Improved artificial immune system algorithm for type-2 fuzzy flexible job shop scheduling problem

Junqing Li, *Member, IEEE*, Zheng-min Liu, Chengdong Li, *Member, IEEE*, and Zhixin Zheng

***Abstract*—In practical applications, particularly in flexible manufacturing systems, there is a high level of uncertainty. A type-2 fuzzy logic system (T2FS) has several parameters and an enhanced ability to handle high levels of uncertainty. This study proposes an improved artificial immune system (IAIS) algorithm to solve a special case of the flexible job shop scheduling problem (FJSP), where the processing time of each job is a nonsymmetric triangular interval T2FS (IT2FS) value. First, a novel affinity calculation method considering the IT2FS values is developed. Then, four problem-specific initialization heuristics are designed to enhance both quality and diversity. To enhance the exploitation abilities, six local search approaches are conducted for the routing and scheduling vectors respectively. Next, a simulated annealing method is embedded to accept antibodies with low affinity, which can enhance the exploration abilities of the algorithm. Moreover, a novel population diversity heuristic is presented to eliminate antibodies with high crowding values. Five efficient algorithms are selected for a detailed comparison, and the simulation results demonstrate that the proposed IAIS algorithm is effective for IT2FS FJSPs.**

***Index Terms*—flexible job shop;type-2 fuzzy processing time; artificial immune system; energy consumption**

I. Introduction

THE scheduling problem is an important research topic, as it is often a key challenge in many applications [1]-[9] The typical scheduling problems include the flow shop problem (FSP) [1], [2], hybrid flow shop (HFS) problem [3], [4], job shop problem (JSP) [5], flexible job shop problem (FJSP) [6], and distributed flow shop problems (DFSP) [7], [8]. FJSPs are more practical than the other types of scheduling problems because they consider more realistic constraints. In an FJSP, n jobs are scheduled on m machines, where each job has its own route at each stage, and each stage has a set of candidate machines. The machine flexibility enhances the system performance; however, it increases the complexity of the problem. Due to the problem complexity, meta-heuristics have been investigated for FJSPs, such as the evolutionary optimization method [6], genetic algorithm (GA) [10], [11], artificial bee colony (ABC) algorithm [12], and particle swarm optimization (PSO) [13]. Recently, an accelerated simulated annealing (ASA) algorithm [14], a discrete Jaya algorithm [15], a multiagent and bargaining-game-based heuristic [16], two-phase meta-heuristic (TPM) [17], and a modified iterated greedy (MIG) algorithm [18] have been proposed to solve FJSPs.

There are various realistic constraints for applying the FJSP in an industrial production system, such as sequence-dependent setup times [19], and fuzzy processing time constraints. In particular, the fuzzy processing time in uncertain environments is a commonly used realistic constraint. Kacem et al. developed an approach hybridization of evolutionary algorithms and fuzzy logic for the problem [20]. Subsequently, various meta-heuristics have been studied, such as the co-evolutionary GA by Lei [21], chemical-reaction optimization (CRO) by Li and Pan [22], the ABC algorithm [23], [24], [25], hybrid biogeography-based optimization by Lin [26], estimation of distribution algorithm (EDA) by Liu et al. [27], the tabu search (TS) algorithm by Palacios et al. [28], and the memetic algorithm (MA) [29].The hybridization of various heuristics can generally improve the algorithm performance, and examples include the backtracking search based hyper-heuristic [30] and the hybrid cooperative coevolution algorithm (hCEA) [31]. In addition, several studies have applied fuzzy FJSPs in realistic applications, including a semiconductor manufacturing system [32], a typical industrial production system with crane transportation [33], and a green manufacturing system [34].

Many types of realistic applications should consider the fuzzy features due to the uncertainty of the system, such as traffic lights systems[35], and robotic path planning systems[36]. However, in most realistic applications, especially the industrial production systems, the level of uncertainty is high. Compared to T1FSs, a type-2 fuzzy logic system (T2FS) has more parameters and enhanced abilities to handle a high level of uncertainty[37], [38], [39]. Naturally, T2FLSs have exhibited better performance than T1FLSs in many research fields, especially in modeling, prediction, and control applications [40], [41], [42], [43], [44]. For example, the fuzzy logic approach for dynamic parameter adaptation was investigated by Melin et al. [45], Castillo et al. [46], and Olivas et al. [47]. The image processing based on generalized type-2 fuzzy logic was conducted by Melin et al. [48]. The multi-objective scheduling of tasks in an Industry 4.0 ecosystem with interval type-2 fuzzy timing constraints was studied by Shukla et al. [49]. The other typical applications include noise robustness of type-2 fuzzy logic controllers [50], linguistic

This research is partially supported by National Science Foundation of China under Grant 61773192, 61803192, Shandong Province Higher Educational Science and Technology Program (J17KZ005), and major basic research projects in Shandong (ZR2018ZB0419).

Jun-qing Li, is with the School of Information and Engineering, Shandong Normal University, Jinan 250014, China, and also with the School of Computer Science, Liaocheng University, Liaocheng 252059, China. (e-mail: lijunqing@lcu-cs.com).

Zheng-min Liu, with the School of Management Science and Engineering, Shandong University of Finance and Economics, Shandong, China.

Cheng-dong Li is with the School of Information and Electrical Engineering, Shandong Jianzhu University, Jinan, 252101, China.

Zhixin Zheng is with the School of Computer Science, Liaocheng University, Liaocheng 252059, China.

word modelling[51], and time series prediction [52].

Nevertheless, existing studies mainly consider a Type-1 fuzzy logic system (T1FLS) in the FJSP, where each value can be represented by a triangular fuzzy number (TFN) and other types of numbers. The uncertainty levels in realistic applications are generally too high to be represented by a deterministic value or a triangular fuzzy number (TFN) value. For example, the processing time for any operation in the steelmaking system is generally determined by many realistic factors. Considering all of the factors, the processing times should be set as type-2 fuzzy values to handle high levels of uncertainty of the realistic system. Therefore, to model and simulate realistic production processes, T2FSs should be considered in the FJSP, which is a challenging task. During recent years, we have considered developing many types of efficient optimization algorithms, such as the fruit fly optimization algorithm (FOA) for the realistic hybrid flowshop rescheduling problem in steelmaking systems [4], hybrid artificial bee colony algorithms for a parallel batching distributed flowshop problem [8], Pareto-based discrete artificial bee colony algorithm for multi-objective FJSPs [12], and chemical-reaction optimization for fuzzy job shop scheduling problems [22]. In addition, we have also investigated the theory and applications of the type-2 fuzzy system [43], such as the monotonicity of interval type-2 fuzzy logic systems, a fast learning method for data-driven design of interval T2FSs [44], and the applications to linguistic word modelling [51]. However, based on the literature review presented above, there is no existing literature that consider the interval type-2 fuzzy system in the FJSPs.

Recently, a meta-heuristic called artificial immune system (AIS) has been applied for realistic optimization problems[53]. Jiao and Wang developed an immune genetic algorithm (IGA) and applied it to solve the traveling salesman problem (TSP) [54]. Several studies have used the AIS algorithm to solve the flow shop scheduling problems, such as the blocking flow shop problems [55], and the distributed flow shop [56]. To consider more flexible features of realistic applications, Engin and Döyen constructed a method based on AIS to solve HFSs [57]. The assembly HFS problem was solved in [58], [59], and the AIS algorithm has been used to solve FJSPs [60], [61].

Keeping the above consideration in mind, this study proposes an improved version of the AIS, named IAIS, to solve the interval T2FS (IT2FS) FJSP problems. Our main contributions are as follows: (1) The IT2FS FJSP is firstly investigated by an improved AIS algorithm, where the processing time, the fuzzy starting time, and the fuzzy completion time of each operation are set as interval type-2 fuzzy values. (2) considering the IT2FS features, a novel affinity calculation heuristic is developed to set the IT2FS starting time for each operation; (3) two objectives are considered simultaneously, including minimization of the maximum IT2FS completion time and the energy consumption; (4) four problem-specific initialization heuristics are designed to enhance both quality and diversity; (5) six local search approaches are conducted for the routing and scheduling vectors respectively, to enhance the exploitation abilities; (6) a simulated annealing method is embedded to accept antibodies with low affinity, which can enhance the exploration abilities of the algorithm; and (7) a novel population diversity heuristic is presented to eliminate antibodies with high crowding values.

The remainder of this paper is organized as follows: In Section II, we introduce the related methods, including IT2FS concepts and the canonical AIS algorithm. In Section III, we briefly introduce the considered problem, and in Section IV we describe the proposed algorithm. In Section V, we present experimental comparisons and analysis are presented in Section V. Finally, in Section VI, we give the conclusions and ideas for future work.

II. Related methods

A. IT2FS basic concepts

A type-2 fuzzy set, say $\tilde{A}$, is generally characterized by a membership function $\mu_{\tilde{A}}(x,u)$, where $x \in X$, and $u \in [0,1]$. Based on the membership function, the definition of the T2FS is given as follows:

$$\tilde{A} = \{((x,u), \mu_{\tilde{A}}(x,u)) | \forall x \in X, \forall u \in [0,1]\}, 0 \le \mu_{\tilde{A}}(x,u) \le 1 \quad (1)$$

An interval T2FS (IT2FS), denoted as $\tilde{A}_I$, is a special case of T2FS, which is generally characterized as [39], [49]

$$\tilde{A}_I = \{((x,u),1) | \forall x \in X, \forall u \in [0,1]\} \quad (2)$$

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

$$\text{FOU}(\tilde{A}) = \cup_{x \in X} J_x \quad (3)$$

The upper membership function (UMF) and lower membership function (LMF) of $\tilde{A}$ are denoted $\bar{\mu}_{\tilde{A}}(x)$ and $\underline{\mu}_{\tilde{A}}(x)$, $\forall x \in X$, respectively.

$$\bar{\mu}_{\tilde{A}}(x) \equiv \overline{\text{FOU}(\tilde{A})} \ \ \forall x \in X \quad (4)$$

$$\underline{\mu}_{\tilde{A}}(x) \equiv \underline{\text{FOU}(\tilde{A})} \ \ \forall x \in X \quad (5)$$

The following formulation gives a simple way to calculate the centroid of a nonsymmetric triangular interval T2FS, where the lower membership function is not symmetric. Fig. 1 presents a chart for this type of system.

$$C_{\tilde{A}} \approx [m - \frac{(b-a_r)(a_l+2a_r+b)}{6(a_l+a_r)}, m + \frac{(b-a_l)(2a_l+a_r+b)}{6(a_l+a_r)}] \quad (6)$$

B. IT2FS operators

(1) Addition operator

Given two interval type-2 fuzzy variables $\tilde{A} = (a_1, a_2, a_3, a_4, a_5)$ and $\tilde{B} = (b_1, b_2, b_3, b_4, b_5)$, then the addition operator is expressed as follows:

$$\tilde{A} + \tilde{B} = (a_1+b_1, a_2+b_2, a_3+b_3, a_4+b_4, a_5+b_5) \quad (7)$$

(2) Ranking operator

Given two interval type-2 fuzzy variables $\tilde{A} = (a_1, a_2, a_3, a_4, a_5)$ and $\tilde{B} = (b_1, b_2, b_3, b_4, b_5)$, calculate the centroid values $C_{\tilde{A}}$ and $C_{\tilde{B}}$ for the two variables $\tilde{A}$ and $\tilde{B}$, respectively.

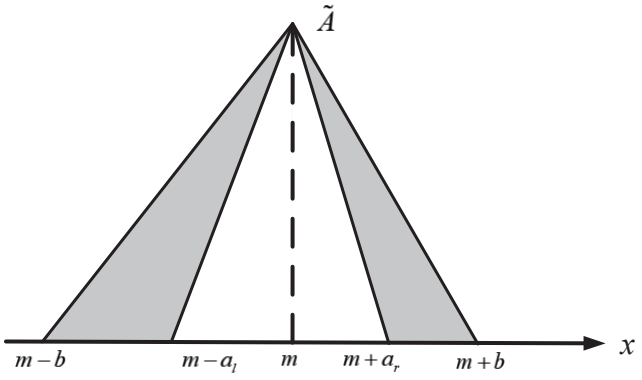


Fig. 1. Nonsymmetrical triangular interval T2FS (Type I).

$$C_{\tilde{A}}=[C^U_{\tilde{A}}, C^L_{\tilde{A}}] \approx \left[\begin{array}{l} a_3 - \frac{(a_5-a_4)(a_5+2a_4-a_2-2a_3)}{6(a_4-a_2)}, \\ a_3 + \frac{(a_5+a_2-2a_3)(a_5+a_4-2a_2)}{6(a_4-a_2)} \end{array} \right. \quad (8)$$

$$C_{\tilde{B}}=[C^U_{\tilde{B}}, C^L_{\tilde{B}}] \approx \left[\begin{array}{l} b_3 - \frac{(b_5-b_4)(b_5+2b_4-b_2-2b_3)}{6(b_4-b_2)}, \\ b_3 + \frac{(b_5+b_2-2b_3)(b_5+b_4-2b_2)}{6(b_4-b_2)} \end{array} \right. \quad (9)$$

Condition 1. If $C_{\tilde{A}}$ >(<) $C_{\tilde{B}}$, then $\tilde{A}$ >(<) $\tilde{B}$; otherwise, check condition 2.

Condition 2. If a_3 >(<) b_3 , then $\tilde{A}$ >(<) $\tilde{B}$; otherwise, check condition 3.

Condition 2. If $(a_5 - a_1)$ >(<) $(b_5 - b_1)$, then $\tilde{A}$ >(<) $\tilde{B}$.

For example, consider two type-2 fuzzy numbers, $\widetilde{T}_1$ =(2, 4, 6, 8, 12), and $\widetilde{T}_2$=(1, 5, 8, 12, 15). According to the above ranking method, $C_{\widetilde{T}_1}$ = [4, 8], $(C^U_{\widetilde{T}_1} + C^L_{\widetilde{T}_1})/2$ = (4+8)/2 = 6; $C_{\widetilde{T}_2}$ = [6.71, 9.62], $(C^U_{\widetilde{T}_2} + C^L_{\widetilde{T}_2})/2$= (6.71+9.62)/2 = 8.17. Therefore, $\widetilde{T}_1 < \widetilde{T}_2$.

C. The canonical artificial immune algorithm

In an AIS, there is a population of antibodies, each of which represents a solution. The antibody with high solution quality is assigned a higher affinity.

After generating the initial population, a certain number of antibodies are initially selected and used to generate clone antibodies, which is similar to the mutation procedure in a GA. Unlike the mutation operator in a GA, antibodies with higher affinity have a larger number of clones. Therefore, a larger number of searching tasks are performed around the promising space, and the exploitation abilities of the algorithm are enhanced. Next, the newly generated clones are evaluated and used to update the parent clones. This process is repeated until one of the termination conditions is met.

Algorithm 1: The canonical AIS

Input: input parameters
Output: the best solution found so far.
1 Initialize the antibodies;
2 Calculate the objective value of each antibodies;
3 Choose some antibodies and clone them according to their affinities;
4 Perform the mutation operator on the cloned antibodies to obtain new antibodies;
5 Update the antibodies according to the greedy rule;
6 Record the best antibody;
7 **if** *the stooping criterion is satisfied* **then**
8 | end the algorithm;
9 **else**
10 | go to step 2.
11 **end**

III. Problem description

Considering a realistic production procedure in a typical steelmaking system, as shown in Fig. 2, the molten iron in a device, called a torpedo car (TPC), is generally considered as scheduling unit and is called a charge or a job in the system. The charge is processed in five stages. Each charge or job has its own route dynamically assigned according to the current device state; that is, the route differs for different jobs. In each stage, each charge selects one machine from the candidate machine set; machine selection is thus flexible. Based on the above analysis of scheduling in a steelmaking system, we can model this scheduling as an FJSP problem.

Most of the current studies consider a deterministic processing environment [6], [13], [14], [16] or a type-1 fuzzy scenario [20], [21], [22]. In the deterministic environment, the common assumption is to set the processing time for each operation as a deterministic value. In a type-1 fuzzy scenario, the processing time is generally set as a TFN value. Due to the uncertainty in realistic applications, the processing time of a deterministic value or a TFN value cannot adapt to the realistic production systems. For example, the processing time for any operation in the steelmaking system are determined by many realistic factors, such as the temperature of the current processing machine, the fatigue degree of the device, and the proficient degree of the manpower. Considering all of the factors, the processing time of any operation should be set as a type-2 fuzzy value to handle high levels of uncertainty of the realistic system.

In addition, a crane is considered to transport each operation after its completion in a previous stage. This study aims to minimize the weighted value of the makespan and the energy consumption during machine processing and crane transportation. Based on the features of the problem, there are four main types of tasks: (1) how to assign a suitable machine for each operation; (2) how to schedule all assigned jobs for each machine; (3) how to determine the fuzzy starting and completion times of each operation; and (4) how to determine the crane transportation routes to minimize the objective considered in this study.

A. Assumptions

- Each job has a given number of operations and should be processed on one machine selected from a set of suitable machines.
- Each machine is available and should be powered on continuously. Preventive maintenance and machine breakdown activities are not considered.

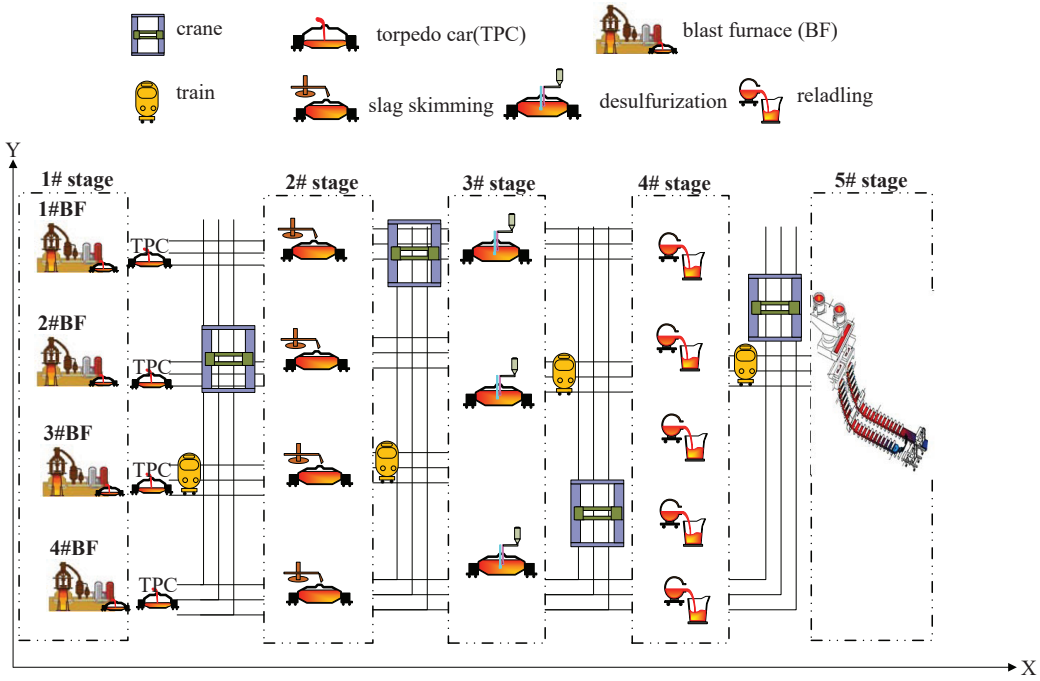


Fig. 2. Typical FJSP in a steelmaking system.

- Pre-emption is not available.
- Overlapping is not permitted during crane transportation.
- The crane device works continuously without any malfunction.
- Sufficient buffers between any two machines can store the completed operations while waiting for the crane transportation or successor machine.
- The arrival time of the operations by transportation using the crane device must be equal to or greater than the available time of the successor machine.
- No crane is required for different operations belonging to the same job on the same machine.
- No crane is required for the first operation of each job.
- The processing time of each charge or job is a type-2 fuzzy value rather than a given value.

B. Notations

Indices:

. m: The number of machines.
. n: The number of jobs.
. j: The index of jobs.
. i: The index of operation.
. k: The index of machines.

Variables:

. $O_{i,j}$: The i^{th} operation of job j, i=1,2,,θ_j.
. $\widetilde{T}s_{i,j}$: The fuzzy starting time of $O_{i,j}$, represented by a type-2 fuzzy value: $\widetilde{T}s_{i,j}$ =(ts_1, ts_2, ts_3, ts_4, ts_5).
. $\widetilde{T}c_{i,j}$: The fuzzy completion time of $O_{i,j}$, represented by a type-2 fuzzy value: $\widetilde{T}c_{i,j}$ =(tc_1, tc_2, tc_3, tc_4, tc_5).
. E_{mp}: Total energy consumption of the machining process.
. E_{ct}: Total energy consumption of the crane transportation.
. E_{no}: Total energy consumption of the no-load operation of the crane.
. E_{ns}: Total energy consumption of the no-load standby of the crane.
. E_{ls}: Total energy consumption of the load standby of the crane.
. E_{lo}: Total energy consumption of the load operation of the crane.
. $E_{no}(O_{i,j})$: No-load operation energy consumption of $O_{i,j}$.
. $E_{ns}(O_{i,j})$: No-load standby energy consumption of $O_{i,j}$.
. $E_{ls}(O_{i,j})$: Load standby energy consumption of $O_{i,j}$.
. $E_{lo}(O_{i,j})$: Load operation energy consumption of $O_{i,j}$.
. $\widetilde{T}_{ls}(O_{i,j})$: Load standby time of $O_{i,j}$, represented by a type-2 fuzzy value: $\widetilde{T}_{ls}(O_{i,j})$=($tls_1$, tls_2, tls_3, tls_4, tls_5).
. $\widetilde{T}_{lo}(O_{i,j})$: Load operation time of $O_{i,j}$, represented by a type-2 fuzzy value: $\widetilde{T}_{lo}(O_{i,j})$=($tlo_1$, tlo_2, tlo_3, tlo_4, tlo_5).
. $\widetilde{T}_{lf}(O_{i,j})$: Load lifting time of $O_{i,j}$, represented by a type-2 fuzzy value: $\widetilde{T}_{lf}(O_{i,j})$=($tlf_1$, tlf_2, tlf_3, tlf_4, tlf_5).
. $\Psi < O_{i1,j1}, O_{i,j} >$: The relationship set when the crane is to serve $O_{i,j}$ after completing the service for $O_{i1,j1}$.
. $\Phi < O_{i-1,j}, O_{i,j}, k, k2 >$: The relationship set when the crane is to serve $O_{i,j}$ on machine k_2 after completing the predecessor operation $O_{i-1,j}$ on machine k.
. $\Theta < O_{i2,j2}, O_{i,j} >$: if $O_{i,j}$ is the immediate successor operation of $O_{i2,j2}$ being processed on the same machine.
. L: A very large number.

Parameters

. θ_j: The operation number of job j, j=1,2,,n.
. $\widetilde{T}_{i,j,k}$: The fuzzy processing time of $O_{i,j}$ on machine k, represented by a type-2 fuzzy value: $\widetilde{T}_{i,j,k}$ =(p_1, p_2, p_3, p_4, p_5).
. Ep_k: Processing power of machine k.

Decision variables:

- $y_{i1,j1,i2,j2}$: A binary value set to 1 if $< O_{i2,j2}, O_{i,j} > \in \Psi$; otherwise is set to 0.
- $x_{i,j,k}$: A binary value set to 1 when $O_{i,j}$ is assigned to machine k; otherwise $x_{i,j,k}$ is set to 0..
- $z_{i-1,j,k,i,j,k2}$: A binary value set to 1 if $< O_{i-1,j}, O_{i,j}, k, k2 > \in \Phi$; otherwise $z_{i-1,j,k,i,j,k2}$ is set to 0.
- $u_{i2,j2,i,j}$: A binary value set to 1 if $< O_{i2,j2}, O_{i,j} > \in \Theta$; otherwise $u_{i2,j2,i,j}$ is set to 0.

C. Energy consumption of machine processing

Let $E_{mp}(O_{i,j})$ represent the machine processing energy consumption for operation $O_{i,j}$ and let E_{mp} represent the total machine processing energy consumption. The two variables can be calculated as follows:

$$E_{mp}(O_{i,j}) = \sum_{k=1}^{m} Ep_k \cdot \widetilde{T}_{i,j,k} \cdot x_{i,j,k} \tag{10}$$

$$E_{mp} = \sum_{j=1}^{n} \sum_{i=1}^{\theta_j} E_{mp}(O_{i,j}) \tag{11}$$

D. Energy consumption during crane transportation

Similar to [21], the following energy consumptions are considered:

$$E_{no} = \sum_{j=1}^{J} \sum_{j1=1}^{J} \sum_{i=2}^{\theta_j} \sum_{i1=1}^{\theta_j} E_{no}\left(O_{i,j}\right) \cdot y_{i1,j1,i,j} \tag{12}$$

$$E_{ns} = \sum_{j=1}^{J} \sum_{j1=1}^{J} \sum_{i=2}^{\theta_j} \sum_{i1=1}^{\theta_j} E_{ns}\left(O_{i,j}\right) \cdot y_{i1,j1,i,j} \tag{13}$$

$$E_{ls} = \sum_{j=1}^{J} \sum_{i=1}^{\theta_j} \sum_{k=1}^{K} E_{ls}(O_{i,j}) \cdot x_{i,j,k} \tag{14}$$

$$E_{lo} = \sum_{j=1}^{J} \sum_{i=1}^{\theta_j} \sum_{k=1}^{K} E_{lo}(O_{i,j}) \cdot x_{i,j,k} \tag{15}$$

Considering the four types of transportation energy consumption, the total energy consumption during the entire transportation process can be calculated as follows:

$$E_{ct} = E_{no} + E_{ns} + E_{ls} + E_{lo} \tag{16}$$

E. The formulation of the fCFJSP

$$\min f = \omega_1 \cdot f_1 + \omega_2 \cdot f_2 \tag{17}$$

$$\min f_1 = \max(\widetilde{T}c_{\theta j,j}), j = 1, 2, ..., n \tag{18}$$

$$\min f_2 = E_{mp} + E_{ct} \tag{19}$$

$$\widetilde{T}s_{i,j} \geq \widetilde{T}c_{i-1,j} \tag{20}$$

$$(\widetilde{T}s_{i,j} - \widetilde{T}c_{i2,j2}) \cdot u_{i2,j2,i,j} \geq 0 \tag{21}$$

$$\sum_{k=1}^{K} x_{i,j,k} = 1 \tag{22}$$

$$P_i = K^1 \tag{23}$$

$$\widetilde{T}c_{i-1,j} + \widetilde{T}_{ls}(O_{i,j}) - \widetilde{T}s_{i2,j2} \geq 0 \tag{24}$$

$$P_{nl}(O_{i,j}) = P_l(O_{i,j}) = P_l(O_{i1,j1}),$$
$$\forall < O_{i1,j1}, O_{i,j} > \in \Psi,$$
$$\wedge x_{i,j,k} = x_{i-1,j,k} = 1 \tag{25}$$

$$x_{i,j,k} = \begin{cases} 1, & if\ O_{i,j}\ is\ processed\ on\ machine\ k \\ 0, & otherwise \end{cases} \tag{26}$$

$$y_{i1,j1,i2,j2} = \begin{cases} 1, & if\ < O_{i1,j1}, O_{i2,j2} > \in \Psi \\ 0 \end{cases} \tag{27}$$

$$z_{i-1,j,k,i,j,k2} = \begin{cases} 1, & if < O_{i-1,j,k}, O_{i,j,k2} > \in \Phi \\ 0, & otherwise \end{cases} \tag{28}$$

$$u_{i2,j2,i,j} = \begin{cases} 1, & if\ < O_{i2,j2}, O_{i,j} > \in \Theta \\ 0, & otherwise \end{cases} \tag{29}$$

Constraints (16)-(18) describe the total objective of this problem. Constraint (19) ensures the precedence relation of consecutive operations of the same job. The processed relation of two immediate jobs on the same machine is ensured by constraint (20), while constraint (21) ensures that each operation can select only one machine. Constraint (22) sets the initial position for the crane, and constraint (23) specifies that the transportation of the successor operation must wait for the following machine to become idle. Constraint (24) guarantees that no crane transportation is required if no machine has changed between consecutive operations of the same job. The range of variables is described in constraints (25)-(28).

F. Illustrative example

To illustrate the problem, Fig. 3 presents a Gantt chart of an example of the considered problem. There are six jobs to be processed on five machines, and one crane device to complete the transportation tasks between any two machines. The IT2FS starting and completion times for each job are represented by two nonsymmetrical triangular interval IT2FS values, and a pair of values representing the job and operation numbers. For example, the last operation being processed on M_5 is $O_{18,6}$, which is represented by two nonsymmetrical triangular interval IT2FS values. The IT2FS value under the line denotes the fuzzy starting time for $O_{18,6}$, and the IT2FS value above the line tells the fuzzy completion time for $O_{18,6}$. Each IT2FS value is represented by two overlapped triangles. The interval spaces between the two triangles filled with purple colors represent the area between the UMF and LMF for the operation $O_{18,6}$. Moreover, the two pairs of five values represented the interval type-2 fuzzy values for the starting

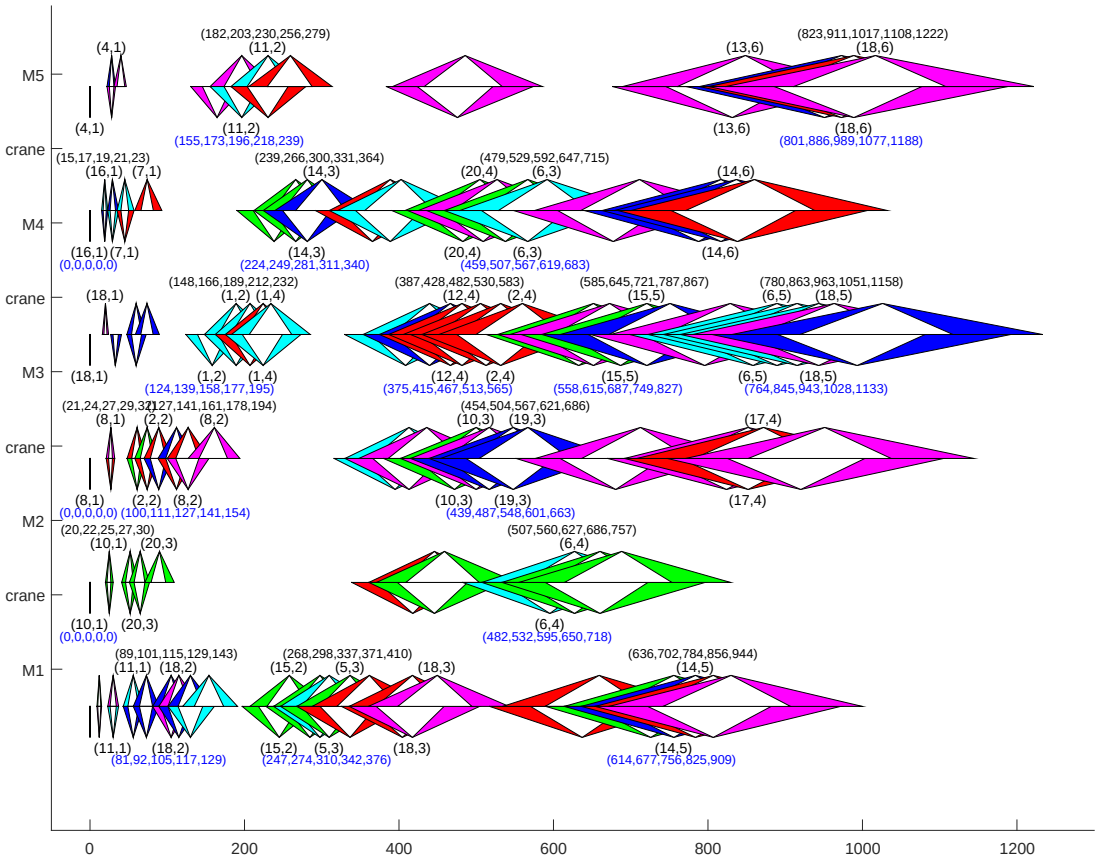


Fig. 3. A Gantt chart for an example problem.

and completion times. For example, the value (801, 886, 989, 1077, 1188) means the IT2FS value for the fuzzy starting time for $O_{18,6}$, and the value (823, 911, 1017, 1108, 1222) is for the fuzzy completion time for $O_{18,6}$. After computing the IT2FS values of the completion time for each operation, the maximum completion time for the system will be obtained, and the objective function can be collected.

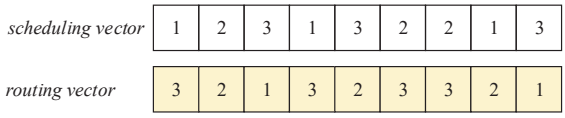


Fig. 4. Example of solution representation.

IV. Proposed Algorithm

A. Framework of IAIS

In this section, we describe the detailed components of the proposed IAIS algorithm. We present step-by-step descriptions of the solution representation and decoding mechanism, the affinity calculation, initialization strategy, selection and clone method, mutation operators, local search heuristics and the population diversity steps. The framework of the IAIS is described in Algorithm 2.

B. Solution representation

Similar to other studies, in this study, we also use a two-dimensional vector to represent each antibody or solution. The routing vector is used to indicate the assigned machine for each operation, and the scheduling vector is used to report the processing sequence for all operations. The two vectors are set with the same length, which equals the total number of operations.

For example, Fig. 4 provides a solution representation, in which the operations given in the scheduling vector are O_{11},O_{12},O_{13},O_{21},O_{23},O_{22},O_{32},O_{31},O_{33}.

The routing vector indicates that the first position is for operation O_{11}, where machine M_3 is assigned to process it. Then, machine M_2 is selected from processing O_{12}, and so on. The last position in the routing vector is for O_{33}, which is processed on machine M_1.

Algorithm 2: The Framework of IAIS

Input: input parameters
Output: the best solution found so far.
Let P_{size} be the population size, and P_{size}=$n_c(n_c+1)/2$ where n_c is the clone number;
Initialize P_{size} antibodies (c.f. subsection IV-D.);
Calculate the objective value of each antibody in the initial population (c.f. subsection IV-C.);
Choose n_c antibodies and clone them according to their affinities (c.f. subsection IV-E.);
Perform the mutation operator on the cloned antibodies to obtain new antibodies (c.f. subsection IV-F.);
Perform the population diversity to suppress the population by eliminating the antibodies with high crowding values (c.f. subsection IV-G.);
Perform the SA-based exploration search (c.f. subsection IV-H.);
Record the best antibody;
if *the stooping criterion is satisfied* **then**
  end the algorithm;
else
  go to step 2.
end

C. IT2FS-based affinity calculation heuristic

Given a scheduling vector $\prod := (\pi_1, \pi_2, ..., \pi_n)$ consisting of n operations and a routing vector $\Re := (\gamma_1, \gamma_2, ..., \gamma_n)$ consisting of n operations, the decoding method involves first determining the operation number for each position π_i, $i = 1, 2, ..., n$ in the routing vector. Then, for the operation number, the machine number should be selected at position γ_i, $i = 1, 2, ..., n$ in $\Re$. Considering the IT2FS processing time, the IT2FS starting time for each assigned operation should be set with respect to the scheduling vector.

The decoding steps are as follows: (1) determine the IT2FS starting time for each operation considering both the completion time of the previous operation and the machine idle time, and consider the crane transportation time; and (2) calculate the objective values, including the IT2FS makespan, and the energy consumptions.

Algorithm 3 presents the detailed steps of the affinity calculation heuristic.

Algorithm 3: IT2FS-based affinity calculation heuristic

Input: two dimensional vectors for a solution
Output: the IT2FS starting time for each operation.
Initialize a machine idle time vector, named M_{idle}, with the initial values 0 for each machine;
for *each operation* $O_{i,j}$ *in the scheduling vector* **do**
  Find the assigned machine k for processing $O_{i,j}$ in the routing vector;
  Let $\widetilde{T}c_{i-1,j}$ be the IT2FS completion time of $O_{i-1,j}$, if i=1, then set $\widetilde{T}c_{i-1,j}$ =0;
  Let $\widetilde{T}_k$ be the idle time for the machine k ;
  Let $\widetilde{T}s_{i,j} = \widetilde{\max}\{\widetilde{T}c_{i-1,j}, \widetilde{T}_k\}$, where $\widetilde{\max}\{\}$ means the maximum IT2FS values by using the IT2FS ranking operator discussed in Section II-B;
  Let $\widetilde{T}c_{i,j}$= $\widetilde{T}s_{i,j}$+$\widetilde{T}_{i,j,k}$, where the addition operator can be found in Section II-B;
  Set the machine idle time $\widetilde{T}_k$=$\widetilde{T}c_{i,j}$.
end

D. Initialization strategy

To obtain an initial population with high quality and diversity, four types of initialization strategies are embedded in the proposed IAIS algorithm as follows:

IS1: Random initialization rule. This rule is simple and aims to generate a solution with high diversity. For the scheduling vector: (1) initialize an empty scheduling vector, and add the number for each job j n_j times; and (2) randomly rearrange the sequence for all numbers in the scheduling vector. For the routing vector: (1) initialize an empty routing vector; and (2) select each operation $O_{i,j}$ in the scheduling vector, randomly select one available machine $M_{i,j}$ for $O_{i,j}$, and store $M_{i,j}$ in the routing vector.

IS2: Global minimum workload rule. For each operation, select an available machine with the minimum workload. If more than one machine has the same workload, then select the machine with the minimum processing time for the operation.

IS3: Global minimum energy consumption rule. For each operation, select an available machine with the minimum energy consumptions. If more than one machine has the same energy consumptions, then select the machine with the minimum processing time for the operation.

IS4:Local minimum processing time rule. For each operation, select an available machine with the minimum processing time.

It is obvious that the computational complexity of the four initialization strategy is $O(nm)$.

Based on these four initialization strategies, a simple initialization procedure is as follows:

Step 1. Apply the four problem-specific initialization rules (i.e., IS2, IS3, and IS4) to generate four solutions.

Step 2. While the initial population size is less than P_{size}, generate an initial solution by performing the IS1 strategy.

E. Selection and cloning

In the proposed algorithm, similar to [55], rather than select the entire population to perform mutation, n_c antibodies are selected to generate cloned antibodies. To maintain meaningful information in the current population, a native way is to increase the number of clone antibodies for a solution with higher affinity. Detailed steps to select and clone are as follows:

Step 1. Sequence the entire population according to affinity, that is, an antibody with higher affinity is selected for cloning.

Step 2. The clone number of each n_c selected antibody is calculated as $(n_c - k + 1)$, where k denotes the antibody with the k^{th} highest affinity function.

Step 3. All $n_c(n_c + 1)/2$ antibodies construct a set named CP.

F. Mutation

For each cloning antibody, the exploitation process should be applied to explore its neighboring search space. A common way is to use either a swap mutation or insertion mutation. In this subsection, by considering the problem features and objective information, several local search operators are developed.

1) Mutation for routing vector:

LS1: Local search for the busiest machine: (1) For a solution X, compute the workload for all machines. To compute the workload for each operation $O_{i,j}$ on machine k, first, let $\tilde{S}_i$ be the IT2FS starting time and $\tilde{E}_i$ be the completion time of $O_{i,j}$. Let $P_i = \tilde{E}_i$ - $\tilde{S}_i$ and $\tilde{C}_i$ be the centroid of P_i. Then, the workload of $O_{i,j}$ on machine k is set to $\tilde{C}_i$. (2) Select machine M with the maximum workload; (3) Randomly select an operation $O_{i,j}$ on machine M. If $O_{i,j}$ has more than one candidate machine, then randomly replace it with another machine; and (4) Evaluate the newly-generated neighboring solution and replace the current solution.

LS2: Local search for a machine with the highest energy consumption: (1) For a solution X, compute the energy consumption for all machines; (2) Select machine M with the maximum energy consumption; (3) Randomly select an operation $O_{i,j}$ on M. If $O_{i,j}$ has more than one candidate machine, then randomly replace it with another machine; and (4) Evaluate the newly-generated neighboring solution and replace the current solution.

LS3: Local search for a random operation: (1) For a solution X, randomly select one operation $O_{i,j}$ on the scheduling vector; (2) If $O_{i,j}$ has more than one candidate machine, then randomly replace it with another machine; and (3) Evaluate the newly-generated neighboring solution and replace the current solution if the latter is better.

It should be noted that, the main computational burden of the three mutation operators LS1 to LS3 consumed during the replacement of another candidate machine for the selected operations. It is obvious that the computational complexity of the three mutation operators for routing vector is $O(n^2)$.

2) Mutation for scheduling vector:

For the scheduling vector, we present the following two types of mutation operators.

LS5: Two-point swap (TPS) operator. The TPS operator generates a neighboring solution by swapping two selected jobs. Fig. 5 (a) illustrates the procedure of the TPS operator.

LS6: Two-point insertion (TPI) operator. The TPI operator generates a neighboring solution by inserting one job before the position of another selected job. Fig. 5 (b) illustrates the procedure of the TPI operator.

It should be noted that, the main computational burden of the two mutation operators LS5 to LS6 consumed during the computation of the objective functions. It is obvious that the computational complexity of the two mutation operators for scheduling vector is $O(nm)$.

After executing the mutation operators randomly, each cloning antibody generates a neighboring antibody. These neighboring antibodies construct a set named NP.

G. Population diversity

To maintain the diversity of the population, the population is reduced by eliminating antibodies with high crowding values. The crowding value is defined as a value that represents the crowding degree among neighboring antibodies. Given a population of antibodies P := {X_1,X_2,...,X_n}, the crowding value of X_i is calculated as follows:

$$X_i = \begin{cases} 1 - \frac{(f_{i+1} - f_{i-1})}{(f_{max} - f_{min})}, i = 2, 3, ..., n-1. \\ 1 - \frac{(f_2 - f_1)}{(f_{max} - f_{min})}, i = 1. \\ 1 - \frac{(f_n - f_{n-1})}{(f_{max} - f_{min})}, i = n. \end{cases} \quad (30)$$

where f_{max} and f_{min} are the maximum and minimum affinity values among all antibodies, respectively. For example, given three solutions $X_1 = (\prod_1, \Re_1)$, $X_2 = (\prod_2, \Re_2)$, and $X_3 = (\prod_3, \Re_3)$, these solutions are sorted according to the non-increasing order of their affinity. The crowding value of X_1 is calculated by $(f_2 - f_1)/(f_{max} - f_{min})$, where f_1 and f_2 represent the affinity values of X_1 and X_2. Then, the crowding value of X_2 is calculated by $(f_3 - f_1)/(f_{max} - f_{min})$, and that of X_3 is calculated by $(f_3 - f_2)/(f_{max} - f_{min})$.

Based on the crowding degree values, the population diversity mechanism is presented in Algorithm 4.

Algorithm 4: Population diversity procedure

Input: the current population
Output: the population after suppression.

1 Merge all of the antibodies in NP and CP, and the number of the antibodies is $n_c + n_c(n_c + 1)/2$;
2 Sequence all the $n_c + n_c(n_c + 1)/2$ antibodies according to the non-increasing order of affinity function values of them, and store them in a set named MP;
3 Sort all the antibodies in MP according to the non-increasing order of their affinities;
4 Calculate the crowding degree value CD_i for the antibody X_i;
5 Eliminate the antibodies with the crowding degree values bigger than a given threshold value CR_{max};
6 After the suppression process, let the size of the current population is P_{sc}. if P_{sc}<P_{size} then perform the following step P_{size}-P_{sc} times;
7 To apply the mutation operator for the best antibody found so far, and store the neighboring solution into the current population.

From Algorithm 4, we find that: (1) the computational burden of Steps 2 and 3 is $log_2^{P_{size}}$; (2) the computational complexity of Steps 6 to 7 is $O(n^2m)$. Therefore, the computational complexity of Algorithm 4 is $O(n^2m)$.

H. SA-based exploration heuristic

In this study, we embed the SA-based acceptance method in the proposed IAIS algorithm to enhance the exploration abilities of the algorithm. The temperature for the SA algorithm is set to $\frac{T \cdot \sum_{k=1}^{m} \sum_{j=1}^{n} \sum_{i=1}^{\theta_j} T_{i,j,k}}{n \cdot m \cdot 10}$, where T is set as a parameter for the proposed algorithm.

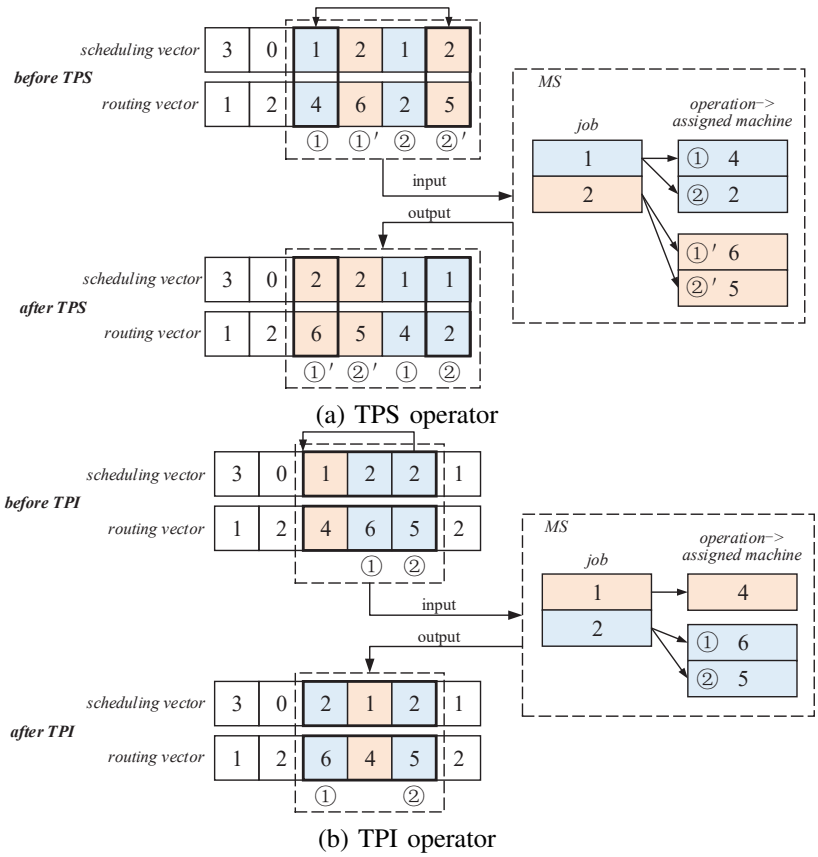


Fig. 5. Mutation operators.

Detailed steps of the SA-based exploration heuristic are provided in Algorithm 5. The computational complexity of Algorithm 5 is $O(n^2m)$.

Algorithm 5: SA-based exploration heuristic

Input: the best solution found so far
Output: the best solution after applying the SA-based exploration method

Let $count = 1$;
for $count < n/4$ **do**
  Let ss be the best antibody found so far;
  Peform the local search operator on ss to generate a new antibody ss';
  Let $\Delta = ss'$-ss;
  if $\Delta < 0$ **then**
    Replace the best solution with ss' ;
  else
    Let $rr = e^{\Delta/Temperature}$;
    Let $Accept_p = 1/rr$;
    Generate a random number r_1 range in [0,1];
    if $r_1 < Accept_p$ **then**
      Replace the best solution with ss' ;
    end
  end
end

V. Experimental results

In Section IV, the proposed algorithm has been described in detail, where the IT2FS operators are used to schedule the operations and compute the objectives (see details in Section II and III). In this section, we present detailed comparisons to evaluate the performance of the proposed IAIS algorithm discussed in Section IV. The compared algorithms are coded in C++ on an Intel Core *i*7 3.4-GHz PC with 16 GB memory. It should be noted that 30 independent runs for each instance is commonly used to make fair comparisons in many references[24]. Therefore, to verify the effectiveness and efficiency of the proposed algorithm, after 30 independent runs, the resulting best solutions were collected for performance comparisons.

The compared algorithms include two types: (1) population-based algorithms, including the genetic algorithm with glowworm swarm optimization (GA-GSO) algorithm (Liu et al., 2019) [33], the discrete ABC (DABC) algorithm (Gao et al., 2016) [24], the TPM algorithm (Lei et al., 2019) [17], and the hCEA (Sun et al., 2019) [31]; and (2) local-search-based methods, including the ASA algorithm (CruzChávez et al., 2017) [14]. All the above-listed algorithms are used to solve various FJSPs. For example, the GA-GSO algorithm was developed to solve FJSPs with crane transportation, which is also examined in this study. DABC, TPM, and hCEA were designed for Type-1 fuzzy FJSPs, and the two local search methods, MIG and ASA, were used to solve classical FJSPs. Note that we

were unable to find any algorithm for solving IT2FS FJSPs. Therefore, we implemented all of the compared algorithms to solve the considered problem, and their parameters were set according to their respective literature. The relative percentage increase (RPI) value was used as the performance measure, which is calculated as follows:

$$RPI(C) = (f_c - f_b)/f_b * 100 \quad (31)$$

where f_b is the best result obtained by the seven algorithms and f_c is the best result for each instance obtained by a given algorithm.

A. Experimental instances

Based on the realistic instances obtained from[33], we randomly generated 30 different scales of instances, where n =20, 30, 40, 50, 80, 100, and m = 6, 7, 8, 9, 10. In addition, the total number of operations of each job was distributed uniformly in the interval $[m/2, m]$. The instances can be found on the following website: http://ischedulings.com/data/T2FJSP_instances.rar. The results obtained by the proposed IAIS algorithm can be found on the following website: http://ischedulings.com/data/TFS_IAIS_results.rar.

B. Experimental parameters

Three parameters for the proposed algorithms included a temperature-related parameter for the SA-based exploration heuristic (T), the clone number (nc), and the crowding threshold value for the population diversity (CR_{max}). The stop criterion was set to CPU = $30\times\lceil n/50\rceil$s. The design of experiments (DOE) approach was used to calibrate the parameters, and the parameter levels were as follows:

. T=0.1, 0.3, 0.5, 0.8.
. n_c=5, 10, 15, 20.
. CR_{max}= 0.5, 0.7, 0.8, 0.9.

To calibrate the parameters fairly, we randomly generated another set of instances. The results were collected after 30 independent runs for each instance. Plots of the factor-level trends with the selection of parameters T, n_c, and CR_{max} are shown in Fig. 6, where T = 0.5, n_c = 10, and CR_{max} = 0.8 yielded a much better fitness value than the other values.

C. Efficiency of initialization heuristic

The initialization heuristic discussed in Section IV-D was used to enhance the quality and diversity of the initial population. To evaluate the initialization heuristic, we coded two types of IAIS algorithms: IAIS-NI with a random initialization heuristic and IAIS with all components discussed in Section IV. All other components of the two compared algorithms were set to the same values.

Table I presents the results of solving the given 30 instances using the two compared algorithms. The first column displays the instance name, while the second column displays the best fitness value for each instance. The following two columns describe the fitness values collected by IAIS-NI and IAIS, respectively, while the RPI values obtained by the two compared

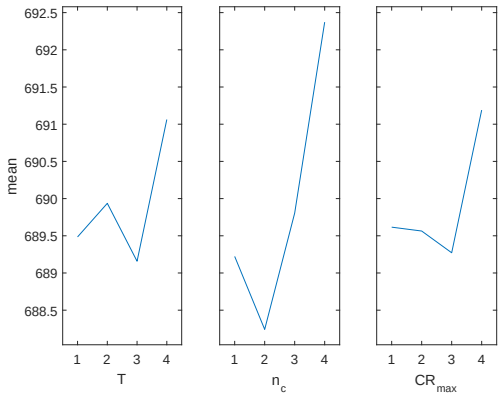


Fig. 6. Factor-level trends of parameters .

algorithms are provided in the last two columns, respectively. The two compared algorithms were performed with CPU = $30\times\lceil n/50\rceil$s for each instance.

The comparison results indicate that (1) IAIS obtained better results for 30 instances; (2) from the RPI values listed in the last two columns and the last row in the table, IAIS obtained a significantly better result than IAIS-NI; and (3) in summary, the proposed initialization heuristic was effective for the considered problem.

Multifactor analysis of variance (ANOVA) was also used to evaluate whether the comparison results were significant. Fig. 7 (a) reveals that the resulting p-value was close to zero, which reveals that the difference was significant after applying the proposed initialization heuristic.

D. Efficiency of mutation heuristics

The mutation heuristic discussed in Section IV-F is another contribution of the proposed algorithm. To evaluate its performance, we also coded two types of IAIS algorithms: IAIS-NM with a random mutation heuristic and IAIS with all components discussed in Section IV. All other components of the two compared algorithms were identical. Table II provides the results of solving the given 30 instances using the two compared algorithms. The two compared algorithms were performed with CPU = $30\times\lceil n/50\rceil$s for each instance.

It can be observed from the comparison result that: (1) the total number of superior results obtained by IAIS was 29, which is obviously better than that obtained by IAIS-NM; and (2) from the RPI values listed in the last two columns, IAIS obtained an average value of 0.01, which is significantly better than the result obtained by IAIS-NM.

Fig. 7 (b) provides ANOVA comparisons of the two methods. The figure indicates that there are significant differences between the compared methods and that the proposed mutation heuristic enhances the performance of the proposed algorithm.

E. Effectiveness of population diversity heuristics

To evaluate the performance of the population diversity heuristic discussed in Section IV-D, we coded two types of IAIS algorithms: IAIS-ND without the population diversity

TABLE I
COMPARISON RESULTS BETWEEN IAIS-NI AND IAIS.

Instance	best	Fitness		RPI	
		IAIS-NI	IAIS	IAIS-NI	IAIS
j20m6	453.493	465.347	453.493	2.61	0.00*
j20m7	406.93	434.88	406.93	6.87	0.00*
j20m8	553.54	600.97	553.54	8.57	0.00*
j20m9	556.04	572.86	556.04	3.02	0.00*
j20m10	646.88	680.96	646.88	5.27	0.00*
j30m6	645.94	680.95	645.94	5.42	0.00*
j30m7	672.03	692.12	672.03	2.99	0.00*
j30m8	814.91	840.96	814.91	3.20	0.00*
j30m9	829.58	859.26	829.58	3.58	0.00*
j30m10	971.25	1006.49	971.25	3.63	0.00*
j40m6	908.22	981.51	908.22	8.07	0.00*
j40m7	863.35	906.07	863.35	4.95	0.00*
j40m8	1184.42	1235.31	1184.42	4.30	0.00*
j40m9	978.75	1034.98	978.75	5.74	0.00*
j40m10	1158.46	1250.43	1158.46	7.94	0.00*
j50m6	1115.64	1165.69	1115.64	4.49	0.00*
j50m7	1077.8	1141.39	1077.8	5.90	0.00*
j50m8	1376.33	1446.57	1376.33	5.10	0.00*
j50m9	1378.36	1441.56	1378.36	4.59	0.00*
j50m10	1779.67	1875.26	1779.67	5.37	0.00*
j80m6	1868.2	2010.21	1868.2	7.60	0.00*
j80m7	1839.61	1897.68	1839.61	3.16	0.00*
j80m8	2112.36	2299.03	2112.36	8.84	0.00*
j80m9	2078.4	2279.86	2078.4	9.69	0.00*
j80m10	2436.08	2629.47	2436.08	7.94	0.00*
j100m6	2420.02	2578.77	2420.02	6.56	0.00*
j100m7	2010.49	2133.46	2010.49	6.12	0.00*
j100m8	2766.95	2991.43	2766.95	8.11	0.00*
j100m9	2688.4	2923.97	2688.4	8.76	0.00*
j100m10	3538.56	3843.82	3538.56	8.63	0.00*
mean	1404.36	1496.71	1404.36	5.90	0.00*

-*means the better values

TABLE II
RESULTS OF COMPARISON BETWEEN IAIS-NM AND IAIS.

Instance	best	Fitness		RPI	
		IAIS-NM	IAIS	IAIS-NM	IAIS
j20m6	453.49	454.32	453.49	0.18	0.00*
j20m7	406.93	411.22	406.93	1.05	0.00*
j20m8	553.54	578.93	553.54	4.59	0.00*
j20m9	556.04	568.81	556.04	2.30	0.00*
j20m10	646.88	673.94	646.88	4.18	0.00*
j30m6	643.05	643.05	645.94	0.00*	0.45
j30m7	672.03	678.01	672.03	0.89	0.00*
j30m8	814.91	846.06	814.91	3.82	0.00*
j30m9	829.58	861.84	829.58	3.89	0.00*
j30m10	971.25	1056.27	971.25	8.75	0.00*
j40m6	908.22	956.86	908.22	5.36	0.00*
j40m7	863.35	870.48	863.35	0.83	0.00*
j40m8	1184.42	1271.62	1184.42	7.36	0.00*
j40m9	978.75	1021.32	978.75	4.35	0.00*
j40m10	1158.46	1313.49	1158.46	13.38	0.00*
j50m6	1115.64	1145.26	1115.64	2.65	0.00*
j50m7	1077.8	1113.39	1077.8	3.30	0.00*
j50m8	1376.33	1519.87	1376.33	10.43	0.00*
j50m9	1378.36	1482.74	1378.36	7.57	0.00*
j50m10	1779.67	2014.04	1779.67	13.17	0.00*
j80m6	1868.2	1921.44	1868.2	2.85	0.00*
j80m7	1839.61	1910.42	1839.61	3.85	0.00*
j80m8	2112.36	2385.39	2112.36	12.93	0.00*
j80m9	2078.4	2310.36	2078.4	11.16	0.00*
j80m10	2436.08	2768.17	2436.08	13.63	0.00*
j100m6	2420.02	2563.19	2420.02	5.92	0.00*
j100m7	2010.49	2083.48	2010.49	3.63	0.00*
j100m8	2766.95	3045.31	2766.95	10.06	0.00*
j100m9	2688.4	3047.42	2688.4	13.35	0.00*
j100m10	3538.56	4169.73	3538.56	17.84	0.00*
mean	1404.26	1522.88	1404.36	6.44	0.01*

-*means the better values

heuristic and IAIS with all components discussed in Section IV.

Table III reveals that the proposed algorithm with the population diversity heuristic was efficient, and the results presented in Fig. 7 (c) demonstrate that the IAIS with the population diversity heuristic exhibited significantly better performance. It can be thus concluded that the proposed population diversity heuristic enhances the search ability of the proposed algorithm.

F. Effectiveness of SA-based exploration heuristics

In this subsection, we present our evaluation of the SA-based exploration heuristic discussed in Section IV-H. We coded two types of IAIS algorithms: IAIS-NS without the SA-based exploration heuristic and IAIS with all components discussed in section IV. The results provided in Table IV and Fig. 7 (d) demonstrate that the two compared methods are significantly different, which further support the effectiveness of the proposed SA-based exploration heuristic.

G. Test on the four manufacturing instances

This section investigates the extension version of the four optimization problems collected from the Ref [33]. The four typical complex production problems are collected from a large cement equipment manufacturing company located in Tianjin, China. The company mainly produces large complex cement equipment, including rotary kilns, vertical mills, roll squeezers. In the extension version of problems, the IT2FS

TABLE III
RESULTS OF COMPARISON BETWEEN IAIS-ND AND IAIS.

Instance	best	Fitness		RPI	
		IAIS-ND	IAIS	IAIS-ND	IAIS
j20m6	452.31	452.31	453.49	0.00*	0.26
j20m7	406.93	408.58	406.93	0.41	0.00*
j20m8	553.54	571.82	553.54	3.30	0.00*
j20m9	549.55	549.55	556.04	0.00*	1.18
j20m10	638.69	638.69	646.88	0.00*	1.28
j30m6	645.94	646.75	645.94	0.13	0.00*
j30m7	650.56	650.56	672.03	0.00*	3.30
j30m8	812.57	812.57	814.91	0.00*	0.29
j30m9	829.58	831.07	829.58	0.18	0.00*
j30m10	971.25	973.93	971.25	0.28	0.00*
j40m6	908.22	929.16	908.22	2.31	0.00*
j40m7	857.32	857.32	863.35	0.00*	0.70
j40m8	1174.2	1174.2	1184.42	0.00*	0.87
j40m9	978.75	984.63	978.75	0.60	0.00*
j40m10	1158.46	1188.87	1158.46	2.63	0.00*
j50m6	1110.5	1110.5	1115.64	0.00*	0.46
j50m7	1067.25	1067.25	1077.8	0.00*	0.99
j50m8	1376.33	1408.96	1376.33	2.37	0.00*
j50m9	1378.36	1380.00*	1378.36	0.12	0.00*
j50m10	1779.67	1919.3	1779.67	7.85	0.00*
j80m6	1868.2	1876.01	1868.2	0.42	0.00*
j80m7	1822.75	1822.75	1839.61	0.00*	0.92
j80m8	2112.36	2173.64	2112.36	2.90	0.00*
j80m9	2078.4	2145.67	2078.4	3.24	0.00*
j80m10	2436.08	2586.68	2436.08	6.18	0.00*
j100m6	2420.02	2423.9	2420.02	0.16	0.00*
j100m7	2010.49	2025.04	2010.49	0.72	0.00*
j100m8	2766.95	2877.55	2766.95	4.00	0.00*
j100m9	2688.4	2847.13	2688.4	5.90	0.00*
j100m10	3538.56	4074.39	3538.56	15.14	0.00*
mean	1401.41	1446.96	1404.36	1.96	0.34*

-*means the better values

TABLE IV
RESULTS OF COMPARISON BETWEEN IAIS-NS AND IAIS.

Instance	best	Fitness		RPI	
		IAIS-NS	IAIS	IAIS-NS	IAIS
j20m6	453.49	459.96	453.49	1.43	0.00*
j20m7	406.93	413.53	406.93	1.62	0.00*
j20m8	553.54	589.53	553.54	6.50	0.00*
j20m9	556.04	562.82	556.04	1.22	0.00*
j20m10	646.88	654.3	646.88	1.15	0.00*
j30m6	641.54	641.54	645.94	0.00*	0.69
j30m7	660.76	660.76	672.03	0.00*	1.71
j30m8	791.36	791.36	814.91	0.00*	2.98
j30m9	825.38	825.38	829.58	0.00*	0.51
j30m10	971.25	975.49	971.25	0.44	0.00*
j40m6	908.22	939.05	908.22	3.39	0.00*
j40m7	858.49	858.49	863.35	0.00*	0.57
j40m8	1184.42	1186.43	1184.42	0.17	0.00*
j40m9	978.75	994.48	978.75	1.61	0.00*
j40m10	1158.46	1183.94	1158.46	2.20	0.00*
j50m6	1115.64	1119.22	1115.64	0.32	0.00*
j50m7	1077.8	1083.15	1077.8	0.50	0.00*
j50m8	1376.33	1396.11	1376.33	1.44	0.00*
j50m9	1378.36	1402.19	1378.36	1.73	0.00*
j50m10	1779.67	1815.08	1779.67	1.99	0.00*
j80m6	1868.2	1901.72	1868.2	1.79	0.00*
j80m7	1828.66	1828.66	1839.61	0.00*	0.60
j80m8	2112.36	2170.15	2112.36	2.74	0.00*
j80m9	2078.4	2116.36	2078.4	1.83	0.00*
j80m10	2436.08	2476.91	2436.08	1.68	0.00*
j100m6	2420.02	2426.53	2420.02	0.27	0.00*
j100m7	2010.49	2029.54	2010.49	0.95	0.00*
j100m8	2766.95	2823.54	2766.95	2.05	0.00*
j100m9	2688.4	2751.19	2688.4	2.34	0.00*
j100m10	3538.56	3639.02	3538.56	2.84	0.00*
mean	1402.38	1423.88	1404.36	1.41	0.23

-*means the better values

processing times collected from the realistic production system are embedded. The four instances are given in the website http://ischedulings.com/data/, where each line tells the IT2FS processing time for each operation on each machine.

The GA-GSO in [33] is selected as the compared algorithm, and the comparison results are collected in Table V. The first column in the table reports the weight values, from 0.1 to 0.9. The second column gives the instance numbers, from 1 to 4. The next column lists the best values collected from the two compared algorithms. The fitness values obtained by IAIS and GA-GSO are reported in the following two columns, respectively. Then, the last columns list the RPI values for the two compared algorithms.

It can be concluded from the table that: (1) the proposed IAIS algorithm obtained 26 better solutions out of the given 36 instances, which is obviously better than GA-GSO; and (2) the average performance from the last line in the table shows that IAIS is better than the compared algorithm. For example, the average value obtained by IAIS is 0.17, which is about 0.28 times of the results collected by GA-GSO algorithm.

Fig. 7 (e) gives the ANOVA comparisons of the four manufacturing instances considering IAIS and GA-GSO. It also shows that the two compared methods are significantly different and the proposed IAIS algorithm is better.

H. Comparison with other algorithms

To further evaluate the performance of the proposed IAIS algorithm, we selected the following algorithms for comparison: GA-GSO, ASA, DABC, TPM, and hCEA. For each compared algorithm, the results obtained after 30 independently runs are used to make detailed comparisons. The best value for each instance is collected by the results after performing each algorithm with CPU = $30 \times \lceil n/50 \rceil$s. Note that all the above selected algorithms are not used to solve the considered IT2FS FJSPs. Because there is no existing algorithms for solving the considered problem, we coded the above selected algorithms to solve IT2FS FJSPs. To make a fair comparison, we implement all the compared algorithms to include all feasible components from their respective literature, except the encoding, decoding, and the IT2FS-based affinity calculation heuristic discussed in Section IV.

The comparison results for the given 30 instances under =0.1, 0.5, and 0.9, are given in Table V, VI, and VII, respectively. In these three tables, the first column describes the instance scales, and the second column gives the fitness values obtained by the proposed IAIS algorithm. The following five columns list the gap values for the other five compared algorithms, i.e., GA-GSO, DABC, ASA, TPM, and hCEA, respectively. The gap values are calculated as follows:

$$Gap_{compare} = (F_{compare} - F_{IAIS})/F_{IAIS} \times 100 \qquad (32)$$

It can be concluded from three tables that: (1) considering the situation where ω=0.1, compared with the other five algorithms, the performance of IAIS increases by 5.07%, 8.32%, 0.68%, 4.58%, and 3.61%, respectively; (2) when =0.5, compared with the other five algorithms, the performance of IAIS increases by 6.44%, 10.76%, 1.00%, 5.72%, and 4.71%, respectively; (3) considering the situation where =0.9, compared with the other five algorithms, the performance of IGABC increases by 13.70%, 22.82%, 1.96%, 12.59%, and 9.89%, respectively; and (4) in a nutshell, the proposed IAIS is efficient compared with the other five efficient algorithms, especially for the relative large scale problems.

In addition, Fig. 8 shows the distribution of solutions with different weights optimized by the six compared algorithms. As observed from Fig. 8, IAIS shows competitive performance considered makespan, energy consumption, and the weighted fitness value.

Fig. 9 shows the comparison of the convergence curves for different scale of problems, including "j20m6", "j20m10", "j30m10", "j50m10", and "j100m10". It can be concluded from the convergence curves that the proposed IAIS algorithm shows better convergence abilities for various IT2FS FJSPs. Fig.10 represents the Gantt chart for the best solution obtained by IAIS for "j20m6", where each operation is represented by two nonsymmetrical triangular interval T2FS values, i.e., the type-2 fuzzy starting time and completion time. The type-2 fuzzy starting time is illustrated under the line of the machine, while the type-2 fuzzy completion time is given upper the line of the machine. For example, on machine M_6, the first operation is $O_{3,1}$, which is represented two nonsymmetrical triangular interval T2FS values. The Gantt chart also verify the effectiveness of the proposed algorithm.

TABLE V
COMPARISONS OF THE FOUR TYPICAL COMPLEX PROBLEMS WITH IT2FS PROCESSING TIMES.

w	Instance	Best	fitness		RPI	
			LAIS	GA-GSO	LAIS	GA-GSO
0.10	1.00	507.06	507.23	507.06	0.03	0.00*
	2.00	624.35	624.35	626.92	0.00*	0.41
	3.00	696.15	696.89	696.15	0.11	0.00*
	4.00	716.92	716.92	720.54	0.00*	0.50
0.20	1.00	464.21	464.40	464.21	0.04	0.00*
	2.00	575.39	577.40	575.39	0.35	0.00*
	3.00	640.08	640.08	644.76	0.00*	0.73
	4.00	661.91	661.91	666.63	0.00*	0.71
0.30	1.00	419.38	419.38	419.38	0.00*	0.00*
	2.00	523.26	523.26	523.77	0.00*	0.10
	3.00	585.20	585.20	592.75	0.00*	1.29
	4.00	601.31	601.31	604.66	0.00*	0.56
0.40	1.00	374.56	374.56	376.69	0.00*	0.57
	2.00	466.23	466.23	470.16	0.00*	0.84
	3.00	528.78	528.78	535.53	0.00*	1.28
	4.00	539.36	545.42	539.36	1.12	0.00*
0.50	1.00	330.66	330.90	330.66	0.07	0.00*
	2.00	416.29	416.29	417.51	0.00*	0.29
	3.00	465.42	465.42	473.55	0.00*	1.75
	4.00	476.64	476.64	479.02	0.00*	0.50
0.60	1.00	283.60	283.60	283.89	0.00*	0.10
	2.00	357.18	357.18	358.76	0.00*	0.44
	3.00	405.09	405.09	407.64	0.00*	0.63
	4.00	413.31	413.31	414.65	0.00*	0.32
0.70	1.00	236.00	239.09	236.00	1.31	0.00*
	2.00	298.05	298.05	300.81	0.00*	0.93
	3.00	332.56	332.56	339.97	0.00*	2.23
	4.00	341.31	341.31	344.94	0.00*	1.06
0.80	1.00	187.38	188.90	187.38	0.81	0.00*
	2.00	235.76	235.76	236.91	0.00*	0.49
	3.00	261.68	261.68	262.51	0.00*	0.32
	4.00	267.42	267.42	272.30	0.00*	1.83
0.90	1.00	137.36	137.36	139.26	0.00*	1.39
	2.00	170.31	170.31	172.38	0.00*	1.21
	3.00	188.56	192.66	188.56	2.17	0.00*
	4.00	197.64	197.64	200.09	0.00*	1.24
Mean		414.62	415.12	416.96	0.17*	0.60

VI. CONCLUSIONS

This study considered the crane transportation flexible job shop scheduling problem, where the type-2 fuzzy processing time constraint is studied. The objective is to minimize the weighted sum of type-2 fuzzy makespan and the energy consumption. All these interval type-2 fuzzy values are used to schedule all operations according to the ranking operator discussed in Section II-B, and then the maximum completion time is obtained as one part of the objective functions.

To the best of my knowledge, this study is the first work to consider this type of optimization problem. We firstly present the mathematical model of the problem. Then, considering the type-2 fuzzy values, a novel affinity calculation heuristic is proposed. An efficient initialization heuristic with four different rules are embedded to generate an initial population with quality and diversity. Six types of mutation methods are presented to perform search tasks in different searching spaces. Moreover, a novel population diversity method is conducted to erase solutions with crowding degree values. Finally, the SA-based exploration heuristic enhances the global search abilities of the proposed algorithm. The proposed IAIS algorithm is used to generate a feasible and optimal solution for the considered problem, while considering the processing time as a IT2FS value rather than a deterministic value or a TFN value. Therefore, the proposed algorithm considering the IT2FS values can be adapt to solving the complex flexible job shop scheduling problem under high level of uncertainty.

In our future work, we will consider following tasks: (1) apply the IT2FS FJSPs in different types of realistic applications, and to consider other realistic constraints, such as operation related setup time and distributed processing features; (2) combine with other efficient heuristics, such as species driven methods and big data driven methods, and thus enhance the performance of the proposed algorithm; (3) embed the multi-objective optimization methods, such as Pareto-based algorithms, multi-objective evolutionary algorithm based on decomposition (MOEA/D), and therefore develop an efficient multi-objective algorithm for the IT2FS FJSPs; and (4) consider other types of IT2FS operators in the proposed algorithm, such as IT2FS addition operator and IT2FS ranking operator, and therefore, the proposed algorithm can adapt to other types of applications.

REFERENCES

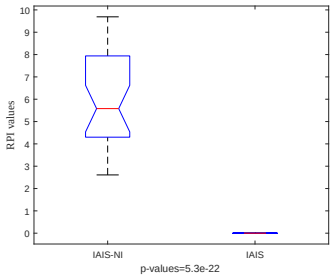
[1] M. Nawaz, E. Enscore Jr, and I. Ham, "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem," Omega, vol. 11, no. 1, pp. 91-95, 1983.

TABLE VI
RESULTS OF COMPARISON OF SIX ALGORITHMS (ω=0.1).

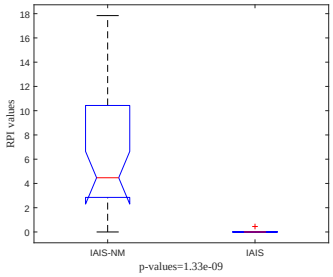
Inst	IAIS	Gap				
		GA-GSO	DABC	ASA	TPM	hCEA
j20m6	1791.51	2.25	1.95	-0.17	1.40	1.66
j20m7	1674.55	4.13	3.05	2.48	3.62	2.40
j20m8	2406.09	4.74	5.50	1.92	3.98	3.78
j20m9	2511.51	2.85	5.93	-0.07	4.65	3.15
j20m10	2823.41	6.43	7.36	0.82	7.27	5.58
j30m6	2422.84	2.30	3.60	0.75	2.53	2.30
j30m7	2743.26	1.29	3.16	-0.73	1.71	0.16
j30m8	3428.94	3.62	6.08	0.42	3.08	3.00
j30m9	3686.22	3.83	5.81	-0.74	3.85	3.35
j30m10	4243.04	8.05	10.66	0.44	7.06	5.07
j40m6	3566.96	3.12	4.84	-1.25	2.71	2.38
j40m7	3581.58	4.26	6.23	0.75	3.60	2.12
j40m8	5099.02	4.73	9.66	-0.22	4.46	3.64
j40m9	4300.79	5.70	8.45	-0.42	4.95	3.42
j40m10	5130.61	7.06	9.87	0.93	6.64	5.41
j50m6	4294.95	4.51	5.89	1.10	3.13	2.89
j50m7	4389.91	3.63	6.50	0.11	2.09	1.42
j50m8	5839.98	6.54	8.83	0.81	6.00	4.17
j50m9	5981.52	6.97	9.62	1.00	4.46	4.95
j50m10	7576.99	10.32	14.72	1.79	9.38	8.36
j80m6	7327.82	2.99	8.34	0.10	3.31	2.61
j80m7	7593.14	3.37	7.43	0.48	3.32	1.39
j80m8	9136.47	5.57	10.68	1.02	4.34	4.50
j80m9	8822.86	6.41	14.03	0.56	5.83	3.40
j80m10	10292.10	7.35	11.90	0.78	6.74	5.67
j100m6	9730.68	3.45	8.56	0.90	2.77	2.63
j100m7	8361.90	3.80	9.38	0.32	2.79	2.07
j100m8	11310.00	6.51	12.13	1.00	5.41	4.34
j100m9	10998.00	6.65	13.35	2.34	7.17	5.45
j100m10	13970.50	9.73	16.09	3.30	9.07	6.94
mean	5834.57	5.07	8.32	0.68	4.58	3.61

TABLE VII
RESULTS OF COMPARISON OF SIX ALGORITHMS (ω=0.5).

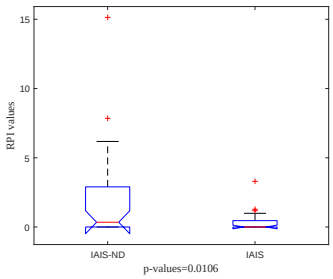
Inst	IAIS	Gap				
		GA-GSO	DABC	ASA	TPM	hCEA
j20m6	1147.21	2.15	2.97	0.09	3.20	1.83
j20m7	1080.39	3.46	3.69	-0.13	3.11	2.26
j20m8	1550.84	4.30	8.00	1.77	4.52	5.36
j20m9	1596.00	4.47	5.77	-0.05	3.99	4.37
j20m10	1789.42	7.11	9.85	0.48	9.31	7.24
j30m6	1547.01	6.64	7.92	3.81	4.34	6.20
j30m7	1744.75	4.20	5.43	0.88	2.81	3.40
j30m8	2167.98	6.69	8.67	0.04	5.99	5.32
j30m9	2296.90	7.96	10.01	0.25	4.48	3.10
j30m10	2674.48	9.76	13.23	1.10	9.44	8.15
j40m6	2310.11	4.40	5.69	0.24	2.49	2.56
j40m7	2266.90	5.03	8.28	1.24	4.89	3.54
j40m8	3225.54	6.61	10.76	-0.09	5.40	4.05
j40m9	2701.55	7.90	12.81	1.19	7.03	5.90
j40m10	3236.92	7.31	10.57	0.40	6.92	4.97
j50m6	2794.49	2.79	6.81	-0.65	2.26	2.79
j50m7	2776.94	3.90	9.53	0.45	6.24	2.30
j50m8	3672.63	7.52	12.97	2.23	8.16	5.50
j50m9	3714.25	9.66	14.16	2.30	7.61	7.40
j50m10	4753.49	14.41	20.64	2.81	11.75	9.70
j80m6	4717.78	3.41	9.71	0.76	3.57	2.30
j80m7	4807.19	2.56	7.96	0.83	3.83	2.22
j80m8	5774.17	6.69	13.60	0.14	5.08	4.99
j80m9	5596.16	8.69	18.05	0.18	7.52	5.59
j80m10	6499.63	9.75	17.09	1.38	8.24	7.09
j100m6	6228.39	3.96	8.98	1.24	3.72	2.06
j100m7	5307.45	4.92	9.95	0.78	3.73	2.07
j100m8	7184.37	7.82	15.89	1.94	6.75	6.32
j100m9	7039.44	8.01	14.93	1.53	7.11	5.38
j100m10	9046.69	11.17	18.98	2.96	8.25	7.41
mean	3708.30	6.44	10.76	1.00	5.72	4.71



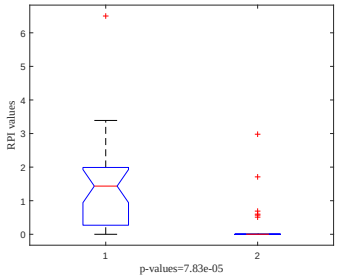
(a) ANOVA result for IAIS-NI and IAIS



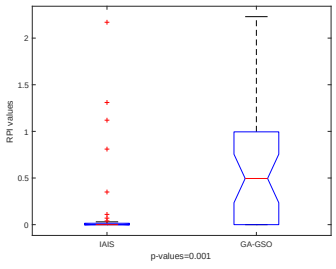
(b) ANOVA result for IAIS-NM and IAIS



(c) ANOVA result for IAIS-ND and IAIS



(d) ANOVA result for IAIS-NS and IAIS



(e) ANOVA comparisons of the four manufacturing instances

Fig. 7. ANOVA results for comparison.

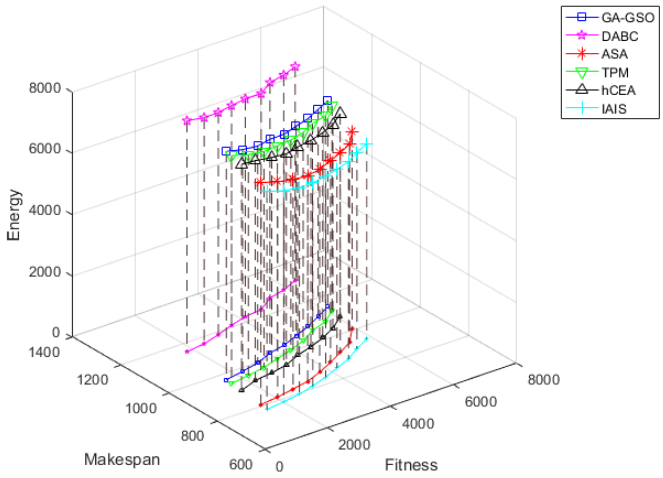
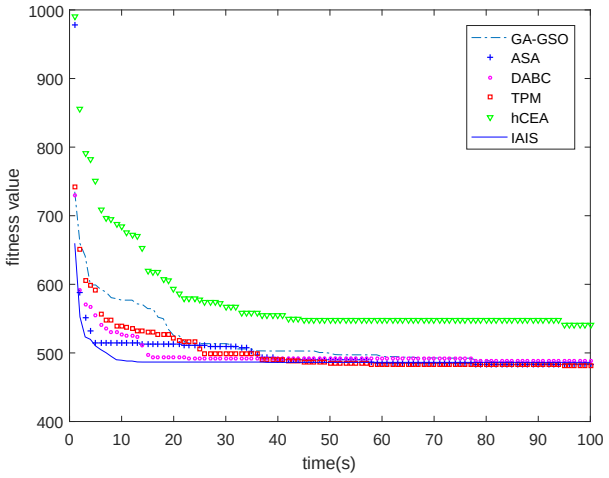


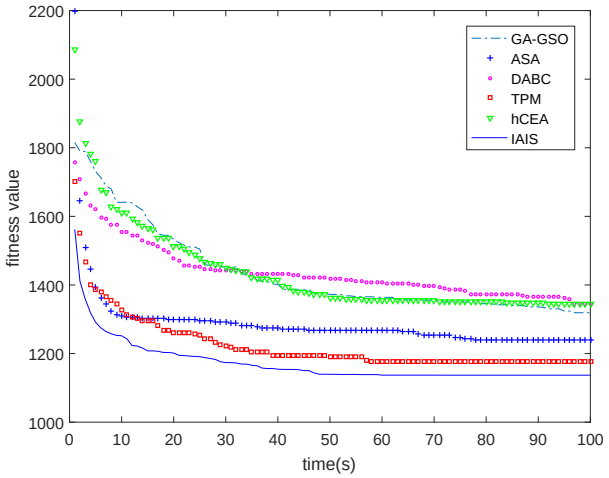
Fig. 8. Distribution of solutions with different weights.

TABLE VIII
RESULTS OF COMPARISON OF SIX ALGORITHMS (ω=0.9).

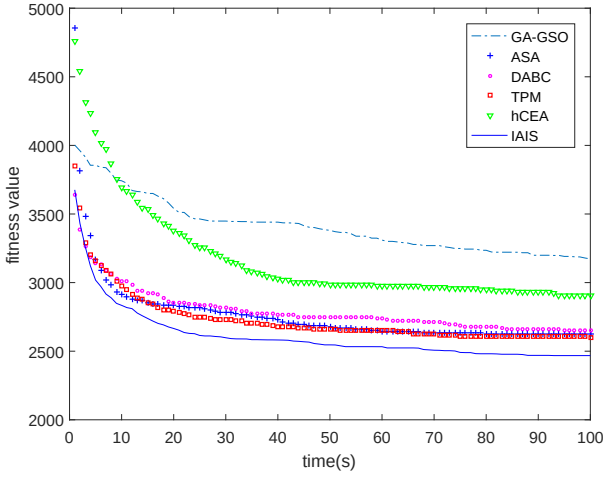
Inst	IAIS	Gap				
		GA-GSO	DABC	ASA	TPM	hCEA
j20m6	461.32	2.13	3.97	-0.21	6.47	4.96
j20m7	414.30	3.44	9.86	1.19	8.34	6.24
j20m8	600.67	12.48	12.95	-2.88	8.86	5.99
j20m9	558.48	16.79	14.61	1.43	15.65	15.48
j20m10	665.17	22.19	20.21	0.08	18.60	16.80
j30m6	653.78	6.34	7.62	-0.32	5.77	4.18
j30m7	670.77	7.99	10.28	-0.78	9.44	6.02
j30m8	821.44	13.44	18.41	1.39	12.63	8.56
j30m9	835.71	18.31	21.78	0.57	16.52	14.72
j30m10	988.17	17.67	29.20	2.39	20.67	15.67
j40m6	943.94	7.89	9.63	1.03	4.75	5.74
j40m7	848.86	10.61	18.13	3.59	9.94	11.32
j40m8	1195.59	14.04	22.94	2.49	13.30	8.58
j40m9	989.01	15.50	30.45	1.48	16.85	12.99
j40m10	1187.33	16.30	27.65	2.74	17.27	13.66
j50m6	1129.48	7.44	10.86	0.68	5.12	5.85
j50m7	1080.80	11.41	16.48	1.49	9.16	7.79
j50m8	1410.89	16.11	27.35	0.72	14.71	11.93
j50m9	1396.87	19.99	29.22	1.99	18.06	13.89
j50m10	1790.63	29.60	43.75	2.74	24.87	21.83
j80m6	1867.50	9.56	17.10	3.56	6.58	5.55
j80m7	1837.52	7.24	17.22	0.14	5.79	5.96
j80m8	2106.11	17.20	32.58	4.89	14.25	10.70
j80m9	2101.24	18.59	38.40	5.48	15.53	12.29
j80m10	2446.29	19.03	36.23	3.57	18.13	15.93
j100m6	2416.05	8.34	20.83	1.63	7.85	7.36
j100m7	2011.03	10.55	23.48	4.01	7.77	6.46
j100m8	2770.80	16.33	32.65	3.60	14.07	7.32
j100m9	2705.95	16.04	33.92	4.82	15.34	6.96
j100m10	3584.08	18.55	46.80	5.39	15.30	5.98
mean	1416.33	13.70	22.82	1.96	12.59	9.89



(a) Convergence curve for "j30m10"



(b) Convergence curve for "j50m6"



(c) Convergence curve for "j100m6"

Fig. 9. Comparisons of the convergence abilities.

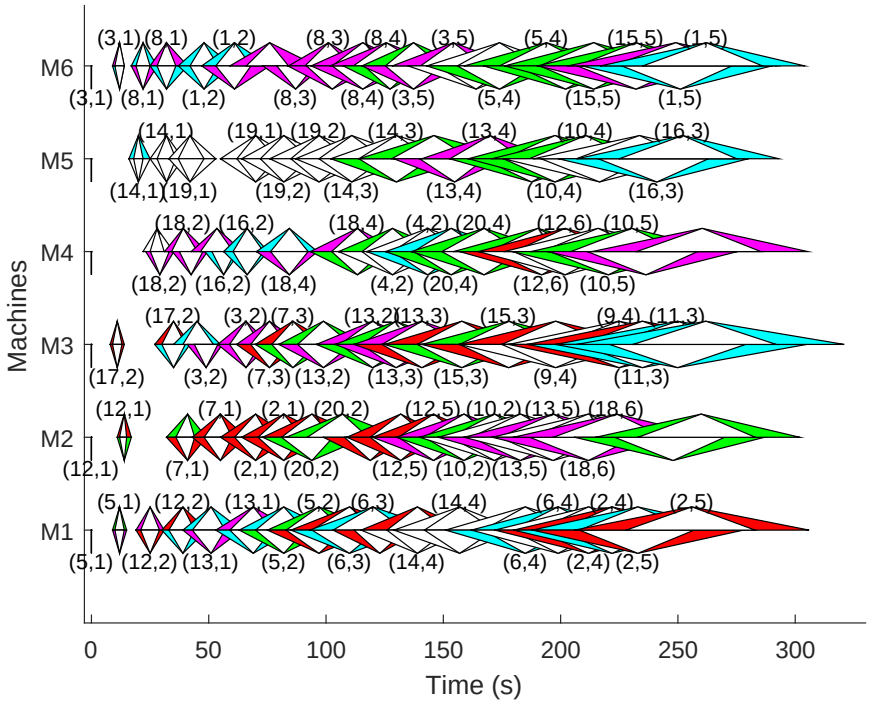


Fig. 10. Gantt chart for the best solution for "j20m6".

[2] V. Santucci, M. Baioletti, and A. Milani, "Algebraic differential evolution algorithm for the permutation flowshop scheduling problem with total flowtime criterion," IEEE Transactions on Evolutionary Computation, vol. 20, no. 5, pp. 682-694, 2015.

[3] R. Ruiz and J. A. Vázquez-Rodríguez, "The hybrid flow shop scheduling problem," European journal of operational research, vol. 205, no. 1, pp. 1-18, 2010.

[4] J. Li, Q. Pan, and K. Mao, "A hybrid fruit fly optimization algorithm for the realistic hybrid flowshop rescheduling problem in steelmaking systems," IEEE Transactions on Automation Science, vol. 13, no. 2, pp. 932-949, 2016.

[5] P. Van Laarhoven, E. Aarts, and J. Lenstra, "Job shop scheduling by simulated annealing," Operations research, vol. 40, no. 1, pp. 113-125, 1992.

[6] I. Kacem, S. Hammadi, and P. Borne, "Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems," IEEE Transactions on Systems, Man, Cybernetics, Part C, vol. 32, no. 1, pp. 1-13, 2002.

[7] R. Ruiz, Q. Pan, and B. Naderi, "Iterated Greedy methods for the distributed permutation flowshop scheduling problem." Omega, vol. 83, PP. 213-222, 2019.

[8] J. Li, M. Song, L. Wang, P. Duan, H. Sang, Y. Han, Q. Pan, "Hybrid Artificial Bee Colony Algorithm for a Parallel Batching Distributed Flow-Shop Problem With Deteriorating Jobs," IEEE transactions on cybernetics, 2019, doi: 10.1109/TCYB.2019.2943606.

[9] L. Zhou, L. Zhang, L. Ren, and J. Wang, "Real-time Scheduling of Cloud Manufacturing Services Based on Dynamic Data-Driven Simulation," IEEE Transactions on Industrial Informatics, vol. 15, no. 9, pp. 5042-5051, 2019.

[10] M. Jensen, "Generating robust and flexible job shop schedules using genetic algorithms," IEEE Transactions on Evolutionary Computation, vol. 7, no. 3, pp. 275-288, 2003.

[11] F. Pezzella, G. Morganti, and G. Ciaschetti, "A genetic algorithm for the flexible job-shop scheduling problem," Computers Operations Research, vol. 35, no. 10, pp. 3202-3212, 2008.

[12] J. Li, Q. Pan, and K. Gao, "Pareto-based discrete artificial bee colony algorithm for multi-objective flexible job shop scheduling problems," The International Journal of Advanced Manufacturing Technology, vol. 55, no. 9-12, pp. 1159-1169, 2011.

[13] M. Nouiri, A. Bekrar, A. Jemai, D. Trentesaux, A. Ammari, and S. Niar, "Two stage particle swarm optimization to solve the flexible job shop predictive scheduling problem considering possible machine breakdowns," Computers and Industrial Engineering, vol. 112, pp. 595-606, 2017.

[14] M. CruzChávez, M. MartínezRangel, and M. CruzRosales, "Accelerated simulated annealing algorithm applied to the flexible job shop scheduling problem," International Transactions in Operational Research, vol. 24, no. 5, pp. 1119-1137, 2017.

[15] 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 Trans Cybern, vol. 49, no. 5, pp. 1944-1955, May 2019.

[16] J. Wang, Y. Zhang, Y. Liu, and N. Wu, "Multiagent and Bargaining-Game-Based Real-Time Scheduling for Internet of Things-Enabled Flexible Job Shop," IEEE Internet of Things Journal, vol. 6, no. 2, pp. 2518-2531, 2019.

[17] D. Lei, M. Li, and L. Wang, "A Two-Phase Meta-Heuristic for Multiobjective Flexible Job Shop Scheduling Problem With Total Energy Consumption Threshold," IEEE Trans Cybern, vol. 49, no. 3, pp. 1097-1109, Mar 2019.

[18] G. Aqel, X. Li, and L. Gao, "A Modified Iterated Greedy Algorithm for Flexible Job Shop Scheduling Problem," Chinese Journal of Mechanical Engineering, vol. 32, no. 1, 2019, doi: 10.1186/s10033-019-0337-7.

[19] L. Shen, S. Dauzère-Pérès, and J. Neufeld, "Solving the flexible job shop scheduling problem with sequence-dependent setup times," European Journal of Operational Research, vol. 265, no. 2, pp. 503-516, 2018.

[20] I. Kacem, S. Hammadi, and P. Borne, "Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic," Mathematics computers in simulation, vol. 60, no. 3-5, pp. 245-276, 2002.

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

[22] J. Li and Q. Pan, "Chemical-reaction optimization for solving fuzzy job-shop scheduling problem with flexible maintenance activities," International Journal of Production Economics, vol. 145, no. 1, pp. 4-17, 2013.

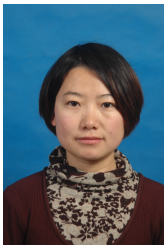
[23] L. Wang, G. Zhou, Y. Xu, and M. Liu, "A hybrid artificial bee colony algorithm for the fuzzy flexible job-shop scheduling problem,"

International Journal of Production Research, vol. 51, no. 12, pp. 3593-3608, 2013.
[24] K. Gao, P. Suganthan, Q. Pan, M. Tasgetiren, and A. Sadollah, "Artificial bee colony algorithm for scheduling and rescheduling fuzzy flexible job shop problem with new job insertion," Knowledge-Based Systems, vol. 109, pp. 1-16, 2016.
[25] Y. Zhong, F. Yang, and F. Liu, "Solving multi-objective fuzzy flexible job shop scheduling problem using MABC algorithm," Journal of Intelligent & Fuzzy Systems, vol. 36, no. 2, pp. 1455-1473, 2019.
[26] J. Lin, "A hybrid biogeography-based optimization for the fuzzy flexible job-shop scheduling problem," Knowledge-Based Systems, vol. 78, pp. 59-74, 2015.
[27] B. Liu, Y. Fan, and Y. Liu, "A fast estimation of distribution algorithm for dynamic fuzzy flexible job-shop scheduling problem," Computers and Industrial Engineering, vol. 87, pp. 193-201, 2015.
[28] J. Palacios, M. González, C. Vela, I. González-Rodríguez, and J. Puente, "Genetic tabu search for the fuzzy flexible job shop problem," Computers Operations Research, vol. 54, pp. 74-89, 2015.
[29] C. Wang, N. Tian, Z. Ji, and Y. Wang, "Multi-objective fuzzy flexible job shop scheduling using memetic algorithm," Journal of Statistical Computation and Simulation, vol. 87, no. 14, pp. 2828-2846, 2017.
[30] 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.
[31] 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.
[32] T. Jamrus, C. Chien, M. Gen, and K. Sethanan, "Hybrid Particle Swarm Optimization Combined With Genetic Operators for Flexible Job-Shop Scheduling Under Uncertain Processing Time for Semiconductor Manufacturing," IEEE Transactions on Semiconductor Manufacturing, vol. 31, no. 1, pp. 32-41, 2018.
[33] Z. Liu, S. Guo, and L. Wang, "Integrated green scheduling optimization of flexible job shop and crane transportation considering comprehensive energy consumption," Journal of Cleaner Production, vol. 211, pp. 765-786, 2019.
[34] M. Hemmati Far, H. Haleh, and A. Saghaei, "A fuzzy bi-objective flexible cell scheduling optimization model under green and energy-efficient strategy using Pareto-based algorithms: SATPSPGA, SANRGA, and NSGA-II," The International Journal of Advanced Manufacturing Technology, vol. 105, no. 9, pp. 3853-3879, 2019.
[35] R. Gil, Z. Johanyak, T. Kovacs, "Surrogate Model based Optimization of Traffic Lights Cycles and Green Period Ratios using Microscopic Simulation and Fuzzy Rule Interpolation. International Journal of Artificial Intelligence, vol. 16, pp. 20-40, 2018.
[36] R. Precup, E. Voisan, E. Petriu, M. Tomescu, R. David, A. Szedlak-Stinean, R. Roman, "Grey Wolf Optimizer-Based Approaches to Path Planning and Fuzzy Logic-based Tracking Control for Mobile Robots," International Journal of Computers Communications & Control, vol. 15, doi: 10.15837/ijccc.2020.3.3844.
[37] P. Melin, O. Castillo, "A review on type-2 fuzzy logic applications in clustering, classification and pattern recognition," Applied Soft Computing, vol. 21, pp. 568-577, 2014.
[38] D. Wu, J. Mendel, "Similarity Measures for Closed General Type-2 Fuzzy Sets: Overview, Comparisons, and a Geometric Approach," IEEE Transactions on Fuzzy Systems, vol. 27, pp. 515-526, 2019.
[39] J. Mendel, X. Liu, "Simplified interval type-2 fuzzy logic systems", IEEE Transactions on Fuzzy Systems, vol. 21, pp. 10561069, 2013.
[40] J. Mendel and H. Wu, "Type-2 fuzzistics for symmetric interval type-2 fuzzy sets: Part 1, forward problems," IEEE Transactions on Fuzzy Systems, vol. 14, no. 6, pp. 781-792, 2006.
[41] J. Mendel and H. Wu, "Type-2 Fuzzistics for Nonsymmetric Interval Type-2 Fuzzy Sets: Forward Problems," IEEE Transactions on Fuzzy Systems, vol. 15, no. 5, pp. 916-930, 2007.
[42] J. Figueroa-García and G. Hernández, "A method for solving linear programming models with interval type-2 fuzzy constraints," Pesquisa Operacional, vol. 34, no. 1, pp. 73-89, 2014.
[43] C. Li, J. Yi, and G. Zhang, "On the Monotonicity of Interval Type-2 Fuzzy Logic Systems," IEEE Transactions on Fuzzy Systems, vol. 22, no. 5, pp. 1197-1212, 2014.
[44] C. Li, G. Zhang, J. Yi, F. Shang, and J. Gao, "A fast learning method for data-driven design of interval type-2 fuzzy logic system," Journal of Intelligent & Fuzzy Systems, vol. 32, no. 3, pp. 2705-2715, 2017.
[45] P. Melin, F. Olivas, O. Castillo, F. Valdez, J. Soria, M. Valdez,"Optimal design of fuzzy classification systems using PSO with dynamic parameter adaptation through fuzzy logic," Expert Systems With Applications, vol. 40, pp. 3196-3206, 2013.
[46] O. Castillo, L. Amadorangulo, "A generalized type-2 fuzzy logic approach for dynamic parameter adaptation in bee colony optimization applied to fuzzy controller design," Information Sciences, vol., pp. 476-496, 2017.
[47] F. Olivas, F. Valdez, P. Melin, A. Sombra, O. Castillo, "Interval type-2 fuzzy logic for dynamic parameter adaptation in a modified gravitational search algorithm," Information Sciences, vol. 476, pp. 159-175, 2019.
[48] P. Melin, C. Gonzalez, J. Castro, O. Mendoza, O. Castillo, "Edge-Detection Method for Image Processing Based on Generalized Type-2 Fuzzy Logic," IEEE Transactions on Fuzzy Systems, vol. 22, pp. 1515-1525, 2014.
[49] A. Shukla, R. Nath, P. Muhuri, Q. 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, pp. 103257, doi: 10.1016/j.engappai.2019.103257, 2020.
[50] E. Ontiverosrobles, P. Melin, O. Castillo, "Comparative analysis of noise robustness of type 2 fuzzy logic controllers," Kybernetika, vol. 54, pp. 175-201, 2018.
[51] C. Li, J. Yi, H. Wang, G. Zhang, and J. Li, "Interval data driven construction of shadowed sets with application to linguistic word modelling," Information Sciences, vol. 507, pp. 503-521, 2020.
[52] J. Soto, P. Melin, O. Castillo, "A New Approach for Time Series Prediction Using Ensembles of IT2FNN Models with Optimization of Fuzzy Integrators," International Journal of Fuzzy Systems, vol. 20, pp. 701-728, 2018.
[53] S. Hofmeyr and S. Forrest, "Architecture for an artificial immune system," Evolutionary computation, vol. 8, no. 4, pp. 443-473, 2000.
[54] L. Jiao and L. Wang, "A novel genetic algorithm based on immunity," IEEE Transactions on Systems, Man,Cybernetics-part A: systems and humans, vol. 30, no. 5, pp. 552-561, 2000.
[55] S. Lin and K. Ying, "Minimizing makespan in a blocking flowshop using a revised artificial immune system algorithm," Omega, vol. 41, no. 2, pp. 383-389, 2013.
[56] Y. Xu, L. Wang, S. Wang, and M. Liu, "An effective hybrid immune algorithm for solving the distributed permutation flow-shop scheduling problem," Engineering Optimization, vol. 46, no. 9, pp. 1269-1283, 2013.
[57] O. Engin and A. Döyen, "A new approach to solve hybrid flow shop scheduling problems by artificial immune system," Future Generation Computer Systems, vol. 20, no. 6, pp. 1083-1095, 2004.
[58] G. Komaki, E. Teymourian, and V. Kayvanfar, "Minimising makespan in the two-stage assembly hybrid flow shop scheduling problem using artificial immune systems," International Journal of Production Research, vol. 54, no. 4, pp. 963-983, 2016.
[59] T. Chung and F. Chen, "A complete immunoglobulin-based artificial immune system algorithm for two-stage assembly flowshop scheduling problem with part splitting and distinct due windows," International Journal of Production Research, pp. 1-19, 2019, doi: 10.1080/00207543.2019.1577565.
[60] A. Bagheri, M. Zandieh, I. Mahdavi, and M. Yazdani, "An artificial immune algorithm for the flexible job-shop scheduling problem," Future Generation Computer Systems, vol. 26, no. 4, pp. 533-541, 2010.
[61] X. Wang, L. Gao, C. Zhang, and X. Shao, "A multi-objective genetic algorithm based on immune and entropy principle for flexible job-shop scheduling problem," The International Journal of Advanced Manufacturing Technology, vol. 51, no. 5-8, pp. 757-767, 2010.

Jun-qing Li (M'08) received the Master degree of computer science and technology in 2004 from Shandong Economic University, Shandong, China, and Ph.D. degree in 2016 from Northeastern University, Shenyang, China. Since 2004, he was with School of Computer, Liaocheng University. Since 2017, he has been with School of Information Science and Engineering, Shandong Normal University, where he became a Professor in 2017. His current research interests include intelligent optimization, fuzzy processing technology, and scheduling. He has authored more than 50 refereed papers.

Zheng-min Liu received the Ph.D. degree in management science and engineering, from Shandong University of Finance and Economics, Shandong, China. He is currently a Professor with the School of Management Science and Engineering, Shandong University of Finance and Economics, Shandong, China. He has authored or coauthored more than 20 publications. His research interests include information fusion, fuzzy decision making and their applications.

Chengdong Li (M'12) received the B.S. and M.S. degrees from Shandong University, Jinan, China, in 2004 and 2007, respectively, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2010. He is currently a Professor with the School of Information and Electrical Engineering, Shandong Jianzhu University. His major research interests include fuzzy logic theory and applications, and other computational intelligence methods. He has authored more than 80 papers in international journals and conferences. He has been the Program Committee Member of several international conferences and the Reviewer for several international conferences and journals.



Zhi-xin Zheng Zhi-xin Zheng received the bachelors degree in management from Henan Polytechnic University, Jiaozuo, China, in 2005, and the master's degree in management from the Shandong University of Finance and Economics, Jinan, China, in 2008. She is currently a lecturer with College of Computer Science, Liaocheng University. Her major research interests include intelligent optimization algorithm theory and applications in energy optimization problems. She has authored more than 10 papers in international journals and conferences.