COMPLEX SYSTEM MODELING AND SIMULATION
ISSN 2096-9929 01/06 pp 113−129
Volume 2, Number 2, June 2022
DOI: 10.23919/CSMS.2022.0002

Q-Learning-Based Teaching-Learning Optimization for Distributed Two-Stage Hybrid Flow Shop Scheduling with Fuzzy Processing Time

Bingjie Xi and Deming Lei*

Abstract: Two-stage hybrid flow shop scheduling has been extensively considered in single-factory settings. However, the distributed two-stage hybrid flow shop scheduling problem (DTHFSP) with fuzzy processing time is seldom investigated in multiple factories. Furthermore, the integration of reinforcement learning and metaheuristic is seldom applied to solve DTHFSP. In the current study, DTHFSP with fuzzy processing time was investigated, and a novel Q-learning-based teaching-learning based optimization (QTLBO) was constructed to minimize makespan. Several teachers were recruited for this study. The teacher phase, learner phase, teacher's self-learning phase, and learner's self-learning phase were designed. The Q-learning algorithm was implemented by 9 states, 4 actions defined as combinations of the above phases, a reward, and an adaptive action selection, which were applied to dynamically adjust the algorithm structure. A number of experiments were conducted. The computational results demonstrate that the new strategies of QTLBO are effective; furthermore, it presents promising results on the considered DTHFSP.

Key words: teaching-learning based optimization; Q-learning algorithm; two-stage hybrid flow shop scheduling; fuzzy processing time

1 Introduction

A two-stage hybrid flow shop scheduling problem (THFSP) consists of two stages, at least one of which consists of parallel machines. This NP-hard problem is a special case of the hybrid flow shop scheduling problem (HFSP). This topic has attracted much attention in the past decade, and a number of results have been obtained in single-factory[1−14] and multifactory settings[15, 16].

Various methods, such as the exact, heuristic, and metaheuristic methods, have been applied to solve THFSP in a single-factory setting. For instance, Allaoui and Artiba[1] studied the problem with availability constraint by using an exact method and several heuristics to minimize makespan. Yang[2] developed some heuristics for particular cases of the problem with dedicated machines. Wang and Liu[3] presented a branch-and-bound based heuristic for solving the problem with dedicated machines. Wang et al.[4] proposed a problem-tailored constructive heuristic with local search for bi-criteria energy-efficient THFSP. Feng et al.[5] developed an exact method and two heuristics for THFSP with interval processing time, while Wang and Liu[6] proposed a genetic algorithm for no-wait THFSP. Tan et al.[7] presented a hybrid decomposition approach with variable neighborhood search for the problem with batch processing machines. Fan et al.[8] presented a mutant firefly algorithm to optimize the simultaneous rate for the arrival and on-time delivery rate. Past studies also solved the THFSP by using other metaheuristics, such as artificial immune system[9], tabu search[4, 10], estimation of distribution algorithm[11], artificial bee colony[12], and imperialist competitive algorithm[13]. Lang et al.[14] investigated the neuro-evolution of augmenting topology to generate and parameterize artificial neural networks on

• Bingjie Xi and Deming Lei are with the School of Automation, Wuhan University of Technology, Wuhan 430070, China. E-mail: deminglei11@163.com.
* To whom correspondence should be addressed.
Manuscript received: 2021-10-21; revised: 2022-01-12; accepted: 2022-01-19

determining allocation and sequencing decisions in THFSP with family setup times.

In recent years, production has transferred from single factories to multiple factories with the further development of globalization. As a result, distributed scheduling problems in multiple factories have become the main topic of production scheduling in recent years. The distributed two-stage hybrid flow shop scheduling problem (DTHFSP) is also considered in this work. Lei and Wang[15] and Cai et al.[16] proposed a shuffled frog-leaping algorithm with memeplex grouping and a collaborative variable search (CVS), respectively for DTHFSP with makespan minimization and DTHFSP with sequence-dependent setup time (SDST) and fuzzy processing time.

As stated above, previous works have mainly investigated the THFSP in single-factory settings, while the DTHFSP has been seldom studied. The THFSP exists in many real-life manufacturing processes, with extensive applications in multifactory production in many manufacturing industries, such as semiconductors. To quickly respond to rapid changes in customer demands and global market requirements, it is necessary to extend the THFSP by considering the DTHFSP. However, uncertainty extensively exists in real-life production processes, and these cannot be neglected intentionally because the obtained schedule would not be able to meet the requirements of real-life production once uncertainty is neglected. Moreover, neglecting uncertainty in multiple factories can lead to more losses than neglecting it in a single factory, so it is important to deal with the DTHFSP with uncertainty.

Uncertainty is often depicted by fuzzy theory. Fuzzy scheduling problems have been considered in single[17−25] and multiple factories[16, 26, 27]. For example, Shao et al.[26] studied a distributed fuzzy blocking flow shop scheduling problem in multiple homogeneous factories, after which they presented two constructive heuristics and two iterated greedy to minimize fuzzy makespan. Zheng et al.[27] dealt with multi-objective fuzzy distributed hybrid flow shop scheduling and developed a cooperative coevolution algorithm with an iterated greedy estimation of distribution algorithm. The DTHFSP with fuzzy processing time is solved by using CVS[16]. Obviously, fuzzy distributed scheduling does not attract sufficient attention, and the DTHFSP with fuzzy processing time is seldom considered.

Meanwhile, teaching-learning based optimization (TLBO) is a population-based algorithm inspired by the act of passing on knowledge from a teacher to a student within a classroom. The TLBO possesses a simple structure and fewer parameters and is easily understood and implemented[28]. In recent years, the TLBO has been successfully used to solve various production scheduling problems in single and multiple factories[29−36]. The competitive performances of the TLBO have been verified and tested in these works.

At the same time, the integration of reinforcement learning (RL) into metaheuristics has also become a hot topic, with some studies investigating production scheduling in recent years[37−41]. The integration of RL and metaheuristic can lead to the dynamic selection of search operators or adaptively adjusted parameter settings, among others[42, 43]. As a result, integrating RL with metaheuristic can improve the performance of the latter, making it an effective approach to obtaining high quality solutions.

The extensive applications of the TLBO to scheduling and the positive impact of the integration of RL and metaheuristic motivate us to construct a Q-learning-based teaching-learning based optimization (QTLBO) to solve the DTHFSP with fuzzy processing time. Moreover, RL is used to dynamically adjust the TLBO's algorithm structure, thereby resulting in new applications of RL with metaheuristics.

In this study, the DTHFSP with fuzzy processing time is studied, and a novel algorithm called the QTLBO is constructed through the integration of the Q-learning algorithm and the TLBO to minimize makespan. We recruited a group of teachers and adopted four phases: teacher, learner, teacher's self-learning, and learner's self-learning. The Q-learning algorithm is constructed by 9 states, 4 actions, a reward, and an adaptive action selection to dynamically adjust the algorithm structure of the QTLBO. After conducting extensive experiments, the computational results demonstrate that the new strategies are effective and efficient. Furthermore, the QTLBO is a competitive algorithm for the considered problem.

The remainder of the paper is organized as follows. The problem description is given in Section 2, followed by an introduction to the TLBO and Q-learning in Section 3. Section 4 presents the QTLBO for the problem. Numerical experiments on the QTLBO are reported in Section 5, and the conclusions are

summarized in Section 6. Some topics for future research are also proposed in Section 6.

2 Problem Description

The DTHFSP with fuzzy processing time consists of n jobs $J_1, J_2, \ldots, J_n$ processed by F factories located in different sites. Each factory f possesses a two-stage hybrid flow shop with m_f identical parallel machines $M_{1,S_f+1}, M_{1,S_f+2}, \ldots, M_{1,S_f+m_f}$ at the first stage and a single machine $M_{2,f}$ at the second stage, $S_1 = 0$. Here $S_f = \sum_{l=1}^{f-1} m_l, f > 1$, $W = S_{F+1}$ indicates the total number of parallel machines. All jobs are available at time zero.

The triangular fuzzy number (TFN) is used to depict the processing time and setup time. Specifically, $p_{ilf} = \left(p_{ilf}^1, p_{ilf}^2, p_{ilf}^3\right)$ is the processing time of job J_i at stage l of factory f. SDST is considered. Meanwhile, $u_{jilf} = \left(u_{jisf}^1, u_{jisf}^2, u_{jilf}^3\right)$ denotes the setup time for job J_i on each machine of stage l in factory f when job J_j is processed directly before it on the same machine. At the same time, $u_{0ilf} = \left(u_{0isf}^1, u_{0isf}^2, u_{0ilf}^3\right)$ is the initial fuzzy setup time of the first job J_i processed on the machine at stage l of factory f.

The DTHFSP has some constraints on jobs and machines: (1) each machine can process at most one operation at a time, (2) no jobs may be processed on more than one machine at a time, (3) operations cannot be interrupted, and (4) all machines are available at all times, that is, maintenance, failure, and breakdown are not considered. Figure 1 shows the layout of the DTHFSP.

The DTHFSP possesses three subproblems: the factory assignment used to decide which jobs are assigned to each factory, machine assignment, and scheduling. Cai et al.[16] analyzed the relationships among these subproblems and combined two assignment problems into machine assignments.

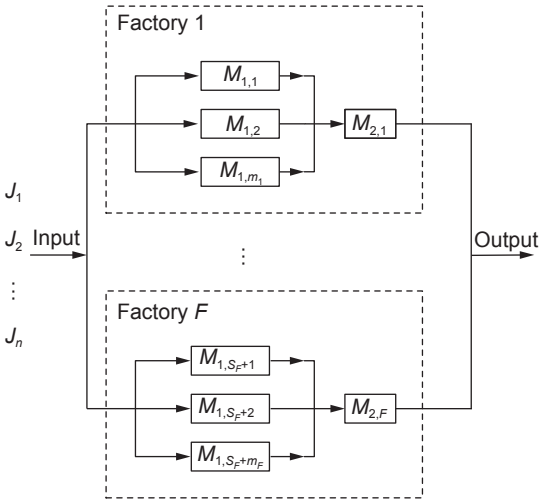

Fig. 1 Layout of the distributed two-stage hybrid flow shop scheduling problem (DTHFSP).

The goal of the problem is to minimize makespan until all constraints are met:

$$C_{\max} = \max_{i=1,2,\ldots,n} \{C_i\} \tag{1}$$

where $C_{\max} = \left(C_{\max}^1, C_{\max}^2, C_{\max}^3\right)$ is the fuzzy makespan, and C_i indicates the fuzzy completion time of job J_i.

When the TFN is used to represent the processing data of the scheduling problem, three operators are often used to build a schedule: addition operator, max operator, and ranking operator.

The addition operator is used to compute the completion time of the job and is defined for TFNs $B = (b_1, b_2, b_3)$ and $H = (h_1, h_2, h_3)$ by

$$B + H = (b_1 + h_1, b_2 + h_2, b_3 + h_3) \tag{2}$$

The ranking operator is about the comparison regarding fuzzy completion time. If $c_1(B) > c_1(H)$ or $c_1(B) = c_1(H)$ and $c_2(B) > c_2(H)$ or $c_1(B) = c_1(H)$, $c_2(B) = c_2(H)$, and $c_3(B) > c_3(H)$, then $B > H$, where $c_3(B) = b_3 - b_1$, $c_1(B) = (b_1 + 2b_2 + b_3)/4$, and $c_2(B) = b_2$.

The max operator is applied to decide the fuzzy beginning time and is given by

$$\text{if } B > H, \text{then } B \vee H = B; \text{else } B \vee H = H \tag{3}$$

Cai et al.[16] also considered the above problem and provided an example and a schedule. With respect to makespan, each job J_i has a fuzzy completion time C_i, then $C_{\max} = C_1 \vee C_2 \vee \cdots \vee C_n$.

3 Introduction to the TLBO and the Q-Learning Algorithm

The search process of the TLBO acts on population P, which is considered as a class of learners and has a teacher phase and a learner phase. In the teacher phase, a teacher $x_{teacher}$, which is the best solution obtained so far, passes its knowledge to learners, and a solution x is modified by

$$x_{new} = x + rand \times (x_{teacher} - T \times x_{mean}) \tag{4}$$

where T is a teaching factor that can have a value of either 1 or 2, x_{mean} is the current mean value of all solutions, and *rand* is a random number following

uniform distribution in [0, 1].

In the learner phase, a learner x increases its knowledge by interacting with its classmate x_j :

$$x_{new} = \begin{cases} x + rand \times (x - x_j), & \text{if } f(x) < f(x_j); \\ x + rand \times (x_j - x), & \text{else} \end{cases} \tag{5}$$

where $f(x)$ indicates the fitness of x.

In the above phases, x_{new} is accepted if x_{new} gives a better objective than x.

Equations (4) and (5) cannot directly generate feasible solutions to scheduling problems because of their combinatorial feature. The TLBO is often discrete when it is used to solve scheduling problems[29−37]. In the current study, The QTLBO is also discrete.

In recent years, RL has been applied to solve a wide variety of complex problems[37−40, 42−45], and its main components are a learning agent, an environment, states, actions, and rewards. The Q-learning algorithm[46] is the most commonly used RL algorithm. It works by learning an action-value function that expresses the cumulative reward of taking a certain action in a given state. It has been successfully integrated with metaheuristics for production scheduling[37−40]. Its simplest form is defined by

$$\begin{aligned} &Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \\ &\alpha\left(r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right) \end{aligned} \tag{6}$$

where α denotes the learning rate, often set to 0.1 (this setting is adopted in this study), γ indicates the discount factor, r_{t+1} is the reward by taking a_t on s_t, and $\max_a Q(s_{t+1}, a)$ represents the largest Q value in the Q-table at state s_{t+1}.

Action selection is executed based on the Q-table. Initially, all elements of the Q-table are zero, which means that the agent does not have any learning experience. Here, ε-greedy is often used, and this is expressed as follows: if a random number $rand < \varepsilon$, then randomly select an action a; otherwise, select an action a that maximizes Q values, that is, $a = \arg\max_{a'} Q(s_t, a')$.

4 QTLBO for the DTHFSP with Fuzzy Processing Time

The integration of RL and the TLBO is hardly investigated in the literature. Thus, the current paper proposed an effective way to combine an RL algorithm named Q-learning and the TLBO to solve the considered DTHFSP.

4.1 Coding and decoding

For the DTHFSP with fuzzy processing time, it is required to optimize scheduling and machine assignment of the first stage because factory assignment can be inferred from machine assignment[16]. Thus, a two-string representation is used as a solution to the problem.

For the DTHFSP with n jobs and W parallel machines at stage 1, a solution is denoted by a machine assignment string $\left[M_{1,\theta_1}, M_{1,\theta_2}, \ldots, M_{1,\theta_n}\right]$ and a scheduling string $[q_1, q_2, \ldots, q_n]$, where machine M_{1,θ_i} is allocated for job J_i, and q_i is the real number of job J_i. The second string is a random key; job permutation is obtained by sorting all its genes. Figure 2 shows the coding strings.

For example, a solution has a string [0.2,0.3,0.45, 0.55,0.09] and a string $[M_{1,2}, M_{1,3}, M_{1,1}, M_{1,3}, M_{1,4}]$. In the machine assignment string, J_1 is allocated on $M_{1,1}$, J_2 is on $M_{1,3}$, and so on, job permutation [5,1,2,3,4] is obtained by sorting all genes in the ascending order of q_i.

The decoding procedure is shown as follows. In each factory f, for each parallel machine $M_{1,k}$, all its allocated jobs are decided on according to the machine assignment string. Then, a permutation of these jobs is obtained by sorting these jobs in the ascending order of their q_i. Next, the first job of the permutation is initiated, all the jobs of the permutation on $M_{1,k}$ are processed sequentially, and the processing of each job on $M_{2,f}$ is completed after the processing at stage 1 is finished. On $M_{1,3}$, J_2 and J_4 are processed and a permutation of these jobs is presented[2, 4].

Meanwhile, in the decoding process, the beginning time of a job on a machine is computed by using the max operator. The completion time is calculated with the addition operator, and the ranking operator compares the makespan of each solution in order to decide the elite solution, among others. The initial population P with N solutions is randomly produced. All solutions are sorted in the ascending order of their makespan.

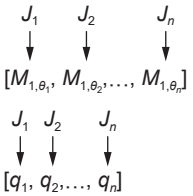

Fig. 2 Descriptions on coding strings.

4.2 Four phases of the QTLBO

A group G of teachers is constructed by selecting $\beta \times N$ best solutions from population P, where β is a real number. Each member of G acts as a teacher. Aside from the teacher and learner phases, the teacher's and learner's self-learning are also added. Thus, a total of four phases were used.

The teacher phase is shown below. For each learner $x \in P \backslash G$, the following steps are repeated η_1 times. First, a teacher $x_{teacher} \in G$ is randomly selected. Second, for x and $x_{teacher}$, a two-point crossover of the machine assignment string between them is executed. If the new solution x_{new} is better than x, then x_{new} substitutes for x; else do a two-point crossover of scheduling string between them and replace x with x_{new} if x_{new} is better than x.

Next, the learner phase is described as follows. For each learner $x \in P \backslash G$, repeat the following steps η_2 times, randomly select $\delta \in [1, N]$ and a solution x_j, $j \in [1, \delta]$, and execute a two-point crossover between x and x_j. Two cases are considered. In the first case, only the two-point crossover of the machine assignment string is done. In the second case, crossovers between x and x_j are executed as in the teacher phase, where η_i is an integer. A teaching-learning ration μ is defined as $\eta_1 : \eta_2$.

The two other phases are based on neighborhood searches NS_1 and NS_2, which are constructed by using six neighborhood structures. Neighborhood structure $\mathcal{N}_1$ produces new solutions by removing a randomly selected job J_i from the factory with the biggest completion time to the factory with the shortest completion time. $\mathcal{N}_2$ is depicted as follows. First, randomly choose jobs J_i, J_j from the factory f with the biggest completion time, then swap M_{1,θ_i} and M_{1,θ_j} on the machine assignment string. When a job is removed from the factory with the biggest completion time in $\mathcal{N}_1$ or two jobs are swapped in $\mathcal{N}_2$, there is a high probability that the makespan will diminish.

When J_j is selected at random from a randomly chosen factory (not factory f in $\mathcal{N}_2$), $\mathcal{N}_3$ is thus obtained. Meanwhile, $\mathcal{N}_4$ is used to generate solutions by exchanging two randomly selected genes in the scheduling string. $\mathcal{N}_5$ is shown as follows: randomly decide q_i and q_j, and insert q_i into position $j-1$ in scheduling string if $j > 1$. If $j = 1$, insert q_i into position j in scheduling string. In addition, $\mathcal{N}_6$ is depicted as follows: randomly select q_i, q_j, and q_k in the scheduling string and set a new sequence for them. For example, $q_1 = 0.8$, $q_3 = 0.75$, and $q_5 = 0.78$; hence, the new sequence may be $q_1 = 0.78$, $q_3 = 0.8$, and $q_5 = 0.75$. Thus, $\mathcal{N}_4$, $\mathcal{N}_5$, and $\mathcal{N}_6$ act on the scheduling string. NS_1 is shown as follows. For solution x, let $g = 1$, and produce a solution $x_{new} \in \mathcal{N}_g(x)$. If x_{new} is better than x, then $x = x_{new}$; else, $g = g + 1$. Next, obtain $x_{new} \in \mathcal{N}_g(x)$. If x_{new} is better than x, then $x = x_{new}$; else $g = g + 1$. Finally, generate $x_{new} \in \mathcal{N}_g(x)$ and $x = x_{new}$ if x_{new} is better than x. NS_2 differs from NS_1 in that g is initially set to be 4 and not 1 initially.

The teacher's self-learning phase is executed as follows. For each $x_{teacher} \in G$, execute NS_1 η_2 times. The learner's self-learning phase is depicted as follows: for each learner x, execute one of the following two cases η_2 times: Case 1—only NS_1 is used, and Case 2— NS_1 and NS_2 are done sequentially.

4.3 Q-learning algorithm

In this study, the Q-learning algorithm is integrated with the TLBO to dynamically select some phases. To realize this goal, the population evaluation results are used to depict state s_t, and a combination of some phases is applied to describe action a_t. As a result, the action selection will lead to the dynamical selection of the algorithm structure.

Three indices are used to evaluate population P. First, $trial^*$ is used to describe the global best solution $x_{gb}^t \in P$. Initially, $trial^* = 0$. If x_{gb}^t is not updated on generation t, $trial^* = trial^* + 1$; otherwise, $trial^* = 0$. Discrete degree D^t on generation t is defined by

$$D^t = \frac{2\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\left|C_{\max}^{i,t} - C_{\max}^{j,t}\right|}{N(N-1)} \tag{7}$$

where $C_{\max}^{i,t}$ indicates the makespan of solution $x_i \in P$ on generation t.

Genotype difference GD^t of $\beta \times N$ best solutions in P is shown below. The set Θ is the set of $\beta \times N$ solutions of P with the smallest makespan. For each $x \in \Theta$, job permutation is first obtained by sorting all genes of scheduling string, then let $\eta_x^t = 0$, for each $y \in \Theta, y \neq x$, starting with the first position of the machine assignment string of x and y. For each position, x and y have different genes, then $\eta_x^t = \eta_x^t + 1$; the same process is done on the job permutation of x and y.

$$GD^t = \sum\nolimits_{x \in \Theta} \eta_x^t / |\Theta| \tag{8}$$

where η_x^t is an integer related to x on generation t.

As shown in Table 1, nine states are defined based on the above three indices, where $\Delta D^t = D^t - D^{t-1}$, $\Delta GD^t = GD^t - GD^{t-1}$.

Four actions are given by using some phases of the QTLBO. Action $a(1)$ is composed of the teacher and learner phases. Action $a(2)$ consists of the teacher phase, the first case of the learner's self-learning phase, and the learner phase. In the teacher phase of $a(2)$, only a two-point crossover of the machine assignment string is executed. Action $a(3)$ is made up of the teacher phase, the first case of the learner's self-learning phase, and the second case of the learner's phase. When $a(4)$ is executed, the teacher's self-learning phase, the teacher phase, and the second case of the other two phases are applied sequentially. When other actions like $a(1)$ are completed, their own phases are also executed sequentially.

Here, the reward is defined by

$$r_{t+1} = \frac{w_1 \times \rho_1 + w_2 \times \rho_2 + w_3 \times \rho_3}{\rho_1 + \rho_2 + \rho_3} \tag{9}$$

where $\rho_1 = \left(f\left(x_{gb}^t\right) - f\left(x_{gb}^{t+1}\right)\right)/f\left(x_{gb}^t\right)$, $\rho_2 = \Delta D^{t+1}/D^t$, $\rho_3 = \Delta GD^{t+1}/GD^t$, and w_i is the real number, $w_1 + w_2 + w_3 = 1$.

We believe that the elite solution is more important than the other two indices used for the state, so $w_1 > w_2, w_1 > w_3$. We set $w_1 = 0.5, w_2 = 0.3, w_3 = 0.2$ by experiments.

An adaptive ε-greedy selection is proposed to decide action on generation t. Here ε is defined as $\varepsilon = \varepsilon_0 - (\varepsilon_0 - 0.01) \times R_{cur}/R_{\max}$, where $R_{\max}$ is the running time as a stopping condition, and R_{cur} indicates the current time. In addition, 0.01 is used to guarantee $\varepsilon \geqslant 0.01$.

4.4 Algorithm description

The above Q-learning algorithm is the main part of the QTLBO and is used to decide its algorithm structure. This is a new application of Q-learning to metaheuristics. Yet, this feature is hardly considered in the previous TLBOs.

Table 1 Station set.

State	Description
1	$trial^* = 0, \Delta D^t > 0$
2	$trial^* = 0, \Delta D^t < 0$
3	$trial^* > 0, \Delta D^t = 0$
4	$trial^* > 0, \Delta D^t > 0, \Delta GD^t < 0$
5	$trial^* > 0, \Delta D^t > 0, \Delta GD^t = 0$
6	$trial^* > 0, \Delta D^t > 0, \Delta GD^t > 0$
7	$trial^* > 0, \Delta D^t < 0, \Delta GD^t < 0$
8	$trial^* > 0, \Delta D^t < 0, \Delta GD^t = 0$
9	$trial^* > 0, \Delta D^t < 0, \Delta GD^t > 0$

The QTLBO consists of the initialization, sorting, Q-learning step, and execution steps of the chosen action. The latter three steps are repeated until the stopping condition is met. The flow chart of the QTLBO is shown in Fig. 3. Meanwhile, the sorting step is used to sort all solutions of the population in the ascending order of their makespan, which decides the group of teachers.

Unlike the previous TLBO, sorting and Q-learning are applied to choose an action, and the chosen action includes some phases of the QTLBO.

5 Computational Experiment

Extensive experiments were conducted to evaluate the performance of the QTLBO for the DTHFSP with fuzzy processing time and SDST. All experiments were implemented by using Microsoft Visual C++ 2019 and run on an 8.0 GB RAM 2.4 GHz CPU PC.

5.1 Instance and comparative algorithms

One hundred instances were generated according to Cai et al.[16] Tables 2 and 3 show the information on these instances.

Ying and Lin[47] studied distributed HFSP with

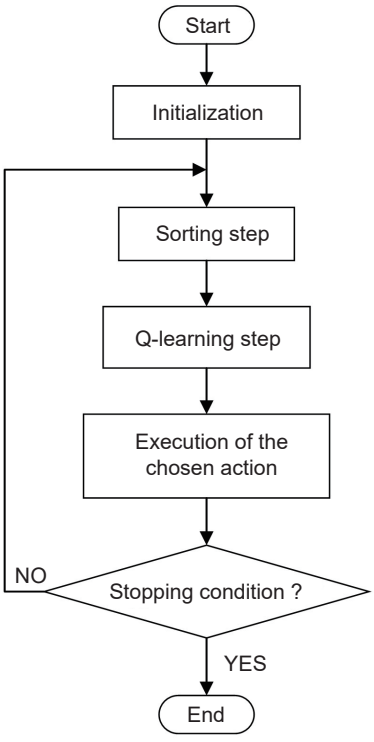

Fig. 3 Algorithm flow chart.

Table 2 Factory information on 100 instances.

Instance	F	m_f
1−10	2	3,3
11−20	2	2,4
21−31	3	3,3,3
32−42	3	2,3,4
43−53	4	2,2,2,2
54−64	4	2,2,3,3
65−73	4	2,3,4,5
74−82	5	3,3,3,3,3
83−91	5	2,3,3,3,4
92−100	5	2,3,4,5,6

Table 3 Information of number of jobs on 100 instances.

Instance	n
1,11,21,32,43,54	30
2,12,22,33,44,55	40
3,13,23,34,45,56,65,74,83,92	50
4,14,24,35,46,57,66,75,84,93	60
5,15,25,36,47,58,67,76,85,94	70
6,16,26,37,48,59,68,77,86,95	80
7,17,27,38,49,60,69,78,87,96	90
8,18,28,39,50,61,70,79,88,97	100
9,19,29,40,51,62,71,80,89,98	120
10,20,30,41,52,63,72,81,90,99	150
31,42,53,64,73,82,91,100	200

multiprocessor tasks and presented a self-tuning iterated greedy (SIG) to minimize makespan. Li et al.[48] developed an improved artificial bee colony (IABC) for distributed HFSP with SDST and unrelated parallel machines. The proposed IABC adopts the factory assignment rule, the greedy iterative strategy, and a hybrid search strategy. Li et al.[49] developed an improved brainstorm optimization algorithm for type-2 fuzzy distributed hybrid flow shop scheduling. These algorithms can be directly used to solve the DTHFSP with fuzzy processing time and yield superior performance. Thus, they were chosen as comparative algorithms in the current study.

To show the new strategies, such as the usage of the Q-learning algorithm, a TLBO is constructed by removing the Q-learning algorithm from the QTLBO and sorting steps. In the TLBO, there is only one teacher, which is the best solution in population P. The teacher phase and learner phase are executed.

5.2 Parameter settings

We set running time $R_{\max}$ to be $0.2\times n\times m$ s. The QTLBO, TLBO, and three comparative algorithms converge fully when the above CPU time is used. Thus, we selected the above time as the stopping condition.

Other parameters of the QTLBO were as follows: N, γ, ε_0, β, and μ. The Taguchi method was used to decide the settings for these parameters. Instance 69 was chosen. The levels of parameters are given in Table 4, and the orthogonal array $L_{27}(3^5)$ is executed. The QTLBO with each parameter combination was run independently 10 times for instance 69. The results of $c_1(Min)$ and S/N ratio are given in Fig. 4, where S/N ratio is $-10\log_{10}(c_1(Min)^2)$, and Min is the best solution obtained in 10 runs. Given that Min is the TFN and cannot be used to compute S/N ratio; thus, so $c_1(Min)$ is used.

As shown in Fig. 4, when the levels of N, γ, ε_0, β,

Table 4 Parameters and their levels.

Factor level	Parameter				
	N	γ	ε_0	β	μ
1	60	0.85	0.4	0.1	1:1
2	80	0.9	0.5	0.15	5:2
3	100	0.95	0.6	0.2	5:1

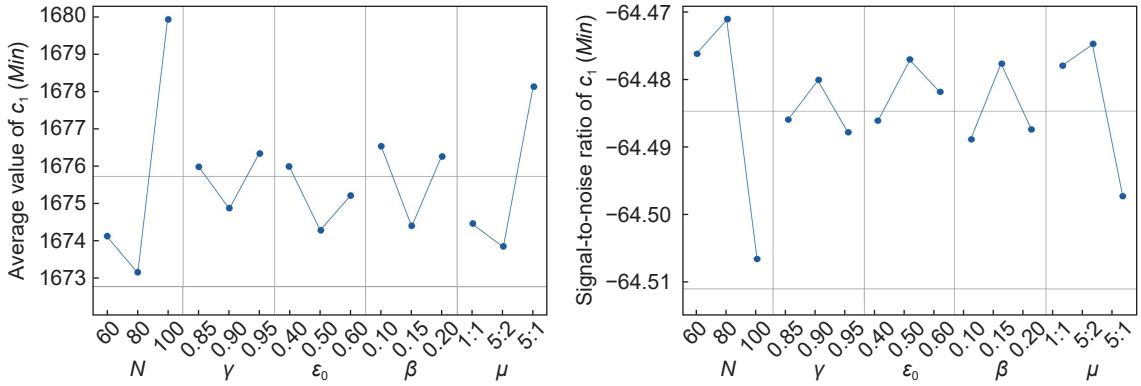

Fig. 4 Results on average value and S/N ratio on $c_1(Min)$.

and μ are 2, the QTLBO produces smaller average $c_1(Min)$ and bigger S/N ratio than the QTLBO with other levels. The suggested settings are $N = 80$, $\gamma = 0.9$, $\varepsilon_0 = 0.5$, $\beta = 0.15$, and $\mu = 5:2$.

The above settings were obtained by $c_1(Min)$. We tested these settings on all instances and found that the above settings can produce better results than the other settings; thus, the above settings were adopted.

Except the stopping condition, the TLBO only has $N = 80$. For three comparative algorithms, all parameters except the stopping condition were directly selected from previous works[45−47].

We ran the QTLBO, TLBO, and three comparative algorithms 10 times on each instance. Tables 5−10 show the computational results of all algorithms, in which *Max* indicates the worst one of 10 elite solutions obtained in 10 runs, and *Avg* is the average value of the 10 elite solutions. Table 11 presents the results of the paired sample t-test, where the significance level is 0.05. Figure 5 describes the convergence curves of all algorithms on four instances, while Fig. 6 describes the best schedule of instance 1 obtained by the QTLBO.

As shown in Tables 5−10, the QTLBO produces better *Min* than the TLBO on all instances; moreover, there are significant differences between the *Min* of the QTLBO and the *Min* of the TLBO. Notably, the former performs better than the latter in terms of convergence. The same conclusion also can be drawn from Table 11 and Fig. 5. The QTLBO also obtains smaller *Avg* and *Max* than the TLBO on all instances. Furthermore, the QTLBO possesses significant performance advantages in terms of average results and stability. Obviously, the new strategies of the QTLBO have had a positive impact on its effective and efficient performance.

When the QTLBO is compared with the IABC, it can be found from Tables 7 and 8 that the QTLBO obtains smaller *Min* than the IABC in 92 of 100 instances. This means that the IABC generates better *Min* than the QTLBO on only 8 instances. Meanwhile, the QTLBO outperforms the IABC in terms of convergence. As shown in Tables 5, 6, 9, and 10, the QTLBO produces smaller *Max* than the IABC on 89 instances and better *Avg* than the IABC on 85 instances. The QTLBO also performs better than the IABC in terms of average results and stability. It can be seen from Table 5–10 that the QTLBO has better performance than the SIG and IBSO. The SIG and IBSO cannot produce better solutions than the QTLBO on all instances. Moreover, the *Min* of QTLBO is significantly smaller than that of the SIG and IBSO. Finally, the QTLBO performs better than the SIG and IBSO in relation to the other two metrics.

Table 11 and Fig. 5 present the convergence difference between the QTLBO and its comparative algorithm. The superior performance of the QTLBO mainly results from its integration with the Q-learning algorithm and four phases. These four phases are a combination of global search and neighborhood search. The Q-learning algorithm is used to dynamically select the algorithm structure with some phases. The exploration is intensified because of the dynamical selection of algorithm structure, and the possibility of falling local optima also can be diminished notably. Thus, it can be concluded that the QTLBO is an effective method for the DTHFSP with fuzzy processing time.

6 Conclusion

Many studies have investigated the integration of RL and metaheuristic for production scheduling. This study provides a new path to integrate RL with the TLBO. Unlike existing works, this study applies an RL algorithm named Q-learning to dynamically adjust the algorithm structure of the QTLBO. In the QTLBO, a group of teachers was used. The teacher phase, learner phase, teacher's self-learning phase, and learner's self-learning phase were designed. Then, the Q-learning algorithm was implemented by 9 states, 4 actions defined as combinations of the above phases, a reward, and an adaptive action selection, after which they are applied to dynamically adjust algorithm structure. A number of experiments were conducted. The computational results demonstrate that the new strategies of the QTLBO are effective and that the QTLBO provides promising results on the considered DTHFSP.

Distributed scheduling has also been extensively considered; however, distributed scheduling with uncertainty is not studied fully. Thus, in future works, we will attempt to solve the distributed scheduling problem with uncertainty by using various metaheuristics. Previous works have also mainly used many kinds of RL algorithms, particularly Q-learning. Related to this, the integration of other RL algorithms with metaheuristics for production scheduling can also be the subject of our future research.

Table 5 Computational results of all algorithms on *Avg* for instances 1−50.

Instance	QTLBO	TLBO	IABC	SIG	IBSO
1	(1021.2,1131.6,1246.4)	(1062.7,1174.5,1347.9)	(1022.6,1133,1247.6)	(1100,1202.8,1354.1)	(1052.6,1160.3,1295.9)
2	(1383.5,1531,1720.7)	(1445.5,1592.3,1854.1)	(1382.8,1534.4,1719.9)	(1459.3,1603.6,1805.3)	(1426.6,1578,1768.4)
3	(1653.5,1844.4,2084.6)	(1748.7,1937.1,2253.8)	(1653.3,1845.9,2086.6)	(1774.6,1961.5,2236.7)	(1732.1,1919.5,2159.8)
4	(1995.9,2194.6,2491.6)	(2100.2,2313.9,2692.9)	(1993.5,2193.8,2486.2)	(2124.2,2330.3,2641.6)	(2077.9,2293.3,2574.1)
5	(2325.4,2556,2871.1)	(2433.7,2681.4,3087.8)	(2329.1,2547.5,2884.7)	(2455.2,2702.2,3038)	(2423.8,2667.4,2983.1)
6	(2645.5,2939.6,3289.3)	(2774.8,3081,3544.1)	(2674.2,2969.9,3304.6)	(2786.6,3090.8,3510.9)	(2755,3057.6,3452.9)
7	(2998.9,3330.2,3685.3)	(3133.7,3476.6,3956.5)	(3135.1,3475.3,3896.9)	(3114.5,3459.5,3897.1)	(3119.5,3453.2,3854)
8	(3308.8,3684.4,4161.3)	(3475.5,3865,4433.4)	(3509.1,3876.4,4393.6)	(3513.2,3886.2,4408.4)	(3455.2,3840.4,4351.4)
9	(4032.9,4446.2,4983.6)	(4184.9,4609.2,5298.9)	(4212.2,4632.6,5240.7)	(4226.1,4649.6,5304.3)	(4182.9,4606.8,5216.7)
10	(5004.6,5535.8,6221.1)	(5209.7,5757.3,6583.6)	(5249.7,5770.7,6557.6)	(5249.9,5784.3,6564.9)	(5215,5749.4,6522.1)
11	(1044.7,1142.7,1276.6)	(1092.3,1198.1,1412.4)	(1045.2,1146.6,1284.1)	(1102,1199,1347)	(1068.3,1172.6,1316.6)
12	(1353.6,1486.2,1689.2)	(1427.7,1573.9,1843.8)	(1345.6,1503.8,1699.8)	(1445.6,1588,1804.2)	(1394.4,1547.3,1755.4)
13	(1676.1,1846.6,2070.5)	(1763.7,1954.4,2256.8)	(1652.7,1842.3,2042.4)	(1771.8,1955.2,2192.6)	(1718.7,1910.4,2138.4)
14	(1986.3,2222,2500.2)	(2150.1,2370.5,2732.4)	(2008.2,2228.4,2509.4)	(2152.3,2368.4,2685.1)	(2097.9,2320.6,2619.2)
15	(2330.4,2585.4,2911.4)	(2490.2,2749.8,3146.4)	(2359.9,2594,2896.4)	(2479.7,2736,3080.6)	(2451.5,2698.4,3008.8)
16	(2623.9,2895.1,3243.5)	(2831.9,3110,3546.2)	(2667.3,2959.3,3281.4)	(2809.7,3097.3,3494.7)	(2762.2,3045.1,3411.4)
17	(2985.5,3290.5,3719.1)	(3170.5,3502,4036)	(3150.2,3480.2,3934.3)	(3161.5,3493.4,3959.9)	(3120.4,3451.3,3901.1)
18	(3300.4,3665.5,4083.3)	(3527.3,3882.1,4437.7)	(3519.4,3868.9,4386.5)	(3534.5,3880.8,4391.4)	(3489.3,3838.5,4340.7)
19	(3951,4365.5,4889.9)	(4229.5,4653.7,5304.4)	(4193.2,4618.3,5198.7)	(4197.4,4639.7,5224.9)	(4179.7,4597.2,5148)
20	(4976.6,5494.2,6103.4)	(5266.5,5811.4,6650.1)	(5212.2,5749.2,6504.1)	(5225.2,5773.7,6559.9)	(5197.7,5740.3,6456)
21	(696.5,769.4,864.3)	(749.6,824.9,984.3)	(710.3,784.1,851.8)	(762.4,846.6,951.9)	(734.3,809.5,892.7)
22	(924,1004.7,1113.8)	(978.1,1076.5,1271.8)	(913.1,1015.7,1113.8)	(995.7,1094.8,1232.2)	(956.3,1059.4,1168.2)
23	(1118.1,1250.1,1406.2)	(1207.6,1342.1,1599)	(1119.6,1262.4,1436.4)	(1223.6,1360.9,1548)	(1186.1,1320.3,1502.5)
24	(1336.3,1491.3,1659.3)	(1450.9,1595.7,1848.5)	(1355.8,1507,1648.8)	(1463.7,1614.8,1822.9)	(1434.6,1575.9,1756.6)
25	(1551.4,1720.9,1909.3)	(1658.8,1842.6,2149.3)	(1553,1728.9,1931.4)	(1678,1857,2115.1)	(1637.1,1818.8,2067.6)
26	(1757.3,1956.1,2176.3)	(1922.9,2110.3,2454.5)	(1820,2011.8,2266.6)	(1932.2,2136.7,2433.6)	(1883.6,2074,2361.9)
27	(1967.1,2197.7,2473.3)	(2147.3,2352.1,2724.6)	(2141.3,2357.5,2670)	(2160.9,2382,2693.5)	(2112.5,2336.5,2620.6)
28	(2202.2,2452.2,2752.5)	(2353.2,2594.7,3017.3)	(2357.3,2605.5,2972.5)	(2396.7,2640.2,3005.3)	(2327.9,2581,2925.4)
29	(2665.1,2951.2,3299.1)	(2817.1,3099.8,3543.8)	(2828.2,3116.1,3498.9)	(2842.3,3138.5,3534.2)	(2793.7,3081.4,3457.7)
30	(3344.7,3701.8,4138.8)	(3501.4,3861.8,4418.4)	(3525.6,3889.7,4379.2)	(3542.6,3906.9,4421.9)	(3497.5,3845.9,4335.3)
31	(4421.4,4925.6,5556.8)	(4651.5,5161,5912.7)	(4699.6,5193.7,5877.3)	(4735.9,5239.4,5930.1)	(4665.8,5153.8,5820.9)
32	(703.9,784.5,863)	(757.6,837.1,1026)	(696.4,775.8,876.2)	(767.8,842.6,961.2)	(726.9,807.3,911.1)
33	(914.4,1024,1136.4)	(966.7,1072.7,1276.1)	(920.5,1033.1,1162.6)	(1004.5,1113.4,1252.5)	(965,1068.9,1205.8)
34	(1118.9,1251.5,1410.5)	(1201.1,1335.9,1579.3)	(1103.9,1237.6,1348)	(1227.4,1350.1,1513.8)	(1175.4,1302.7,1442.5)
35	(1326.1,1463.2,1638.3)	(1459,1608.6,1892.8)	(1323.4,1474.5,1666.7)	(1427,1579.7,1792.7)	(1398.2,1558.8,1747.9)
36	(1560.7,1739,1959.3)	(1689.7,1858,2159.3)	(1551.6,1725.6,1909.2)	(1691.1,1862.3,2112.6)	(1649.9,1821.7,2035)
37	(1805.2,1981.8,2225.4)	(1918.4,2112,2443.4)	(1786.2,1995.7,2236)	(1922.9,2128.9,2410.3)	(1859.2,2067.9,2333.6)
38	(1992.5,2209.4,2463.3)	(2160.8,2384.4,2750.1)	(2121.4,2332.5,2621.3)	(2160.6,2367.9,2668.1)	(2089.6,2306.8,2596.4)
39	(2259.4,2497.9,2763.8)	(2385.7,2626.4,3038.8)	(2364.2,2614.3,2940.4)	(2376.8,2623.8,2943.3)	(2328.9,2575.4,2904)
40	(2665.8,2943.1,3291.4)	(2863.2,3160.6,3628.6)	(2843.5,3133.7,3492.7)	(2871.9,3153.7,3554.2)	(2794.7,3068.3,3482.7)
41	(3364.6,3728.9,4112.2)	(3558.4,3898.1,4480.2)	(3495.6,3865.8,4375.4)	(3524.9,3912.8,4394.9)	(3468.8,3840.8,4315.1)
42	(4449.2,4924.2,5502.7)	(4711.5,5188.1,5970.5)	(4700,5181.8,5856.2)	(4723.1,5199.1,5860.9)	(4673.7,5148.5,5781.9)
43	(540,606.5,661)	(588.4,652.4,785.1)	(540.5,608.5,687.5)	(603.6,671.3,764)	(569.2,629.2,715.3)
44	(696.8,761.6,860.3)	(749.2,830.9,1016.3)	(696.4,770.4,865.2)	(788.1,858.3,979.2)	(744,816.7,923.8)
45	(855.9,953.3,1072.3)	(937.7,1034.7,1220.8)	(872,959.5,1058.7)	(972.2,1065.5,1204.9)	(919.4,1014.6,1133.5)
46	(1021.2,1132.7,1276.4)	(1101,1213.2,1442)	(1003,1111.7,1242.6)	(1111.4,1223.9,1384.8)	(1080.3,1192.4,1342.1)
47	(1180.9,1309.1,1449.2)	(1278.7,1422.5,1648.6)	(1192.9,1332.5,1499.2)	(1304,1438.6,1618.6)	(1253.5,1379.7,1560.5)
48	(1345.7,1488.5,1664.4)	(1449.2,1589.5,1837.5)	(1333.9,1485.2,1660)	(1476,1623.5,1834.5)	(1417.6,1566.6,1768.3)
49	(1495.6,1664.4,1866.7)	(1625,1793.7,2094)	(1637.5,1813.6,2034.8)	(1661.6,1839.1,2079.1)	(1600.7,1774.8,2006.6)
50	(1670.3,1852.3,2065.8)	(1790.4,1980.2,2291.7)	(1796.4,1988.8,2247.4)	(1818.5,2018.2,2282.5)	(1755.4,1945.2,2212.3)

Table 6 Computational results of all algorithms on *Avg* for instances 51−100.

Instance	QTLBO	TLBO	IABC	SIG	IBSO
51	(2043.6,2244.6,2506.5)	(2148.2,2378.4,2756.6)	(2140,2372.5,2677.2)	(2187.3,2415.2,2733.8)	(2117.5,2331.4,2635.2)
52	(2505.9,2785.5,3108.2)	(2667.2,2951.6,3404.8)	(2688.9,2960.2,3315.9)	(2716.3,2993.3,3365.1)	(2639.8,2921.6,3246.1)
53	(3352.4,3708,4154.6)	(3546.5,3918.7,4501.9)	(3551.1,3919.3,4442.4)	(3575.3,3945.9,4488.6)	(3497.4,3869,4331.5)
54	(541.8,605.7,662.4)	(602.3,666.6,817.4)	(556.8,612.3,662.4)	(604,667.4,739.3)	(567.3,629.2,696.1)
55	(695.545,764.909,855)	(745.6,822.5,996.4)	(695.9,762.8,859)	(772.9,859.7,981.3)	(736.2,806.1,919.3)
56	(865.1,960,1052.1)	(938.2,1033.6,1234.3)	(859,953,1048.9)	(960.5,1056.4,1198.3)	(915.1,1009.1,1127.3)
57	(1018.1,1121.2,1267.4)	(1115.3,1222.5,1433.4)	(1000.1,1120,1243.1)	(1121,1229.8,1387)	(1078,1190.3,1340.6)
58	(1174.1,1306,1450.7)	(1274.3,1414.8,1666.2)	(1171,1308.4,1459.8)	(1308.5,1453.9,1632.9)	(1256.8,1394.6,1549.9)
59	(1350.1,1485.4,1662.4)	(1446.9,1606.2,1861.2)	(1351.3,1490,1656.9)	(1484.4,1634.3,1826.6)	(1420.6,1562.9,1749.8)
60	(1504.7,1663.8,1863)	(1616.7,1784.6,2093.3)	(1627,1793.9,2033.6)	(1650.2,1813.1,2056.7)	(1591.5,1761.9,1989.7)
61	(1679,1857.7,2070.6)	(1804.9,1992.3,2315.8)	(1794.6,1977.4,2236.4)	(1843.7,2028.8,2306.7)	(1762.3,1942,2194.9)
62	(2022.5,2248.4,2522.4)	(2138.7,2375.8,2789.2)	(2160.6,2385.2,2675.9)	(2189.9,2425,2734.8)	(2113.6,2339.2,2650.6)
63	(2523.6,2789.4,3113)	(2674.1,2949.4,3408.8)	(2694.1,2970.6,3328.7)	(2695.8,2979.9,3380.2)	(2645.7,2920.5,3292.4)
64	(3374.7,3727.6,4166.5)	(3566.3,3926.4,4521.6)	(3573.8,3946.7,4464.4)	(3645.6,4025.7,4561.6)	(3528.8,3893.3,4427.5)
65	(855.6,954.3,1052.8)	(933.9,1025.9,1233.8)	(867.8,958.4,1037.3)	(963,1063.1,1190.4)	(911.2,1004.4,1102.7)
66	(1012.8,1118.2,1266.2)	(1110.4,1226.6,1462.3)	(1004.1,1115.8,1237.4)	(1117.4,1233.8,1386.8)	(1079.2,1190.9,1342)
67	(1175.4,1301.1,1453.6)	(1294.4,1435.1,1669.5)	(1169.9,1312.2,1435.8)	(1300.7,1425.4,1609.2)	(1246.7,1376.8,1563.7)
68	(1337.9,1482.4,1662.7)	(1444.8,1592.4,1855.6)	(1380.8,1525.8,1710.4)	(1491,1642.5,1869.9)	(1432.5,1581.9,1778)
69	(1510.2,1672.8,1861.1)	(1638.6,1805.6,2100.1)	(1638.4,1793.2,2023.3)	(1659,1828.6,2070.8)	(1601.5,1768.3,1989.3)
70	(1661.5,1844.7,2059.1)	(1804.4,1996.4,2333.9)	(1792.1,1982.6,2238.5)	(1835.7,2039.8,2310.2)	(1767,1952.2,2205.8)
71	(2026.6,2251.8,2540)	(2167.4,2394.4,2777.6)	(2148.5,2377.7,2664.9)	(2187.2,2418.1,2722.6)	(2111.4,2331.9,2632.5)
72	(2505.9,2774.9,3092.4)	(2685.1,2963.7,3420)	(2684.2,2962.6,3346.2)	(2711.3,2996.7,3413.6)	(2634.5,2913.4,3307.9)
73	(3331.7,3695.2,4132.4)	(3571,3941.8,4565.4)	(3545.5,3926.3,4446.8)	(3577.1,3950.7,4488.3)	(3523.2,3888.4,4370.7)
74	(683.3,765.8,851.3)	(756.2,832.3,1006.1)	(678.6,758.2,847.6)	(769.2,846.4,957.3)	(737.1,810.4,918.8)
75	(815.2,908.8,1003.3)	(907.7,996.6,1196.3)	(814.3,905.9,1028.4)	(930.7,1023.2,1164)	(876.3,967.6,1100.1)
76	(951.3,1054.9,1191.7)	(1042.4,1149.1,1364.2)	(947.5,1049.6,1179.7)	(1069.4,1185.4,1333.6)	(1012.8,1124.1,1268.5)
77	(1075.2,1192.1,1323.5)	(1183.3,1305,1576.7)	(1132.9,1252.8,1379.2)	(1227.3,1343.6,1527)	(1150.8,1269.9,1430.8)
78	(1214.5,1348.5,1502.2)	(1315.4,1458.1,1711)	(1323.8,1457.6,1644.7)	(1349.2,1481.8,1686.5)	(1289.3,1427.1,1613.4)
79	(1350.6,1495,1664.5)	(1470.1,1617.2,1886.9)	(1467,1618.8,1832.2)	(1478.4,1633.2,1864.9)	(1436.6,1588.3,1786.9)
80	(1608.1,1790.5,2007.3)	(1737.6,1908.1,2231)	(1753,1933.1,2195)	(1809.9,1998.1,2268.3)	(1716.1,1890.9,2141.7)
81	(2031.6,2245.6,2516.7)	(2162,2375.1,2745.5)	(2182.1,2405.8,2707.5)	(2240.4,2461.1,2760.1)	(2141.7,2362.2,2640.8)
82	(2707.4,2995.4,3368.4)	(2845.1,3145.1,3613.5)	(2950.2,3245.5,3664.4)	(2915.5,3214,3643.2)	(2856.1,3134,3526.4)
83	(687.7,765.8,862.2)	(768.3,850.9,1021.1)	(698.2,762.4,840.4)	(805.4,890.5,1002)	(742.1,814.1,916.4)
84	(815.4,904.7,1003.5)	(906.2,995.6,1193.7)	(813.1,907.2,1009.2)	(944.4,1047.8,1193.8)	(884.5,970.7,1096)
85	(946.7,1049.9,1174.1)	(1039.1,1156.4,1398.7)	(943.6,1043.1,1159.8)	(1066.6,1184.1,1352)	(1010.8,1117.4,1265.1)
86	(1079.1,1197.4,1327.1)	(1178.9,1309.5,1549.4)	(1131.9,1251.7,1407)	(1200.9,1324.3,1484.8)	(1150.8,1276.8,1436.1)
87	(1211,1346.8,1501.8)	(1329.6,1465,1709.5)	(1311.9,1450.7,1642.9)	(1346.3,1477.1,1678.6)	(1282.9,1424.3,1591.9)
88	(1340.3,1503.6,1686.1)	(1462.8,1614.3,1902.9)	(1473.3,1628.5,1840.8)	(1491.2,1649.4,1861.4)	(1436.7,1595.1,1797.7)
89	(1605.5,1788.7,1990.1)	(1750.6,1926,2250.8)	(1769.7,1949.8,2204.4)	(1809.9,1991.4,2252.9)	(1718.9,1896.1,2129.9)
90	(2017,2237.2,2486.3)	(2174,2387.5,2778.2)	(2180.2,2394.1,2707.7)	(2254.9,2472,2803)	(2132.3,2351.7,2645.6)
91	(2705,3000.1,3371.3)	(2866.5,3167.1,3641.6)	(2914,3205.1,3631.8)	(2947.8,3253.1,3703.9)	(2832.7,3127.1,3548.4)
92	(690.2,762.9,841)	(766.7,840,1033.5)	(678.7,752.3,842.4)	(799.9,883,996.1)	(733,806.3,915.4)
93	(815.2,903.3,1005.9)	(905,1001.1,1184.6)	(823.6,910.8,999.4)	(933.9,1028.9,1168.4)	(886.5,973,1092.6)
94	(950.4,1054,1167.2)	(1052.1,1158.5,1379.7)	(940.6,1035,1159.4)	(1065.3,1165.4,1322.2)	(1007.3,1112.5,1253.2)
95	(1086.2,1196.7,1336.8)	(1199.8,1323.3,1554.5)	(1151.9,1275.1,1430.8)	(1234.3,1363.7,1551.7)	(1152.6,1274.4,1438.3)
96	(1229.2,1359.8,1502.8)	(1326.5,1465,1743.5)	(1321.3,1454.8,1641.4)	(1372.1,1511.1,1715.1)	(1295.8,1422.1,1608.8)
97	(1347.7,1490.4,1642.8)	(1459.8,1620.2,1921.7)	(1458.7,1606.6,1831.5)	(1501.5,1660.1,1866.2)	(1433.6,1584.4,1779.3)
98	(1611.9,1794,2020.7)	(1760.8,1941.1,2264.9)	(1767.4,1941.6,2228.1)	(1804.3,1991.4,2281.7)	(1724,1898.1,2160.1)
99	(2028.8,2236.5,2494.1)	(2195.9,2419.3,2824.3)	(2154.8,2390.7,2701.3)	(2174.6,2419.4,2733.8)	(2112.3,2346.1,2645.2)
100	(2722.2,3012.2,3359.3)	(2952,3243.5,3742.6)	(2935.4,3231.7,3685.9)	(2917.4,3209.6,3650.3)	(2828.7,3119,3512.4)

Table 7 Computational results of all algorithms on *Min* for instances 1−50.

Instance	QTLBO	TLBO	IABC	SIG	IBSO
1	(1019,1132,1245)	(1053,1164,1324)	(1020,1131,1248)	(1079,1187,1344)	(1045,1153,1289)
2	(1380,1531,1712)	(1432,1585,1844)	(1379,1536,1713)	(1451,1583,1780)	(1431,1565,1768)
3	(1651,1836,2091)	(1745,1917,2252)	(1649,1845,2079)	(1749,1934,2182)	(1711,1908,2165)
4	(1997,2187,2497)	(2099,2313,2639)	(1980,2209,2453)	(2094,2307,2626)	(2077,2295,2538)
5	(2322,2542,2885)	(2415,2651,3096)	(2328,2545,2876)	(2413,2659,3012)	(2404,2641,2960)
6	(2639,2926,3299)	(2749,3058,3544)	(2667,2967,3290)	(2747,3047,3444)	(2753,3041,3433)
7	(2990,3313,3663)	(3113,3474,3906)	(3124,3464,3899)	(3070,3409,3810)	(3090,3445,3845)
8	(3287,3686,4133)	(3478,3843,4427)	(3514,3887,4329)	(3469,3821,4339)	(3460,3842,4282)
9	(4025,4451,4940)	(4152,4567,5289)	(4173,4595,5239)	(4162,4599,5155)	(4172,4600,5182)
10	(4995,5499,6176)	(5163,5740,6518)	(5298,5761,6465)	(5215,5768,6525)	(5162,5730,6548)
11	(1038,1140,1284)	(1074,1190,1406)	(1039,1146,1281)	(1075,1184,1323)	(1050,1162,1304)
12	(1353,1483,1684)	(1401,1565,1801)	(1345,1499,1699)	(1407,1561,1740)	(1393,1548,1728)
13	(1662,1840,2087)	(1715,1900,2198)	(1665,1847,2007)	(1756,1942,2109)	(1727,1902,2087)
14	(1988,2224,2479)	(2121,2343,2704)	(2010,2225,2495)	(2115,2339,2620)	(2082,2314,2602)
15	(2336,2586,2894)	(2436,2714,3164)	(2363,2587,2892)	(2462,2699,2985)	(2452,2687,2998)
16	(2620,2887,3247)	(2776,3067,3540)	(2658,2964,3250)	(2784,3085,3436)	(2742,3051,3357)
17	(2947,3277,3752)	(3159,3475,4003)	(3132,3469,3914)	(3075,3396,3911)	(3084,3424,3940)
18	(3300,3642,4069)	(3493,3856,4409)	(3493,3840,4396)	(3513,3865,4335)	(3474,3826,4325)
19	(3928,4340,4858)	(4192,4588,5293)	(4195,4599,5200)	(4151,4605,5162)	(4157,4590,5109)
20	(4938,5446,6106)	(5194,5735,6570)	(5196,5729,6507)	(5170,5749,6532)	(5162,5724,6444)
21	(700,764,863)	(746,810,964)	(709,782,848)	(751,829,926)	(718,805,907)
22	(918,1002,1119)	(967,1062,1274)	(904,1006,1128)	(985,1080,1200)	(953,1056,1156)
23	(1116,1248,1402)	(1185,1324,1573)	(1115,1258,1423)	(1175,1331,1516)	(1188,1323,1478)
24	(1340,1494,1627)	(1445,1567,1848)	(1359,1502,1634)	(1436,1570,1783)	(1416,1577,1737)
25	(1550,1712,1903)	(1661,1819,2124)	(1532,1712,1961)	(1643,1839,2084)	(1644,1819,2036)
26	(1756,1936,2189)	(1903,2107,2425)	(1797,1991,2295)	(1907,2091,2398)	(1859,2053,2385)
27	(1958,2183,2499)	(2110,2325,2723)	(2106,2342,2708)	(2112,2328,2631)	(2099,2334,2602)
28	(2201,2444,2739)	(2309,2577,3003)	(2366,2589,2916)	(2370,2599,2951)	(2332,2581,2888)
29	(2663,2951,3266)	(2752,3054,3525)	(2835,3114,3472)	(2825,3125,3459)	(2805,3088,3402)
30	(3373,3700,4070)	(3492,3829,4365)	(3529,3876,4347)	(3519,3896,4417)	(3480,3849,4315)
31	(4398,4917,5542)	(4612,5120,5892)	(4699,5163,5823)	(4699,5199,5852)	(4668,5140,5777)
32	(699,776,873)	(739,831,1026)	(696,771,875)	(743,824,919)	(726,806,904)
33	(906,1018,1150)	(965,1072,1233)	(915,1030,1154)	(991,1097,1223)	(971,1066,1184)
34	(1092,1251,1432)	(1159,1321,1588)	(1103,1233,1348)	(1214,1332,1455)	(1178,1296,1424)
35	(1321,1461,1624)	(1459,1604,1873)	(1313,1461,1684)	(1396,1559,1764)	(1387,1547,1756)
36	(1555,1732,1956)	(1671,1842,2109)	(1541,1710,1905)	(1657,1848,2081)	(1617,1801,2069)
37	(1818,1980,2202)	(1896,2098,2419)	(1784,1994,2211)	(1885,2084,2392)	(1843,2052,2342)
38	(1956,2194,2500)	(2139,2358,2765)	(2123,2318,2590)	(2118,2339,2632)	(2069,2297,2568)
39	(2246,2506,2736)	(2368,2588,3030)	(2352,2605,2915)	(2335,2579,2901)	(2329,2574,2850)
40	(2655,2950,3268)	(2865,3131,3566)	(2850,3123,3428)	(2807,3108,3461)	(2771,3052,3449)
41	(3355,3735,4088)	(3546,3892,4420)	(3493,3843,4343)	(3471,3865,4370)	(3471,3833,4296)
42	(4449,4910,5493)	(4680,5167,5914)	(4658,5183,5833)	(4650,5144,5836)	(4654,5129,5758)
43	(545,603,655)	(584,645,769)	(546,605,678)	(583,634,730)	(561,622,707)
44	(695,757,860)	(755,827,982)	(694,770,854)	(775,845,942)	(734,814,913)
45	(855,952,1066)	(927,1020,1217)	(846,960,1043)	(930,1029,1133)	(904,1006,1131)
46	(1018,1131,1273)	(1085,1197,1403)	(996,1103,1245)	(1087,1186,1351)	(1063,1198,1334)
47	(1183,1290,1465)	(1272,1410,1625)	(1188,1341,1469)	(1289,1394,1582)	(1247,1370,1553)
48	(1336,1478,1670)	(1428,1588,1798)	(1338,1483,1637)	(1435,1602,1790)	(1400,1558,1763)
49	(1491,1665,1850)	(1620,1766,2073)	(1628,1793,2057)	(1651,1804,2034)	(1593,1772,1991)
50	(1685,1849,2009)	(1766,1962,2264)	(1789,1982,2232)	(1786,1987,2201)	(1738,1942,2198)

Table 8 Computational results of all algorithms on *Min* for instances 51−100.

Instance	QTLBO	TLBO	IABC	SIG	IBSO
51	(2040,2222,2494)	(2128,2377,2719)	(2118,2346,2688)	(2129,2364,2641)	(2115,2305,2645)
52	(2468,2768,3147)	(2642,2901,3432)	(2672,2939,3284)	(2636,2908,3294)	(2636,2919,3209)
53	(3313,3674,4198)	(3539,3905,4419)	(3504,3907,4446)	(3538,3898,4409)	(3470,3864,4270)
54	(540,600,671)	(584,654,825)	(567,608,646)	(597,649,699)	(554,623,699)
55	(690,756,863)	(732,805,978)	(694,763,847)	(760,825,920)	(732,796,919)
56	(857,962,1041)	(909,1009,1231)	(863,941,1054)	(925,1003,1179)	(906,1006,1122)
57	(1006,1125,1254)	(1111,1214,1368)	(992,1117,1229)	(1097,1220,1366)	(1068,1192,1312)
58	(1157,1301,1463)	(1279,1388,1657)	(1161,1305,1450)	(1259,1410,1588)	(1260,1380,1553)
59	(1350,1473,1660)	(1423,1588,1858)	(1347,1488,1636)	(1471,1597,1800)	(1411,1539,1762)
60	(1497,1660,1856)	(1601,1756,2023)	(1641,1779,2015)	(1606,1796,2045)	(1588,1751,1963)
61	(1670,1853,2066)	(1795,1982,2300)	(1784,1953,2249)	(1789,1974,2205)	(1748,1929,2180)
62	(2000,2243,2512)	(2114,2351,2784)	(2143,2373,2666)	(2152,2374,2669)	(2097,2329,2643)
63	(2480,2767,3142)	(2674,2912,3369)	(2667,2949,3361)	(2655,2951,3292)	(2651,2934,3233)
64	(3376,3731,4116)	(3506,3897,4505)	(3523,3911,4426)	(3531,3920,4438)	(3493,3881,4419)
65	(851,938,1081)	(923,1006,1192)	(860,960,1021)	(911,1013,1133)	(914,1001,1083)
66	(1006,1120,1257)	(1089,1213,1459)	(1000,1112,1230)	(1108,1205,1324)	(1048,1186,1338)
67	(1179,1294,1454)	(1274,1421,1639)	(1163,1307,1438)	(1260,1395,1569)	(1241,1373,1560)
68	(1328,1471,1656)	(1420,1576,1843)	(1380,1518,1681)	(1443,1594,1801)	(1442,1570,1743)
69	(1496,1654,1895)	(1615,1809,2074)	(1614,1778,2007)	(1635,1790,2077)	(1599,1761,1981)
70	(1651,1834,2055)	(1754,1980,2310)	(1770,1970,2227)	(1806,2001,2270)	(1764,1938,2186)
71	(2017,2224,2571)	(2105,2360,2780)	(2159,2373,2624)	(2120,2355,2607)	(2079,2325,2614)
72	(2486,2762,3096)	(2632,2934,3414)	(2691,2961,3282)	(2669,2932,3333)	(2601,2888,3333)
73	(3295,3677,4091)	(3566,3910,4483)	(3522,3917,4423)	(3512,3848,4451)	(3485,3895,4357)
74	(680,759,859)	(748,825,1004)	(668,750,861)	(751,822,944)	(730,803,894)
75	(806,909,999)	(877,980,1152)	(806,903,1026)	(897,989,1117)	(860,958,1120)
76	(945,1048,1195)	(1042,1141,1329)	(930,1044,1169)	(1038,1139,1262)	(1019,1117,1257)
77	(1072,1191,1309)	(1155,1274,1539)	(1117,1233,1372)	(1195,1300,1441)	(1126,1266,1420)
78	(1206,1340,1508)	(1297,1431,1701)	(1295,1448,1634)	(1311,1435,1635)	(1264,1418,1628)
79	(1355,1494,1638)	(1443,1593,1828)	(1462,1610,1802)	(1431,1578,1815)	(1434,1575,1781)
80	(1612,1774,1996)	(1726,1886,2197)	(1734,1916,2197)	(1719,1921,2191)	(1716,1882,2125)
81	(2002,2223,2567)	(2113,2351,2729)	(2171,2405,2661)	(2148,2383,2643)	(2107,2362,2641)
82	(2705,2976,3351)	(2846,3120,3576)	(2914,3192,3548)	(2843,3173,3589)	(2854,3116,3534)
83	(690,768,846)	(761,843,998)	(699,760,829)	(748,840,927)	(731,807,926)
84	(806,902,1009)	(901,983,1205)	(808,908,995)	(889,985,1116)	(886,961,1100)
85	(954,1043,1167)	(1027,1126,1403)	(956,1045,1113)	(1033,1133,1288)	(1002,1117,1239)
86	(1091,1196,1301)	(1147,1288,1545)	(1125,1243,1391)	(1148,1278,1462)	(1125,1261,1435)
87	(1219,1348,1478)	(1318,1448,1691)	(1306,1448,1615)	(1313,1427,1603)	(1262,1409,1610)
88	(1334,1497,1684)	(1422,1597,1906)	(1459,1614,1814)	(1456,1602,1807)	(1418,1603,1770)
89	(1617,1774,1973)	(1719,1901,2204)	(1744,1938,2189)	(1742,1940,2179)	(1729,1880,2088)
90	(2022,2227,2468)	(2143,2373,2713)	(2183,2384,2684)	(2155,2343,2672)	(2120,2345,2610)
91	(2713,2986,3345)	(2859,3142,3636)	(2864,3160,3573)	(2886,3142,3592)	(2806,3104,3563)
92	(702,769,806)	(739,809,1008)	(675,751,838)	(760,825,926)	(733,804,902)
93	(798,902,1010)	(888,992,1169)	(820,909,987)	(888,988,1128)	(891,969,1067)
94	(970,1055,1135)	(1023,1136,1320)	(958,1030,1140)	(1031,1143,1307)	(998,1099,1260)
95	(1079,1197,1322)	(1176,1300,1511)	(1154,1270,1410)	(1152,1271,1460)	(1137,1275,1403)
96	(1213,1355,1496)	(1306,1451,1734)	(1307,1444,1625)	(1293,1435,1650)	(1284,1418,1603)
97	(1336,1486,1647)	(1444,1592,1908)	(1451,1598,1823)	(1486,1606,1805)	(1415,1580,1783)
98	(1601,1791,2006)	(1722,1911,2266)	(1755,1916,2219)	(1739,1910,2187)	(1714,1900,2126)
99	(2014,2238,2476)	(2151,2392,2818)	(2167,2385,2651)	(2153,2380,2704)	(2070,2336,2653)
100	(2718,3022,3288)	(2845,3147,3685)	(2872,3181,3621)	(2825,3130,3558)	(2845,3105,3487)

Table 9 Computational results of all algorithms on Max for instances 1−50.

Instance	QTLBO	TLBO	IABC	SIG	IBSO
1	(1022,1133,1247)	(1075,1181,1368)	(1024,1136,1249)	(1132,1222,1399)	(1057,1164,1306)
2	(1381,1534,1726)	(1449,1606,1884)	(1387,1538,1721)	(1484,1631,1855)	(1446,1581,1779)
3	(1652,1846,2102)	(1760,1956,2243)	(1656,1854,2099)	(1795,1971,2293)	(1745,1927,2160)
4	(2008,2190,2498)	(2099,2339,2732)	(2006,2195,2505)	(2147,2344,2673)	(2081,2307,2586)
5	(2329,2550,2894)	(2443,2697,3096)	(2332,2552,2886)	(2475,2729,3044)	(2452,2676,2971)
6	(2651,2945,3334)	(2810,3104,3570)	(2681,2986,3348)	(2824,3151,3659)	(2764,3055,3519)
7	(3008,3341,3694)	(3121,3497,3998)	(3163,3490,3879)	(3161,3497,3944)	(3144,3458,3851)
8	(3307,3704,4189)	(3514,3869,4475)	(3520,3885,4431)	(3579,3970,4400)	(3453,3857,4365)
9	(4037,4446,5013)	(4226,4653,5342)	(4226,4635,5296)	(4261,4686,5372)	(4156,4612,5281)
10	(5036,5550,6241)	(5255,5782,6605)	(5253,5780,6607)	(5269,5812,6670)	(5215,5768,6559)
11	(1047,1149,1271)	(1109,1209,1416)	(1041,1154,1287)	(1106,1210,1362)	(1068,1173,1337)
12	(1361,1491,1681)	(1446,1595,1861)	(1359,1516,1684)	(1461,1607,1851)	(1398,1556,1750)
13	(1674,1848,2084)	(1794,1989,2270)	(1647,1840,2066)	(1783,1985,2281)	(1732,1916,2158)
14	(1988,2225,2506)	(2180,2393,2768)	(2027,2231,2510)	(2190,2406,2750)	(2087,2328,2645)
15	(2344,2591,2928)	(2496,2784,3196)	(2342,2603,2909)	(2499,2774,3161)	(2437,2710,3045)
16	(2627,2914,3259)	(2893,3150,3573)	(2675,2956,3313)	(2852,3113,3524)	(2781,3066,3416)
17	(2958,3291,3782)	(3210,3536,4045)	(3175,3508,3921)	(3218,3538,3977)	(3129,3462,3924)
18	(3314,3686,4099)	(3553,3914,4468)	(3513,3870,4438)	(3563,3917,4428)	(3537,3860,4297)
19	(3973,4376,4918)	(4267,4706,5332)	(4207,4635,5202)	(4224,4685,5306)	(4196,4601,5186)
20	(5000,5497,6158)	(5320,5843,6761)	(5216,5768,6508)	(5236,5812,6622)	(5215,5752,6469)
21	(694,769,880)	(734,831,1023)	(713,786,853)	(784,873,967)	(738,810,896)
22	(922,1007,1128)	(986,1082,1322)	(907,1015,1134)	(1020,1125,1256)	(963,1067,1165)
23	(1123,1246,1422)	(1218,1354,1616)	(1130,1280,1424)	(1258,1383,1590)	(1184,1321,1524)
24	(1332,1497,1665)	(1493,1624,1807)	(1347,1505,1682)	(1442,1620,1894)	(1441,1579,1774)
25	(1557,1726,1917)	(1688,1864,2153)	(1555,1728,1958)	(1734,1893,2149)	(1649,1819,2080)
26	(1775,1962,2185)	(1936,2114,2517)	(1822,2002,2330)	(1957,2167,2480)	(1901,2098,2352)
27	(1980,2208,2454)	(2177,2360,2777)	(2178,2374,2642)	(2248,2443,2763)	(2108,2352,2655)
28	(2230,2461,2737)	(2407,2607,3025)	(2370,2612,2976)	(2419,2678,3023)	(2300,2584,2967)
29	(2657,2960,3322)	(2842,3118,3585)	(2845,3123,3512)	(2851,3188,3586)	(2790,3079,3498)
30	(3340,3688,4209)	(3560,3896,4436)	(3526,3900,4394)	(3566,3940,4498)	(3513,3844,4358)
31	(4465,4948,5521)	(4664,5211,5950)	(4747,5194,5920)	(4776,5292,5953)	(4671,5163,5860)
32	(704,781,880)	(775,852,1023)	(683,778,901)	(803,850,986)	(745,810,903)
33	(917,1029,1132)	(990,1097,1280)	(919,1039,1171)	(1017,1134,1284)	(970,1078,1206)
34	(1139,1253,1399)	(1200,1345,1603)	(1115,1243,1357)	(1253,1383,1543)	(1184,1318,1434)
35	(1337,1464,1642)	(1486,1636,1921)	(1324,1465,1716)	(1447,1586,1848)	(1401,1567,1757)
36	(1579,1736,1959)	(1711,1870,2196)	(1564,1732,1902)	(1711,1884,2172)	(1659,1826,2059)
37	(1811,1994,2224)	(1941,2131,2461)	(1791,1997,2257)	(1977,2181,2440)	(1862,2072,2351)
38	(2007,2218,2464)	(2189,2415,2767)	(2111,2354,2632)	(2159,2401,2744)	(2106,2314,2598)
39	(2249,2490,2812)	(2413,2647,3071)	(2382,2602,2981)	(2403,2658,2961)	(2334,2582,2938)
40	(2681,2956,3278)	(2902,3178,3654)	(2858,3161,3491)	(2987,3251,3561)	(2813,3093,3462)
41	(3375,3744,4090)	(3575,3907,4512)	(3506,3875,4419)	(3542,3943,4465)	(3479,3856,4334)
42	(4467,4941,5533)	(4743,5217,5990)	(4692,5203,5884)	(4709,5207,5994)	(4671,5163,5809)
43	(538,610,660)	(598,662,799)	(533,607,715)	(700,761,851)	(576,632,729)
44	(708,766,850)	(757,841,1025)	(689,774,874)	(793,874,1026)	(750,824,926)
45	(849,954,1101)	(947,1039,1235)	(873,976,1056)	(1089,1166,1316)	(916,1011,1167)
46	(1026,1136,1277)	(1101,1226,1470)	(1018,1115,1255)	(1151,1263,1448)	(1077,1196,1360)
47	(1165,1316,1464)	(1305,1437,1640)	(1205,1337,1487)	(1376,1526,1755)	(1235,1386,1597)
48	(1357,1501,1648)	(1464,1595,1862)	(1339,1478,1694)	(1529,1681,1914)	(1446,1577,1748)
49	(1502,1673,1871)	(1630,1811,2146)	(1632,1813,2080)	(1678,1865,2137)	(1618,1777,2009)
50	(1662,1859,2095)	(1818,2003,2296)	(1809,1988,2282)	(1908,2062,2340)	(1765,1948,2240)

Table 10 Computational results of all algorithms on *Max* for instances 51−100.

Instance	QTLBO	TLBO	IABC	SIG	IBSO
51	(2034,2243,2564)	(2167,2397,2782)	(2148,2403,2656)	(2307,2582,2921)	(2125,2352,2624)
52	(2487,2789,3160)	(2715,2975,3412)	(2691,2983,3323)	(2932,3208,3631)	(2666,2943,3230)
53	(3359,3735,4174)	(3583,3939,4515)	(3557,3919,4496)	(3635,3988,4543)	(3490,3900,4367)
54	(541,606,669)	(617,691,815)	(554,617,682)	(629,701,771)	(581,625,708)
55	(709,765,850)	(756,840,990)	(709,766,860)	(835,928,1061)	(740,810,929)
56	(860,953,1084)	(967,1053,1246)	(872,949,1093)	(1000,1117,1235)	(904,1017,1146)
57	(1033,1127,1254)	(1120,1236,1474)	(1008,1128,1258)	(1145,1256,1401)	(1080,1185,1372)
58	(1178,1310,1449)	(1290,1434,1673)	(1170,1315,1471)	(1392,1525,1710)	(1281,1407,1532)
59	(1359,1494,1659)	(1480,1627,1878)	(1378,1503,1665)	(1534,1682,1867)	(1435,1567,1772)
60	(1525,1686,1830)	(1628,1812,2169)	(1614,1800,2058)	(1677,1849,2083)	(1602,1761,2001)
61	(1706,1864,2058)	(1800,1999,2347)	(1808,1988,2273)	(1982,2210,2589)	(1753,1952,2228)
62	(2027,2267,2506)	(2170,2404,2785)	(2158,2388,2702)	(2353,2571,2932)	(2125,2346,2658)
63	(2524,2802,3134)	(2709,2988,3425)	(2690,2987,3344)	(2710,3017,3414)	(2639,2913,3344)
64	(3428,3747,4129)	(3609,3972,4555)	(3609,3982,4443)	(3889,4306,4818)	(3547,3903,4420)
65	(856,956,1056)	(945,1027,1262)	(881,964,1037)	(1031,1142,1240)	(903,1010,1128)
66	(1021,1121,1272)	(1131,1230,1475)	(1007,1126,1230)	(1152,1253,1398)	(1096,1199,1338)
67	(1178,1307,1450)	(1310,1456,1696)	(1181,1313,1445)	(1319,1453,1633)	(1242,1386,1568)
68	(1348,1505,1639)	(1469,1608,1868)	(1379,1532,1732)	(1597,1778,2055)	(1432,1587,1802)
69	(1501,1676,1877)	(1641,1811,2161)	(1645,1802,2043)	(1723,1921,2182)	(1598,1771,2015)
70	(1656,1853,2065)	(1821,2032,2384)	(1800,1997,2249)	(1887,2075,2369)	(1778,1964,2198)
71	(2044,2252,2546)	(2194,2422,2787)	(2164,2392,2655)	(2295,2535,2923)	(2130,2357,2620)
72	(2509,2787,3095)	(2710,2973,3440)	(2691,2971,3424)	(2824,3133,3581)	(2637,2930,3321)
73	(3365,3726,4152)	(3608,3994,4676)	(3567,3937,4464)	(3674,4045,4591)	(3531,3900,4379)
74	(702,769,840)	(762,845,1029)	(687,760,854)	(778,861,1013)	(738,821,920)
75	(831,919,982)	(923,1014,1220)	(820,909,1036)	(970,1076,1220)	(879,985,1092)
76	(946,1053,1221)	(1055,1158,1402)	(956,1064,1194)	(1130,1255,1429)	(1014,1138,1262)
77	(1096,1211,1297)	(1212,1318,1605)	(1143,1271,1389)	(1330,1445,1650)	(1173,1272,1428)
78	(1217,1344,1531)	(1319,1460,1753)	(1314,1459,1703)	(1439,1588,1771)	(1300,1441,1597)
79	(1362,1502,1677)	(1493,1632,1923)	(1480,1648,1891)	(1514,1662,1917)	(1451,1589,1789)
80	(1627,1799,2002)	(1764,1947,2219)	(1767,1945,2217)	(1972,2181,2425)	(1701,1896,2188)
81	(2053,2259,2530)	(2183,2392,2800)	(2197,2435,2752)	(2384,2597,2970)	(2154,2369,2646)
82	(2724,3030,3337)	(2855,3152,3667)	(3072,3373,3849)	(2959,3246,3687)	(2835,3138,3571)
83	(687,765,876)	(778,864,1035)	(701,768,842)	(874,993,1128)	(738,813,937)
84	(823,911,992)	(941,1017,1162)	(820,912,1013)	(1071,1173,1346)	(876,980,1102)
85	(961,1053,1169)	(1050,1173,1423)	(953,1045,1186)	(1199,1318,1525)	(1027,1112,1281)
86	(1079,1194,1351)	(1198,1329,1565)	(1151,1280,1409)	(1296,1407,1569)	(1161,1292,1416)
87	(1227,1343,1510)	(1337,1482,1756)	(1347,1483,1663)	(1470,1599,1819)	(1294,1436,1595)
88	(1345,1515,1687)	(1457,1624,1943)	(1480,1653,1906)	(1555,1718,1967)	(1425,1591,1836)
89	(1612,1794,2007)	(1780,1934,2280)	(1804,1970,2256)	(1880,2039,2315)	(1722,1896,2165)
90	(2025,2248,2508)	(2215,2400,2825)	(2189,2407,2714)	(2419,2660,3018)	(2148,2352,2659)
91	(2720,3002,3395)	(2855,3188,3673)	(3043,3347,3743)	(3070,3418,3860)	(2834,3132,3575)
92	(709,772,813)	(795,853,1045)	(678,754,847)	(917,1009,1137)	(728,805,943)
93	(812,914,999)	(904,1017,1199)	(815,914,1013)	(993,1087,1225)	(900,977,1092)
94	(937,1056,1196)	(1086,1172,1412)	(946,1038,1177)	(1098,1207,1386)	(1008,1118,1266)
95	(1101,1195,1347)	(1235,1364,1600)	(1169,1283,1426)	(1392,1522,1722)	(1162,1272,1466)
96	(1240,1369,1507)	(1354,1486,1771)	(1325,1479,1667)	(1490,1638,1898)	(1303,1424,1626)
97	(1352,1497,1653)	(1473,1656,1966)	(1471,1619,1842)	(1575,1770,2007)	(1439,1580,1808)
98	(1627,1812,2004)	(1798,1991,2287)	(1813,1981,2244)	(1931,2118,2436)	(1703,1898,2220)
99	(2033,2241,2502)	(2229,2461,2817)	(2174,2427,2690)	(2207,2455,2770)	(2131,2351,2673)
100	(2748,3019,3402)	−305,033,403,835	(3083,3405,3881)	(3006,3294,3791)	(2823,3125,3570)

Table 11 Results of the paired sample t-test.

t-test	*p*-value (*Avg*)	*p*-value (*Max*)	*p*-value (*Min*)
t-test(QTLBO,TLBO)	0.0000	0.0000	0.0000
t-test(QTLBO,IABC)	0.0000	0.0000	0.0000
t-test(QTLBO,SIG)	0.0000	0.0000	0.0000
t-test(QTLBO,IBSO)	0.0000	0.0000	0.0000

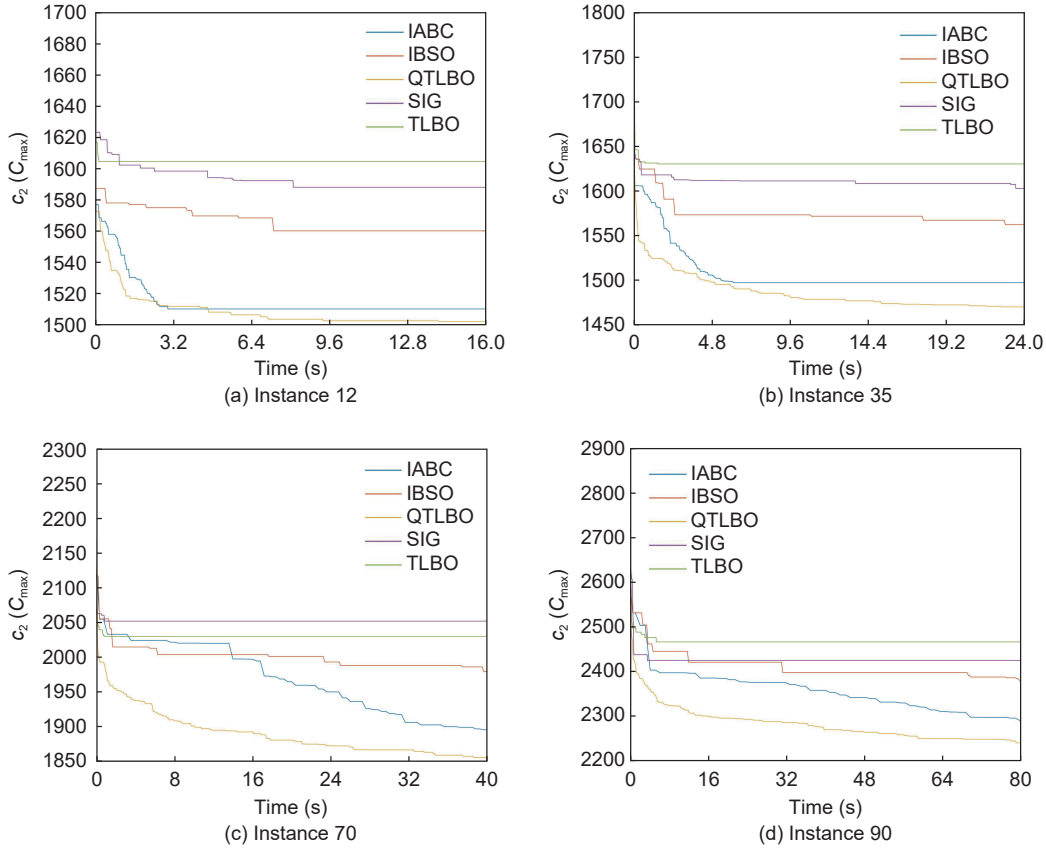

Fig. 5 Convergence curves of all algorithms on four instances.

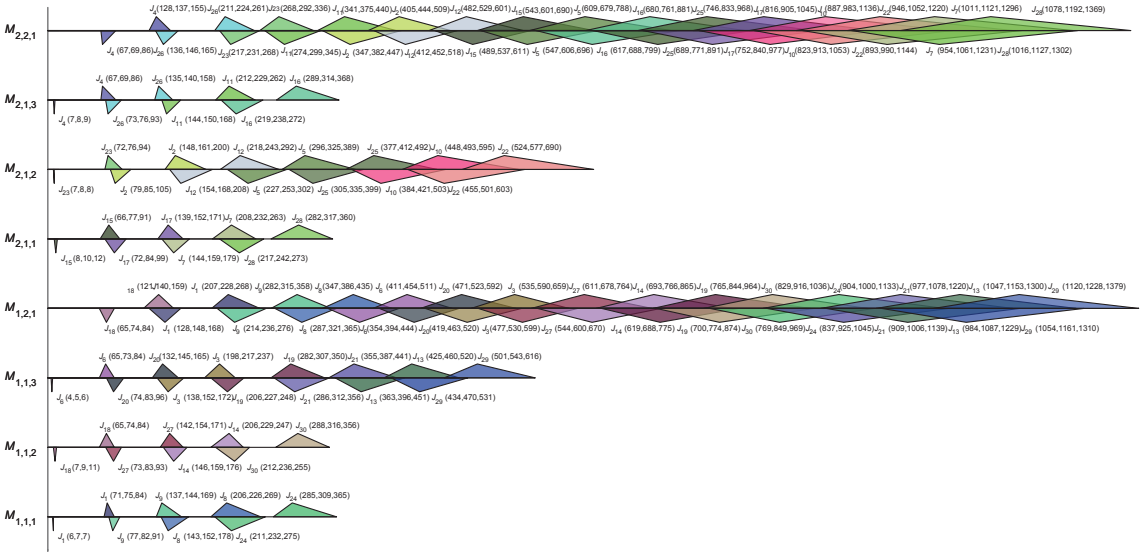

Fig. 6 Gantt chart of the best solution of instance 1 obtained by Q-learning-based teaching-learning based optimization (QTLBO).

References

[1] H. Allaoui and A. Artiba, Scheduling two-stage hybrid flow shop with availability constraints, *Comput. Oper. Res.*, vol. 33, no. 5, pp. 1399–1419, 2006.

[2] J. Yang, Minimizing total completion time in two-stage hybrid flow shop with dedicated machines, *Comput. Oper. Res.*, vol. 38, no. 7, pp. 1045–1053, 2011.

[3] S. Wang and M. Liu, A heuristic method for two-stage hybrid flow shop with dedicated machines, *Comput. Oper. Res.*, vol. 40, no. 1, pp. 438–450, 2013.

[4] S. J. Wang, X. D. Wang, F. Chu, and J. B. Yu, An energy-efficient two-stage hybrid flow shop scheduling problem in a glass production, *Int. J. Prod. Res.*, vol. 58, no. 8, pp. 2283–2314, 2020.

[5] X. Feng, F. Zheng, and Y. Xu, Robust scheduling of a two-stage hybrid flow shop with uncertain interval processing times, *Int. J. Prod. Res.*, vol. 54, no. 12, pp. 3706–3717, 2016.

[6] S. Wang and M. Liu, A genetic algorithm for two-stage no-wait hybrid flow shop scheduling problem, *Comput. Oper. Res.*, vol. 40, no. 4, pp. 1064–1075, 2013.

[7] Y. Tan, L. Mönch, and J. W. Fowler, A hybrid scheduling approach for a two-stage flexible flow shop with batch processing machines, *J. Scheduling.*, vol. 21, no. 2, pp. 209–226, 2017.

[8] B. Fan, W. Yang, and Z. Zhang, Solving the two-stage hybrid flow shop scheduling problem based on mutant firefly algorithm, *J. Amb. Intel. Hum. Comp.*, vol. 10, no. 3, pp. 979–990, 2019.

[9] G. M. Komaki, E. Teymourian, and V. Kayvanfar, Minimising makespan in the two-stage assembly hybrid flow shop scheduling problem using artificial immune systems, *Int. J. Prod. Res.*, vol. 54, no. 4, pp. 963–983, 2016.

[10] S. Wang and M. Liu, Two-stage hybrid flow shop scheduling with preventive maintenance using multi-objective tabu search method, *Int. J. Prod. Res.*, vol. 52, no. 5, pp. 1495–1508, 2014.

[11] S. W. Liu, J. Pei, H. Cheng, X. B. Liu, and P. M. Pardalos, Two-stage hybrid flow shop scheduling on parallel batching machines considering a job-dependent deteriorating effect and no-identical job sizes, *Appl. Soft Comput.*, vol. 84, p. 105701, 2019.

[12] O. Kheirandish, R. Tavakkoli-Moghaddam, and M. Karimi-Nasab, An artificial bee colony algorithm for a two-stage hybrid flowshop scheduling problem with multilevel product structures and requirement operations, *Int. J. Comput. Integ. M.*, vol. 28, no. 5, pp. 437–450, 2015.

[13] M. Rabiee, R. S. Rad, M. Mazinani, and R. Shafaei, An intelligent hybrid meta-heuristic for solving a case of no-wait two-stage flexible flow shop scheduling problem with unrelated parallel machines, *Int. J. Adv. Manuf. Tech.*, vol. 71, nos. 5–8, pp. 1229–1245, 2014.

[14] S. Lang, T. Reggelin, J. Schmidt, M. Müller, and A. Nahhas, NeuroEvolution of augmenting topologies for solving a two-stage hybrid flow shop scheduling problem: A comparison of different solution strategies, *Expert Syst. Appl.*, vol. 172, p. 114666, 2021.

[15] D. M. Lei and T. Wang, Solving distributed two-stage hybrid flowshop scheduling using a shuffled frog-leaping algorithm with memeplex grouping, *Eng. Optimiz.*, vol. 52, no. 9, pp. 1461–1474, 2020.

[16] J. C. Cai, R. Zhou, and D. M. Lei, Fuzzy distributed two-stage hybrid flow shop scheduling problem with setup time: Collabortive variable search, *J. Intell. Fuzzy Syst.*, vol. 38, no. 3, pp. 3189–3199, 2020.

[17] C. R. Vela, S. Afsar, and J. J. Palacios, Evolutionary tabu search for flexible due-date statisfaction in fuzzy job shop scheduling, *Comput. Oper. Res.*, vol. 119, p. 104931, 2020.

[18] Y. Dorfeshan, R. Tavakkoli-Moghaddam, S. M. Mousavi, and B. Vahedi-Nouri, A new weighted distance-based approximation methodology for flow shop scheduling group decisions under the interval-valued fuzzy processing time, *Appl. Soft Comput.*, vol. 91, p. 106248, 2020.

[19] L. Sun, L. Lin, M. Gen, and H. J. Li, A hybrid cooperative coevolution algorithm for fuzzy flexible job shop scheduling, *IEEE T. Fuzzy Syst.*, vol. 27, no. 5, pp. 1008–1022, 2019.

[20] J. Lin, Backtracking search based hyper-heuristic for the flexible job-shop scheduling problem with fuzzy processing time, *Eng. Appl. Artif. Intell.*, vol. 77, pp. 186–196, 2019.

[21] K. Z. Gao, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, An effective discrete harmony search algorithm for flexible job shop scheduling problem with fuzzy processing time, *Int. J. of Prod. Res.*, vol. 53, no. 19, pp. 5896–5911, 2015.

[22] J. Q. Li and Q. K. Pan, Chemical-reaction optimization for solving fuzzy job-shop scheduling problem with flexible maintenance activities, *Int. J. of Prod. Econ.*, vol. 145, no. 1, pp. 4–17, 2013.

[23] L. Wang, G. Zhou, Y. Xu, and M. Liu, A hybrid artificial bee colony algorithm for the fuzzy flexible job-shop scheduling problem, *Int. J. of Prod. Econ.*, vol. 51, no. 12, pp. 3593–3608, 2013.

[24] D. M. Lei, Co-evolutionary genetic algorithm for fuzzy flexible job shop scheduling, *Appl. Soft Comput.*, vol. 12, no. 8, pp. 2237–2245, 2012.

[25] K. Li, J. F. Chen, H. Fu, Z. H. Jia, and W. Z. Fu, Uniform parallel machine scheduling with fuzzy processing times under resource consumption constraint, *Appl. Soft Comput.*, vol. 82, p. 105585, 2019.

[26] Z. S. Shao, W. S. Shao, and D. C. Pi, Effective heuristics and metaheuristics for the distributed fuzzy blocking flow-shop scheduling problem, *Swarm and EComput.*, vol. 59, p. 100747, 2020.

[27] J. Zheng, L. Wang, and J. -J. Wang, A cooperative coevolution algorithm for multi-objective fuzzy distributed hybrid flow shop, *Knowl. Based Syst.*, vol. 194, p. 105536, 2020.

[28] R. V. Rao, V. J. Savsani, and D. P. Vakharia, Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems, *Computer-Aided Design*, vol. 43, no. 3, pp. 303–315, 2011.

[29] W. W. Lin, D. Y. Yu, C. Y. Zhang, X. Liu, S. Q. Zhang, Y. H. Tian, S. Q. Liu, and Z. P. Xie, A multi-objective teaching-learning-based optimization algorithm to scheduling in turning processes for minimizing makespan and carbon footprint, *J. Clean. Prod.*, vol. 101, pp. 337–347, 2015.

[30] J. Q. Li, Q. K. Pan, and K. Mao, A discrete teaching-learning-based optimisation algorithm for realistic flowshop scheduling problems, *Eng. Appl. Artif. Intell.*, vol. 37, pp. 279–292, 2015.

[31] D. M. Lei, L. Gao, and Y. L. Zheng, A novel teaching-learning-based optimization algorithm for energy-efficient scheduling in hybrid flow shop, *IEEE T. Eng. Manage.*, vol. 65, no. 2, pp. 330–340, 2018.

[32] Y. Xu, L. Wang, S. Y. Wang, and M. Liu, An effective teaching-learning-based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time, *Neurocomputing*, vol. 148, pp. 260–268, 2015.

[33] W. S. Shao, D. C. Pi, and Z. S. Shao, A hybrid discrete teaching-learning based meta-heuristic for solving no-idle flow shop scheduling problem with total tardiness criterion, *Comput. Oper. Res.*, vol. 94, pp. 89–105, 2018.

[34] A. Mishra and D. Shrivastava, A TLBO and a Jaya heuristics for permutation flow shop scheduling to minimize the sum of inventory holding and batch delay costs, *Comput. Ind. Eng.*, vol. 124, pp. 509–522, 2018.

[35] R. Buddala and S. S. Mahapatra, Two-stage teaching-learning-based optimization method for flexible job-shop scheduling under machine breakdown, *Int. J. of Adv. Manuf. Tech.*, vol. 100, nos. 5–8, pp. 1419–1432, 2019.

[36] D. M. Lei, B. Su, and M. Li, Cooperated teaching-learning-based optimisation for distributed two-stage assembly flow shop scheduling, *Int. J. of Prod. Res.*, vol. 59, no. 23, pp. 7232–7245, 2020.

[37] R. Chen, B. Yang, S. Li, and S. Wang, A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem, *Comput. Ind. Eng.*, vol. 149, p. 106778, 2020.

[38] Z. Cao, C. Lin, M. Zhou, and R. Huang, Scheduling semiconductor testing facility by using cuckoo search algorithm with reinforcement learning and surrogate modeling, *IEEE T. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 825–837, 2018.

[39] Z. Cao, C. Lin, and M. Zhou, A knowledge-based cuckoo search algorithm to schedule a flexible job shop with sequencing flexibility, *IEEE T. Autom. Sci. Eng.*, vol. 18, no. 1, pp. 56–69, 2019.

[40] H. Öztop, M. F. Tasgetiren, L. Kandiller, and Q. K. Pan, A novel general variable neighborhood search through Q-learning for no-idle flowshop scheduling, in *Proc. IEEE Congress on Evolutionary Computation (CEC)*, Glasgow, UK, 2020, pp. 1–8.

[41] J. Wang, D. M. Lei, and J. C. Cai, An adaptive artificial bee colony with reinforcement learning for distributed three-stage assembly scheduling with maintenance, *Appl. Soft. Comput.*, vol. 117, p. 108371, 2021.

[42] L. Wang, Z. X. Pan, and J. J. Wang, A review of reinforcement learning based intelligent optimization for manufacturing scheduling, *Complex. Syst. Mod. Sim.*, vol. 1, no. 4, pp. 257–270, 2021.

[43] M. Karimi-Mamaghan, M. Mohammadi, P. Meyer, A. M. Karimi-Mamaghan, and E. -G. Talbi, Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art, *Eur. J. Oper. Res.*, vol. 296, no. 2, pp. 393–422, 2022.

[44] Q. L. Zhou, A novel movies recommendation algorithm based on reinforcement learning with DDPG policy, *Int. J. Intell. Comput. Cybernetics*, vol. 13, no. 1, pp. 67–79, 2020.

[45] M. H. Pandit, R. N. Mir, M. A. Chishti, Adaptive task scheduling in IoT using reinforcement learning, *Int. J. Intell. Comput. Cybernetics*, vol. 13, no. 2, pp. 261–282, 2020.

[46] J. C. H. Watkins and P. Dayan, Q-learning, *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, 1992.

[47] K. C. Ying and S.W. Lin, Minimizing makespan for the distributed hybrid flowshop scheduling problem with multiprocessor tasks, *Expert. Syst. Appl.*, vol. 92, pp. 132–141, 2018.

[48] Y. L. Li, X. Y. Li, L. Gao, and L. L. Meng, An improved artificial bee colony algorithm for distributed heterogeneous hybrid flowshop scheduling problem with sequence-dependent setup times, *Comput. Ind. Eng.*, vol. 147, p. 106638, 2020.

[49] J. Q. Li, J. K. Li, L. J. Zhang, H. Y. Sang, Y. Y. Han, and Q. D. Chen, Solving type-2 fuzzy distributed hybrid flowshop scheduling using an improved brain storm optimization algorithm, *Int. J. Fuzzy Syst.*, vol. 23, pp. 1194–1212, 2021.

Deming Lei received the MS degree in applied mathematics from Xi'an Jiaotong University, Xi'an, China, in 1996 and the PhD degree in automation science and engineering from Shanghai Jiao Tong University, Shanghai, China, in 2005. He is currently a professor with the School of Automation, Wuhan University of Technology, Wuhan, China. He has published over 100 journal papers. His current research interests include intelligent system optimization and control, and production scheduling.

Bingjie Xi received the bachelor degree from Wuhan University of Technology in 2019. She is currently pursuing the PhD degree in Wuhan University of Technology. Her current reserch interests are intelligent optimizatio and scheduling. She has published a paper on fuzzy distributed two-stage hybrid flow shop scheduling in the *Journal of Intelligent and Fuzzy Systems*.