A Reinforcement-Learning-Based 3-D Estimation of Distribution Algorithm for Fuzzy Distributed Hybrid Flow-Shop Scheduling Considering On-Time-Delivery

Libao Deng, *Member, IEEE*, Yuanzhu Di, and Ling Wang, *Member, IEEE*

***Abstract*—With the increasing level of mass-customization and globalization of competition, environmentally friendly production scheduling for distributed manufacturing considering customer satisfaction has received growing attention. Meanwhile, uncertain scheduling is becoming a force to be considered within intelligent manufacturing industries. However, little research has been found that surveyed the uncertain distributed scheduling considering both energy consumption and customer satisfaction. In this article, the fuzzy distributed hybrid flow-shop scheduling problem considering on-time delivery (FDHFSP-OTD) is addressed, and a 3-D estimation of distribution algorithm (EDA) with reinforcement learning (RL) is proposed to minimize the makespan and total energy consumption while maximizing delivery accuracy. First, two heuristics and a random method are designed and used cooperatively for initialization. Next, an EDA with a 3-D probability matrix is innovated to generate offspring. Then, a biased decoding method based on *Q*-learning is proposed to adjust the direction of evolution self-adaptively. Moreover, a local intensification strategy is employed for further enhancement of elite solutions. The effect of major parameters is analyzed and the best combination of values is determined through extensive experiments. The numerical results prove the effectiveness of each specially designed strategy and method, and the comparisons with existing algorithms demonstrate the high-potential of the 3D-EDA/RL in solving the FDHFSP-OTD.**

***Index Terms*—Distributed hybrid flow shop scheduling problem, estimation of distribution algorithm (EDA), fuzzy scheduling, on-time delivery, *Q*-learning.**

Manuscript received 1 August 2023; revised 17 October 2023; accepted 19 November 2023. Date of publication 21 December 2023; date of current version 17 January 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62176075; in part by the National Key Research and Development Program of China under Grant 2022YFB3304002; and in part by the Natural Science Foundation of Shandong Province, China, under Grant ZR2021MF063. This article was recommended by Associate Editor X. Liu. *(Corresponding authors: Libao Deng; Ling Wang.)*

Libao Deng and Yuanzhu Di are with the School of Information Science and Engineering, Harbin Institute of Technology, Weihai 264209, China (e-mail: denglibao_paper@163.com).

Ling Wang is with the Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: wangling@tsinghua.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCYB.2023.3336656.

Digital Object Identifier 10.1109/TCYB.2023.3336656

I. Introduction

With the persistent trend of mass-customization and the growth in the variety of products, a growing number of manufacturing companies have focused on improving customer satisfaction [1]. For decades, the tardiness criterion has been used to measure the customer satisfaction [2]. However, in real-world situations, not only tardiness but also earliness have an influence on service quality. Extra warehousing cost and unexpected deterioration of the product take place when the order is completed ahead of the schedule, while a delayed delivery may lead to weak customer retention and brand reputation [3]. Thus, the accuracy of delivery plays a fundamental role in competitive markets.

To improve the delivery accuracy, many studies have focused on scheduling problems with earliness and tardiness penalties, also known as just-in-time (JIT) production. Most of the early studies on JIT production paid attention to single-machine scheduling [4], [5], [6]. As the manufacturing process scales up, the job-shop scheduling problem (JSP) with earliness and tardiness costs has emerged [7], [8], [9]. However, most recent attention has focused on JIT production in the flow-shop. In [10], a total of five objectives, including earliness and tardiness penalties, in a steelmaking casing system are optimized. Tardiness in hybrid flowshop manufacturing systems is addressed in [11] and [12] has focused on distributed flowshop group scheduling problems (DFGSPs).

As the most energy-consuming and CO_2 emission entity, over 70% of power plants in China are coal-fired [13]. Recently, a growing number of manufacturers have realized the effectiveness of energy-efficient scheduling for reducing carbon emissions [14]. Meanwhile, distributed manufacturing has been applied widely in various fields, including automotive, steel-making and chemical processing [15]. During the past few decades, various evolutionary algorithms, such as the iterated greedy algorithm [16], Jaya algorithm [17], and memetic algorithm [18], have been proposed to solve distributed shop scheduling problems (DSSPs) with different constraints. In addition, DSSPs under various environments have been studied, such as distributed parallel machine scheduling problems [19], DFGSPs [20], distributed heterogeneous hybrid flowshop scheduling problems (FSP) [21], distributed no-wait FSP [22], and distributed hybrid differentiation FSP [23].

Moreover, with the characteristics of distributed flow-shop and parallel machines, the distributed hybrid flowshop scheduling problem (DHFSP) has recently emerged as a practical production model recently. Both Jiang et al. [24] and Li et al. [25] scheduled the energy-aware DHFSP with multiprocessor tasks with multiobjective evolutionary algorithms.

Uncertainties always exist in real-life manufacturing. Solutions to uncertain scheduling problems can offer decision-makers deeper insight into the production as compared to those of deterministic scheduling problems. As a common uncertain scheduling problem, the fuzzy flexible job-shop scheduling problem (FFJSP) has drawn considerable attention. In most of the existing research, the processing time is regarded as a triangular fuzzy number [26], [27]. In addition, the customer satisfaction with the FFJSP is considered in some research [28], [29]. Recently, the type-2 fuzzy logic system has been used to represent processing time [30], [31]. Furthermore, other categories of scheduling problems with fuzzy elements have recently been studied, such as flow-shop group scheduling [32] and DHFSP [33], [34]. Cai et al. [33] investigated the fuzzy distributed two-stage hybrid flowshop scheduling problem with setup time regarding the total agreement index and makespan as optimization objectives. The Q-learning method is used in [34] to minimize makespan. However, there has been little discussion about the fuzzy DHFSP considering both customer satisfaction and energy consumption, of which the representation of customer satisfaction with fuzzy processing time needs to be settled, apart from subproblems such as factory assignment, machine selection and job sequencing. Thus, effective algorithms need to be developed in order to solve such a challenging problem.

By estimating the probability distribution of the fittest individuals and sampling the induced model, the estimation of distribution algorithm (EDA) completes the evolution from one generation to the next one [35]. The utilization of the probability distribution is able to break through the special value restriction of JSPs; thus, the algorithm can capture the essence of better scheduling plans. In the past few decades, EDAs have been extensively used in diverse scheduling problems, including FSPs [36], FJSPs [37] and distributed assembly permutation FSPs [38]. As an aspect of machine learning, reinforcement learning (RL) has attracted considerable attention. Recently, some techniques from RL have been used to solve scheduling problems [39], [40], [41], [42]. Q-learning is employed in [43] to select an appropriate low-level heuristic for solving energy-efficient distributed blocking FSPs. RL is used for parameter selection in [30]. A parameter adaptation strategy based on Q-learning is used in [27]. Wang and Wang [44] proposed an RL-based policy agent to refine the solutions by selecting an appropriate improvement operator. Overall, the combination of RL and EAs has offered a promising method for solving scheduling problems. Therefore, this study attempts to contribute to this growing area of research by embedding RL technology into the EDA.

The major objective of this study is to investigate the scheduling methods based on RL for fuzzy DHFSPs considering delivery accuracy. To minimize both fuzzy makespan and total energy consumption, while maximizing the delivery accuracy, a 3-D EDA based on reinforcement learning (3D-EDA/RL) is proposed. The delivery accuracy is calculated via fuzzy relative entropy (FRE), and the lower bounds of all objectives are combined as the reference point. Two heuristics are designed for hybrid initialization, and a 3-D probability matrix and a self-adaptive biased decoding method based on Q-learning are proposed. Additionally, a local intensification strategy is designed for further enhancement. Finally, extensive comparisons and tests are carried out to verify the performance of the proposed 3D-EDA/RL in solving fuzzy distributed hybrid flow-shop scheduling problem considering on-time delivery (FDHFSP-OTD).

The remainder of this article is divided into four parts. Section II describes FDHFSP-OTD in detail and provides the calculation of the reference point. The mathematical model and the proposed 3D-EDA/RL is introduced in Section III. The numerical results and discussion are shown in Section IV. Finally, the conclusion gives a brief summary and critique of the findings in Section V.

II. Fuzzy Distributed Hybrid Flow-Shop Scheduling Problem Considering On-Time Delivery

A. Fuzzy Set

Fuzzy set is a mathematical model of vague quantitative data, also known as an extension and gross oversimplification of classical sets [45]. Generally, a fuzzy set $\tilde{A}$ can be represented mathematically as $\tilde{A} = \{(x, \mu_{\tilde{A}})|x \in U\}$, where $\mu_{\tilde{A}}(x)$ is the degree of membership of x in $\tilde{A}$. Triangular fuzzy numbers (TFN) are widely used to represent uncertain or incomplete information in decision-making and expert systems [46], represented by a triplet $\tilde{A} = (a_1, a_2, a_3)$, where a_1 and a_3 denote the minimal and maximal value of x, a_2 is the most likely value of x.

Let $\tilde{A} = (a_1, a_2, a_3)$ and $\tilde{B} = (b_1, b_2, b_3)$ be two TFNs. According to $\tilde{A} \pm \tilde{B} = (a_1 \pm b_1, a_2 \pm b_2, a_3 \pm b_3)$, the sum and difference of them are also TFNs. Besides, the relationship between $\tilde{A}$ and $\tilde{B}$ can be considered as $\tilde{A} > \tilde{B}$ if either of the following three conditions is met, where $c_1(\tilde{A}) = (a_1 + 2 \times a_2 + a_3)/4$, $c_2(\tilde{A}) = a_2$ and $c_3(\tilde{A}) = a_3 - a_1$.

1) *Condition 1:* $c_1(\tilde{A}) > c_1(\tilde{B})$.
2) *Condition 2:* $c_1(\tilde{A}) = c_1(\tilde{B})$ and $c_2(\tilde{A}) > c_2(\tilde{B})$.
3) *Condition 3:* $c_1(\tilde{A}) = c_1(\tilde{B})$ and $c_2(\tilde{A}) = c_2(\tilde{B})$ and $c_3(\tilde{A}) > c_3(\tilde{B})$.

In this article, the fuzzy processing time is represented as TFN. With the basis above, the relationship between fuzzy makespan obtained by different scheduling plans can be determined through comparison.

B. Fuzzy Relative Entropy

Suppose $\tilde{X} = \{\mu_1(X), \ldots, \mu_n(X)\}$ and $\tilde{Y} = \{\mu_1(Y), \ldots, \mu_n(Y)\}$ are two n-dimensional fuzzy sets, where $\mu_i(X)$ and $\mu_i(Y)$ are the ith membership values of fuzzy sets $\tilde{X}$ and $\tilde{Y}$, respectively. The similarity relationship between $\tilde{X}$ and $\tilde{Y}$ can be measured by the FRE coefficient, which can be calculated as follows.

First, the information entropies of $\tilde{X}$ and $\tilde{Y}$, denoted by $I(X)$ and $I(Y)$, are calculated as

$$I(\cdot) = -N\sum_{i=1}^{n}\mu_i(\cdot)\ln\mu_i(\cdot) + [1-\mu_i(\cdot)]\ln[1-\mu_i(\cdot)] \quad (1)$$

where $N = 1/(n\ln 2)$ denotes the normalization factor.

Next, the partial entropy can be calculated as $I_Y(X) = -\sum_{i=1}^{n}\mu_i(Y)\ln\mu_i(X) + [1-\mu_i(Y)]\ln[1-\mu_i(X)]$ and $I_X(Y) = -\sum_{i=1}^{n}\mu_i(X)\ln\mu_i(Y) + [1-\mu_i(X)]\ln[1-\mu_i(Y)]$. Then, the FRE between $\tilde{X}$ and $\tilde{Y}$ can be calculated as $I(X,Y) = I_Y(X) + I_X(Y)$. Finally, the FRE coefficient between $\tilde{X}$ and $\tilde{Y}$ is calculated as

$$I_e(X,Y) = \frac{1}{N}\frac{I(X)+I(Y)}{I(X,Y)} \quad (2)$$

where $0 \le I_e(X,Y) = I_e(Y,X) \le 1$. The value of $I_e(X,Y)$ is equal to 1 if and only if $\tilde{X}$ is the same as $\tilde{Y}$.

As a result, the larger the value of I_e is, the higher similarity exists between $\tilde{X}$ and $\tilde{Y}$.

C. Problem Description

The notation used for problem description and algorithm introduction in this article is represented as follows.

Notation	Description
F	Number of factories.
f	Index of factories, $f \in \{1,2,\cdots,F\}$.
n	Number of jobs.
j	Index of jobs, $j \in \{1,2,\cdots,n\}$.
s	Number of stages.
k	Index of stages, $k \in \{1,2,\cdots,s\}$.
$m_{f,k}$	Number of machines for stage k in factory f.
i	Index of machines, $i \in \{1,2,\cdots,m_{f,k}\}$.
$\widetilde{P}_{j,k}$	Fuzzy processing time of job j at stage k, $\widetilde{P}_{j,k} = (P^1_{j,k}, P^2_{j,k}, P^3_{j,k})$.
$EP_{f,k,i}$	Energy consumption of machine i per unit time at stage k in factory f in processing mode.
EI	Energy consumption per unit time in idle mode.
$\widetilde{D}$	Fuzzy due date, $\widetilde{D} = (D^1, D^2, D^3)$.
$\widetilde{ECP}_{f,k,i}$	Fuzzy energy consumption of machine i in processing mode at stage k in factory f.
$\widetilde{ECI}_{f,k,i}$	Fuzzy energy consumption of machine i in idle mode at stage k in factory f.
$\widetilde{S}_{j,k}$	Fuzzy begin time of processing job j at stage k.
$\widetilde{C}_{j,k}$	Fuzzy completion time of processing job j at stage k.
$\widetilde{MS}$	Fuzzy makespan.
$\widetilde{TEC}$	Fuzzy total energy consumption.
A_d	Delivery accuracy.
$x_{j,f}$	Binary variable whose value equals to 1 when job j is assigned to factory f or 0 otherwise.
$y_{j,f,k,i}$	Binary variable whose value equals to 1 when job j is assigned to machine i at stage k in factory f or 0 otherwise.
$z_{j,j',f,k,i}$	Binary variable whose value equals to 1 when job j is processed before j' on machine i at stage k in factory f or 0 otherwise.

Generally, DHFSP means a group of n jobs, each of which possesses s stages, have to be completed by a set of machines, which distribute in F different factories. Each job needs to be allotted to one of the factories before processing. As a result, DHFSP can be considered to consist of three subproblems, including factory allocation for each job, machine assignment in each factory, and processing sequence on each machine.

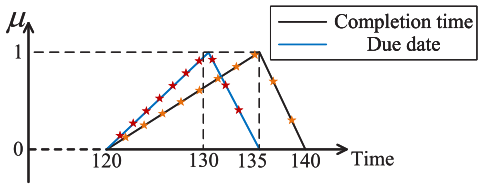


Fig. 1. Fuzzy completion time and fuzzy due date.

On account of the machine wear and the different situation of jobs in real-life manufacturing, the actual processing time may float around the reference value provided by the nameplate. Thus, the processing time can be described by a mathematical model of vague quantitative, namely, fuzzy processing time. Due to the fuzzy processing time, the makespan and total energy consumption cannot be represented as fixed numbers. Therefore, in fuzzy DHFSP, both the makespan and total energy consumption are TFNs, denoted as $\widetilde{MS}$ and $\widetilde{TEC}$, respectively.

According to the minimum value of makespan, the delivery time provided by costumers, and the maximum earliness allowed by product or storage, a schedule with the latest start time of processing can be obtained, which is capable of minimizing the occupation of machines while maximizing the delivery accuracy. However, in most of the real-life circumstances, both the due dates given by the costumers and the maximum earliness allowed are time horizons rather than exact dates. As a result, due dates can also be represented as TFNs. In this article, the delivery accuracy is calculated on the base of FRE coefficient between fuzzy makespan $\widetilde{MS}$ and fuzzy due date $\widetilde{D}$. Since both $\widetilde{MS}$ and $\widetilde{TEC}$ are minimal optimization objectives, while a larger value of $I_e(MS,D)$ means better delivery, the value of A_d is calculated as $A_d = 1 - I_e(MS,D)$.

For example, the $\widetilde{MS}$ and $\widetilde{D}$ of an instance are shown in Fig. 1. Each fuzzy set is sampled ten times with equal intervals in time domain, which are represented as red and yellow stars. Thus, the x coordinate of the first sampled node of the fuzzy due date, shown as the far left red star, can be calculated as $(135-120/11)+120 = 121.36$, while its y coordinate is calculated as $(1/130-120)\times(121.36-120) = 0.136$. Thus, the $\mu_i(\cdot)$ value of the certain node is 0.136. Similarly, the $\mu_i(\cdot)$ value of its neighbor can be calculated as $(1/130-120)\times(121.36-120)\times 2 = 0.272$. Therefore, the FRE coefficient between these two fuzzy sets can be calculated as $I_e(MS,D) = 0.9367$. Then, the delivery accuracy is calculated as $A_d = 1 - I_e(MS,D) = 0.0633$.

The mixed integer linear programming model for FDHFSP-OTD minimizing $\widetilde{MS}$, $\widetilde{TEC}$, and A_d is formulated as follows:

$$\min \quad \left(\widetilde{MS}, \widetilde{TEC}, A_d\right) \quad (3)$$

$$\text{Subject to:} \quad \sum_{f=1}^{F} x_{j,f} = 1 \quad \forall j \quad (4)$$

$$\sum_{i=1}^{m_{f,k}} y_{j,f,k,i} = 1 \quad \forall f,j,k \quad (5)$$

$$\widetilde{S}_{j,1} \ge 0 \quad \forall j \quad (6)$$

$$\widetilde{C}_{j,k} = \widetilde{S}_{j,k} + \sum_{f=1}^{F}\sum_{i=1}^{m_{f,k}} \widetilde{P}_{j,k} \quad \forall j,k \tag{7}$$

$$\widetilde{S}_{j,k+1} \geq \widetilde{C}_{j,k} \quad \forall j \; \forall k \in \{1,2,\ldots,s-1\} \tag{8}$$

$$z_{j,j',f,k,i} + z_{j',j,f,k,i} \leq 1 \quad \forall j \neq j', f, k, i \in \{1,2,\ldots,m_{f,k}\} \tag{9}$$

$$z_{j,j',f,k,i} + z_{j',j,f,k,i} \geq y_{j,f,k,i} + y_{j',f,k,i} - 1 \quad \forall j \neq j', f, k, i \tag{10}$$

$$\widetilde{S}_{j',k} - \widetilde{C}_{j,k} + 10^6 \times (3 - y_{j,f,k,i} - y_{j',f,k,i} - z_{j,j',f,k,i}) \geq 0 \quad \forall j \neq j', f, k, i \tag{11}$$

$$\widetilde{MS} \geq \widetilde{C}_{j,s} \quad \forall j \tag{12}$$

$$\widetilde{ECP}_{f,k,i} = \sum_{j=1}^{n} y_{j,f,k,i} \times EP_{f,k,i} \times \widetilde{P}_{j,k} \quad \forall f, k, i \in \{1,2,\ldots,m_{f,k}\} \tag{13}$$

$$\widetilde{ECI}_{f,k,i} = EI \times [\max_{j}(\widetilde{C}_{j,k} \times y_{j,f,k,i}) - \min_{j}(\widetilde{S}_{j,k+1} \times y_{j,f,k,i}) - \sum_{j=1}^{n} y_{j,f,k,i} \times \widetilde{P}_{j,k}] \quad \forall f, k, i \in \{1,2,\ldots,m_{f,k}\} \tag{14}$$

$$\widetilde{TEC} = \sum_{f=1}^{F}\sum_{k=1}^{s}\sum_{i=1}^{m_{f,k}} (\widetilde{ECP}_{f,k,i} + \widetilde{ECI}_{f,k,i}) \tag{15}$$

where (4) demonstrates the three optimization objectives of this article, (11) ensures that each job can only be allotted to one factory, (5) assures that each job can only be processed by one machine for each stage, (6) limits the starting time of each job is later than time 0, (7) and (8) provide the description of fuzzy completion time and guarantee that the processing sequence of each job should follow its natural order, (9)–(11) ensure that the largest number of jobs being processed by each machine is one, (12) describes the calculation of fuzzy makespan, while the fuzzy total energy consumption is calculated via (13)–(15).

D. Lower Bounds

Reference points are widely used in the calculation of evaluation metrics of multiobjective optimization algorithms, such as generational distance (GD), inverted generational distance (IGD), convergence metric, and hypervolume (HV). The presetting method of reference points plays a significant role in algorithm design. In this article, lower bounds of three objectives are combined and used as the reference point.

According to the calculation of global makespan lower bounds on HFSP proposed in [47] and the mathematical proof of the connection between the lower bounds on DHFSP and HFSP presented in [44], the global fuzzy makespan lower bounds of FDHFSP, denoted as $\widetilde{LB}_{MS} = (LB^1_{MS}, LB^2_{MS}, LB^3_{MS})$, can be calculated as $\widetilde{LB}_{MS} = \max\{\widetilde{LB}(0), \max_k \widetilde{LB}(k)\}$, where $\widetilde{LB}(0)$ is the longest total fuzzy processing time of each job. Besides, $\widetilde{LB}(k)$ denotes the local fuzzy makespan lower bound of each stage k, which is obtained via

$$\widetilde{LB}(k) = \frac{1}{M(k)}\left[\sum_{y=1}^{M(k)} \widetilde{LSA}(y,k) + \sum_{j=1}^{n} \widetilde{P}_{j,k} + \sum_{y=1}^{M(k)} \widetilde{RSA}(y,k)\right] \tag{16}$$

where $M(k)$ is the total number of available machines at stage k in all factories. For each stage k, $\widetilde{LSA}(y,k)$ is the yth element in $\widetilde{LSA}(k)$, which denotes the $\widetilde{LS}(j,k)$ values sorting in ascending order. $\widetilde{LS}(j,k)$ is the sum of fuzzy processing times for job j from stage 1 up through stage $(k-1)$. $\widetilde{RSA}(y,k)$ is defined in the same manner and $\widetilde{RS}(j,k)$ means the sum of fuzzy processing times for job j starting from stage $(k+1)$.

The global lower bounds of TEC, indicated as $\widetilde{TEC} = (LB^1_{TEC}, LB^2_{TEC}, LB^3_{TEC})$ can be obtained by assigning each job to the certain machine with minimal energy consumption all over the factories for each stage while no energy consumption during the standby mode is taken into consideration.

Ideally, the end product is delivered accurately and thus the lower bound of delivery accuracy is set as 0. Therefore, the reference point of FDHFSP can be represented as $[LB^3_{MS}, LB^3_{TEC}, 0]$.

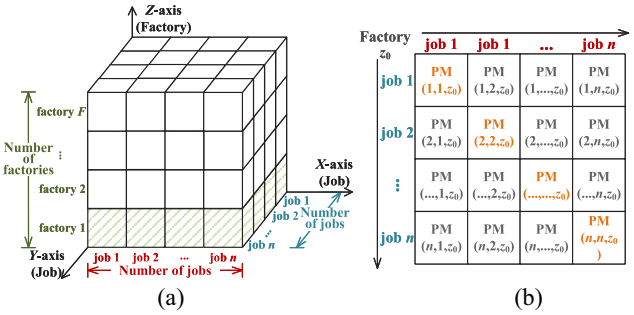
III. 3D-EDA/RL for FDHFSP-OTD

The classical EDAs refer to stochastic optimization methods in which the evolutionary direction of the population is guided by constructing and analyzing probabilistic models of promising candidate solutions [48]. The 3-D probability matrix is capable of making an explicit connection between the traditional EDA and FDHFSP, while the inclusion of RL can dynamically adjust the direction of evolution dynamically. In this section, four vital components in the design of 3D-EDA/RL, including hybrid initialization, a 3-D probability matrix, RL-based biased decoding and local intensification, are presented in detail.

A. Solution Representation

According to the description of FDHFSP-OTD, its solution should cover the allocation of factory for each job, the processing sequence of jobs in each factory, and the machine selection for each stage of each job. In this article, the permutation encoding method is adopted to represent the allocation of factory for each job and the processing sequence of jobs in each factory, while the machine selection is obtained during decoding operation, which is proposed in Section III-E. The solution of FDHFSP-OTD can be represented as $\{n_1,\ldots,n_F,\pi_1,\ldots,\pi_F\}$, where $n_f, f \in \{1,2,\ldots,F\}$ denotes the number of jobs allocated to factory f, while $\pi_f, f \in \{1,2,\ldots,F\}$ represents the processing sequence of jobs in factory f.

For example, a feasible solution of an instance with three factories and nine jobs can be represented as a set of numbers: 2 4 3 9 1 3 6 4 8 2 7 5, from which can be seen that job 9 and job 1 are processed in factory 1 sequentially, while job 3, job 6, job 4, and job 8 are assigned to factory 2 sequentially, and the first job processed in factory 3 is job 2, followed by job 7 and job 5.



(a) (b)

Fig. 2. Structure of 3-D probability matrix. (a) Whole 3-D version. (b) 2-D version of a certain factory.

B. Hybrid Initialization

In the past few decades, the Nawaz–Enscore–Ham (NEH) heuristic proposed in [49] has been proven to be effective for solving FSPs, and its variants have been frequently used for solving DFSPs and DHFSPs recently [16], [17], [43], [44]. Therefore, the NEH heuristic is used as the basis of the two proposed initialization methods.

A NEH heuristic with biased optimization (NEHBO) is designed to produce initial solutions with lower $\widetilde{MS}$ and $\widetilde{TEC}$. First, insert each job in the initial job sequence, which is randomly generated, into all positions in all factories. Then, the $\widetilde{MS}$ and $\widetilde{TEC}$ of each insertion are calculated according to the biased decoding method. Finally, the insertion with the lowest $\widetilde{MS}$ and $\widetilde{TEC}$ is selected. Another NEH heuristic based on lower bounds (NEHLBs) aims at balancing the production pressure among different factories. For each job, the current lower bounds of each factory are calculated, and the certain one with the smallest value is chosen. Then, a possible position in a certain factory is randomly selected for the job.

The whole population is divided into three equal parts, which are generated through NEHBO, NEHLB and random methods, respectively. The proposed hybrid initialization method is expected to balance the diversity and quality of the initial population.

C. Three-Dimensional Probability Matrix

EDA has been used for solving different kinds of job-shop scheduling problems in the past few years [36], [37], [38]. However, the machine assignment and job sequence are usually represented by different multiple probability matrices, which easily cause loss of useful information related to better performed solutions. Therefore, a 3-D probability matrix is designed to describe the information supported by the solutions in this article.

The structure of the proposed 3-D probability matrix is illustrated in Fig. 2 (a), in which both x-axis and y-axis represent the jobs, while the z-axis denotes the factories. Therefore, the xy-plane is a square with a side length of n, as shown in Fig. 2(b). Positions (x_0, y_0, z_0) in the 3-D probability matrix with $x_0, y_0 \in \{1, 2, \ldots, n\}, x_0 \neq y_0$ and $z_0 \in \{1, 2, \ldots, F\}$ represent the priority of different jobs in the same factory, while positions (x_0, x_0, z_0) denote the probability of factory allocation of each job. To be specific, $PM(1, 1, z_0)$ is the probability of job 1 being assigned to factory z_0, $PM(1, 2, z_0)$ denotes the probability that both jobs 1 and 2 are assigned to factory z_0 while job 1 is processed before job 2 at stage 1.

Algorithm 1: Update Strategy of 3-D Probability Matrix

Input: The set of non-dominated solutions $E(g)$ of generation g;
Output: The updated 3D probability matrix of the current generation PM_g.

```
for e = 1, 2, ..., N_E (N_E is the size of E(g)) do
    for f = 1, 2, ..., F do
        for j_f = 1, 2, ..., n_f (n_f is the number of jobs assigned to factory f) do
            δ(j_f, j_f, f) = 1;
            for j'_f = 1, 2, ..., n_f, j'_f ≠ j_f do
                if job j'_f is processed before job j_f then
                    δ(j'_f, j_f, f) = 1;
                end
            end
        end
    end
end
Update the 3D population matrix PM_g according to Eq. (18);
Normalize the 3D population matrix PM_g according to Eq. (19);
```

In this article, the 3-D probability matrix PM is updated according to the nondominated solutions in each generation, the main steps are represented in Algorithm 1. The initial value of $PM_0(x_j, y_{j'}, f)$ in 3-D probability matrix is defined as (17), where $j, j' \in \{1, 2, \ldots, N\}$ and $f \in \{1, 2, \ldots, F\}$

$$PM_0(x_j, y_{j'}, f) = \begin{cases} 0.5, & \text{if } j \neq j' \\ 1, & \text{if } j = j'. \end{cases} \tag{17}$$

A 3-D incremental matrix Θ_e is obtained according to each elite individual pop_e in $E(g)$, $e \in 1, 2, \ldots, N_E$. The value of $\Theta_e(x_j, y_{j'}, f)$, where $j, j' \in \{1, 2, \ldots, n_F\}$ and $j_1 \neq j_2$, is set as 1 if job j is processed before j' in factory f, and 0 otherwise, while the value of $\Theta_e(x_j, y_j, f)$ is set as 1 if job j is allocated to factory f.

Then, the 3-D probability matrix is updated and normalized via (19) and (18), respectively

$$PM_g = \beta \times \frac{\sum_{e=1}^{N_E} \Theta_e}{N_E} + (1 - \beta) \times PM_{g-1} \tag{18}$$

$$PM_g(x, y, z) = \begin{cases} \frac{PM_g(x,y,z)}{\sum_{z=1}^{F} PM(x,y,z)}, & \text{if } x = y \\ \frac{PM_g(x,y,z)}{\sum_{x=1}^{n}\sum_{y=1}^{n} PM(x,y,z)}, & \text{if } x \neq y. \end{cases} \tag{19}$$

D. Offspring Generation

After the 3-D probability matrix is updated, offspring are generated by sampling from PM according to roulette wheel selection. The main method is represented in Algorithm 2.

The two main steps of the sampling operation are selecting the factory for each job, and sorting jobs in each factory. Specifically, an example is used to illustrate the main steps of offspring generation, which is shown in Fig. 3. When selecting the factory for each job j, the one with $\max_f PM(j, j, f)$ is chosen. Then, the roulette wheel selection strategy is used to sequence jobs allocated to the same factory according to the value of $PM(j, j', f)$.

Algorithm 2: Sampling Strategy for Offspring Generation

Input: The updated 3-D probability matrix PM_g, population size NP;
Output: The offspring population.

```
for index = 1, 2, ..., NP do
    for j = 1, 2, ..., n do
        The factory with max_f PM(j, j, f) is selected;
    end
    for f = 1, 2, ..., F do
        Record the number of jobs in factory f as n_f;
        for index_f = 1, 2, ..., n_f do
            Choose a job via roulette wheel selection strategy;
            Remove the chosen job;
            Normalize the processing priority value of the
            remaining jobs;
        end
    end
end
```

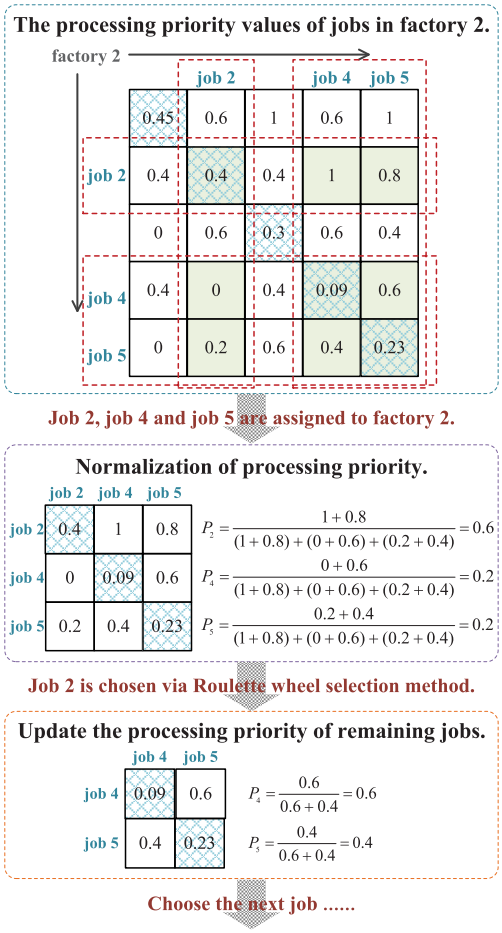


Fig. 3. Main steps of offsprings generating.

E. RL-Based Biased Decoding

Given the factory assignment and job sequence, the decoding operation is needed to obtain the complete schedule, thus the quality of solution highly depends on the design of decoding strategies. In this article, an importance vector $\mathbf{w} = [w_1, w_2]$, where $w_1 + w_2 = 1$, is used in the decoding strategy to balance the optimization of $\widetilde{MS}$ and $\widetilde{TEC}$.

Algorithm 3: Basic Biased Decoding

Input: A feasible solution $\{N_1, \ldots, N_F, \Pi_1, \ldots, \Pi_F\}$, the importance vector $[w_1, w_2]$;
Output: A complete schedule of the solution.

```
for f = 1, 2, ..., F do
    Record the number of jobs assigned to factory f as n_f;
    for k = 1, 2, ..., s do
        for j_f = 1, 2, ..., n_f do
            Assign job j_f to the last of all available machines;
            Calculate the fuzzy completion time and fuzzy
            energy consumption of each machine;
            if rand ≤ w_1 then
                The certain machine with the minimum
                completion time is selected;
            else
                The certain machine with the minimum energy
                consumption is selected;
            end
        end
        Rerank the jobs according to their completion time of
        stage k;
    end
end
```

1) Basic Biased Decoding: The main method of the basic biased decoding strategy is described in Algorithm 3. For each stage, the job is first assigned to the last position of every available machine. Next, the value of fuzzy completion time and fuzzy energy consumption are calculated. Then, the certain machine with the minimum completion time or the minimum energy consumption is chosen via roulette wheel selection mechanism according to the value of **w**.

2) Self-Adaptive Strategy Based on RL: The value of w_1 and w_2 has a great influence on the evolutionary direction, thus impacting the quality of solutions. An importance vector with fixed value may guide the population toward a monotonous evolutionary direction, hence the algorithm is more likely to be trapped in the local optimum. Therefore, an importance vector with the ability of adjusting numeric value according to the status of the evolutionary process is expected. In order to address the issues above, a self-adaptive strategy based on RL is designed for importance vectors.

Model: In RL, the agent selects an action A_g based on the current state S_g in generation g first. Then, the importance vector updates according to the chosen action and is utilized in the next generation $g + 1$. Next, the state changes into S_{g+1} after evolving over generation $g + 1$, a reward R_g is calculated according to changes of evolutionary state and given to the agent. Finally, the Q-table is updated as $Q(S_g, A_g) = Q(S_g, A_g) + \alpha \times [R_g + \gamma \cdot \max_a Q(S_{g+1}, a) - Q(S_g, A_g)]$, where α and γ denote learning rate and discount factor, respectively [50].

State: In this article, the evolutionary rates of $\widetilde{MS}$ and $\widetilde{TEC}$, denoted as Δ_{MS} and Δ_{TEC}, along with the HV indicator, are employed to represent the state. The evolutionary rate Δ_{obj} reflects the numerical change of the objective between two adjacent generations, which can be calculated as

$$\Delta_{obj} = \frac{\sum_{p=1}^{NP} obj_p^3(g) - \sum_{p=1}^{NP} obj_p^3(g-1)}{\sum_{p=1}^{NP} obj_p^3(g-1)} \tag{20}$$

Algorithm 4: Local Intensification

Input: The current population *Pop*, population size *NP*;
Output: A new population Pop_{New}.

```
Select all the non-dominated solutions from Pop through non-dominated sorting;
for e = 1, 2, ..., N_E do
    Select the factory f_c with the maximum completion time;
    Select the factory f_e with the minimum completion time;
    Randomly select a job j_c from factory f_c and remove it;
    Insert job j_c into all possible positions in factory f_e;
    for each insertion do
        Calculate the value of objectives via biased decoding mechanism;
    end
    Select the best one as a neighbour;
end
Select the best NP individuals from the union of Pop and neighbours;
```

where $obj_p^3(g)$ denotes MS_p^3 or TEC_p^3 of the pth individual in generation g, while $obj_p^3(g-1)$ is obtained from generation $(g-1)$. Since both $\widetilde{MS}$ and $\widetilde{TEC}$ are minimal objectives, Δ_{MS} and Δ_{TEC} are expected to be positive numbers.

By comparing $\mathrm{HV}(g)$ and $\mathrm{HV}(g-1)$, Δ_{MS} and Δ_{TEC}, $\min(\Delta_{MS}, \Delta_{TEC})$ and 0, a total of eight different combinations can be achieved, including the following.

1) $\mathrm{HV}(g) \le \mathrm{HV}(g-1)$, $\Delta_{MS} \le \Delta_{TEC}$, $\Delta_{MS} < 0$.
2) $\mathrm{HV}(g) \le \mathrm{HV}(g-1)$, $0 \le \Delta_{MS} \le \Delta_{TEC}$.
3) $\mathrm{HV}(g) \le \mathrm{HV}(g-1)$, $\Delta_{MS} > \Delta_{TEC}$, $\Delta_{TEC} < 0$.
4) $\mathrm{HV}(g) \le \mathrm{HV}(g-1)$, $\Delta_{MS} > \Delta_{TEC} \ge 0$.
5) $\mathrm{HV}(g) > \mathrm{HV}(g-1)$, $\Delta_{MS} \le \Delta_{TEC}$, $\Delta_{MS} < 0$.
6) $\mathrm{HV}(g) > \mathrm{HV}(g-1)$, $0 \le \Delta_{MS} \le \Delta_{TEC}$.
7) $\mathrm{HV}(g) > \mathrm{HV}(g-1)$, $\Delta_{MS} > \Delta_{TEC}$, $\Delta_{TEC} < 0$.
8) $\mathrm{HV}(g) > \mathrm{HV}(g-1)$, $\Delta_{MS} > \Delta_{TEC} \ge 0$.

The above comparing results are regarded as states of Q-learning.

Action: Four actions are designed for importance vector $\mathbf{w} = [w_1, w_2]$, including: 1) increase the value of w_1; 2) increase the value of w_2; 3) keep both w_1 and w_2 unchanged; and 4) regenerate $\mathbf{w}$ as $[0.5, 0.5]$.

Reward: Taking the characteristics of multiobjective optimization problem into consideration, the reward is set as 5 if $\mathrm{HV}(g) > \mathrm{HV}(g-1)$ and 0 otherwise.

F. Local Intensification

To further enhance the quality of solutions, a local intensification is designed and carried out on nondominated individuals at the end of each generation. The main method of local intensification is described in Algorithm 4.

For each nondominated individual, the factories with the maximum and minimum completion time are first selected, denoted as critical factory f_c and the easiest factory f_e, respectively. Next, a job assigned to f_c is chosen randomly, which is then inserted into all possible positions in f_e. After that, the biased decoding strategy is conducted on each insertion and the best one is kept as a neighbor. Finally, a total of *NP* best individuals are selected from the union of the current population and neighbors.

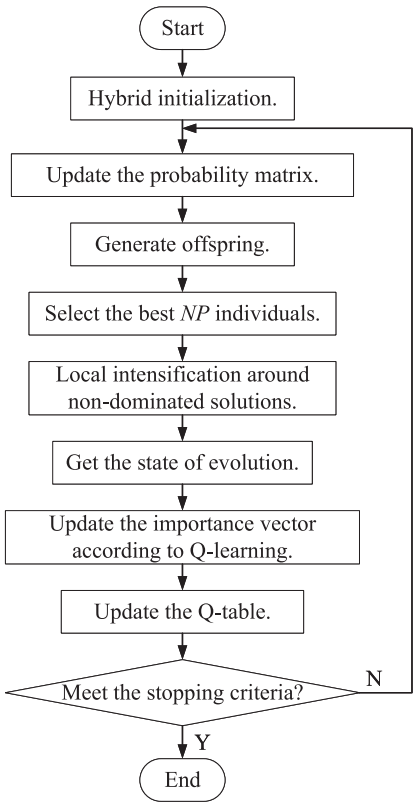


Fig. 4. Framework of 3D-EDA/RL for FDHFSP-OTD.

G. Framework of 3D-EDA/RL

The framework of 3D-EDA/RL is illustrated in Fig. 4. A 3-D probability matrix is employed in the EDA to take full advantage of the scheduling information carried by elite solutions, and a self-adaptive biased decoding mechanism is designed to adjust the direction of evolution on the basis of the exploration and learning abilities of RL. Moreover, problem-specific strategies, including hybrid initialization, updating of the probability matrix, and local intensification, are implemented in a cooperative method. The performance of such a properly designed algorithm in solving FDHFSP-OTD is highly expected.

H. Computational Complexity Analysis

According to Fig. 4, the main steps of each generation in the proposed 3D-EDA/RL consist of updating the probability matrix, generating offspring, RL-based biased decoding method and local intensification, which are considered to analyze the computational complexity.

As shown in Algorithm 1, the computational complexity of probability matrix updating is $O(N_E \times F \times n_f^2)$, where N_E denotes the number of nondominated solutions, F is the number of factories, and n_f represents the job assigned to factory f. According to Algorithm 2, the sampling strategy of offspring generation mainly consists of selecting a factory for each job, and decide the job sequence in each factory. The computational complexity of factory selecting is $O(NP \times n)$, while job sequencing has a computational complexity of $O(F \times n_f)$, where *NP* denotes the population size and n is

the number of jobs. Since the number of jobs assigned to each factory n_f is smaller than total number of jobs n, and the population size in this article is set much larger than factory number in this article, the computational complexity of the offspring generation can be recorded as $O(NP \times n)$. As shown in Algorithm 3, the computational complexity of RL-based biased decoding method is $O(F \times s \times n_f)$, in which s is the number of stages. Similarly, according to Algorithm 4, the computational complexity of local intensification can be calculated as $O(N_E \times n_i \times F \times s \times n_f)$, where n_i is the number of jobs in factory f_e, and $f_e < n$ since each factory has at least one job. Therefore, the overall computational complexity of the proposed LTCSO/D is $O(N_E \times n_i \times F \times s \times n_f)$.

IV. Numerical Results and Comparisons

A. Experimental Settings

In order to test the performance of 3D-EDA/RL on solving FDHFSP-OTD, the following instances are generated and used. In this article, the scale of an instance is controlled by F, n, and m. In order to generate practical instances, F is chosen from $\{3, 4, 5\}$, n is from $\{30, 50, 80\}$ and s is from $\{3, 4, 5\}$. For each combination of F, n and s, 10 test instances are generated. Thus, a total of $3 \times 3 \times 3 \times 10 = 270$ instances with 27 different scales are used. Furthermore, the fuzzification as follows is utilized to generate fuzzy processing time. For each TFN, $P^2_{j,k}$ is evenly sampled from instances provided in [44] and [24], which are designed for energy-aware DHFSP. Then the numerical range of $P^1_{j,k}$ and $P^3_{j,k}$ can be obtained as (21) and (22). The number of $m_{f,k}$ is randomly taken from $\{1, 2, 3, 4, 5\}$, while the power consumption of the processing mode is randomly sampled from $\{5, 6, 7, 8, 9, 10\}$ and the power consumption of the idle mode is set as 1

$$P^1_{j,k} \in \begin{cases} \left(0, \frac{P^2_{j,k}}{2}\right), & \text{if } P^2_{j,k} \le 12 \\ \left(P^2_{j,k} - 6, P^2_{j,k}\right), & \text{otherwise} \end{cases} \tag{21}$$

$$P^3_{j,k} \in \begin{cases} \left(P^2_{j,k}, \frac{3}{2}P^2_{j,k}\right), & \text{if } P^2_{j,k} \le 12 \\ \left(P^2_{j,k}, P^2_{j,k} + 6\right), & \text{otherwise.} \end{cases} \tag{22}$$

All the compared algorithms are set to stop when the maximum running time is up to $(0.1 \times F \times n \times m)$ seconds. All algorithms are implemented through MATLAB2021b and conducted on the same computer with 12th Gen Intel Core i7-12700K, 3.61 GHz, 32-GB RAM, and Windows 10 OS. In order to evaluate the performance of each algorithm, three commonly used metrics, including HV [51], C metric (CM) [52] and GD [53], are utilized. When calculating HV and GD, all solutions are first normalized, with the combination of the minimum value of each objective as the lower bound, and the combination of the maximum value of each objective as the upper bound.

B. Parameter Setting

Four important parameters are included in the proposed 3-D-EDA/RL: 1) weight factor in updating the probability matrix (β); 2) learning rate in Q-learning (α); 3) deteriorating factor in Q-learning (γ); and 4) probability of exploration in ϵ-greedy selection strategy (ϵ). The design of experiments (DOE) [54] is employed to investigate the effects of these parameters on 3D-EDA/RL and determine the most effective combination.

TABLE I
Orthogonal Arrays and HVs

Experiment Number	Factor Level β	α	γ	ϵ	HV
1	1	1	1	1	0.8549
2	1	2	2	2	0.8523
3	1	3	3	3	0.8553
4	1	4	4	4	0.8535
5	2	1	2	3	0.8523
6	2	2	1	4	0.8546
7	2	3	4	1	0.8537
8	2	4	3	2	0.8547
9	3	1	3	4	0.8573
10	3	2	4	3	0.8532
11	3	3	1	2	0.8538
12	3	4	2	1	0.8553
13	4	1	4	2	0.8594
14	4	2	3	1	0.8588
15	4	3	2	4	0.8590
16	4	4	1	3	0.8600

TABLE II
Average HVs and the Rank of Parameters

Level	β	α	γ	ϵ
1	0.8540	**0.8560**	0.8558	0.8557
2	0.8538	0.8547	0.8547	0.8551
3	0.8549	0.8554	**0.8565**	0.8552
4	**0.8593**	0.8559	0.8549	**0.8561**
Delta	0.0055	0.0013	0.0018	0.0010
Rank	1	3	2	4

Four levels are set for each parameter, more precisely, $\beta \in \{0.7, 0.75, 0.8, 0.85\}$, $\alpha \in \{0.1, 0.15, 0.2, 0.25\}$, $\gamma \in \{0.8, 0.85, 0.9, 0.95\}$, and $\epsilon \in \{0.8, 0.85, 0.9, 0.95\}$. Therefore, according to the orthogonal array $L_{16}(4^4)$, 16 different parameter combinations of $(\beta, \alpha, \gamma, \epsilon)$ are tested on a total of 27 instances, which consists of every combination of F, n, and s. For each instance, 3D-EDA/RL with each combination runs ten times independently, and Table I provides the average value of HV.

For each level of each parameter, the average HV is calculated and the certain level with the largest HV value is chosen. The average HVs and the rank of the parameters are shown in Table II, in which Delta denotes the difference between the largest and the smallest HVs of each parameter, and a larger Delta means a more significant effect of a certain parameter has on the algorithm. Meanwhile, the trends of factor levels trend are illustrated in Fig. 5.

It is obvious that β has the greatest influence on the performance of 3D-EDA/RL, followed by γ, α and ϵ. β denotes the proportion that PM learns from the elite individuals when updating, therefore a small value of β may lead to a loss of information. γ determines the importance of future

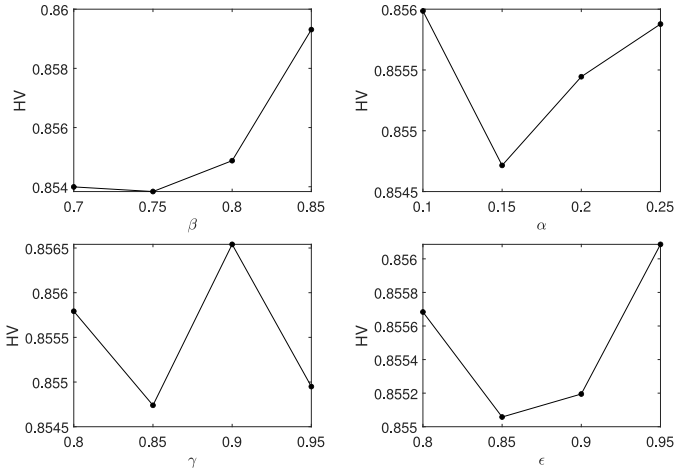


Fig. 5. Trends of factor levels.

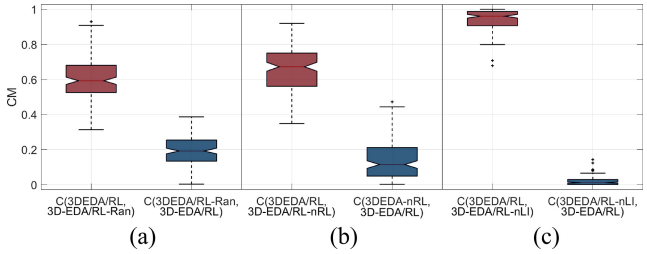


Fig. 6. Boxplots of C Metric of 3D-EDA/RL, 3D-EDA/RL-Ran, 3D-EDA-nRL and 3D-EDA/RL-nLI.

rewards, and a small value of it causes more consideration about the immediate reward, which may result in local optimal solutions. α is the learning rate in Q-learning, and the Q-table will updates slowly with a low value of learning rate; therefore more iterations are needed. ϵ is used to balance exploration and exploitation when choosing an action for the current state according to Q-values. A low ϵ means that the agent has more chances to explore during the exploitation and needs more time to find the optimal solution. Considering to the results of the DOE given in Table II, the values of the parameters are set as $\beta = 0.85$, $\alpha = 0.1$, $\gamma = 0.9$, and $\epsilon = 0.95$.

C. Effect of Algorithm Designs

To investigate the effectiveness of the problem-specific operations in 3D-EDA/RL, including hybrid initialization, RL-based biased decoding and local intensification, the proposed algorithm is compared to its variants 3D-EDA/RL-Ran, 3D-EDA-nRL and 3D-EDA/RL-nLI, respectively. For each instance, each algorithm runs 10 times independently, which provides ten groups of *CM* values. The p-value of the sign test with a 95% confidence level is calculated via the signtest function according to ten groups of *CM* values on each instance, and then averaged for each scale, in purpose to analyze the differences between the two compared algorithms.

The initial population in 3D-EDA/RL-Ran is all generated randomly without heuristic methods. The first-in-first-out decoding rule is used in 3D-EDA-nRL instead of the RL-based biased decoding method, while the local intensification part is absent in 3D-EDA/RL-nLI. The boxplots of *CM* values obtained by 3D-EDA/RL and its variants are shown in Fig. 6, from which it can be seen that 3D-EDA/RL outperforms 3D-EDA/RL-Ran, 3D-EDA-nRL and 3D-EDA/RL-nLI. Therefore, it can be concluded that the hybrid initialization considerably enhances the quality of the initial population. The RL-based biased decoding method can efficiently support 3D-EDA/RL by adjusting the importance vector, while the proposed local intensification operator has a significant influence on enhancing the performance of 3D-EDA/RL.

TABLE III
CM VALUES OF 3D-EDA/RL AND ITS VARIANTS ON INSTANCES WITH DIFFERENT f
$C_1 = C$(3D-EDA/RL,3D-EDA/RL-RAN),
$C'_1 = C$(3D-EDA/RL-RAN,3D-EDA/RL),
$C_2 = C$(3D-EDA/RL,3D-EDA-NRL),
$C'_2 = C$(3D-EDA-NRL,3D-EDA/RL),
$C_3 = C$(3D-EDA/RL,3D-EDA/RL-NLI),
$C'_3 = C$(3D-EDA/RL-NLI,3D-EDA/RL)

f	C_1	C'_1	p	C_2	C'_2	p	C_3	C'_3	p
3	**0.63**	0.17	0.02	**0.67**	0.12	0.00	**0.92**	0.03	0.00
4	**0.54**	0.23	0.02	**0.64**	0.15	0.02	**0.95**	0.02	0.00
5	**0.64**	0.19	0.00	**0.68**	0.15	0.02	**0.96**	0.02	0.00

TABLE IV
CM VALUES OF 3D-EDA/RL AND ITS VARIANTS ON INSTANCES WITH DIFFERENT n AND s

n	s	C_1	C'_1	p	C_2	C'_2	p	C_3	C'_3	p
30	4	**0.54**	0.25	0.02	**0.56**	0.22	0.02	**0.92**	0.03	0.00
	5	**0.53**	0.27	0.02	**0.55**	0.29	0.00	**0.88**	0.05	0.00
	6	**0.57**	0.23	0.02	**0.56**	0.24	0.00	**0.93**	0.02	0.00
50	4	**0.60**	0.18	0.02	**0.63**	0.12	0.02	**0.96**	0.01	0.00
	5	**0.58**	0.20	0.02	**0.67**	0.13	0.00	**0.93**	0.03	0.00
	6	**0.61**	0.18	0.00	**0.70**	0.11	0.02	**0.96**	0.01	0.00
80	4	**0.62**	0.15	0.00	**0.72**	0.07	0.02	**0.98**	0.00	0.00
	5	**0.64**	0.16	0.00	**0.77**	0.04	0.02	**0.93**	0.02	0.00
	6	**0.70**	0.14	0.02	**0.81**	0.04	0.02	**0.98**	0.01	0.00

The average *CM* values of the instances with the same value f are shown in Table III, while the effectiveness of each strategy is verified from the perspective of jobs and states in Table IV. It is obvious that despite different numbers of distributed factories and different scales of orders, C(3D-EDA/RL, 3D-EDA-nRL), C(3D-EDA/RL, 3D-EDA/RL-nLI) and C(3D-EDA/RL, 3D-EDA/RL-nLI) are larger than C(3D-EDA/RL-Ran, 3D-EDA/RL), C(3D-EDA-nRL, 3D-EDA/RL), and C(3D-EDA/RL-nLI, 3D-EDA/RL), which suggests that most of the nondominated solutions obtained by 3D-EDA/RL are better than those provided by its variants. Moreover, the p-values are all smaller than 0.05, which proves that all the strategies can significantly improve the ability of the algorithm for scheduling FDHFSP with different scales of orders and different distributions.

D. Comparisons to Other Algorithms

To the best of our knowledge, there is currently no research on FDHFSP-OTD. Therefore, two state-of-the-art algorithms, i.e., the DCMA [23] and BCMA [18], and two classical multiobjective optimization algorithms, including the NSGA-II [55] and MOEA/D [56], are used for comparison. DCMA

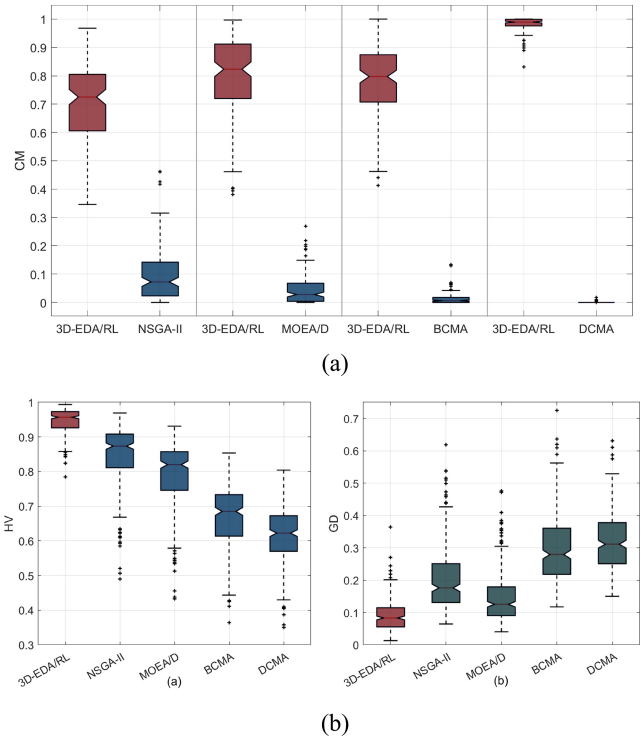


Fig. 7. Comparisons Results Between 3D-EDA/RL and NSGA-II, MOEA/D, BCMA, DCMA. (a) Boxplots of *CM*. (b) Boxplots of HV and GD.

TABLE V
CM VALUES OF 3D-EDA/RL AND NSGA-II, MOEA/D, BCMA AND DCMA ON INSTANCES WITH DIFFERENT f
$C_4 = C$(3D-EDA/RL,NSGA-II), $C'_4 = C$(NSGA-II,3D-EDA/RL), $C_5 = C$(3D-EDA/RL,MOEA/D), $C'_5 = C$(MOEA/D,3D-EDA/RL), $C_6 = C$(3D-EDA/RL,BCMA), $C'_6 = C$(BCMA,3D-EDA/RL), $C_7 = C$(3D-EDA/RL,DCMA), $C'_7 = C$(DCMA,3D-EDA/RL)

f	C_4	C'_4	p	C_5	C'_5	p	C_6	C'_6	p	C_7	C'_7	p
3	**0.69**	0.08	0.02	**0.77**	0.04	0.00	**0.75**	0.01	0.00	**0.98**	0.00	0.00
4	**0.72**	0.09	0.00	**0.81**	0.05	0.00	**0.78**	0.01	0.00	**0.98**	0.00	0.00
5	**0.68**	0.13	0.02	**0.80**	0.05	0.00	**0.82**	0.02	0.00	**0.99**	0.00	0.00

was proposed in 2022 for DHDFSP, while BCMA was proposed in 2021 for DHFSP, both of which are designed for similar problems and thus can be implemented to solve FDHFSP-OTD with minor adjustments. Parameters in all algorithms are set as suggested in the corresponding references.

First, the C metric is calculated and analyzed to evaluate the performances of all the algorithms, and a nonparametric test is carried out to determine whether significant differences exist between different algorithms, in which the confidence level is set as 95%. Fig. 7(a) provides the boxplots of all *CM* values. It is noticeable that C(3D-EDA/RL, NSGA-II), C(3D-EDA/RL, MOEA/D), C(3D-EDA/RL, BCMA), and C(3D-EDA/RL, DCMA) are much larger than C(NSGA-II, 3D-EDA/RL), C(MOEA/D, 3D-EDA/RL), C(BCMA, 3D-EDA/RL), and C(DCMA, 3D-EDA/RL), which indicates that the the quality of nondominated solutions obtained by 3D-EDA/RL are higher than those of the other algorithms.

Next, the ability of all algorithms for scheduling the FDHFSP with different dispersion and scales of orders is analyzed. The average *CM* values of the instances with the same value f are shown in Table V and Table VI provides the average *CM* values of algorithms on instances of different order sizes. Obviously, 3D-EDA/RL achieves higher *CM* value while the p values are all lower than 0.05 at the same time. The results demonstrate that 3D-EDA/RL performs better when solving FDHFSP-OTD no matter the dispersion and size of order, especially on large-scaled instances.

TABLE VI
CM VALUES OF 3D-EDA/RL AND NSGA-II, MOEA/D, BCMA AND DCMA ON INSTANCES WITH DIFFERENT n AND s

n	s	C_4	C'_4	p	C_5	C'_5	p	C_6	C'_6	p	C_7	C'_7	p
30	4	**0.63**	0.17	0.00	**0.75**	0.07	0.00	**0.79**	0.00	0.02	**0.98**	0.00	0.00
	5	**0.56**	0.20	0.02	**0.68**	0.09	0.00	**0.68**	0.05	0.00	**0.95**	0.00	0.00
	6	**0.61**	0.19	0.02	**0.67**	0.10	0.00	**0.79**	0.02	0.00	**0.99**	0.00	0.00
50	4	**0.74**	0.07	0.00	**0.83**	0.02	0.00	**0.83**	0.01	0.00	**0.99**	0.00	0.00
	5	**0.64**	0.09	0.00	**0.75**	0.05	0.00	**0.72**	0.01	0.00	**0.98**	0.00	0.00
	6	**0.73**	0.07	0.00	**0.79**	0.05	0.00	**0.87**	0.01	0.00	**0.99**	0.00	0.00
80	4	**0.81**	0.02	0.02	**0.90**	0.01	0.00	**0.85**	0.00	0.00	**0.99**	0.00	0.00
	5	**0.72**	0.04	0.02	**0.85**	0.02	0.00	**0.66**	0.00	0.00	**0.96**	0.00	0.00
	6	**0.83**	0.02	0.02	**0.90**	0.01	0.00	**0.86**	0.00	0.00	**0.99**	0.00	0.00

TABLE VII
HV AND GD VALUES OF 3D-EDA/RL, NSGA-II, MOEA/D, BCMA AND DCMA ON INSTANCES WITH DIFFERENT f

Metrics	f	3D-EDA/RL	NSGA-II	MOEA/D	BCMA	DCMA
HV	3	**0.94**	0.81	0.77	0.65	0.61
	4	**0.95**	0.84	0.78	0.66	0.61
	5	**0.95**	0.88	0.82	0.69	0.63
GD	3	**0.10**	0.23	0.17	0.30	0.33
	4	**0.09**	0.21	0.15	0.30	0.32
	5	**0.08**	0.17	0.12	0.28	0.31

Then, the HV and GD values are used to evaluate the performance of each algorithm. For each instance, the objective values are normalized with the largest value and the lowest value obtained by all the algorithms as higher bounds and lower bounds, respectively. Fig. 7(b) illustrates the boxplots of two evaluation indicators on all instances, which suggests that 3D-EDA/RL results in higher HV and lower GD on all scales of instances.

Furthermore, the performance of the algorithms from the perspective of the dispersion of factories and the scale of orders are shown in Tables VII and VIII. It is obvious that 3D-EDA/RL has a much better performance than the other four algorithms, regardless of the dispersion of the factories and the scales of the orders.

The Pareto set approximations obtained by five algorithms on an instance with three factories, 50 jobs, and four stages are illustrated in Fig. 8(a), while nondominated solutions on instance with five factories, 30 jobs, and four stages are shown in Fig. 8(b). It can be observed that the distribution of Pareto set approximations achieved by 3D-EDA/RL are more even than the other ones, which proves the ability of 3D-EDA/RL in shortening the processing time, reducing total energy consumption and maintaining the customer satisfaction.

Comparing with the existing algorithms mentioned above, the superiority of the proposed 3D-EDA/RL is significant and steady, which is mainly due to the following features

TABLE VIII
HV AND GD VALUES OF 3D-EDA/RL, NSGA-II, MOEA/D, BCMA AND DCMA ON INSTANCES WITH DIFFERENT n AND s

Metrics	n	s	3D-EDA/RL	NSGA-II	MOEA/D	BCMA	DCMA
HV	30	4	**0.94**	0.84	0.78	0.65	0.63
		5	**0.93**	0.87	0.82	0.66	0.58
		6	**0.92**	0.89	0.84	0.67	0.64
	50	4	**0.95**	0.84	0.79	0.68	0.63
		5	**0.95**	0.84	0.78	0.66	0.58
		6	**0.95**	0.87	0.82	0.68	0.62
	80	4	**0.96**	0.82	0.76	0.69	0.63
		5	**0.96**	0.80	0.74	0.63	0.57
		6	**0.96**	0.85	0.80	0.69	0.64
GD	30	4	**0.11**	0.22	0.16	0.31	0.30
		5	**0.11**	0.20	0.14	0.30	0.35
		6	**0.11**	0.19	0.14	0.31	0.31
	50	4	**0.08**	0.20	0.15	0.29	0.29
		5	**0.09**	0.21	0.15	0.28	0.34
		6	**0.08**	0.19	0.12	0.30	0.32
	80	4	**0.08**	0.20	0.15	0.29	0.30
		5	**0.09**	0.24	0.17	0.30	0.35
		6	**0.07**	0.18	0.12	0.28	0.31

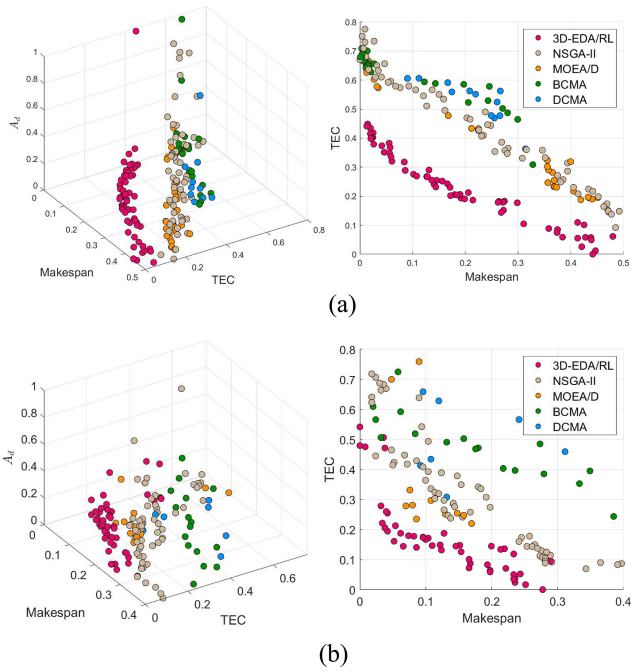


Fig. 8. Distribution of nondominated solutions obtained by 3D-EDA/RL, NSGA-II, MOEA/D, BCMA and DCMA. (a) Instance with 3 factories, 50 jobs and 4 stages. (b) Instance with 5 factories, 30 jobs and 4 stages.

and novelties. First, two heuristics based on the NEH are used to generate the initial population, which enhances the quality and diversity of the population. Next, a 3-D probability matrix is designed to describe the elite solutions without losing too much useful information, while the other four algorithms handle the machine assignment and job sequencing separately. Then, the RL-based decoding method can efficiently adjust the importance vector to maintain a balance between the evolution of different objectives. Finally, the local intensification is able to obtain better solutions around the nondominated solutions, which further improves the performance of the proposed algorithm. Therefore, the results and analysis above lead to the conclusion that 3D-EDA/RL provides solutions with shorter makespan, higher delivery accuracy, and lower energy consumption for FDHFSP-OTD.

V. CONCLUSION AND FUTURE WORK

This article aims to solve the fuzzy distributed hybrid flow-shop scheduling problem, regarding the makespan, total energy consumption, and delivery accuracy as optimization objectives, by designing a 3-D EDA based on RL. The effects of important parameters on the proposed algorithm are analyzed, and the effects of each problem-specific strategy are verified through experiments and tests. Comprehensive comparisons and analysis demonstrate that the 3D-EDA/RL is able to achieve superior performance on all the instances, especially on large-scale instances. Therefore, it can be concluded that the cooperation of well-designed strategies and the EDA is effective for solving scheduling problems with high complexity, while the embedding of RL can further enhance the ability to search for better solutions.

In the future, the 3D-EDA/RL can be expanded to address distributed scheduling problems with constraints such as type-2 fuzzy processing time, sequence-dependent setup time, and limited buffers, by adding corresponding constraints. In addition, objectives with higher complexity and more practical significance can be considered in the proposed algorithm, including different due dates for each job, time-of-use electricity costs, and transportation costs. However, when facing other kinds of manufacturing methods, such as no-idle or no-wait scheduling problems, scheduling problems with lot streaming or heterogeneous factories, the mathematical model proposed in this article is no longer suitable and needs modification. Furthermore, it is worth developing the integration of traditional evolutionary algorithms with machine learning techniques and applying them to solve practical engineering issues.

REFERENCES

[1] N. Boysen, S. Emde, M. Hoeck, and M. Kauderer, "Part logistics in the automotive industry: Decision problems, literature review and research agenda," *Eur. J. Oper. Res.*, vol. 242, no. 1, pp. 107–120, 2015.
[2] K. R. Baker and G. D. Scudder, "Sequencing with earliness and tardiness penalties: A review," *Oper. Res.*, vol. 38, no. 1, pp. 22–36, 1990.
[3] M. Mahnam, G. Moslehi, and S. M. T. F. Ghomi, "Single machine scheduling with unequal release times and idle insert for minimizing the sum of maximum earliness and tardiness," *Math. Comput. Model.*, vol. 57, no. 9, pp. 2549–2563, 2013.
[4] J. B. Sidney, "Optimal single-machine scheduling with earliness and tardiness penalties," *Oper. Res.*, vol. 25, no. 1, pp. 62–69, 1977.
[5] G. Wan and B. P. C. Yen, "Single machine scheduling to minimize total weighted earliness subject to minimal number of tardy jobs," *Eur. J. Oper. Res.*, vol. 195, no. 1, pp. 89–97, 2009.
[6] R. Braune, G. Zäpfel, and M. Affenzeller, "An exact approach for single machine subproblems in shifting bottleneck procedures for job shops with total weighted tardiness objective," *Eur. J. Oper. Res.*, vol. 218, no. 1, pp. 76–85, 2012.
[7] J. Kelbel and Z. Hanzálek, "Solving production scheduling with earliness/tardiness penalties by constraint programming," *J. Intell. Manuf.*, vol. 22, no. 4, pp. 553–562, 2011.

[8] M. M. Ahmadian, A. Salehipour, and T. Cheng, "A meta-heuristic to solve the just-in-time job-shop scheduling problem," *Eur. J. Oper. Res.*, vol. 288, no. 1, pp. 14–29, 2021.
[9] Z. Wu and M. X. Weng, "Multiagent scheduling method with earliness and tardiness objectives in flexible job shops," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 2, pp. 293–301, Apr. 2005.
[10] J. Li, Q.-K. Pan, and K. Mao, "A hybrid fruit fly optimization algorithm for the realistic hybrid flowshop rescheduling problem in steelmaking systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 932–949, Apr. 2016.
[11] Y. Fu, H. Wang, J. Wang, and X. Pu, "Multiobjective modeling and optimization for scheduling a stochastic hybrid flow shop with maximizing processing quality and minimizing total tardiness," *IEEE Syst. J.*, vol. 15, no. 3, pp. 4696–4707, Sep. 2021.
[12] Z. Wang, Q. Pan, L. Gao, and Y. Wang, "An effective two-stage iterated greedy algorithm to minimize total tardiness for the distributed flowshop group scheduling problem," *Swarm Evol.*, vol. 74, Oct. 2022, Art. no. 101143.
[13] Y. Wang et al., "Carbon peak and carbon neutrality in China: Goals, implementation path and prospects," *China Geol.*, vol. 4, no. 4, pp. 720–746, 2021.
[14] K. Gao, Y. Huang, A. Sadollah, and L. Wang, "A review of energy-efficient scheduling in intelligent production systems," *Complex Intell. Syst.*, vol. 6, no. 2, pp. 237–249, 2020.
[15] Y. Fu, Y. Hou, Z. Wang, X. Wu, K. Gao, and L. Wang, "Distributed scheduling problems in intelligent manufacturing systems," *Tsinghua Sci. Technol.*, vol. 26, no. 5, pp. 625–645, 2021.
[16] F. Zhao, Z. Xu, L. Wang, N. Zhu, T. Xu, and J. Jonrinaldi, "A population-based iterated greedy algorithm for distributed assembly no-wait flow-shop scheduling problem," *IEEE Trans. Ind. Informat.*, vol. 19, no. 5, pp. 6692–6705, May 2023.
[17] F. Zhao, R. Ma, and L. Wang, "A self-learning discrete Jaya algorithm for multiobjective energy-efficient distributed no-idle flow-shop scheduling problem in heterogeneous factory system," *IEEE Trans. Cybern.*, vol. 52, no. 12, pp. 12675–12686, Dec. 2022.
[18] J.-J. Wang and L. Wang, "A bi-population cooperative memetic algorithm for distributed hybrid flow-shop scheduling," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 5, no. 6, pp. 947–961, Dec. 2021.
[19] Z. Pan, D. Lei, and L. Wang, "A knowledge-based two-population optimization algorithm for distributed energy-efficient parallel machines scheduling," *IEEE Trans. Cybern.*, vol. 52, no. 6, pp. 5051–5063, Jun. 2022.
[20] Q. Pan, L. Gao, and L. Wang, "An effective cooperative co-evolutionary algorithm for distributed flowshop group scheduling problems," *IEEE Trans. Cybern.*, vol. 52, no. 7, pp. 5999–6012, Jul. 2022.
[21] W. Shao, Z. Shao, and D. Pi, "An ant colony optimization behavior-based MOEA/D for distributed heterogeneous hybrid flow shop scheduling problem under nonidentical time-of-use electricity tariffs," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 4, pp. 3379–3394, Oct. 2022.
[22] F. Zhao, T. Jiang, and L. Wang, "A reinforcement learning driven cooperative meta-heuristic algorithm for energy-efficient distributed no-wait flow-shop scheduling with sequence-dependent setup time," *IEEE Trans Ind. Informat.*, vol. 19, no. 7, pp. 8427–8440, Jul. 2023.
[23] G. Zhang, B. Liu, L. Wang, D. Yu, and K. Xing, "Distributed co-evolutionary memetic algorithm for distributed hybrid differentiation flowshop scheduling problem," *IEEE Trans. Evol. Comput.*, vol. 26, no. 5, pp. 1043–1057, Oct. 2022.
[24] E. Jiang, L. Wang, and J. Wang, "Decomposition-based multi-objective optimization for energy-aware distributed hybrid flow shop scheduling with multiprocessor tasks," *Tsinghua Sci. Technol.*, vol. 26, no. 5, pp. 646–663, 2021.
[25] J.-Q. Li, X.-L. Chen, P.-Y. Duan, and J.-H. Mou, "KMOEA: A knowledge-based multiobjective algorithm for distributed hybrid flow shop in a prefabricated system," *IEEE Trans. Ind. Informat.*, vol. 18, no. 8, pp. 5318–5329, Aug. 2022.
[26] D. Gao, G. Wang, and W. Pedrycz, "Solving fuzzy job-shop scheduling problem using DE algorithm improved by a selection mechanism," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 12, pp. 3265–3275, Dec. 2020.
[27] R. Li, W. Gong, and C. Lu, "A reinforcement learning based RMOEA/D for bi-objective fuzzy flexible job shop scheduling," *Expert Syst. Appl.*, vol. 203, Oct. 2022, Art. no. 117380.
[28] Z. Pan, D. Lei, and L. Wang, "A bi-population evolutionary algorithm with feedback for energy-efficient fuzzy flexible job shop scheduling," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 52, no. 8, pp. 5295–5307, Aug. 2022.
[29] G. Wang, D. Gao, and W. Pedrycz, "Solving multiobjective fuzzy job-shop scheduling problem by a hybrid adaptive differential evolution algorithm," *IEEE Trans. Ind. Informat.*, vol. 18, no. 12, pp. 8519–8528, Dec. 2022.
[30] R. Li, W. Gong, C. Lu, and L. Wang, "A learning-based memetic algorithm for energy-efficient flexible job-shop scheduling with type-2 fuzzy processing time," *IEEE Trans. Evol. Comput.*, vol. 27, no. 3, pp. 610–620, Jun. 2023.
[31] J. Li, Z. Liu, C. Li, and Z. Zheng, "Improved artificial immune system algorithm for type-2 fuzzy flexible job shop scheduling problem," *IEEE Trans. Fuzzy Syst.*, vol. 29, no. 11, pp. 3234–3248, Nov. 2021.
[32] Y. Dorfeshan, R. T. Moghaddam, S. Mousavi, and B. V. 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, Jun. 2020, Art. no. 106248.
[33] J. Cai, R. Zhou, and D. Lei, "Fuzzy distributed two-stage hybrid flow shop scheduling problem with setup time: Collaborative variable search," *J. Intell. Fuzzy Syst.*, vol. 38, no. 3, pp. 3189–3199, 2020.
[34] B. Xi and D. Lei, "Q-learning-based teaching-learning optimization for distributed two-stage hybrid flow shop scheduling with fuzzy processing time," *Complex System Model. Simulat.*, vol. 2, no. 2, pp. 113–129, 2022.
[35] P. Larraanaga and J. A. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Norwell, MA, USA: Kluwer Academic Publ., 2001.
[36] B. Qian, Z.-Q. Zhang, R. Hu, H.-P. Jin, and J.-B. Yang, "A matrix-cube-based estimation of distribution algorithm for no-wait flow-shop scheduling with sequence-dependent setup times and release times," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 3, pp. 1492–1503, Mar. 2023.
[37] Y. Du, J.-Q. Li, X.-L. Chen, P.-Y. Duan, and Q.-K. Pan, "Knowledge-based reinforcement learning and estimation of distribution algorithm for flexible job shop scheduling problem," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 7, no. 4, pp. 1036–1050, Aug. 2023.
[38] Z. Zhang, B. Qian, R. Hu, H. Jin, and L. Wang, "A matrix-cube-based estimation of distribution algorithm for the distributed assembly permutation flow-shop scheduling problem," *Swarm Evol.*, vol. 60, Feb. 2021, Art. no. 100785.
[39] F. Zhao, G. Zhou, and L. Wang, "A cooperative scatter search with reinforcement learning mechanism for the distributed permutation flowshop scheduling problem with sequence-dependent setup times," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 8, pp. 4899–4911, Aug. 2023.
[40] L. Wang, Z. Pan, and J. Wang, "A review of reinforcement learning based intelligent optimization for manufacturing scheduling," *Complex Syst. Model. Simulat.*, vol. 1, no. 4, pp. 257–270, Dec. 2021.
[41] H. Li, K. Gao, P. Duan, J. Li, and L. Zhang, "An improved artificial bee colony algorithm with Q-learning for solving permutation flow-shop scheduling problems," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 53, no. 5, pp. 2684–2693, May 2023.
[42] Z. Pan, L. Wang, C. Dong, and J. Chen, "A knowledge-guided end-to-end optimization framework based on reinforcement learning for flow shop scheduling," *IEEE Trans Ind. Informat.*, early access, Jun. 2, 2023, doi: 10.1109/TII.2023.3282313.
[43] F. Zhao, S. Di, and L. Wang, "A hyperheuristic with Q-learning for the multiobjective energy-efficient distributed blocking flow shop scheduling problem," *IEEE Trans. Cybern.*, vol. 53, no. 5, pp. 3337–3350, May 2023.
[44] J. Wang and L. Wang, "A cooperative memetic algorithm with learning-based agent for energy-aware distributed hybrid flow-shop scheduling," *IEEE Trans. Evol. Comput.*, vol. 26, no. 3, pp. 461–475, Jun. 2022.
[45] L. Zadeh, "Fuzzy sets," *Inf. Control*, vol. 8, no. 3, pp. 338–353, 1965.
[46] X. Zhang, W. Ma, and L. Chen, "New similarity of triangular fuzzy number and its application," *Sci. World J.*, vol. 2014, Mar. 2014, Art. no. 215047.
[47] D. Santos, J. Hunsucker, and D. Deal, "Global lower bounds for flow shops with multiple processors," *Eur. J. Oper. Res.*, vol. 80, no. 1, pp. 112–120, 1995.
[48] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm Evol.*, vol. 1, no. 3, pp. 111–128, 2011.
[49] M. Nawaz, E. E. Enscore, and I. Ham, "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem," *Omega*, vol. 11, no. 1, pp. 91–95, 1983.
[50] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction. A Bradford Book*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2018.

[51] L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume," *IEEE Trans. Evol. Comput.*, vol. 10, no. 1, pp. 29–38, Feb. 2006.
[52] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, Jun. 2000.
[53] D. A. van Veldhuizen, "Evolutionary computation and convergence to a Pareto front," 1998. [Online]. Available: https://api.semanticscholar.org/CorpusID:18585705
[54] V. N. Nair et al., "Taguchi's parameter design: A panel discussion," *Technometrics*, vol. 34, no. 2, pp. 127–161, 1992.
[55] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[56] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

Libao Deng (Member, IEEE) was born in Cangzhou, China, in 1981. He received the B.Sc., M.Sc., and Ph.D. degrees from the Harbin Institute of Technology, Harbin, China, in 2004, 2007, and 2012, respectively.

He is currently a Professor with the School of Information Science and Engineering, Harbin Institute of Technology. His research interests are in computational intelligence and optimal scheduling.

Yuanzhu Di was born in Shenyang, China, in 1999. She received the B.Sc. and M.Sc. degrees from the Harbin Institute of Technology, Weihai, China, in 2021 and 2023, respectively, where she is currently pursuing the Ph.D. degree.

Her main research directions include the distributed and green scheduling with computational intelligence.

Ling Wang (Member, IEEE) received the B.Sc. degree in automation and the Ph.D. degree in control theory and control engineering from Tsinghua University, Beijing, China, in 1995 and 1999, respectively.

Since 1999, he has been with the Department of Automation, Tsinghua University, where he became a Full Professor in 2008. He has authored five academic books and more than 300 refereed papers. His current research interests include intelligent optimization and production scheduling.

Prof. Wang was a recipient of the National Natural Science Fund for Distinguished Young Scholars of China, the National Natural Science Award (Second Place) in 2014, the Science and Technology Award of Beijing City in 2008, and the Natural Science Award (the First Place in 2003, and the Second Place in 2007) nominated by the Ministry of Education of China. He is currently the Editor-in-Chief of the *International Journal of Automation and Control* and an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, *Swarm and Evolutionary Computation*, and *Expert Systems With Applications*.