.

$$p_i = \frac{f_i}{\sum_{i=1}^{SN} f_i}. \tag{13}$$

Among them, f_i is the food source, the corresponding fitness value of $\boldsymbol{X}_i$. p_i is the probability value corresponding to $\boldsymbol{X}_i$, the greater the probability, the greater the chance of being followed by the onlooker bee. Once selected, its food source $\boldsymbol{X}_i$ will be updated according to formula 12. The new food source would compare with the X_i through fitness function. If the fitness value of new food source is better than or equal to X_i, the food source will replace the position of X_i and the limits number will be set to 0, otherwise unchanged and the number of *limits* plus 1.

3.4 Scout Bees Phase

If the food source X_i has remained unchanged for the specified number of times (*limits*), the bee will abandon the food source and transform into a scout bee to initialize the X_i again.

$$x_{ij} = LB_j + (UB_j - LB_j) \times r,\ j=1,2,\cdots n. \tag{14}$$

r is a random number in the range of [0,1]. UB_j and LB_j are upper and lower limits respectively.

4 TS-ABC for FJSP

4.1 Encoding Problem

In the TS-ABC algorithm, each solution consists of two vectors: the operation assignment vector and the machine assignment vector. In the operation assignment vector, there are n jobs and each job contain several operations. The length of machine assignment vector is same as the operation assignment vector. Each process can be produced by some candidate machines, through a certain way to select a machine for production. Each dimension in the operation assignment vector represents an operation of job, and each dimension in the machine assignment vector represents the selected machine of the corresponding operation.

We use an example to further illustrate the encoding problem. As shown in Table 2, M_1, M_2, M_3 represent three different machines respectively. The time which machines produce different components is generally different. J_i means job number, such as the job J_1 has three operations. For any operation, we can determine the required number of machines by comparing the production time of different machines. By analyzing the data in Table 1, we can get the possible sequence of operations: (O_{21}, M_2), (O_{11}, M_1), (O_{12}, M_1), (O_{22}, M_3), (O_{31}, M_1), (O_{13}, M_3), (O_{32}, M_3), (O_{23}, M_2). The left of parenthesis is operation and the right represent corresponding machine.

Table 2. The FJSP instance with 3 jobs and 3 machines

		M_1	M_2	M_3
J_1	O_{11}	4	5	-
	O_{12}	3	2	1
	O_{13}	7	9	2
J_2	O_{21}	8	3	5
	O_{22}	7	6	4
	O_{23}	7	3	2
J_3	O_{31}	1	2	3
	O_{32}	3	2	1

4.2 Population Initialization

For the meta-heuristic algorithm, the operation sequence is easy to affect the convergence speed and the result of the algorithm. Therefore, it is necessary to pay attention to the initialization of the algorithm. The operation assignment vector is mainly generated by random arrangement. The methods about machine assignment vector main include random rules, operation minimum time rules and so on. The operation minimum time rule means that in several alternative machines of the same operation, the machine is likely to be selected if the operation time of a machine is minimum. Random rule means that generate the desired machine through the random way in the optional machines. In the case of more optional machines, for example, there will be 8 or even 10 optional machines per process, the paper will use another method: sort the machine assignment vector by time, select the three machines with the shortest operation time and then select a machine using the random method. The sequence is greatly optimized, and the results are greatly reduced.

4.3 Tabu Search

Tabu search is a meta-heuristic algorithm that can be used to solve combinatorial optimization problems. The algorithm has the function of short-term memory and good climbing ability, so as to avoid a lot of invalid computation. Its aspiration criteria can accept differential solutions, thereby expanding local search capabilities of solution. The basic principle of tabu search is to give an initial solution and generate the corresponding neighborhood solution based on the initial solution. Choose an optimal solution from the neighborhood solution as the current solution and the cycle doesn't stop until the convergence condition is satisfied. A tabu list is necessary in order to avoid repetitive operation. If the current solution is in the tabu list, then give up the solution and choose a new one from the neighborhood. If the solution is particularly useful, you can ignore the tabu property, which is the aspiration criteria.

4.4 Pareto Sorting

In order to evaluate the food source searched by bees, after determining the 3 object function values, the non-dominated Pareto sorting and crowding distance were calculated to determine the quality of the food source. First, find the optimal solutions in the population and define them as rank 1. Then remove these non-dominated solutions, find new non-dominated solutions from remaining solutions and define these solutions as rank 2. The process is repeated until all solutions in the population are ranked accordingly.

The next step is to compute the crowding distance of all solutions in the same level. First, define the crowding distance of the first and last individual to infinity. Then all the middle solutions are solved by a formula 15.

$$L_p = \sum_{j=1}^{n} \frac{f_{p+1,j} - f_{p-1,j}}{f_{n,j} - f_{1,j}}. \tag{15}$$

The j represents an object for FJSP. $p \in \{1,2,\ldots,SN\}$ represents pth food source. $f_{p,j}$ indicates the objective function value of pth food source about object j. L_p indicates the crowding distance of pth food source.

Each solution in the population has two attributes, the non-dominated sorting rank and the crowding distance. If the two solutions are in different rank, a higher rank will be preferred. If the two solutions are of the same rank, we can judge the crowding distance of the two solutions, and the solution who is not crowded would be selected.

4.5 Employed Bee Phase

The employed bee $\boldsymbol{X}_i$ at this stage requires the participation of another employed bee $\boldsymbol{X}_{best}$ which is selected randomly from food sources of rank 1. According to the characteristics of FJSP, each solution can be divided into two parts, the operation assignment vector, using IPOX operation, and the machine assignment vector, using MPX operation and mutation operation.

IPOX operation. Randomly generate *n/2* jobs as subset *s1*. Copy the jobs which are contained in *s1* in $\boldsymbol{X}_i$ into $\boldsymbol{X}_{new}$ and keep the original location. And then copy the jobs that are not included in s1 in $\boldsymbol{X}_{best}$ to $\boldsymbol{X}_{new}$. The machines corresponding to each of the operations in the $\boldsymbol{X}_i$ will be copied to $\boldsymbol{X}_{new}$ completely.

MPX operation. Randomly generate a sequence t containing only 0 and 1 with the same length as the machine assignment vector. Select the operations which t corresponding to the number 1 in $\boldsymbol{X}_{best}$. Copy the machines corresponding to the selected operations to $\boldsymbol{X}_{new}$ and retain the original position. Then copy the other machines in $\boldsymbol{X}_i$ to $\boldsymbol{X}_{new}$.

mutation operation. randomly select two operations, reselection operating machine for every operation.

Finally, the new food source $\boldsymbol{X}_{new}$ would compare with old food source $\boldsymbol{X}_i$. If a new food source is better than $\boldsymbol{X}_i$, $\boldsymbol{X}_{new}$ will replace the $\boldsymbol{X}_i$ position, or remain unchanged.

4.6 Onlooker Bee Phase

At this phase, when the bee returns to the nest, each onlooker bee will select an employed bee to follow and the standard of selection is the Roulette. then perform a tabu search operation on the onlooker bees. Among them, generating neighborhood solution is an important step in tabu search, including exchange operation of operation assignment vector and mutation operation of machine assignment vector.

exchange operation. two positions are randomly generated, and the position cannot be the same. Then exchange the pair of operations, and this operation will be executed 2 times.

mutation operation. two operations are randomly generated and reselect operating machine for the two operations.

If the non-dominated sorting rank and crowding distance of the new food source is better than the old food source, old food source would be replaced.

4.7 Scout Bee Phase

The task of this phase is to produce new food sources to replace the old food sources that don’t meet the requirements. As evolution continues, the probability of food source change is gradually reduced. Good genes are gradually inherited by the best food source ($\boldsymbol{X}_{best}$) and then hardly change. In order to improve the diversity of the population, must set parameter *limits*. Based on this, if the food source whose fitness value don’t improve after a certain number *limits* and the crowded distance of food source is too little, food source will be abandoned and the bee will be transformed into a scout bee to regenerate a new food source.

4.8 Hybrid Artificial Bee Colony Algorithm Model

The combination of artificial bee colony algorithm with Pareto sorting strategy and tabu search strategy is proved to have high efficiency. Tabu search which is a kind of local search algorithm can very good to avoid the local optimal solution. The solution can be evaluated well by Pareto sorting and crowding distance calculations. Hybrid artificial bee colony algorithm steps are as follows:

step1: Initialization.
step2: Initialize the population according to encoding scheme.
step3: Calculate the pareto rank and crowded distance of population.
step4: Employed bee phase.
step5: Onlooker bee phase.
step6: Scout bee phase.
step7: Output computational results.

5 Experiment Evaluation and Comparison

The main content of this chapter is to test the effectiveness and performance of TS-ABC algorithm in solving high quality solutions, the algorithm will run on MATLAB v8.4. The algorithm will use two sets of examples, the first group is a Kacem instances, including five examples, respectively, 4×5, 8×8, 10×7, 10×10, 15×10; The second group is the BRdata instances whose author is Brandimarte. the BRdata instances select 10 examples with the number of the job ranging from 10 to 20, the number of machines in the range of 4~15, the number of operations for each job ranging from 5~15. The value of the test is the optimal solution after 10 independent runs.

Each instance performs the following parameters: the number of jobs *n*, the number of machines *m*, the total number of operations *toper*. Other relevant parameters of TS-ABC algorithm are set as follows: the number of iterations *iter_max* = 200, the number of population *SN* = 30, the number of *limits* is 10, the tabu search candidate solution number is 5, the maximum number of tabu search iterations is *n*, tabu list length is *n/*2.

5.1 Kacem Instances

The following data is the comparison of the 5 instances of Kacem, and the comparison objects include the hybrid algorithm of particle swarm optimization and simulated annealing algorithm (PSO+SA) [15], a hybrid algorithm of particle swarm optimization and tabu search (PSO+TS) [16], Pareto-based discrete artificial bee colony algorithm (P-DABC) [17], estimation of distribution algorithms (EDA) [18], discrete artificial bee colony algorithm (DABC) [19]. The comparison results are shown in the following table, each of instances is composed of three columns, from left to right, respectively, the maximum completion time, the maximum machine workload, the total workload. If an algorithm does not have data about an instance, it is represented by the symbol "-".

From the results in Table 3, the following conclusions can be obtained.

Table 3. Results of the five Kacem instances

	4×5			8×8			10×10			10×7			15×10		
PSO+SA	-	-	-	15	12	75	7	6	44	-	-	-	12	11	91
				16	13	73									
PSO+TS	11	10	32	14	12	77	7	6	43	-	-	-	11	11	93
				15	12	75									
P-DABC	11	10	32	14	12	77	7	5	43	11	11	63	12	11	91
	12	8	32	15	12	75	8	7	41	12	11	61	11	11	93
	13	7	33	16	13	73	8	5	42	12	12	60			
EDA	11	10	32	14	12	77	7	6	44	11	11	61	11	10	93
	11	9	34	15	12	75	7	5	43	11	10	62	11	11	91
DABC	11	10	32	14	12	77	7	6	42	11	11	61	11	10	93
	12	8	32	15	12	75	8	5	42	11	10	62	11	11	91
	11	9	34	16	13	73	7	5	43	12	12	60			
TS-ABC	11	10	32	14	12	77	7	6	41	11	10	61	11	11	93
	11	9	34	15	15	75	7	7	41	11	11	62	12	12	96
	12	8	32	15	12	75	7	6	42	11	11	63	12	11	91

Instance 4×5 The column labeled 4×5 in Table 2 display that the three objects including the maximum completion time can reach the best state of other algorithms. The best values obtained by the TS-ABC algorithm for instance 4×5 are as follows:

Solution: *makespan* = 12, W_{wl} = 8, W_{aw} = 32.

Solution: *makespan* = 11, W_{wl} = 9, W_{aw} = 34.

The Gantt chart for the instance obtained by TS-ABC algorithm is shown in Fig. 1. The block with the color represents a piece of work process, the space between the block means the machine is idle.

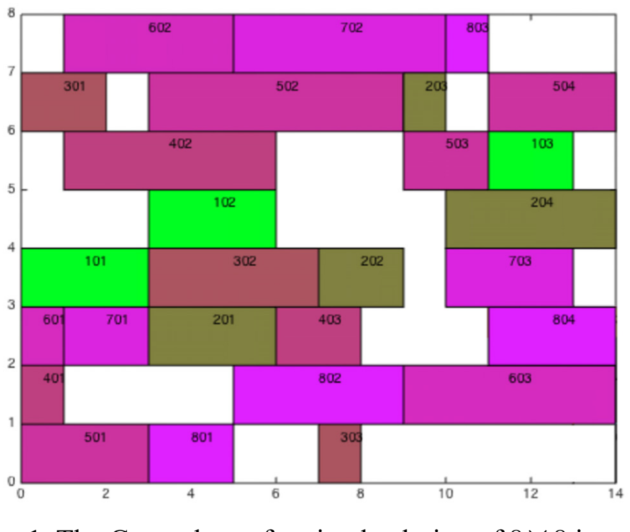

Fig. 1. The Gantt chart of optimal solution of 8×8 instance

Instance 8×8 The best solution obtained by the TS-ABC algorithm for instance 8×8 is given as follows:

Solution: *makespan* = 14, W_{wl} = 12, W_{aw} = 77.

Table 2 show that the TS-ABC algorithm on the instance 8×8 has reached the optimal value compared to other algorithms. the Gantt chart of instance 8×8 obtained by the TS-ABC algorithm is shown in Fig.1.

Instance 10×7 As can be seen from Table 2, it can be found by comparison that the TS-ABC algorithm achieves the optimal value when solving the instance 10×7 of FJSP. the total workload of P-DABC reaches the optimal value the instance 10×7. The optimal solution of this algorithm is shown in the following:

Solution: *makespan* = 11, W_{wl} = 10, W_{aw} = 61.

Instance 10×10 The best results obtained by the TS-ABC algorithm for instance 10×10 are given as follows:

Solution: *makespan* = 7, W_{wl} = 6, W_{aw} = 41.

The column labeled 10×10 in Table 2 display that the makespan and total workload of TS-ABC algorithm can achieve the optimal value. The maximum machine workload of P-DABC have optimal value.

Instance 15×10 The instance15×10 is the most complex Kacem instance. Compared with other algorithms, although TS-ABC is slightly less than EDA, has the same optimal value as DABC, PSO+TS, P-DABC and slightly better than PSO+SA. The relevant Gantt chart is shown in Fig. 2. The best solution obtained by the TS-ABC algorithm for instance 15×10 is as follows:

Solution: *makespan* = 11, W_{wl} = 11, W_{aw} = 93.

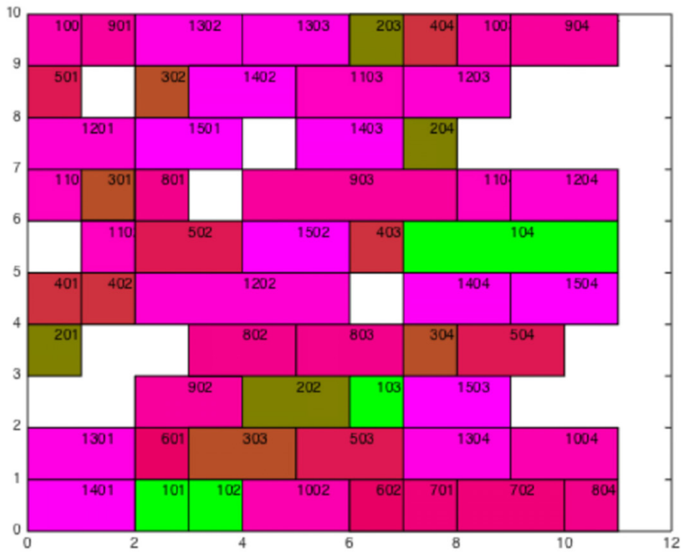

Fig. 2. The Gantt chart of optimal solution of 15×10 instance

From the simulation results, the optimal solution obtained by the TS-ABC algorithm is equal to or better than the optimal solution obtained by the other five algorithms in most cases. For FJSP, it has better ability to find and effectively improve the quality of the solution. For the above five examples, the TS-ABC algorithm can stably find the optimal value or approximate optimal value, indicating the TS-ABC algorithm has higher accuracy and robustness.

5.2 BRdata Instances

This section uses the BRdata instances to test the effectiveness of the TS-ABC algorithm. The algorithms from the existing literature have best performing and are competitive, including ant colony algorithm based on knowledge (KBACO) [20], parallel variable neighborhood search (PVNS) [21], effective artificial bee colony algorithm (EABC) [22], elitist quantum-inspired evolutionary algorithm (EQEA) [23], two step artificial bee colony algorithm (TABC) [24]. As shown in Table 4, the first column is 10 BRdata instances, the second column, m means the number of jobs, n means the number of machines, the third column is the upper and lower bounds of the BRdata instances, the last column is the TS-ABC algorithm in this paper, the other columns are some compared algorithms.

Table 4. Results of the ten BRdata instances

	n×m	BOUND	KBACO	PVNS	EABC	EQEA	TABC	TS-ABC
MK01	10×6	36/42	39	40	40	40	40	40
MK02	10×6	24/32	29	26	26	26	26	26
MK03	15×8	204/211	204	204	204	204	204	204
MK04	15×8	48/81	65	60	60	60	60	60
MK05	15×4	168/186	173	173	172	172	173	171
MK06	10×15	33/86	67	60	60	58	60	60
MK07	20×5	133/157	144	141	139	139	139	138
MK08	20×10	523	523	523	523	523	523	523
MK09	20×10	299/369	311	307	307	307	307	307
MK10	20×15	163/296	229	208	208	212	202	206

Analysis on the Table 3, the TS-ABC algorithm is the most competitive algorithm. In addition to MK1, MK10, the TS-ABC algorithm obtains good results in the other 8 instances. The relevant Gantt chart is shown in Fig. 3. Compared to KBACO, the TS-ABC algorithm obtains better results in 7 instances. TS-ABC algorithm outperforms PVNS in 3 instances. Compared with the EABC algorithm, TS-ABC algorithm is superior in 4 instances. Compared with the TABC and EQEA algorithm, TS-ABC algorithm is superior in 2 instances. The TABC algorithm also has a very good effect except for MK01, MK05, MK07 and TABC achieve the best results in MK10 compared with other algorithms. EQEA algorithm achieve the best results in MK6. The KBACO algorithm achieves the best results in MK01 (makespan = 40, W_{wl} = 39, W_{aw} = 172). PVNS and EABC algorithm get the optimal solution in 6 instances. Therefore, the ABC algorithm is effective and competitive algorithm.

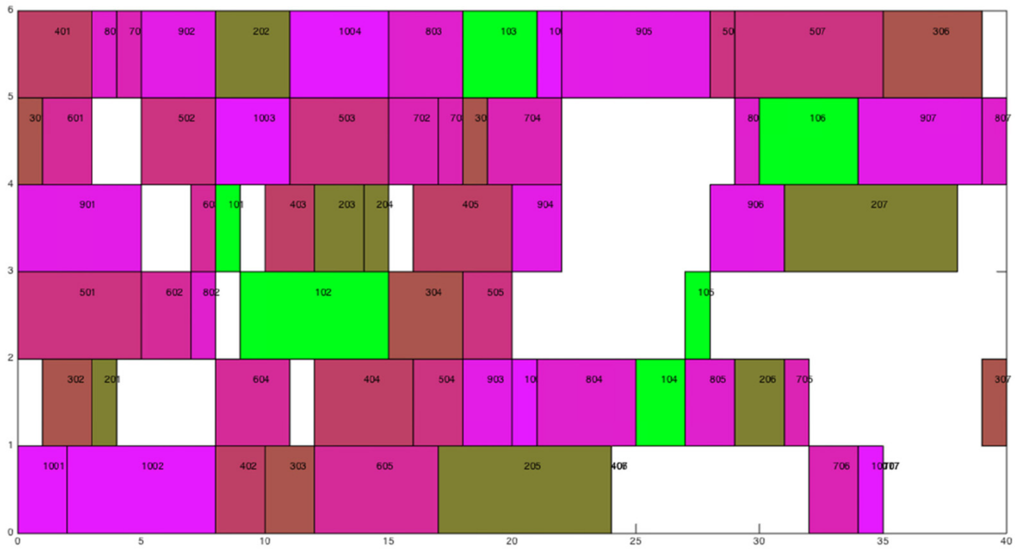

Fig. 3. The Gantt chart of optimal solution of MK1 instance

6 Conclusion and Acknowledge

The goal of this paper was to solve the multi-objective FJSP problem, specifically, there were three objectives, namely, the minimum makespan, the maximum machine workload and the total workload. By using Pareto sorting and crowding distance, the merits of the solution can be well evaluated. In ABC algorithm, the update operation of food source was divided into two sections including machine assignment vector and operation assignment vector. The TS-ABC algorithm based on effective population initialization strategy combined with tabu search strategy can find the optimal solution well in the whole solution space. Several strategies were used in the initial phase of population. In employed bee phase, every solution would execute crossover operation with best solution whose Pareto rank is 1 by using IPOX operation and MPX operation. The generation of neighborhood solutions in onlooker bee phase mainly used exchange operation and mutation operation so that optimal solution was further optimized. Application of tabu search improved local search ability. The ABC algorithm is tested using two benchmark instances. The experimental results and comparisons proved effectiveness and efficiency of the ABC algorithm for solving the FJSP.

The shortcoming of this algorithm is that the TS-ABC algorithm needs further optimization for the data of the kacem example. At the same time, this paper discusses the most widely studied types of flexible scheduling problems in academia, and there is still a certain distance from actual production. A lot of content needs to be expanded, such as considering preparation time and machine qualification. The future work is to consider inserting jobs into the FJSP problem. The ABC algorithm can be applied to other aspects to prove the efficiency of the algorithm. Explore other efficient meta heuristic algorithms.

References

[1] G.H. Zhang, Improved genetic algorithm for the flexible job-shop scheduling problem, Journal of Mechanical Engineering 45(7)(2009) 145-151.

[2] P. Brucker, R. Schlie, Job-shop scheduling with multi-purpose machines, Computing, 45(1990) 369-375.

[3] K.E. Parsopoulos, M.N. Vrahatis, Recent approaches to global optimization problems through Particle Swarm Optimization, Natural Computing 116(2)(2002) 235-306.

[4] L.D. Giovanni, F. Pezzella, An improved genetic algorithm for the distributed and flexible job-shop scheduling problem, European Journal of Operational Research 200(2)(2009) 395-408.

[5] C. Zhang, X. Wang, L. Gao, An improved genetic algorithm for multi-objective flexible job-shop scheduling problem, in: Proc. International Conference on Manufacturing Science and Engineering (ICMSE 2009), 2009.

[6] W. Xia, Z. Wu, An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems, Comput Ind Eng 48(2005) 409-425.

[7] P. Brandimarte, Routing and scheduling in a flexible job shop by tabu search, Annals of Operations Research 41(3)(1993) 157-183.

[8] G. Zhang, X. Shao, P. Li, L. Gao, An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem, Computers & Industrial Engineering 56(4)(2008) 1309-1318.

[9] S. Jia, Z. Hu, Path-relinking tabu search for the multi-objective flexible job shop scheduling problem, Computers & Operations Research 47(9)(2014) 11-26.

[10] Z.L. Du, Y.B. Xu, Z.H. Cui, Solution space distance clustering-variable neighborhood search particle swarm optimization for flexible job shop scheduling problem, Computer Systems & Applications 25(12)(2016) 143-148.

[11] X.L. Wu, S.D. Sun, J.J. Yu, Research on multi-objective optimization for flexible job shop scheduling, Computer Integrated Manufacturing Systems 12(5)(2006) 731-736.

[12] Q. Jia, Y. Guo, Hybridization of ABC and PSO algorithms for improved solutions of RCPSP, Journal Of The Chinese Institute Of Engineers 39(6)(2016) 727-734.

[13] J. Li, Q. Pan, K. Gao, Pareto-based discrete artificial bee colony algorithm for multi-objective flexible job shop scheduling problems, The International Journal of Advanced Manufacturing Technology 55(9)(2011) 1159-1169.

[14] A. Thammano, A. Phu-Ang, A hybrid artificial bee colony algorithm with local search for flexible job-shop scheduling problem., in: Proc. Complex Adaptive Systems Conference, 2013.

[15] A. Banharnsakun, T. Achalakul, B. Sirinaovakul, The best-so-far selection in artificial bee colony algorithm, Applied Soft Computing 11(2)(2010) 2888-2901.

[16] K. Gao, P.N. Suganthan, Q. Pan, M.F. Tasgetiren, A. Sadollah, Artificial bee colony algorithm for scheduling and rescheduling fuzzy flexible job shop problem with new job insertion, Knowledge-Based Systems 109(2016) 1-16.

[17] L. Wang, G. Zhou, Y. Xu, M. Liu, An enhanced Pareto-based artificial bee colony algorithm for the multi-objective flexible job-shop scheduling, The International Journal of Advanced Manufacturing Technology 60(9)(2012) 1111-1123.

[18] S. Wang, L. Wang, Y. Xu, M. Liu, An effective estimation of distribution algorithm for the flexible job-shop scheduling problem with fuzzy processing time, International Journal of Production Research 51(12)(2013) 3778-3793.

[19] J. Li, Q. Pan, M.F. Tasgetiren, A discrete artificial bee colony algorithm for the multi-objective flexible job-shop scheduling problem with maintenance activities, Applied Mathematical Modelling 38(3)(2013) 1111-1132.

[20] L. Xing, Y. Chen, P. Wang, Q. Zhao, J. Xiong, A knowledge-based ant colony optimization for flexible job shop scheduling problems, Applied Soft Computing 10(3)(2009) 888-896.

[21] M. Yazdani, M. Amiri, M. Zandieh, Flexible job-shop scheduling with parallel variable neighborhood search algorithm, Expert Systems with Applications 37(1)(2009) 678-687.

[22] L. Wang, G. Zhou, Y. Xu, S. Wang, M. Liu, An effective artificial bee colony algorithm for the flexible job-shop scheduling problem, The International Journal of Advanced Manufacturing Technology 60(1)(2012) 303-315.

[23] X. Wu, S. Wu, An elitist quantum-inspired evolutionary algorithm for the flexible job-shop scheduling problem, Journal of Intelligent Manufacturing 28(6)(2017) 1441-1457.

[24] K. Gao, P.N. Suganthan, T. Chua, C. Chong, T. Cai, Q. Pan, A two-stage artificial bee colony algorithm scheduling flexible job-shop scheduling problem with new job insertion, Expert Systems with Applications 42(21)(2015) 7652-7663.