A Learning-based Memetic Algorithm for Energy-Efficient Flexible Job Shop Scheduling With Type-2 Fuzzy Processing Time

Rui Li, Wenyin Gong, *Member, IEEE*, Chao Lu, and Ling Wang *Member, IEEE*

Abstract—**Green flexible job shop scheduling problem (FJSP) aims to improve profit and reduce energy consumption for modern manufacturing. Meanwhile, FJSP with type-2 fuzzy processing time is proposed to predict the uncertainty in timing constrain for better simulating the practical production. This study addresses the multi-objective energy-efficient flexible job shop scheduling problem with type-2 processing time (ET2FJSP), where the minimization of makespan and total energy consumption are considered simultaneously. The previous studies do not propose the model verification and energy-saving strategy. Moreover, the best parameters required by an algorithm in different stage is different. Therefore, we propose a mixed integer linear programming model and design a learning-based reference vector memetic algorithm (LRVMA). Its main features are: i) four problem-specific initial rules are presented for initialization to generate diverse solutions; ii) four problem-specific local search methods are incorporated to enhance the exploitation; iii) an effective solution selection method depending on the Tchebycheff decomposition strategy is utilized to balance the convergence and diversity; iv) a reinforcement learning-based parameter selection strategy is proposed to improve the diversity of nondominated solutions, and v) an energy-saving strategy is designed to reduce energy consumption. To verify the effectiveness of LRVMA, it is compared against other related algorithms. The results demonstrate that LRVMA outperforms the compared algorithms for solving ET2FJSP.**

Index Terms—**Flexible job shop scheduling, type-2 fuzzy processing time, energy-efficient, reinforcement learning, memetic algorithm.**

I. Introduction

With the development of economic globalization, modern manufacturing has met unprecedented challenges. Some concepts put forward higher requirements to traditionally flexible manufacturing such as Industrial 4.0 (apply intelligent techniques to promote industrial revolution) and Carbon neutrality (the total amount of greenhouse gas emissions produced by a country, enterprise, or et.al within a certain period shall be offset using afforestation, energy conservation). Classical flexible job shop scheduling problem (FJSP) contains two subproblems including determining the sequence of all operations and assigning a machine for each operation, which is widely applied in large equipment manufacturing. However, the existing FJSP [1], [2], [3], [4] can not satisfy the more personalized market requirement. Among most models, the processing time in FJSP is an ideal fixed value. However, in practical flexible manufacturing, the processing time is uncertain and floats in an interval, which has the lower bound of the earliest start time and upper bound of the latest start time [5]. Moreover, predicting the processing time and determining the schedule is important because the number of operations of large equipment is too much resulting in its production cycle of several months. Furthermore, predicting the range of completion times of large equipment can bring significant profits to enterprise. Thus, a fuzzy processing time was introduced for solving this problem [6]. Besides, the fuzzy processing time applies the fuzzy membership function to efficiently predict the possibility of different processing time. The FJSP with fuzzy processing time (FFJSP) has been studied for several years [7]. Various algorithms have been proposed for the single-objective FFJSP, such as [8], [9], [10], [11], [12], [13]. Besides, recently, the multi-objective FFJSP has also obtained increasing attention [14], [15], [16]. Most of them applied classical triangular fuzzy number (TFN). Nonetheless, with the development of modern manufacturing, traditional TFN can not efficiently represent the high level of uncertainty for the complex and large-scale scheduling system. To achieve automation and accuracy, a type-2 fuzzy processing time (T2FPT) is proposed for better predicting the uncertainty in timing constrain approximately [17]. However, there is little research on FJSP with T2FPT [18], [19].

On the other hand, many industries pay growing attention to green manufacturing due to the serious global warming and frequent occurrence of extreme weather. A series of environmental protection policies have been proposed, and energy-efficient manufacturing has been an emergency topic [20]. Therefore, many scheduling models have been combined with energy-efficient strategies. These green scheduling models have become more important [21], [22], [23], [24]. For example, the energy-aware distributed hybrid flow shop [25], [26], energy-efficient distributed permutation flow shop [27], [28], energy-efficient distributed flow shop with heterogeneous factory [29], energy-efficient distributed flexible job shop scheduling problem [30], and energy-efficient distributed heterogeneous welding flow shop [31]. Furthermore, in modern manufacturing, energy is mainly consumed by machines at running and idle states. There is a large potential space for energy saving, such as turning the machines off during

This work was partly supported by the National Natural Science Foundation of China under Grant Nos. 62076225 and 61873328, and the Natural Science Foundation for Distinguished Young Scholars of Hubei under Grant No. 2019CFA081. *(Corresponding authors: Wenyin Gong & Ling Wang.)*

R. Li, W. Gong, and C. Lu are with School of Computer Science, China University of Geosciences, Wuhan 430074, China. (Email: wygong@cug.edu.cn)

L. Wang is with Department of Automation, Tsinghua University, Beijing 100084, China. (Email: wangling@tsinghua.edu.cn)

idling [32], employing a speed scaling strategy [33], selecting machines with low power consumption [19], and reducing idle time to lower energy consumption. Thus, it is worth studying the energy-efficient FJSP with T2FPT modern flexible manufacturing.

Memetic algorithms (MAs), which refer to the evolutionary search framework incorporating local refinements [34], [35], have been successfully applied to a variety of scheduling problems [36], [37]. Reinforcement learning (RL) has strong decision and optimization ability. Cooperation between RL and evolutionary algorithms is a promising direction to solve complex scheduling problems [38], [39], [40], [41], [42]. Li applies RL to adjust the reference point dynamically [43] and Du utilizes a deep Q-network with 34 states and nine actions to efficiently describe FJSP [44]. However, to the best of our knowledge, studies on the use of RL-based MA for the multi-objective energy-efficient FJSP with T2FPT (ET2FJSP) is scarce.

We solve the ET2FJSP with a novel RL-based MA, where the makespan and total energy consumption (TEC) are optimized simultaneously. First, a MILP model of ET2FJSP is introduced. Then, to effectively solve the problem, a learning-based reference vector MA (LRVMA) is proposed, where i) the reference vectors are used to ensure diversity; ii) an initial heuristic strategy, which combines three initial rules, is designed to get a population with high quality; iii) a new evolution operator is designed to generate offspring; iv) variable neighborhood search (VNS) with four problem-specific local search methods is performed to improve exploitation; v) an energy-saving strategy is presented to reduce the TEC efficiently; and vi) an RL-based parameter adaption strategy is developed to make the algorithm automatically select the best parameter, and an elite archive is utilized to reserve historical elite solutions. To verify the performance of LRVMA, extensive experiments have been carried out and the comparative results prove the effectiveness of the proposed algorithm in solving the ET2FJSP.

The rest of this paper is organized as follows. The MILP model for ET2FJSP is established in Section II. Section III describes the proposed algorithm in detail. The experimental study is provided in Section IV. Finally, Section V concludes this work.

II. Description of ET2FJSP

A. T2FPT basic concepts

A type-2 fuzzy set $\tilde{F}$ consists of elements x and membership function $\mu_{\tilde{F}}(x)$. $\mu_{\tilde{F}}(x)$ means the possibility of x belonging to $\tilde{F}$. All of x belong to a definite set X. The definition of fuzzy set is given as follows:

$$\tilde{F} = \{x, \mu_{\tilde{F}}(x) | \forall x \in X\}, 0 \leqslant \mu_{\tilde{F}}(x) \leqslant 1. \tag{1}$$

The classical set is definite. When $\mu_{\tilde{F}}(x) = 1$, the type-2 fuzzy set is transferred to a classical set.

$$F = \{x, \mu_{\tilde{F}}(x) = 1 | \forall x \in X\}. \tag{2}$$

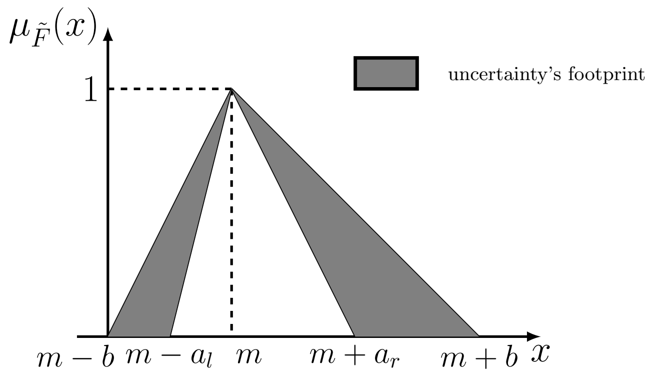

Fig. 1. Nonsymmetrical triangular interval T2FPT (Type-I).

The footprint of uncertainty (FOU) of $\tilde{F}$ is calculated as follows:

$$FOU(\tilde{F}) = \cup_{x \in X} J_x. \tag{3}$$

The upper membership function (UMF) and lower membership function (LMF) of $\tilde{F}$ are donated as $\overline{\mu}_{\tilde{F}}(x)$ and $\underline{\mu}_{\tilde{F}}(x)$, $\forall x \in X$, respectively.

$$\overline{\mu}_{\tilde{F}}(x) \equiv \overline{FOU(\tilde{F})} \quad \forall x \in X. \tag{4}$$

$$\underline{\mu}_{\tilde{F}}(x) \equiv \underline{FOU(\tilde{F})} \quad \forall x \in X. \tag{5}$$

The way to calculate the centroid of nonsymmetric triangular interval T2FPT is given as follows, where the lower membership function is not symmetric. Fig. 1 presents a chart for this type of system.

$$\begin{aligned} C_{\tilde{F}} \approx \Big[& m - \frac{(b-a_r)(a_l+2a_r+b)}{6(a_l+a_r)}, \\ & m + \frac{(b-a_l)(2a_l+a_r+b)}{6(a_l+a_r)} \Big]. \end{aligned} \tag{6}$$

B. T2FPT Operators

Given two interval type-2 fuzzy variables $\tilde{s} = (s_1, s_2, s_3, s_4, s_5)$ and $\tilde{t} = (t_1, t_2, t_3, t_4, t_5)$, then the addition and ranking operators are expressed as follows:

(1) Addition operator.

$$\tilde{s} \pm \tilde{t} = (s_1 \pm t_1, s_2 \pm t_2, s_3 \pm t_3, s_4 \pm t_4, s_5 \pm t_5,). \tag{7}$$

(2) Ranking operator[18], [45], [46].

$$C_{\tilde{s}} = [C_{\tilde{s}}^L, C_{\tilde{s}}^U] \approx \begin{bmatrix} s_3 - \dfrac{(s_5-s_4)(s_5+2s_4-s_2-2s_3)}{6(s_4-s_2)}, \\ s_3 + \dfrac{(s_5+s_2-2s_3)(s_5+s_4-2s_2)}{6(s_4-s_2)} \end{bmatrix} \tag{8}$$

- *Condition 1*: If $C_{\tilde{s}} > (<) C_{\tilde{t}}$, then $\tilde{s} > (<) \tilde{t}$; otherwise, check condition 2.
- *Condition 2*: If $s_3 > (<) t_3$, then $\tilde{s} > (<) \tilde{t}$; otherwise, check condition 3.
- *Condition 3*: If $(s_5 - s_1) > (<) (t_5 - t_1)$, then $\tilde{s} > (<) \tilde{t}$.

For example, there are two T2FPT, $\tilde{A}$=(1, 2, 3, 4, 5) and $\tilde{B}$=(6, 7, 8, 9, 10). Referring to ranking operator, $C_{\tilde{A}} =$

$[1.33, 4.67]$; $C^L_{\tilde{A}}, C^U_{\tilde{A}} = (1.33 + 4.67)/2 = 3$; $C_{\tilde{B}} = [6.33, 9.67]$; $C^L_{\tilde{B}}, C^U_{\tilde{B}} = (6.33 + 9.67)/2 = 8$. Therefore, $C_{\tilde{A}} < C_{\tilde{B}}$.

C. Problem Description

The ET2FJSP can be described as follows. There is a set of n jobs, $I = \{1, 2, ..., i, ..., n\}$ and a set of m machines, $M = \{1, 2, ..., k, ..., m\}$. Each job I_i has an operation set, $J_i = \{O_{i,1}, O_{i,2}, ..., O_{i,j}, ..., O_{i,n_i}\}$. Each operation can be processed on some machines or all machines. The processing time is a T2FPT $\tilde{P}^O_{i,j,k} = (p_1, p_2, p_3, p_4, p_5)$. ET2FJSP includes two subproblems: machine assignment and operation sequencing. The former is that each operation selects a machine from its candidate set. The latter is to schedule all operations on all machines to get a satisfactory schedule. The assumptions of ET2FJSP are given as follows:

- All jobs and machines are available at time zero.
- Each machine can at only process one operation at a time. Interruption are not considered for each operation.
- All the processing dates, such as processing time and energy consumption, are T2FPT.
- Each operation can only be assigned to one machine.
- Transportation time, setup time, and their energy consumption are not considered.

D. MILP Model for ET2FJSP

Before modeling this ET2FJSP, the notations used throughout the study are as follows:

- i, i': indices for jobs, $i \in \{1, .., n\}$;
- j, j': indices for operations of jobs, $i \in \{1, .., n_i\}$;
- k, k': indices for machines, $k \in \{1, .., m\}$;
- t: index for position, $t \in \{1, .., p_k\}$;
- n: total number of jobs;
- m: total number of machines;
- n_i: number of operations of jobs;
- $n_{\max}$: maximum number of operations of all jobs;
- p_k: number of positions of machine M_k and $p_k = \sum_{i \in I} \sum_{j \in J_i} x_{i,j,k}$;
- I: set for jobs and $I = \{1, 2, ..., n\}$;
- M: set of machines and $M = \{1, 2, ..., m\}$;
- J_i: set for the operations of jobs I_i and $J_i = \{1, 2, ..., n_i\}$;
- P_k: set of positions of machine M_k and $P_k = \{1, 2, ...t, ..., p_k\}$;
- $P^{'}_k$: set of top p_k-1 positions of machine M_k and $P^{'}_k = \{1, 2, ...t, ..., p_k - 1\}$;
- $O_{i,j}$: The j_{th} operation of job I_i;
- $\tilde{P}^O_{i,j,k}$: The processing time for operation $O_{i,j}$ job I_i processed by machine M_k, which is a T2FPT: $\tilde{P}_{i,j,k} = (p_1$, p_2, p_3, p_4, $p_5)$;
- $\tilde{S}_{i,j}$: the start time of operation $O_{i,j}$;
- $\tilde{C}_{i,j}$: the completion time of operation $O_{i,j}$;
- $\tilde{B}_{k,t}$: the start time of machine M_k on position t;
- $\tilde{F}_{k,t}$: the completion time of machine M_k on position t;
- Pr^{Idle}_k idle power of machine M_k;
- $Pr^O_{i,j,k}$ processing power of operation $O_{i,j}$ processed by machine M_k;
- EC_W: total processing energy consumption;
- EC_I: total idle energy consumption;
- $E^{off/on}_{i,j,k}$: energy consumption required to use the turn off/on strategy;
- $EMI_{k,t}$: idle energy consumption of machine M_k on position t;
- TEC total energy consumption;
- $C_{\max}$: the makespan of a schedule;
- T_b: the break-even duration;
- T^{off}: allowable times of turn off/on strategy;
- Π: a feasible solution representation;
- L: a large number for maintaining the consistency of the inequality;
- $x_{i,j,k}$ binary constant that takes 1, if operation $O_{i,j}$ processed by machine M_k;
- $\mathbf{X}_{i,j,k,t}$: If operation $O_{i,j}$ is processed on machine M_k in position $P_{k,t}$, the value is set to 1; otherwise, it is set to 0; $P_{k,t}$ is the t_{th} value in set P_k.
- $\mathbf{Y}_{k,t}$: If machine M_k is processing an operation in position t, the value is set to 1; otherwise is set to 0;
- $H_{i,j,k}$: equals 1 if machine M_k is turned off when no operation is processed on it before operation $O_{i,j}$ arrives.

The objectives of ET2FJSP include makespan and TEC, which are elaborated as follows:

1) *Makespan criterion*: Makespan is usually considered as the economic criterion in scheduling problems. That is, makespan can reflect the production benefit of an enterprise to some extent. Thus, the makespan C_{max} objective of ET2FJSP can be defined as:

$$\min \ F_1 = C_{\max} = \max \left\{ \tilde{C}_{i,n_i} \right\}, 1 \leq i \leq N. \tag{9}$$

2) *TEC criterion*: TEC during the manufacturing process can be seen as one key environmental indicator. The TEC criterion reflects the carbon emission of an enterprise at the manufacturing system layer. In this study, the second objective is to minimize TEC through the following formula:

$$\begin{aligned} \min \ F_2 = TEC &= \sum_{i=1}^{M_k} \sum_{j=1}^{\Theta_i} \int_{t=0}^{\tilde{C}_{i,j}} \theta_i(t) dt \\ &= EC_W + EC_I, \end{aligned} \tag{10}$$

$$EC_W = \sum_{i \in I} \sum_{j \in J_i} \sum_{k \in M} \sum_{t \in P_k} Pr^O_{i,j,k} \cdot \tilde{P}^O_{i,j,k} \cdot \mathbf{X}_{i,j,k,t}, \tag{11}$$

$$\begin{aligned} EC_I = \sum_{k \in K} \sum_{t \in P_k} \Big[& Pr^{idle}_{i,j,k} \cdot (\tilde{S}_{i,j} - \tilde{F}_{k,t}) \\ & \cdot (1 - H_{i,j,k}) + E^{off/on}_{i,j,k} \cdot H_{i,j,k} \Big], \end{aligned} \tag{12}$$

where $\theta_i(t)$ is the instantaneous power of machine M_k at time t. In summary, although this problem has been studied in [18], the MILP model of ET2FJSP with two objective functions is first proposed in this study. The MILP model is described as follows:

$$\begin{cases} \min \ F_1 = C_{\max}, \\ \min \ F_2 = TEC. \end{cases} \tag{13}$$

It is worth noting that F_1 has an implicit conflict against F_2. F_1 depends on the operation sequencing and machine selection, whereas, F_2 depends on minimizing processing time and idle time. In other words, blindly reducing F_2 might increase F_1 sharply. By adjusting the processing time, the solution with lower F_2 but higher F_1 than solution with the smallest F_1 can always be found. Thus, the problem researched in this study is a multi-objective optimization problem (MOP).

subject to:

$$\sum_{k\in K}\sum_{t\in P_k} \mathbf{X}_{i,j,k,t} = 1, \forall i \in I, j \in J_i, \tag{14}$$

$$\sum_{k\in K}\sum_{t\in P_k} \mathbf{Y}_{k,t} \leqslant T^{off}, \forall i \in I, j \in J_i, \tag{15}$$

$$\tilde{S}_{i,n_i} + \sum_{k\in K}\sum_{t\in P_k} \tilde{P}^O_{i,n_i,k} \cdot \mathbf{X}_{i,n_i,k,t} \leqslant C_{\max}, \forall i \in I, \tag{16}$$

$$\tilde{S}_{i,j} + \sum_{k\in K}\sum_{t\in P_k} \tilde{P}^O_{i,j,k} \cdot \mathbf{X}_{i,j,k,t} \leqslant \tilde{S}_{i,j+1}, \forall i \in I, j \in J_i - 1, \tag{17}$$

$$\sum_{i\in I}\sum_{j\in n_{\max}} \mathbf{X}_{i,j,k,t} \leqslant 1, \forall k \in K, t \in P_k, \tag{18}$$

$$\sum_{i\in I}\sum_{j\in n_{\max}} \mathbf{X}_{i,j,k,t} \geq \sum_{i\in I}\sum_{j\in n_{max}} \mathbf{X}_{i,j,k,t+1}, \forall k \in K, t \in P_k^{'}, \tag{19}$$

$$\begin{aligned} \tilde{B}_{k,t+1} - \tilde{B}_{k,t} \geq \sum_{i\in I}\sum_{j\in n_{\max}} \mathbf{X}_{i,j,k,t+1} \cdot \tilde{P}^O_{i,j,k} \\ + T_b, \forall k \in K, t \in P_k - 1, \end{aligned} \tag{20}$$

$$\begin{aligned} & EMI_{k,t} + Y_{k,t} * L \geq \Big(\tilde{B}_{k,t+1} - \tilde{B}_{k,t} \\ & - \sum_{i\in I}\sum_{j\in n_{\max}} \mathbf{X}_{i,j,k,t+1} \cdot \tilde{P}^O_{i,j,k}\Big) \cdot Pr_k^{Idle}, \\ & \forall k \in K, t \in P_k^{'}, \end{aligned} \tag{21}$$

$$\tilde{B}_{k,t} \geq \tilde{S}_{i,j} - L\cdot(1-\mathbf{X}_{i,j,k,t+1}), \forall i \in I, j \in J_i, k \in K, t \in P_k, \tag{22}$$

$$\tilde{B}_{k,t} \leqslant \tilde{S}_{i,j} - L\cdot(1-\mathbf{X}_{i,j,k,t+1}), \forall i \in I, j \in J_i, k \in K, t \in P_k, \tag{23}$$

$$0 \leqslant \tilde{S}_{i,j} \leqslant L, i \in I, j \in J_i, \tag{24}$$

$$\tilde{B}_{k,t} \geq 0, k \in K, t \in P_k, \tag{25}$$

$$EMI_{k,t} \geq 0, k \in K, t \in P_k, \tag{26}$$

where Eq. (13) has two objectives, including the makespan and TEC. Eq. (14) guarantees that each job must be processed on one machine at a time. Eq. (15) limits the maximum times of Turn Off/On strategy. Eq. (16) defines the makespan. Eq. (17) guarantees that the proceeding of last operation must have been processed before the current operation starts. Eq. (18) indicates that only one operation can be assigned to a position of a machine. Eq. (19) forces an operation to be assigned to the preceding positions only when they are occupied by other operations. Eq. (20) assures that if a Turn Off/On strategy is implemented between two adjacent positions of a machine, the idle time is no less than the break-even of the machine. Eq. (21) assures the energy consumption between two adjacent positions of a machine. Eqs. (22)-(23) define the relation between machine start time and operation start time. Eqs. (24)-(26) are value range limitations.

III. Our Approach: LRVMA

The proposed LRVMA is a reference-vector-based optimization algorithm and its framework is depicted in Algorithm 1. The motivation for designing LRVMA is as follows: First, the memetic algorithm is popularity due to its simple algorithm structure and excellent performance. Second, the memetic algorithm can balance the global and local searches. These merits drive us to propose an LRVMA for this flexible job shop scheduling problem with type-2 fuzzy processing time. LRVMA involves four innovations: (1) A cooperative initialization heuristic is proposed. This initialization heuristic generates high-quality initial solutions and guarantees the diversity of the population. (2) A knowledge-based local search strategy is designed to improve the exploitation capacity. This local search guides candidate solutions toward potential promising regions by utilizing problem properties. (3) A new energy conservation strategy is embedded into the proposal. This energy-saving will reduces energy consumption slows down machine deterioration. (4) A parameter adaption strategy based on RL is proposed to improve the diversity of nondominated solutions. This strategy makes the algorithm automatically select the most suitable parameter during iteration. Meanwhile, an elite archive is applied to preserve historical elite nondominated solutions to increase the diversity of the final Pareto Front (PF).

A. Encoding and decoding

In this paper, two one-dimensional vectors (OS, MS) are used to represent the solution (or agent) [18]. The operation sequence is used to indicate the processing sequence for all operations. Moreover, machine selection is used to represent the assigned machine for each operation. The two vectors are set with the same length which is equal to the total number of operations. Fig. 2 gives an example of solution representation which has three jobs with different colors and three machines.

Solutions are decoded to assign the appropriate processing time for each operation on its selected machine according to the operation sequence. When a solution is decoded, the OS vector is converted into a sequence of operations. Then each operation is assigned to a selected machine from the MS

Algorithm 1: The Framework of proposed LRVMA.

```
Input: Parameters like PopSize($ps$), Maximum number of function evaluations
       ($MaxNFEs$), discount factor($\gamma$), learning rate ($\alpha$), greedy factor ($\epsilon$)
Output: Nondominated solutions
1  $\mathcal{P} \leftarrow$ initialize population($ps$).
2  t$\leftarrow$ 0, $C_t = C_{t+1} = D_t = D_{t+1} = 0$.
3  $Q(4,4) \leftarrow 0$, $A \leftarrow \{5,10,15,20\}$, $\Omega \leftarrow []$.
4  $[\lambda, NM] \leftarrow$ initialize reference vector($\mathcal{P}_t, ps+1, A(4)$).
5  while $t \leqslant MaxNFEs$ do
6      $\Delta C = C_{t+1} - C_t, \Delta D = D_{t+1} - D_t, C_t = C_{t+1}, D_t = D_{t+1}$
7      $S_t \leftarrow$ GetState($\Delta C$, $\Delta D$).
8      $A_t \leftarrow$ SelectAction($S_t$, $A$, $\epsilon$, $Q$).
9      $z^* \leftarrow (x_0, y_0) \leftarrow$(min$C_{\tilde{F_1}}(\mathcal{P}_t)$,min$C_{\tilde{F_2}}(\mathcal{P}_t)$).
10     for $i$=1 to $ps$ do
11         $X_1 \leftarrow$ Evolution operator($\mathcal{P}_t(i)$).
12         $X_2 \leftarrow$ Variable neighborhood search($X_1$).
13         $X_3 \leftarrow$ Energy saving($X_2$).
14         $x_0 \leftarrow$min$\{C_{\tilde{F_1}}(X_1), C_{\tilde{F_1}}(X_2), C_{\tilde{F_1}}(X_3), x_0\}$.
15         $y_0 \leftarrow$min$\{C_{\tilde{F_2}}(X_1), C_{\tilde{F_2}}(X_2), C_{\tilde{F_2}}(X_3), y_0\}$.
16         for $j$=1 to $A_t$ do
17             $k \leftarrow NM(j)$.
18             Update neighborhood($\lambda_k$,$\mathcal{P}_t(k)$, $X_3$, $z^*$).
19     $PF_t \leftarrow$ Pareto solution set($\mathcal{P}_t$).
20     $\Omega \leftarrow \Omega \cup PF_t$.
21     Delete repeat solution of $\Omega$.
22     $\Omega \leftarrow$ Fast Nondominated Sort($\Omega$).
23     $C_{t+1} \leftarrow GD(PF_t), D_{t+1} \leftarrow Diversity(PF_t)$.
24     $\Delta C = C_{t+1} - C_t, \Delta D = D_{t+1} - D_t$.
25     if $\Delta D_{t+1} > 0$ then
26         $R_t = 10$
27     $S_{t+1} \leftarrow$ GetState($\Delta C$, $\Delta D$).
28     $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha * (R_t + \gamma * maxQ(S_{t+1}, :) - Q(S_t, A_t))$
```

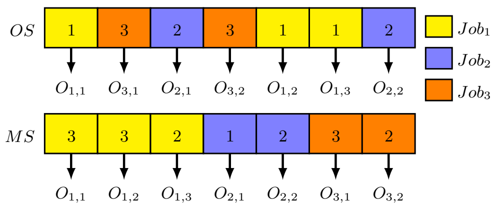

Fig. 2. Solution representation.

vector. Finally, the type-2 fuzzy processing times are assigned to the operation. Each solution (agent) is decoded into a fuzzy schedule. The processing time is a T2FPT.

B. Initialization strategy

A cooperative initialization mechanism inspired by Gao [47] and use four different rules to generate a high-quality population. The procedure is as follows:

- Step 1: For $\lfloor ps/5 \rfloor$ (round down $ps/5$ to an integer) of all solutions, assign all operations with their minimum processing time machine.
- Step 2: For $\lfloor ps/5 \rfloor$ of all solutions, assign all operations with the current minimum workload machine as evenly as possible so that the machine load is balanced.
- Step 3: For $\lfloor ps/5 \rfloor$ of all solutions, assign all operations with the current minimum finish time machine.
- Step 4: For $\lfloor ps/5 \rfloor$ of all solutions, order all operations by the number of selectable machines in ascending. Then, assign all operations with the current minimum workload machine.
- Step 5: The rest of the solutions are randomly generated to improve the diversity of the population.

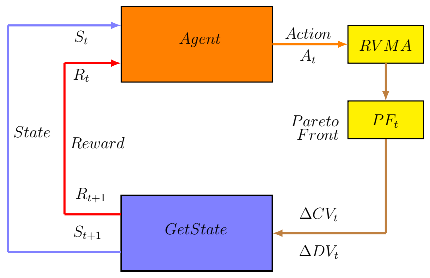

Fig. 3. RL-based parameter selection.

C. Parameter adaption strategy based on Q-learning

1) Motivation: The proposed LRVMA relies on the reference vector, and the diversity of Pareto solutions depends on the parameter (i.e. the size of neighborhood Φ[48]). In the early stage of evolution, the bigger Φ will increase convergence. However, there is no obvious regularity between diversity and Φ. We design a parameter adaption strategy based on Q-learning (Q-PAS) to automatically adjust Φ to increase the diversity of Pareto solutions.

2) Model of Q-PAS: Fig. 3 shows the updated model of the proposed parameter adaption strategy. The agent will select an action A_t based on its state S_t at time t, and A_t is the input parameter Φ of RVMA. Then, after RVMA evolves one generation, the nondominated solution set PF_t can be obtained. Next, according to the metrics change of PF_t, the next state is S_{t+1} and reward is R_{t+1} of agent. Finally, the Q-table is updated (Algorithm 1 line 29). α is the learning rate, which ranges from 0 to 1. γ is the discount factor, which is also in $[0, 1]$. When γ is close to 1, the Q-value is more affected by the future state. In contrast, when γ is close to 0, the Q-value focuses more on the current state.

3) Action and State definition: **Action definition**: Φ has four levels: {5, 10, 15, 20}. At the beginning of the iteration, each agent (solution) selects an action (Φ_i) to determine the size of the neighborhood.

State definition: Under certain conditions the agent selects the parameter Φ_i, and we can get the PF_t after this iteration. The change of the PF_t's convergence and diversity reflects whether this time parameter selection is meaningful. To represent the agent's state, two MOP metrics: $CV = GD(P, P^*) = \frac{\sqrt{\sum_{y \in P} \min_{x \in P^*} dis(x,y)^2}}{|P|}$ and $DV = \Delta = \frac{\sum_{i=1}^{N-1} |d_i - \bar{d}|}{(N-1)\bar{d}}$ are applied to measure the convergence and diversity of PF_t. Those two metrics are referring to [49], where P is the PF_t calculated by the algorithm in each generation and P^* is the predefined reference point. The smaller CV, the better the convergence, and $\Delta CV < 0$. In DV, d_i is the Euclidean distance of two adjacent points in PF_t. $\bar{d}$ is the average value of d_i. The bigger DV, the better diversity, and $\Delta DV > 0$. The steps for selecting the state and action are as follows:

Get state: During the evolution, the ΔCV and ΔDV of PF_t have four combination conditions. **State1:** $\Delta CV > 0$ and $\Delta DV > 0$. **State2:** $\Delta CV > 0$ and $\Delta DV \leqslant 0$; **State3:** $\Delta CV \leqslant 0$ and $\Delta DV > 0$; **State4:** $\Delta CV \leqslant 0$ and $\Delta DV \leqslant 0$; (1) and (2) occur during the early evolution period. (3) and (4) will happen when the population has converged. These four

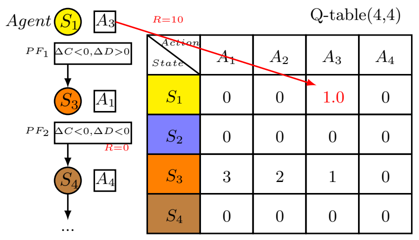

Fig. 4. An example for Q-PAS.

conditions are regarded as four **states** of the agent.

Select action: If $rand > \epsilon$ or all Q-table(S_t,:)=0, then action $A_t = \Phi$(rand), else action select the max Q-value action in Q-table(S_t,:);

4) Reward definition: After executing an action, the agent will get a reward. If $\Delta DV > 0$, $R_t = 10$, else the $R_t = 0$. The parameter adaption strategy is stated in Algorithm 1, rows 5, 8-10, and 25-30.

5) An example: An example for how Q-PAS is executed is given to understand the strategy clearer. In Fig. 4, the initial state of the agent is S_1. Because the row of S_1 in Q-table is empty, an action is randomly selected such as A_3. After inputting A_3 into the evolutionary operator (in line 12-20 in Algorithm 1) , PF_1 can be obtained. Then, the change of convergence ΔCV and diversity ΔDV are calculated. Because $\Delta DV > 0$, Q(1,3) will get a reward $R = 10$ and be updated by the equation in line 30 in Algorithm 1. Because $\Delta CV < 0$ and $\Delta DV > 0$, the next state is S_3. Then, Q(1,3)=Q(1,3)+0.1*(10+0.8*max(Q(3,:))-Q(1,3)) and max(Q(3,:))=0. Thus, Q(1,3)=1. Now, the current state is S_3. Action A_1 with the biggest Q-value is selected. However, the change of diversity ΔDV of PF_2 is less than zero. Thus, no reward can be got and Q(3,1) will not be updated. This above process is repeated until the iteration stop condition is satisfied.

D. Evolution operator

To get a large searching step in objective space, precedence operation crossover (POX) and uniform crossover (UX) are applied [30]. The description is as follows:

POX: (1)Randomly divide the job set into two subsets J_1 and J_2. (2) Select two solutions S_1 and S_2. For each job belonging to J_1, copy its operations into $NewS_1$. For each job belonging to J_2, copy its operations into $NewS_2$. (3) From the part of S_2, copy the operation which does not appear in $NewS_1$ to the vacant positions in $NewS_1$ from left to right according to the order of the sequence in S_2. Do a similar thing to $NewS_2$.

UX: (1)Randomly generate a 0-1 vector whose length equals the total number of operations. (2) Select two solutions M_1 and M_2. Exchange the value in M_1 and M_2, where it is 1 at the same position in 0-1 vector. It is worth noting that UX is applied for machine selection. More details are in [30]

Swap: Randomly select two positions in the operation sequence and exchange the value. **Machine selection mutation**: Randomly select two positions in the machine selection and select a new machine from its candidate set.

E. Variable neighborhood search

Local search is an effective heuristic to improve the quality of solutions in the field of scheduling [29]. In general, makespan depends on the critical operation (i.e., the operation with the maximum completion time), whereas TEC is sensitive to the machine workload and idle time interval of a schedule. Therefore, aiming at the specific problem of ET2FJSP, we designed the following four local search strategies.

Last operation mutation: Find the last finished operation O_{i,n_i}, and move O_{i,n_i} to another machine $M_{k'}$ with minimum processing time.

Maximum workload mutation: Find the maximum workload machine M_k. Randomly select an operation on M_k and move it to another machine $M_{k'}$.

Swap on operation sequence: Randomly choose two positions on the operation sequence and exchange the value.

Insertion on operation sequence: Randomly choose two positions on the operation sequence and insert the latter one in front of the former one.

VNS: Execute each local search strategy once to generate a new solution Π' and replace its parent if possible. Meanwhile, add the Π' to elite archive Ω when it dominates the old one or they do not dominate each other.

F. Energy saving strategy

Energy-saving is a critical step of energy-efficient FJSP. Reducing TEC can improve industry's profit a lot and protect the environment. TEC is sensitive to processing time and idle time. Moreover, if a machine is idle, it will be turned off to save energy until the operation arrives. However, frequently using this turn off/on strategy will damage machine tools, so we present an energy conservation strategy inspired by the literature [25]. The main idea is to rebuild the machine selection vector to reduce the idle time and processing time of each operation. To prove the validity of the energy-saving strategy, two lemmas and their proofs are shown as follows:

Lemma1 Assume Π is the current state of schedule. The arriving operation is $O_{i,j}$. Try assigning $O_{i,j}$ to all machines. If there is no idle time compared with all successor operation $O_{i',j'}$, selecting the machine with minimum processing time will reduce the TEC.

Proof Assume Π_1 is the next state when $O_{i,j}$ selects the machine M_k with minimum processing time. Π_2 is the next state when the M_k is not chosen. $TEC(\Pi_1) = TEC(\Pi) + \tilde{P}_{i,j,k} * Pr^O_{i,j,k}$. $TEC(\Pi_2) = TEC(\Pi) + \tilde{P}_{i,j,k'} * Pr^O_{i,j,k'}$. Because $\tilde{P}_{i,j,k} < \tilde{P}_{i,j,k'}$ and $Pr^O_{i,j,k} = Pr^O_{i,j,k'}$. Thus, $TEC(\Pi_1) < TEC(\Pi_2)$. Therefore, lemma1 is proved.

Lemma2 If idle time occurs when assigning $O_{i,j}$ to all machines, selecting the machine with minimum energy consumption will reduce the TEC.

Proof Assume Pi_1 is the state when $O_{i,j}$ selects the machine M_k with minimum energy consumption. Assume Pi_2 is the state in which $O_{i,j}$ selects the other machine $M_{k'}$. $TEC(\Pi_1) = TEC(\Pi) + TEC_1$. $TEC(\Pi_2) = TEC(\Pi) + TEC_2$. Because $TEC_1 < TEC_2$. Thus, $TEC(\Pi_1) < TEC(\Pi_2)$. Therefore, lemma2 is proved.

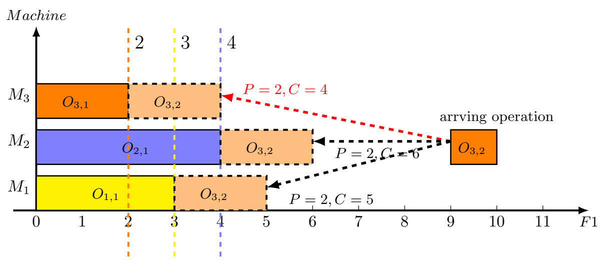

Fig. 5. An example for no idle energy saving

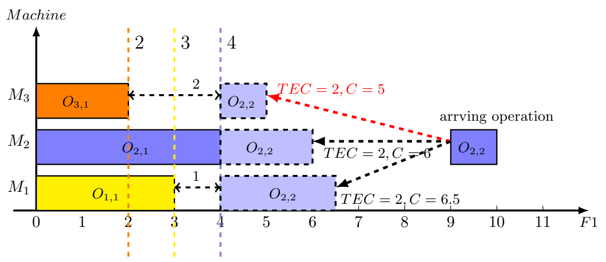

Fig. 6. An example for idle condition energy saving

Fig. 5 shows an example of a no idle situation with three jobs and three machines. For clarity and simplification, a rectangle is used to represent the processing time by transferring T2FPT into its centroid with Eq.(8). Assume that the next operation $O_{3,2}$ arrives and finds that no idle time occurs, for which the finish time of its last operation $C_{3,1}$ is smaller than the other machine's finish time. Moreover, in Fig. 5, we assume the processing time $P_{3,2,k}$ on all machine is the same. Then, the machine with the smallest finish time is selected and M_3 is chosen.

Fig. 6 gives an example of another condition with idle time. $O_{2,2}$ tests each machine, and finds there is idle time on machines M_1 and M_3. To reduce the TEC, the machine with minimal energy consumption should be selected. However, the TEC of all machines is the same. Thus, M_3 with global minimum completion time is selected.

The pseudocode of this strategy is stated in Algorithm 2:

(1) First, machine finish time array $\tilde{F}$, operation finish time array $\tilde{C}$, and operation count OpC are initialized. (2) Second, store the operation index and count into J and OpC. Then, take each operation $O_{I,J(i)}$ from operation sequence, if it is the first operation of job I, then find the machine with minimum finish time. If the number of machine is more than one, then select the machine with minimum T2FPT. (3) Else, if $O(I, J(i))$ is not the first operation of job I, then test all machines to find idle time. (4) If idle time occurs, and find select the minimum energy consumption machine. If there is more than one machine, then select the machine with the minimum finish time after assigning $O_{I,J(i)}$. (5) If idle time exists, the machine with minimum processing time will be chosen. Similarly, if the number of machines is more than one, the machine with the smallest operation completion time $\tilde{C}_{I,J(i)}$ will be chosen.

Algorithm 2: Energy saving strategy.

```
Input: a trial solution containing operation sequence and machine
       selection$\prod(OS, MS)$, elite archive $\Omega$
Output: new solution $\prod'(OS, MS')$,archive $\Omega$.
$\tilde{F} \leftarrow \{\}, \tilde{C} \leftarrow \{\}, OpC \leftarrow zeros\,[1, N]$
for i=1:length($OS$) do
    $OpC(OS(i)) \leftarrow OpC(OS(i)) + 1$
    $J(i) \leftarrow OpC(OS(i))$
for i=1:length($OS$) do
    $I = OS(i)$
    if $J(i)$==1 then
        $MI \leftarrow$ FindMinFinishMachine($\tilde{F}$).
        $MS'(i) \leftarrow$ FindMinProcessMachine($I, J(i), MI$)
        $\tilde{F}_{MS'(i)} \leftarrow \tilde{F}\{MS'(i)\} + \tilde{P}^{O}_{I,J(i),MS'(i)}$
    else
        $Flag \leftarrow$IsIdleTimeExist($\tilde{F}, \tilde{C}_{I,J(i)-1}$).
        if $Flag == 1$ then
            $MS'(i) \leftarrow$ FindMinTECMachine($\tilde{F}, \tilde{C}_{I,J(i)-1}$).
            $\tilde{F}_{MS'(i)} \leftarrow \tilde{C}_{I,J(i)-1} + \tilde{P}^{O}_{I,J(i),MS'(i)}$
        else
            $MS'(i) \leftarrow$ FindMinProcessMachine($MFT, I, J(i)$).
            $\tilde{F}_{MS'(i)} \leftarrow \tilde{F}_{MS'(i)} + \tilde{P}^{O}_{I,J(i),MS'(i)}$
    $\tilde{C}_{I,J(i)} \leftarrow \tilde{F}_{MS'(i)}$
if $\Pi' \succ \Pi$ then
    $\Pi \leftarrow \Pi'$.
    $\Omega \leftarrow \Omega \cup \Pi'$
else if $\Pi'$ and $\Pi$ does not dominate each other then
    $\Omega \leftarrow \Omega \cup \Pi'$
```

G. Update strategy

Diversity is an important metric of MOEAs; uniformly distributed Pareto solutions can provide more and better decisions for industry. Reference vector is a classical method in MOP. Depending on it and the Tchebycheff aggregate function (TAF) [48], solutions can uniformly distribute around the reference vector, which leads to good diversity. TAF merges all objective functions and replaces the strange domination sequence with a wake single-value function so that it can efficiently balance convergence and diversity. The equation of the Tchebycheff aggregate function is as follows:

$$g^{te}(x|\lambda^j, Z^*) = \max_{1 \leqslant i \leqslant m} \{\lambda_i^j |f_i(x) - Z_i^*|\}. \tag{27}$$

Update strategy: Calculate the TAF values of current solution X and all its neighborhood solutions $\mathcal{P}_t(NM)$ sizing A_t. If $g^{te}(X|\lambda^i, Z^*) < g^{te}(\mathcal{P}_t(NM(k))|\lambda^i, Z^*)$, then update the neighborhood solution $\mathcal{P}_t(NM(k))$.

IV. Experimental Results

In Section III, we described our algorithm in detail. In this section, we report on detailed experiments to evaluate the performance of the proposal. LRVMA and the comparison algorithms were coded in MATLAB2016Rb on an Intel Core $i7$ 6700 CPU @ 3.4GHz with 8G RAM. For fairness, all algorithms ran 20 independent times on each instance with the same stop criteria. To verify the convergence and diversity of the proposed algorithm, after 20 independent runs, the mean and standard deviations of metrics were collected.

Three metrics are used to measure the performance of the different algorithms: Hypervolume (HV) [50], Generation

TABLE I
COMPARISON RESULTS WITH THE EXACT CPLEX SOLVER ON SMALL SCALE OF INSTANCES.

Instance	makespan		TEC		time(ms)	
	CPLEX	LRVMA	CPLEX	LRVMA	CPLEX	LRVMA
J3M3O3	202.00	274.00	209.56	219.12	177.00	127.29
J3M4O3	153.00	154.00	168.56	171.52	190.00	132.64
J3M5O3	97.00	97.00	124.56	127.68	191.00	136.02
J4M3O3	173.00	173.00	197.56	196.28	141.00	320.37
J4M4O3	190.00	190.00	232.06	240.14	128.00	151.15
J4M5O3	114.00	160.00	169.56	175.04	156.00	343.73
J5M3O3	188.00	238.00	323.56	324.2	195.00	150.54
J5M4O3	133.00	137.00	180.06	181.46	228.00	943.60
J5M5O3	126.00	137.00	206.56	214.48	260.00	154.63

Distance (GD), and Spread [49]. HV can measure the comprehensive performance of the algorithm. The lower the GD or Spread value, the better the performance of the algorithm. In contrast, the greater the HV value, the better the performance of the algorithm.

A. MILP model validation

A software CPLEX 12.6.3 (trial version) was applied to validate the correction of the proposed MILP model. The MILP model is coded by optimization language which is specific to CPLEX. Moreover, nine small scale of instances were generated for validation. Each instance is named J_M_O which means the number of jobs, the number of machines and the maximum number of operations. The proposed LRVMA is also executed for solving those instances. Table I shows the results. The proposed LRVMA is worse than CPLEX solver because the CPLEX executes the branch and cut algorithm, which is a deterministic and constructive algorithm. However, the results got by CPLEX solver validate the correction of our MILP model. In addition, the results of execution time of finding the best solution are listed in the last two columns. The CPLEX code for the model can be downloaded from https://cuglirui.github.io/Code/EEFJSP.rar

B. Experimental instances

The instances are taken from [18], which can be downloaded from http://ischedulings.com/shares.html. The scales of instances are combined with $n \in \{20, 30, 40, 50, 80, 100\}$ and $m \in \{6, 7, 8, 9, 10\}$. The processing time in each instance is a T2FPT which is $\tilde{P}^{O}_{i,j,k} = (s_1, s_2, s_3, s_4, s_5)$. The power of energy consumption $Pr^{O}_{i,j,k} = 0.5$, $Pr^{Idle}_{k} = 0.04$, and $E^{off/on}_{i,j,k}$=0.06. There are 30 instances in total. Each instance is marked by $JnMm$, which represents the problem with n jobs and m machines.

C. Experimental parameters

The parameter configuration can impact the performance of the algorithm in solving this problem. The proposed LRVMA contains four parameters. The population size ps, discount factor γ, learning rate α, and the greedy factor ϵ. A Taguchi approach of design-of-experiment (DOE)[51] is adopted for testing the best parameter. The parameter level is given as follows:

TABLE II
OVERALL RANKS THROUGH THE FRIEDMAN TEST AMONG ALL LRVMA VARIANTS (LEVEL OF SIGNIFICANCE $\alpha = 0.05$).

MOEAs	HV		GD		Spread	
	rank	p-value	rank	p-value	rank	p-value
LRVMA1	4.967	3.36E-20	4.967	6.54E-20	3.033	1.15E-03
LRVMA2	4.000		4.000		**1.933**	
LRVMA3	1.967		1.933		3.400	
LRVMA4	2.400		2.333		3.233	
LRVMA	**1.667**		**1.767**		3.400	

- $ps = 40, 60, 80, 100;$
- $\gamma = 0.9, 0.8, 0.7, 0.6;$
- $\alpha = 0.1, 0.2, 0.3, 0.4;$
- $\epsilon = 0.95, 0.90, 0.85, 0.80.$

An orthogonal array $L_{16}(4^4)$ was adopted in this calibration experiment. For fairness, each parameter ran 20 independent times. The max generation G=200. We collected the average and standard deviations for 20 runs. Fig.7 shows the main effects plot of four parameters for all metrics. The higher the HV metrics values , the better the performance. In contrast, the lower the GD and Spread metrics, the better the performance. Based on comprehensive observation, the best parametric value configuration was $ps = 100$, $\gamma = 0.8$, $\alpha = 0.1$, and $\epsilon = 0.95$.

D. Effectiveness of each improvement of the LRVMA

To verify the effectiveness of each improvement part of the proposed algorithm, we compared four variants of LRVMA, where LRVMA1 denotes LRVMA without the initialization heuristic; LRVMA2 represents LRVMA without VNS; LRVMA3 denotes LRVMA without the proposed energy-saving strategy, and LRVMA4 donates LRVMA without the proposed Q-PAS.

In the supplementary file, Table S-I lists statistical metric values over 20 times on 30 instances, where symbol ”+/-” represents the variant algorithm is significantly superior/inferior to the proposal. Symbol ”=” represents no significant relationship between the variant algorithm and the proposed algorithm. The best mean values are marked in **bold**. The last row summarizes the number of the symbol ”-/=/+”.

Table II records the Friedman rank test results among all variant algorithms in all instances, where the confidence level $\alpha = 0.05$. Generally, according to the performance of the three metrics, the proposed LRVMA has the highest total ranking, which means LRVMA outperformed its four variants.

TABLE III
OVERALL RANKS THROUGH THE FRIEDMAN TEST AMONG COMPARED ALGORITHMS (LEVEL OF SIGNIFICANCE $\alpha = 0.05$).

MOEAs	HV		GD		Spread	
	rank	p-value	rank	p-value	rank	p-value
NSGA-II	6.933	8.83E-31	6.833	5.60E-26	4.200	3.86E-09
MOEA/D	5.800		5.733		**2.800**	
AR-MOEA	3.700		3.433		3.033	
IAIS	2.233		2.800		4.867	
C-TSEA	4.633		4.300		3.100	
FBEA	3.600		3.633		5.967	
LRVMA	**1.100**		**1.267**		4.033	

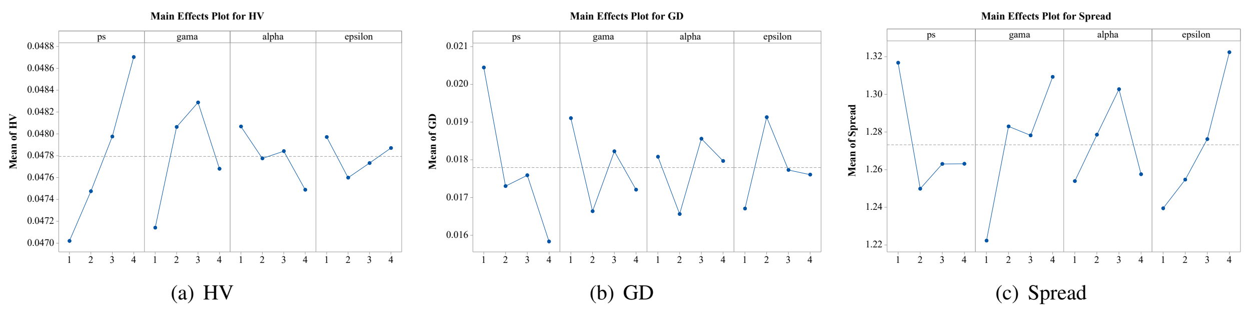

Fig. 7. Main effects plot of three metrics: (a) HV, (b) GD, and (c) Spread.

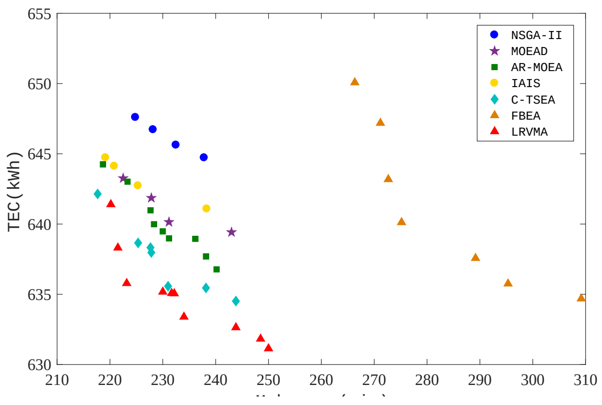

Fig. 8. Pareto Front approximations by different algorithms on J20M6

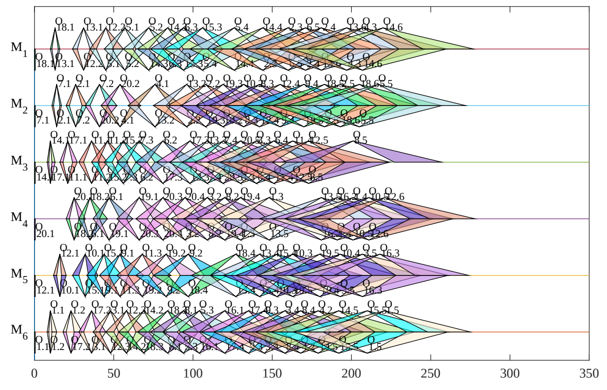

Fig. 9. Gantt chart of solution A with the best makespan on J20M6

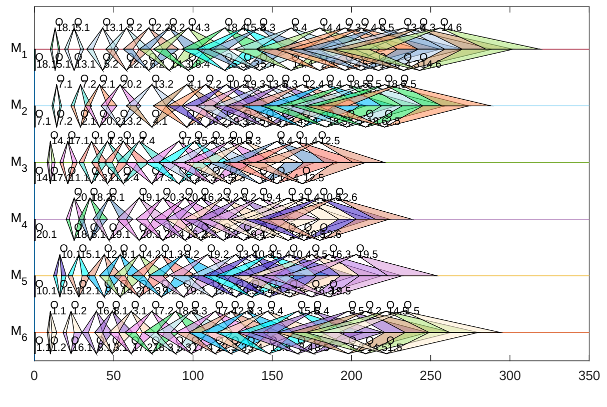

Fig. 10. Gantt chart of solution B with the best total energy consumption on J20M6

E. Comparison and discussion

To further evaluate the effectiveness of the proposed LRVMA, it was compared to the current mainstream MOEAs MOEA/D [48] and NSGA-II [49], the recently proposed MOEAs AR-MOEA [52] and C-TSEA [53], and two new algorithms for ET2FJSP named IAIS [18] and FBEA [6]. The parameters were calibrated as in section IV-C. Due to space limitations, we only provide the final parameter settings of all the compared algorithms as follows: mutation rate $Pm = 0.8$ for all algorithms, neighborhood size $T = 10$ for MOEA/D. The SA-based exploration heuristic (Ts) is 0.5, clone number (nc) is 10 and crowding threshold $CRmax$ is 1×10^{-4}. To conduct a fair comparison, these algorithms used the common termination criterion (i.e., MaxIter=200). They also use the aforementioned encoding and decoding mechanism, and evolution operator (except for FBEA due to its special evolution strategy). All algorithms were implemented them by MATLAB2016Rb. Because of the complexity of instances, 20 independent runs of this comparison experiment were executed in all 30 instances. Table III lists the Friedman rank test results among all comparison algorithms in all instances, where the confidence level $\alpha = 0.05$. As observed in Table III, LRVMA ranks first for the HV and GD metrics but ranks 4th for Spread. However, regarding the Spread, LRVMA is inferior to the MOEAs but superior to the IAIS, NSGA-II and FBEA. Furthermore, Table S-II records statistical results (mean and standard deviations) in 30 instances, where symbol "-/=/+" means significantly inferior, equals, or superior to LRVMA. As observed in the three tables, LRVMA is significantly superior to all comparison algorithms for HV and GD metrics in more than half of the instances. Regarding Spread, there is no significant difference among the algorithms. The success of LRVMA lies in its design. First, a cooperative initialization strategy is used to generate a high-quality population. Second, through RL, the best parameter is selected to improve the diversity of the PF. Third, this VNS consists of four problem-specific local search strategies that enhance convergence. Finally, an energy-saving strategy reduces TEC, which can increase the convergence of the trial solutions towards the PF.

Figures 8, 9, and 10 illustrate the behavior of different algorithms. Fig.8 shows all algorithms' PF with the best HV. To better demonstrate the solutions, each objective value is transformed from T2FPT to the centroid value by Eq. (8). Regarding the convergence and diversity of the PF, LRVMA can obtain better solutions than its competitors, which means

LRVMA can find closer approximations towards the real PF. Meanwhile, we can see from Fig. 8 that there is an obvious conflicting relationship between TEC and makespan. Fig. 9 presents a Gantt chart of solution A with the best makespan ($\tilde{F}_1 = (174, 194, 221, 249, 278)$ min and $\tilde{F}_2 = (504.14, 563.1,$ 649.06, 726.02, 815.64)kWh). Fig. 10 shows the solution with the best TEC ($\tilde{F}_1 = (198, 222, 256, 285, 319)$ min and $\tilde{F}_2 = (491.28,$ 548.36, 633.64, 708.68, 797.4)kWh). Based on the results, the proposed LRVMA can solve ET2FJSP well.

V. Conclusion

Green scheduling has become one of the modern production models in large industries. Although the flexible green scheduling problem with fuzzy processing time has widely appeared in modern manufacturing, FJSP with type2 fuzzy processing time has been seldom researched. RL has become an efficient technique for parameter selection. However, it has not been applied to this problem. Thus, we have proposed a MILP model and designed an LRVMA for this problem. Experimental results show that LRVMA is significantly better than other algorithms in terms of obtaining solutions with better convergence and diversity.

In our future work, we will consider the following tasks: i) applying LRVMA to other scheduling problems such as distributed scheduling; ii) taking realistic constraints in ET2FJSP; and iii) considering transferring small-scale instances to large-scale instances.

The source code of LRVMA is available online at https://wewnyin.github.io/wenyingong/Software/LRVMA-code.zip.

References

[1] F. Zhang, Y. Mei, S. Nguyen, and M. Zhang, "Evolving scheduling heuristics via genetic programming with feature selection in dynamic flexible job-shop scheduling," *IEEE Transactions on Cybernetics*, vol. 51, no. 4, pp. 1797–1811, 2021.

[2] K. Gao, P. N. Suganthan, T. J. Chua, C. S. Chong, T. Cai, and Q. Pan, "A two-stage artificial bee colony algorithm scheduling flexible job-shop scheduling problem with new job insertion," *Expert Systems with Applications*, vol. 42, no. 21, pp. 7652–7663, 2015.

[3] K. Gao, Z. Cao, L. Zhang, Z. Chen, Y. Han, and Q. Pan, "A review on swarm intelligence and evolutionary algorithms for solving flexible job shop scheduling problems," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 4, pp. 904–916, 2019.

[4] K. Gao, F. Yang, M. Zhou, Q. Pan, and P. N. Suganthan, "Flexible job-shop rescheduling for new job insertion by using discrete jaya algorithm," *IEEE Transactions on Cybernetics*, vol. 49, no. 5, pp. 1944–1955, 2019.

[5] D. Gao, G. G. Wang, and W. Pedrycz, "Solving fuzzy job-shop scheduling problem using de algorithm improved by a selection mechanism," *IEEE Transactions on Fuzzy Systems*, vol. 28, no. 12, pp. 3265–3275, 2020.

[6] Z. Pan, D. Lei, and L. Wang, "A bi-population evolutionary algorithm with feedback for energy-efficient fuzzy flexible job shop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–13, 2021.

[7] S. Abdullah and M. Abdolrazzagh-Nezhad, "Fuzzy job-shop scheduling problems: A review," *Information Sciences*, vol. 278, pp. 380–407, 2014.

[8] D. Lei, "A genetic algorithm for flexible job shop scheduling with fuzzy processing time," *International Journal of Production Research*, vol. 48, no. 10, pp. 2995–3013, May 2010.

[9] J. Lin, "A hybrid biogeography-based optimization for the fuzzy flexible job-shop scheduling problem," *Knowledge-Based Systems*, vol. 78, pp. 59–74, 2015.

[10] D. Lei, "Co-evolutionary genetic algorithm for fuzzy flexible job shop scheduling," *Applied Soft Computing*, vol. 12, no. 8, pp. 2237–2245, 2012.

[11] J. Lin, "Backtracking search based hyper-heuristic for the flexible job-shop scheduling problem with fuzzy processing time," *Engineering Applications of Artificial Intelligence*, vol. 77, pp. 186–196, 2019.

[12] L. Sun, L. Lin, M. Gen, and H. Li, "A hybrid cooperative coevolution algorithm for fuzzy flexible job shop scheduling," *IEEE Transactions on Fuzzy Systems*, vol. 27, no. 5, pp. 1008–1022, 2019.

[13] 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," *Applied Soft Computing*, vol. 91, p. 106248, 2020.

[14] J. Jos Palacios, I. Gonzlez-Rodrguez, C. R. Vela, and J. Puente, "Robust multiobjective optimisation for fuzzy job shop problems," *Applied Soft Computing*, vol. 56, pp. 604–616, 2017.

[15] Z. Yuguang, Y. Fan, and L. Feng, "Solving multi-objective fuzzy flexible job shop scheduling problem using mabc algorithm," *Journal of Intelligent and Fuzzy Systems*, vol. 36, no. 2, pp. 1455–1473, 2019.

[16] B. Wang, H. Xie, X. Xia, and X. Zhang, "A nsga-ii algorithm hybridizing local simulated-annealing operators for a bi-criteria robust job-shop scheduling problem under scenarios," *IEEE Transactions on Fuzzy Systems*, vol. 27, no. 5, pp. 1075–1084, 2019.

[17] J. M. Mendel and R. I. B. John, "Type-2 fuzzy sets made simple," *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 2, pp. 117–127, 2002.

[18] J. Li, Z. Liu, C. Li, and Z. Zheng, "Improved artificial immune system algorithm for type-2 fuzzy flexible job shop scheduling problem," *IEEE Transactions on Fuzzy Systems*, pp. 1–1, 2020.

[19] J. Li, J. Li, L. Zhang, H. Sang, Y. Han, and Q. Chen, "Solving type-2 fuzzy distributed hybrid flowshop scheduling using an improved brain storm optimization algorithm," *International Journal of Fuzzy Systems*, vol. 23, no. 4, pp. 1194–1212, 2021.

[20] D. Yksel, M. F. Tagetiren, L. Kandiller, and Q.-K. Pan, "Metaheuristics for energy-efficient no-wait flowshops: A trade-off between makespan and total energy consumption," in *2020 IEEE Congress on Evolutionary Computation (CEC)*, 2020, pp. 1–8.

[21] C. Lu, B. Zhang, L. Gao, J. Yi, and J. Mou, "A knowledge-based multiobjective memetic algorithm for green job shop scheduling with variable machining speeds," *IEEE Systems Journal*, pp. 1–12, 2021.

[22] D. Deliktaş, O. Torkul, and O. Ustun, "A flexible job shop cell scheduling with sequence-dependent family setup times and intercellular transportation times using conic scalarization method," *Intl. Trans. in Op. Res.*, vol. 26, no. 6, pp. 2410–2431, Nov. 2019.

[23] D. Deliktaş, E. Özcan, O. Ustun, and O. Torkul, "Evolutionary algorithms for multi-objective flexible job shop cell scheduling," *Applied Soft Computing*, vol. 113, p. 107890, 2021.

[24] D. Deliktaş, "Self-adaptive memetic algorithms for multi-objective single machine learning-effect scheduling problems with release times," *Flexible Services and Manufacturing Journal*, 2021.

[25] J. Wang and L. Wang, "A cooperative memetic algorithm with learning-based agent for energy-aware distributed hybrid flow-shop scheduling," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2021.

[26] B. Zhang, Q.-K. Pan, L. Gao, L.-L. Meng, X.-Y. Li, and K.-K. Peng, "A three-stage multiobjective approach based on decomposition for an energy-efficient hybrid flow shop scheduling problem," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 12, pp. 4984–4999, 2020.

[27] C. Lu, Y. Huang, L. Meng, L. Gao, B. Zhang, and J. Zhou, "A pareto-based collaborative multi-objective optimization algorithm for energy-efficient scheduling of distributed permutation flow-shop with limited buffers," *Robotics and Computer-Integrated Manufacturing*, vol. 74, p. 102277, 2022.

[28] Y. Pan, K. Gao, Z. Li, and N. Wu, "Improved meta-heuristics for solving distributed lot-streaming permutation flow shop scheduling problems," *IEEE Transactions on Automation Science and Engineering*, pp. 1–11, 2022.

[29] C. Lu, L. Gao, J. Yi, and X. Li, "Energy-efficient scheduling of distributed flow shop with heterogeneous factories: A real-world case from automobile industry in china," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2020.

[30] L. Meng, Y. Ren, B. Zhang, J.-Q. Li, H. Sang, and C. Zhang, "Milp modeling and optimization of energy- efficient distributed flexible job shop scheduling problem," *IEEE Access*, vol. 8, pp. 191 191–191 203, 2020.

[31] G. Wang, X. Li, L. Gao, and P. Li, "Energy-efficient distributed heterogeneous welding flow shop scheduling problem using a modified

moea/d," *Swarm and Evolutionary Computation*, vol. 62, p. 100858, 2021.

[32] Q.-K. Pan, L. Gao, and L. Wang, "An effective cooperative co-evolutionary algorithm for distributed flowshop group scheduling problems," *IEEE Transactions on Cybernetics*, pp. 1–14, 2020.

[33] X. He, Q.-k. Pan, L. Gao, L. Wang, and P. N. Suganthan, "A greedy cooperative co-evolution ary algorithm with problem-specific knowledge for multi-objective flowshop group scheduling problems," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2021.

[34] Y.-S. Ong, M. H. Lim, and X. Chen, "Memetic computationpast, present amp; future [research frontier]," *IEEE Computational Intelligence Magazine*, vol. 5, no. 2, pp. 24–31, 2010.

[35] X. Chen, Y.-S. Ong, M.-H. Lim, and K. C. Tan, "A multi-facet survey on memetic computation," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 5, pp. 591–607, 2011.

[36] S.-M. Tse, Y. Liang, K.-S. Leung, K.-H. Lee, and T. S.-K. Mok, "A memetic algorithm for multiple-drug cancer chemotherapy schedule optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 1, pp. 84–91, 2007.

[37] B. Liu, L. Wang, and Y.-H. Jin, "An effective pso-based memetic algorithm for flow shop scheduling," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 1, pp. 18–27, 2007.

[38] Y.-R. Shiue, K.-C. Lee, and C.-T. Su, "Real-time scheduling for a smart factory using a reinforcement learning approach," *Computers and Industrial Engineering*, vol. 125, pp. 604–614, 2018.

[39] M. Zhao, X. Li, L. Gao, L. Wang, and M. Xiao, "An improved q-learning based rescheduling method for flexible job-shops with machine failures," in *2019 IEEE 15th International Conference on Automation Science and Engineering (CASE)*, 2019, pp. 331–337.

[40] Y.-F. Wang, "Adaptive job shop scheduling strategy based on weighted q-learning algorithm," *Journal of Intelligent Manufacturing*, vol. 31, no. 2, pp. 417–432, 2020.

[41] L. Wang, Z. Pan, and J. Wang, "A review of reinforcement learning based intelligent optimization for manufacturing scheduling," *Complex System Modeling and Simulation*, vol. 1, no. 4, pp. 257–270, 2021.

[42] W. Gong, Z. Liao, X. Mi, L. Wang, and Y. Guo, "Nonlinear equations solving with intelligent optimization algorithms: A survey," *Complex System Modeling and Simulation*, vol. 1, no. 1, pp. 15–32, 2021.

[43] J. Li, X. l. Chen, P. Duan, and J. h. Mou, "Kmoea: A knowledge-based multi-objective algorithm for distributed hybrid flow shop in a prefabricated system," *IEEE Transactions on Industrial Informatics*, pp. 1–1, 2021.

[44] 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 Transactions on Emerging Topics in Computational Intelligence*, pp. 1–15, 2022.

[45] A. K. Shukla, R. Nath, P. K. Muhuri, and Q. M. D. Lohani, "Energy efficient multi-objective scheduling of tasks with interval type-2 fuzzy timing constraints in an industry 4.0 ecosystem," *Engineering Applications of Artificial Intelligence*, vol. 87, p. 103257, 2020.

[46] J. M. Mendel and X. Liu, "Simplified interval type-2 fuzzy logic systems," *IEEE Transactions on Fuzzy Systems*, vol. 21, no. 6, pp. 1056–1069, 2013.

[47] K. Gao, P. N. Suganthan, M. F. Tasgetiren, Q. Pan, and Q. Sun, "Effective ensembles of heuristics for scheduling flexible job shop problem with new job insertion," *Computers & Industrial Engineering*, vol. 90, pp. 107–117, 2015.

[48] Q. Zhang and L. Hui, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[49] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[50] L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 29–38, 2006.

[51] R. C. Van Nostrand, "Design of experiments using the taguchi approach: 16 steps to product and process improvement," *Technometrics*, vol. 44, no. 3, pp. 289–289, Aug. 2002.

[52] Y. Tian, R. Cheng, X. Zhang, F. Cheng, and Y. Jin, "An indicator-based multiobjective evolutionary algorithm with reference point adaptation for better versatility," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 4, pp. 609–622, 2018.

[53] F. Ming, W. Gong, H. Zhen, S. Li, L. Wang, and Z. Liao, "A simple two-stage evolutionary algorithm for constrained multi-objective optimization," *Knowledge-Based Systems*, vol. 228, p. 107263, 2021.

Rui Li received the B.Sc degree in 2016 from China University of Geosciences, Wuhan, China.

He is currently working toward the M.Sc degree. His main research interests include the green shop scheduling problems, evolutionary algorithms and reinforcement learning.

Wenyin Gong (Member, IEEE) received the B.Eng., M.Eng., and Ph.D. degrees in computer science from China University of Geosciences, Wuhan, China, in 2004, 2007, and 2010, respectively.

He is currently a Professor with School of Computer Science, China University of Geosciences, Wuhan, China. His research interests include evolutionary algorithms, evolutionary optimization, and their applications. He has published over 80 research papers in journals and international conferences.

He served as a referee for over 30 international journals, such as IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE TRANSACTIONS ON CYBERNETICS, IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, *IEEE Computational Intelligence Magazine*, *ACM Transactions on Intelligent Systems and Technology*, *Information Sciences*, *European Journal of Operational Research*, *Applied Soft Computing*, *Journal of Power Sources*, and so on.

Chao Lu received the Ph.D. degree in industrial engineering from the Huazhong University of Science and Technology, Wuhan, China, 2017.

He is an associate professor with the School of Computer Science, China University of Geosciences, Wuhan. His research interests include multi-objective evolutionary algorithms and shop scheduling.

Ling Wang (Member, IEEE) received the B.Sc. in automation and Ph.D. 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. His current research interests include intelligent optimization and production scheduling. He has authored five academic books and more than 300 refereed papers.

Professor Wang is 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, the Natural Science Award (First Place in 2003, and Second Place in 2007) nominated by the Ministry of Education of China. Professor Wang now is the Editor-in-Chief of *International Journal of Automation and Control*, and the Associate Editor of IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, *Swarm and Evolutionary Computation*, *Expert Systems with Applications*, ETC.