Computers & Industrial Engineering 168 (2022) 108099

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Self-adaptive multi-objective evolutionary algorithm for flexible job shop scheduling with fuzzy processing time

Rui Li, Wenyin Gong *, Chao Lu

School of Computer Science, China University of Geosciences, Wuhan 430074, China

ARTICLE INFO

Keywords:
Fuzzy flexible job shop scheduling
Multi-objective optimization
Parameter self-adaptation
Triangular fuzzy number

ABSTRACT

With increasing environmental awareness and energy requirement, sustainable manufacturing has attracted growing attention. Meanwhile, there is a high level of uncertainty in practical processing procedure, particularly in flexible manufacturing systems. This study addresses the multi-objective flexible job shop scheduling problem with fuzzy processing time (MOFFJSP) to minimize the makespan and the total workload simultaneously. A mixed integer liner programming model is presented and a hybrid self-adaptive multi-objective evolutionary algorithm based on decomposition (HPEA) is proposed to handle this problem. HPEA has the following features: (i) two problem-specific initial rules considering triangular fuzzy number are presented for hybrid initialization to generate diverse solutions; (ii) five problem-specific local search methods are incorporated to enhance the exploitation; (iii) an effective solution selection method based on Tchebycheff decomposition strategy is utilized to balance the convergence and diversity; and (iv) a parameter selection strategy is proposed to improve the quality of non-dominated solutions. To verify the effectiveness of HPEA, it is compared against other well-known multi-objective optimization algorithms. The results demonstrate that HPEA outperforms these five state-of-the-art multi-objective optimization algorithms in solving MOFFJSP.

1. Introduction

With the development of economic globalization, traditional manufacturing is quite difficult to satisfy the flexible requirement of the market. Whereas, because of lower management risk and higher production profit, flexible manufacturing has become a common modern production (Lu, Gao, Yi, & Li, 2020). In flexible manufacturing, all tasks are assigned to a set of machines to optimize one or more objectives. Meanwhile, on account of various reasons, fuzziness and uncertainty will inevitably appear when processing jobs. So it is necessary to fuzz the processing time. And fuzzy manufacturing has attracted researchers' widespread concern because of its complexity. Consequently, fuzzy scheduling of flexible shop problems has been studied in recent years. In most existing studies, only one objective (i.e., makespan) on fuzzy shop scheduling problems is considered. Nevertheless, in the practical manufacturing environment, there are always more than oneobjectives that need to be optimized. Therefore, the research along multi-objective fuzzy job shop scheduling problem seems more significant for some manufacturing enterprises.

Fuzzy flexible job shop scheduling problem (FFJSP) (Lei, 2010) is a combination of flexible job shop scheduling problem (FJSP) (Brucker & Schlie, 1990) and fuzzy job shop scheduling problem (fJSP) (Abdullah & Abdolrazzagh-Nezhad, 2014). It is very complex and hard to solve and has been proved as an NP-hard problem (Pavlov, Misura, Melnikov, & Mukha, 2019). Thus, many intelligent optimization algorithms (Gong, Liao, Mi, Wang, & Guo, 2021; Dai, Gong, & Gu, 2021) have been proposed to address such scheduling problems, such as genetic algorithm (Sakawa & Mori, 1999), differential evolution (Gao, Wang, & Pedrycz, 2020), ant colony optimization (Jia, Yan, Leung, Li, & Chen, 2019), teaching learning-based optimization (Xu, Wang, Wang, & Liu, 2015), and harmony search (Gao, Suganthan, Pan, & Tasgetiren, 2015b).

This work investigates on multi-objective fuzzy flexible job shop scheduling problem (MOFFJSP). To the best of our knowledge, MOFFJSP has not yet been studied so far. Most previous studies focus on optimizing fuzzy makespan, including decomposition-integration genetic algorithm (DIGA) (Lei, 2010) for FFJSP, coevolutionary genetic (CGA) (Lei, 2012) for FFJSP, hybrid biogeography-based optimization (HBBO) (Lin, 2015) for FFJSP, backtracking search based hyper-heuristic (BS-HH) (Lin, 2019) for FFJSP, hybrid cooperative coevolution algorithm (hCEA)(Sun, Lin, Gen, & Li, 2019) for FFJSP and

* Corresponding author.
E-mail addresses: wygong@cug.edu.cn (W. Gong), luchao@cug.edu.cn (C. Lu).

https://doi.org/10.1016/j.cie.2022.108099
Received 27 November 2021; Received in revised form 18 February 2022; Accepted 10 March 2022
Available online 18 March 2022

weighted distance-based approximation (WBDA) (Dorfeshan, Tavakkoli-Moghaddam, Mousavi, & Vahedi-Nouri, 2020) for FFJSP. As for MOFFJSP, Palacios (Gonzalez-Rodriguez, Vela, & Puente, 2017) applied NSGA-II and dominance-based tabu search (DBTS) to minimize fuzzy makespan and maximize robustness. An improved NSGA-II with critical path and hamming distance for diversity was proposed by Wang (Chun, Na, Zhicheng, & Yan, 2017) for MOFFJSP. Wang (Wang, Tian, Ji, & Wang, 2017) combined NSGA-II and MA to solve robust MOFFJSP efficiently. Yu (Yuguang, Fan, & Feng, 2019) applied a multi-objective artificial bee colony algorithm (MoABC) with basic VNS for MOFFJSP. An NSGA-II algorithm hybridizing local simulated annealing operators (NIISA) was proposed by Wang (Wang, Xie, Xia, & Zhang, 2019), which is better than algorithms. GonzÍçlez (Gonzalez-Rodriguez, Puente, Palacios, & Vela, 2020) developed a novel NSGA-II and use heuristics to reduce energy consumption named (MO-HREC) for MOFFJSP with energy constrain. Li (Li, Liu, Li, & Zheng, 2020) proposed a type-2 fuzzy set and used an improved artificial immune system algorithm (IAIS) to solve MOFFJSP with low energy consumption which is competitive to other algorithms. Inspired by the above-mentioned algorithms, there are some problems that need to be solved. For MOFFJSP, meta-heuristics can get good convergence by excessively adopting local search, but it will sacrifice the diversity of the population. NSGA-II is the most common algorithm in recent studies on MOFFJSP. But the crowding distance strategy seems not a good diversity strategy for discrete problems. After executing a local search, the non-dominated solution may gather in one point, which may get stuck at local optima. So a good diversity strategy should be designed to. Many classical initial methods have been proposed and how to combine their advantages to get a high-quality population is worth studying. Local search is a critical step in discrete scheduling problems and how to organize several problem-specific Local search strategies is worth being considered. According to our experimental results, we found when the algorithm selects different parameters, the best parameter for each FFJSP instance was different. Referring to no free lunch theorems (Wolpert & Macready, 1997), no parameter works best on all instances. Thus, it is essential to make the algorithm automatically choose parameters according to their performances.

Motivated by the above problems, this paper proposed a hybrid MOEA/D with parameter adaption strategy (HPEA) to solve multi-objective FFJSP (MOFFJSP) when minimizing fuzzy makespan and fuzzy total machine workload simultaneously. In HPEA, weight vectors are used to make solutions spread widely and ensure diversity among the candidate solutions. Tchebycheff function is applied to guide the solution converge along weight vectors to balance the convergence and diversity. Then a discrete crossover method, which has a larger step and can exchange the gene more sufficiently, is applied to generate a new solution. Next, an initial method combining three initial rules (MIX3) is designed to get a population with high quality and diversity. Moreover, a variable neighborhood search (VNS) adopting five local search methods is performed to accelerate its convergence. Finally, a parameter adaption strategy (PAS) is developed to make the algorithm automatically select parameters according to their performance. To verify the performance of HPEA, three benchmarks containing 23 FFJSP instances are chosen. In addition, extensive experiments have been carried out in this paper, including parameter calibration in HPEA, the efficiency of each component of HPEA, comparison with some advanced algorithms, the diversity of non-dominated solution set in contrast to other algorithms, and the convergence speed compared with other algorithms.

The main contributions of this paper go in four directions.

(1) An initial strategy is designed to provide a population with high quality and diversity, which combines the advantage of three strategies.
(2) A variable neighborhood search is designed, which adopted five local search methods. It provides guidance to organize local searches to get good results.s
(3) A parameter adaption strategy is dseveloped to ensursse that proposed algorithm dynamically chooses parameters according to their performances.
(4) The performance of HPEA is executed on 23 FFJSP instances with different features. Experimental results show that HPEA is superior to state-of-art algorithms under the condition of faster convergence.

The rest of this paper is organized as follows. Section 2 describes the concept of fuzzy set, fuzzy operators, and MOEA/D. In Section 3, we introduce the modeling of multi-objective FFJSP (MOFFJSP). Our approach HPEA is reported in Section 4 in detail, including initial strategy MIX3, VNS, discrete crossover method, and PSA. Numerical test experiments on HPEA are shown in Section 5 and the conclusions are summarized in Section 6 and some topics of future research are provided.

2. Related work

2.1. Previous works

Table 1 gives the review of literature about MOFFJSP. (Sakawa & Kubota, 2000) first applied GA to solve it and GA is suitable for it. (Chun et al., 2017) used insert decoding method and critical path to efficiently reduce the makespan. (Wang et al., 2017) embed VNS into NSGA-II, which improved the convergence. (Palacios et al., 2017) designed a dominance-based tabu search method, which selected non-improve neighbors to escape from local optima. (Saracoglu & Suer, 2018) used three-phase to simulate manufacturing progress. (Wang et al., 2019) developed simulated annealing as local searching for NSGA-II, which improved the convergence. (Yuguang et al., 2019) proposed multi-objective artificial bee colony algorithm (MoABC) and embedded VNS adopting three local search methods. VNS obtained good searching

Table 1
Literature review.

Researchers	fuzzy number	objective	algorithm
(Sakawa & Kubota, 2000)	TFN	C_{max} and duedates	GA
(Chun et al., 2017)	TFN	C_{max} and *TWL*	NSGA-II
(Wang et al., 2017)	TFN	C_{max} and *AI*	NSGA-II
(Palacios et al., 2017)	TFN	C_{max} and robustness	MoEA
(Saracoglu & Suer, 2018)	duedate	*TFT* and C_{max}	GA
(Wang et al., 2019)	TFN	C_{max} and robustness	NSGA-II
(Yuguang et al., 2019)	TFN	C_{max}, *AI* and *TWL*	ABC
(Gonzalez-Rodriguez et al., 2020)	TFN	*TWT* and *TEC*	NSGA-II
(Li et al., 2020)	IT2FS	C_{max} and *TEC*	AIS
(Pan et al., 2021)	TFN	C_{max} and *TEC*	bi-population EA
Our approach	TFN	C_{max} and *TWL*	MOEA/D

C_{max}: makespan; *TWL*: total machine workload; *TFT*: total flow time; *AI*: agreement index; *TEC*: total energy consumption; *TWT*:total weighted tardiness.

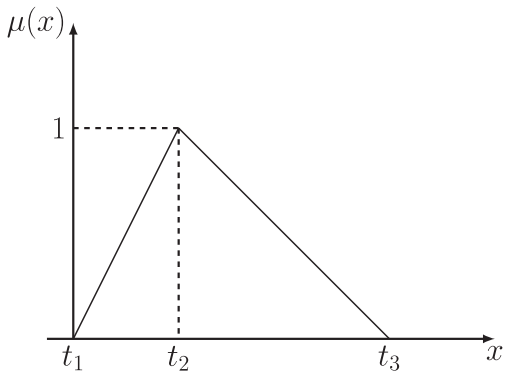


Fig. 1. Triangular membership function.

performance. A different local right shift method is introduced by (Gonzalez-Rodriguez et al., 2020) to optimize objective targeted. (Li et al., 2020) designed an initial strategy combining four rules and randomly choose mutation from six local search methods. (Pan, Lei, & Wang, 2021) proposed a bi-population co-evolutionary algorithm. However, parameter has a great impact on the performance of algorithm, but the adaptive parameter adjustment has not been considered for MOFFJSP, according to previous works. Thus, it is worth for studying a adaptive parameter selection model to increase the convergence and diversity.

2.2. Fuzzy set

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

$$\widetilde{F} = \left\{x, \mu_{\widetilde{F}}(x) \middle| \forall x \in X\right\}, 0 \leqslant \mu_{\widetilde{F}}\left(x\right) \leqslant 1 \tag{1}$$

The classical set is definite. When $\mu_{\widetilde{F}}(x) = 1$, the fuzzy set transfer to classical set.

$$F = \left\{x, \mu_{\widetilde{F}}(x) = 1 \middle| \forall x \in X\right\} \tag{2}$$

Triangle fuzzy number (TFN) is the widely used membership function in scheduling. As shown in Fig. 1, the membership function is similar to the triangle. t_1 is the earliest processing time, t_2 is the most possible processing time and t_3 is the latest processing time. A triple (t_1, t_2, t_3) usually represents a TFN. The definition of TFN membership are given as follows:

$$\mu_{\widetilde{F}}\left(\begin{array}{c} \\ x \\ \\ \end{array}\right) = \begin{cases} 0, & x \leqslant t_1, \\ \dfrac{x - t_1}{t_2 - t_1}, & t_1 < x \leqslant t_2, \\ \dfrac{t_3 - x}{t_3 - t_2}, & t_2 < x < t_3, \\ 0, & x \geqslant t_3. \end{cases} \tag{3}$$

2.3. Fuzzy Operators

(Sakawa & Mori, 1999) defines three operators in a fuzzy set. Addition operator, Ranking operator, and Max operator. Give two TFNs $\widetilde{s} = (s_1, s_2, s_3)$ and $\widetilde{t} = (t_1, t_2, t_3)$, the three operators are computed as follows.(1) Addition operator.

$$\widetilde{s} + \widetilde{t} = \left(s_1 + t_1, s_2 + t_2, s_3 + t_3\right). \tag{4}$$

(2) Ranking operator.

$$f_1\left(\widetilde{x}\right) = \frac{x_1 + 2x_2 + x_3}{4}. \tag{5}$$

Condition 1. if $f_1(\widetilde{s}) > f_1(\widetilde{t}), \widetilde{s} > \widetilde{t}$, otherwise $\widetilde{s} < \widetilde{t}$.

Condition 2. $f_2(\widetilde{x}) = x_2$, when $f_1(\widetilde{s}) = f_1(\widetilde{t})$, if $f_2(\widetilde{s}) > f_2(\widetilde{t})$, then $\widetilde{s} > \widetilde{t}$; otherwise $\widetilde{s} < \widetilde{t}$.

Condition 3. $f_3(\widetilde{x}) = s_3 - s_1$, when $f_2(\widetilde{s}) = f_2(\widetilde{t})$, if $f_3(\widetilde{s}) > f_3(\widetilde{t})$, then $\widetilde{s} > \widetilde{t}$; otherwise $\widetilde{s} < \widetilde{t}$.(3) Max operator. if $\widetilde{s} > \widetilde{t}$, then $\widetilde{s} \vee \widetilde{t} = \widetilde{s}$; otherwise $\widetilde{s} \vee \widetilde{t} = \widetilde{t}$.

3. Problem statement and mathematical modeling

3.1. Problem statement

A flexible job shop scheduling problem with fuzzy processing time (FFJSP) from a real-world manufacturing process can be described as follow. There is a set of n jobs, $I = \{1, 2, \ldots, i, \ldots, n\}$ and a set of m machines, $M = \{1, 2, \ldots, k, \ldots, m\}$. Each job I_i has a operation set J sizing n_i, $J = \{O_{i,1}, O_{i,2}, \ldots, O_{i,j}, \ldots, O_{i,n_i}\}$. Each operation can be processed on part of machines or all machines. And all the processing time of operation $O_{i,j}$ on machine M_k is a TFN $\widetilde{P}^{O}_{i,j,k} = (p_1, p_2, p_3)$. FFJSP includes two subproblems, machine assignment and operation sequencing. The former is that each operation *selects*‘ a machine from a candidate set. The latter is to schedule all operations on all machines to get satisfactory schedules. The assumptions of FFJSP is given as below:

- All jobs, and machines are available at time zero.
- For each machine, it can at most process one operation at the same time. And interruption is not considered for each operation.
- All the processing date such as processing time and energy consumption is TFN.
- Each operation can only be assigned to one machine.
- Transportation time, setup time, and their energy consumption are not considered.

3.2. MILP model for MOFFJSP

Before modeling this MOFFJSP, The notations used throughout the study are as follows:i,i': indices for jobs.

j,j': indices for operations of jobs.

k,k': indices for machines.

t: index for position.

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 $I = \{1,2,\ldots,n\}$.

M: set of machines and $M = \{1,2,\ldots,m\}$.

J_i: set for the operations of jobs I_i and $J_i = \{1,2,\ldots,n_i\}$.

P_k: set of positions of machine M_k in and $P_k = \{1,2,\ldots,p_k\}$.

$P_k^{'}$: set of top p_k-1 positions of machine M_k in and $P_k^{'} = \{1,2,\ldots,p_k-1\}$.

$O_{i,j}$: The j_{th} operation of job I_i.

$\widetilde{P}^{O}_{i,j,k}$: The processing time for operation $O_{i,j}$ job I_i processed by machine M_k, which is a TFN: $\widetilde{P}_{i,j,k} = (p_1,p_2,p_3)$.

$\widetilde{S}_{i,j}$: the start time of operation $O_{i,j}$.

$\widetilde{C}_{i,j}$: the completion time of operation $O_{i,j}$.

$\widetilde{B}_{k,t}$: the start time of machine M_k on position t.

TWL total machine workload.

C_{max}: the makespan of a schedule.

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 by on machine M_k in position P_{kt}, the value is set to 1; otherwise is set to 0.

$\mathbf{Y}_{k,t}$: If machine M_k is processing a operation in position t, the value is set to 1; otherwise is set to 0.

The objectives of MOFFJSP include makespan and total workload (TWL), 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, makespan C_{max} objective of MOFFJSP can be defined below:

$$minF_1 = C_{max} = \max\left\{\widetilde{C}_{i,n_i}\right\}, \forall i \in I. \tag{6}$$

(2) *TWL criterion*: TWL during the manufacturing process can be seen as one key indicator. TWL criterion reflects the degree of machine wear. In this study, the second objective is to minimize TWL through the following formula:

$$minF_2 = \sum_{k\in M}\sum_{t\in P_k}\sum_{i\in I}\sum_{j\in J_i}\widetilde{P}^{O}_{i,j,k}\cdot\mathbf{X}_{i,j,k,t}. \tag{7}$$

In summary, a MILP model for MOFFJSP is as follows:

$$Objectives: \begin{cases} minF_1 = C_{max} \\ minF_2 = TWL \end{cases} \tag{8}$$

Subject to:

$$\sum_{k\in K}\sum_{t\in P_k}\mathbf{X}_{i,j,k,t} = 1, \forall i \in I, j \in J_i \tag{9}$$

$$\sum_{k\in K}\sum_{t\in P_k}\mathbf{Y}_{k,t} \leqslant 1, \forall i \in I, j \in J_i \tag{10}$$

$$\widetilde{S}_{i,n_i} + \sum_{k\in K}\sum_{t\in P_k}\widetilde{P}^{O}_{i,n_i,k}\cdot\mathbf{X}_{i,n_i,k,t} \leqslant C_{max}, \forall i \in I \tag{11}$$

$$\widetilde{S}_{i,j} + \sum_{k\in K}\sum_{t\in P_k}\widetilde{P}^{O}_{i,j,k}\cdot\mathbf{X}_{i,j,k,t} \leqslant \widetilde{S}_{i,j+1}, \forall i \in I, j \in J_i - 1 \tag{12}$$

$$\sum_{i\in I}\sum_{j\in n_{max}}\mathbf{X}_{i,j,k,t} \leqslant 1, \forall k \in K, t \in P_k \tag{13}$$

$$\sum_{i\in I}\sum_{j\in n_{max}}\mathbf{X}_{i,j,k,t} \geqslant \sum_{i\in I}\sum_{j\in n_{max}}\mathbf{X}_{i,j,k,t+1}, \forall k \in K, t \in P_k^{'} \tag{14}$$

$$\widetilde{B}_{k,t+1} - \widetilde{B}_{k,t} \geqslant \sum_{i\in I}\sum_{j\in n_{max}}\mathbf{X}_{i,j,k,t+1}\cdot\widetilde{P}^{O}_{i,j,k}, \forall k \in K, t \in P_k - 1 \tag{15}$$

$$\widetilde{B}_{k,t} \geqslant \widetilde{S}_{i,j} - M\cdot\left(1 - \mathbf{X}_{i,j,k,t+1}\right), \forall i \in I, j \in J_i, k \in K, t \in P_k \tag{16}$$

$$\widetilde{B}_{k,t} \leqslant \widetilde{S}_{i,j} - M\cdot\left(1 - \mathbf{X}_{i,j,k,t+1}\right), \forall i \in I, j \in J_i, k \in K, t \in P_k \tag{17}$$

$$0 \leqslant \widetilde{S}_{i,j} \leqslant M, i \in I, j \in J_i \tag{18}$$

$$\widetilde{B}_{k,t} \geqslant 0, k \in K, t \in P_k \tag{19}$$

Eq. (8) are two objectives including makespan and TEC. Eq. 9,10 guarantees that each job must be processed on one machine at a time. Eq. (11) defines the makespan. Eq. (12)guarantees that the proceeding must have been processed before current operation starting. Eq. (13) indicates that at most one operation can be assigned to a position of a machine. Eq. (14) forces that an operation must be assigned to the preceding positions only when they are occupied by other operations. Eq. (15) assures relationship between two adjacent positions of a machine. Eqs. (16) and (17) define the relation between machine start time and operation start time. Eq. (18) and (19) are value range limitations.

4. Proposed algorithm

4.1. Motivation

In the previous works for MOFFJSP, most of them adopt local search to improve convergence. But continuously executing local search will lead the population to converge into a few points. The solution from the non-dominated solution set will reduce so does the diversity. But crowding distance strategy could not solve this problem efficiently. Because fast non-dominated sorting focus convergence first. MOEA/D applied weight vector and Tchybecheff function and it focuses on both convergence and diversity. The solution always distributes around weight vectors. We also design an initial strategy and VNS to improve the convergence further. However, when MOEA/D set the different number of neighborhood, the best parameter for each FFJSP instance is different. So we applied a history memory to preserve the count of success and failure times for each parameter. According to their performance and transfer memory into a probability of being selected. So that the MOEA/D can choose the best parameter automatically.

Algorithm 1. The Framework of HPEA.

Input: ps, R, $T = \{T_1, ..., Tn\}$. which means population size, mutation rate, and candidate parameter of weight vectors' neighborhood.

Output: the best solution found so far when it stops

Initialize the population $\mathbb{P}$, sizing Np (c.f. Section 4.4);
Calculate fitness function of $\mathbb{P}$ (c.f. Section 3.2);
Generate a uniform spread of Np weights vectors: $\lambda^1, ..., \lambda^{Np}$ and calculate T closest neighborhood $B(i) = \{i_1, ..., i_T n\}, i = 1, ..., Tn$;
Choose the parameter T_i from vector T for each individual in $\mathbb{P}$ (c.f. Section 4.8;
Population evolution. (c.f. Section 4.5);
Update neighborhood solutions space sizing T_i (c.f. Section 4.6);
Perform VNS for each individual. (c.f. Section 4.7);
Adjust the chosen probability of each parameter in T (c.f. Section 4.8);
if *the stopping criterion is satisfied* **then**
    terminate the algorithm.
    else
        go to step 4

4.2. Framework of HPEA

In this section, we describe the detailed components of the proposed HPEA. We present step-by-step descriptions of the solution representation and decoding mechanism, the initialization strategy, crossover and mutation method, variable neighborhood search and parameter adaption strategy steps. The framework of the HPEA is described in Algorithm 1.

4.3. Encoding and decoding

In this paper, two one-dimensional vectors are used to represent the solution. The operation sequence is used to indicate the processing sequence for all operations. And the 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 displays an encoded solution. The solution representation contains two vectors. The operations sequence is $O_{3,1}, O_{2,1}, O_{1,1}, O_{2,2}, O_{1,2}, O_{3,2}, O_{1,3}, O_{3,3}, O_{2,3}$. The machine selection is $M_1, M_3, M_2, M_2, M_2, M_3, M_2, M_2, M_1$. It is a one-to-one correspondence. For example, $O_{3,1}$ selected M_1 and $O_{2,1}$ selected M_3.

The decoding of a solution is to assign the appropriate processing time for each operation on its selected machine according to the operation sequence. When a solution is decoded, the first vector in Fig. 3 is converted into a sequence of operations at frits. Then each operation is assigned to a selected machine from the second vector in Fig. 2. Finally, the fuzzy processing times are assigned to the operation. In this paper, each solution is decoded into a fuzzy schedule. That is the processing time is a TFN.

4.4. Initialization strategy

In this section, three classical initialization strategies and a combination initial strategy are described.

To obtain an initial population with high quality and diversity, the initial strategy should combine different strategies' advantages. Many classical strategies have been proposed, such as Random rule (Gao et al., 2015b), local minimum processing time (LS) rule (Shaheed, Shukor, & Abdullah, 2018), global minimum workload (GW) rule (Li et al., 2020). The descriptions of three strategies are given as follows:

Random: This rule is simple and ensures the initial population has high diversity. (1) repeat each job J_i for Θ_i times to generate a scheduling vector. (2) randomly rearrange the sequence for all operations in the scheduling vector. (3) randomly choose a machine from the operation's candidate set for each operation and generate a routing vector.

LS: This rule aims to reduce the fuzzy makespan (maximum completion time). (1) randomly generate a scheduling vector same as the Random rule. (2) for each operation, choose the minimum processing time machine from the candidate set to generate the routing vector.

GW: This rule focuses on lower the total machine workload. (1) put $O_{1,1}, O_{2,1}, \ldots, O_{n,1}$ into scheduling vector and rearrange the sequence. (2) rearrange the sequence of the rest operation and connect after the former sequence. (3) for each operation, choose an available machine with the minimum workload. If more than one machine has the same workload, then select the machine with minimum processing time for the operation.

Step (1) and (2) in GW are to prevent that one job from continuously appears in the head of the scheduling vector. If the head of the scheduling vector is $O_{1,1}, O_{1,2}, \ldots, O_{1,n_1}$, then according to GW each operation of job 1 will select the minimum workload machine. But at the beginning, each machine's workload is zero, so job J_1 will select the machine in order. That will result in other jobs must wait for O_{1,n_1} to be finished. On account of constraint (11), the following operation must wait for the previous operation to be finished. So O_{1,n_1} will waste too much time.

The advantages of the three strategies are combined to obtain a population of high quality. A method called MIX3 is developed and the description is given in Algorithm 2.

Algorithm 2. MIX3 rule.

Input: Population size, Np.

Output: initialization population

Perform GW to generate an offspring $\mathbb{P}_1$, size $\lfloor Np/3 \rfloor$;
Execute LS to generate an offspring $\mathbb{P}_2$, size $\lfloor Np/3 \rfloor$;
Use Random to generate an offspring $\mathbb{P}_3$, size $\lfloor Np/3 \rfloor$;
Combine $\mathbb{P}_1$, $\mathbb{P}_2$ and $\mathbb{P}_3$ into $Parent$ size $3 \cdot \lfloor Np/3 \rfloor$;
if *size of* $Parent = Np$ **then**
    terminate the algorithm;
    else
        supplement the rest of the solution with Random rule;

4.5. Crossover and mutation

To get a large searching step, a discrete crossover and mutation methods POX (Gao et al., 2015a) for MOFFJSP is applied. Fig. 3 gives two examples. The description is given as follows:

Operation sequence crossover: (1) Randomly divide job set into two subset J_1 and J_2. (2) Select two solutions S_1 and S_2. For each job belonging to J_1, copy their operations into $NewS_1$. And for each job belonging to J_2, copy their operations into $NewS_2$. (3) There exist too much space that *is* not filled with operation in $NewS_1$ and $NewS_2$. From the part of S_2, copy the operation which do 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. And do the similar thing to $NewS_2$. The procedure is illustrated in Fig. 3.

Machine selection crossover: (1) Randomly generate a 0–1 vector which length equals the total number of operations. (2) Select two solution 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. The procedure is illustrated in Fig. 3.

Operation sequence mutation: 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.

4.6. Update strategy

Diversity is an important metric of MOEAs. Uniformly distributed Pareto solutions can provide more and better decision selection for industry. Reference vector is a classical method in MoP. Depending on it and Tchebycheff aggregate function (TAF) (Zhang & Hui, 2007), solutions can uniformly distribute around reference vector, which leads to good diversity. TAF merges all objective functions and replace the strange domination sequence by a wake single-value function and it can efficiently balance the convergence and diversity. The equation of Tchebycheff aggregate function is given as follows:

$$g^{te}\left(x\middle|\lambda^j, Z^*\right) = \max_{1\leqslant i\leqslant m}\left\{\lambda_i^j\middle|f_i\left(x\right) - Z_i^*\middle|\right\} \tag{20}$$

Fig. 4 gives an example of calculating TAF. P_1 bias two objective function values to reference point Z^* and multiply by reference vector λ_i. Every solution in population $\mathscr{P}$ has a reference vector λ_i. Choose the max value of all dimension as the Tchebycheff function value, which is regarded evolutionary direction. The procedure of this update strategy is stated below:

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

Algorithm 3. Variable neighborhood search.

Input: Current solution $\mathbb{P}(i)$, reference point Z^*, weight vector λ_i.

Output: New solution $\mathbb{P}(i)'$ and successful flag F.

Set $F = 0$;
for $k = 1; k < 5; k++$ **do**
    $NewS \leftarrow LS_k(\mathbb{P}(i))$;
    if $g^{te}(NewS|\lambda^i, Z^*) < g^{te}(\mathbb{P}(i)|\lambda^i, Z^*)$ **then**
        $\mathbb{P}(i) = NewS$ and $F = 1$;
    if $F == 1$ **then**
        break;
Calculate the Tchebycheff function of $NewS \in S$;

4.7. Variable neighborhood search

In this section, a variable neighborhood search fixed with five local search methods is proposed. At the beginning of the iteration, the population spreads out to find various solutions with high diversity because of weight vectors. Then VNS is applied to search solution around the current solution to jump out of local optima and improve the convergence. The description of five local search methods is given as follows:

LS1: Find the last finished operation O_{i,Θ_i}, move O_{i,Θ_i} to another machine M' with minimum processing time.
LS2: Randomly select an operation $O_{i,j}$ and move it to another machine with minimum processing time.
LS3: Find the maximum workload machine M. Randomly select a operation which processed by M and move it to another machine M'.
LS4: Randomly choose two positions on the scheduling vector and exchange the value.
LS5: Randomly choose two positions on the scheduling vector and insert the latter one in front of the former one.

We design a VNS method fixed with five local search methods mentioned above. The description is given in Algorithm 3.

We execute each local search for one time to generate a new solutions. Then adjust whether the new solution can update the current solution $\mathbb{P}(i)$. This lets the solution fully exploit in the surrounding, which can help jump out of local optima and improve the convergence.

Fig. 5 gives an example of VNS to illustrate how VNS works. The black dotted arc is the assumed Pareto Front. And P_1, P_2, P_3, P_4, P_5 is the non-dominated solution in Pareto Front. The red point L_1, L_2 are local optima. The black point S_1, S_2, S_3, S_4 are solutions in population. S_1, S_2 fall into local optima L_1 and VNS starts to work. L_1 uses different neighborhood actions to generate some candidate solution. Compare to those candidate solutions, solution might jump out of local optima. The red circle simulates VNS is exploiting around L_1. S_3 dominates L_1 so VNS succeeds. Sometime, local optima can straightly converge such as L_2 to P_5. But S_4's Tchebycheff function might be better than L_2, which is also a successful VNS, because S_4 might converge into P_4.

Algorithm 4. Parameter adaption strategy.

Input: $T = \{T_1, T_2, ..., T_n\}$ parameter candidate set

Output: $P = \{P_1, P_2, ..., P_n\}$ probability of each parameter

Initial all $P(i) = \frac{1}{n}, i = 1, ..., n$;
Execute Roulette Algorithm to assign a parameter $T_i \in T$ to $\mathbb{P}(i) \in \mathbb{P}$;
Count the ns_i and nf_i $i = 1, ..., n$. Save this record into the tail of success memory and failure memory;
if $length(SM) > LP$ **then**
    Delete the first record in SM and FM;
    for $i = 1$ *to* n **do**
        $SR(i) = \sum_{j=1}^{LP} ns_{i,j}$;
        $FR(i) = \sum_{j=1}^{LP} nf_{i,j}$;
        $P(i) = \frac{SR(i)}{SR(i)+FR(i)}$;
    for $i = 1$ *to* n **do**
        $P(i) = \frac{P(i)}{\sum_{i=1}^{n} P(i)}$

4.8. Parameter adaption strategy

The parameter adaption strategy will be reported in this section. The performance of HPEA is impacted by parameter setting. So inspired by (Qin, Huang, & Suganthan, 2009), we design a parameter adaption strategy to let HPEA dynamically choose the best parameter. The procedure is stated as below:

Parameter adaption strategy: (1) Initial a parameter candidate set with n element $T = \{T_1, T_2, ..., T_n\}$. Set a uniform probability for each parameter. (2) Execute roulette algorithm (Qin et al., 2009) to assign a parameter $T_i \in T$ to each individual $\mathbb{P}(i)$ in population $\mathbb{P}$. (3) Count the number of updating old solution successfully and unsuccessfully times ($\mathbf{ns}_i$ and $\mathbf{nf}_i$) of each T_i in this generation. Meanwhile, add this record to the tail of memory as shown in Fig. 6. In addition, as for $ns_{i,G-j}, i = 1, ..., n, j = 1, ..., LP$, G is the number of current iteration.(4) If the length of *SM* and *FM* is bigger than learn-rate *LP*, delete the first record in *SM* and *FM*. Then, update the selection probability P_i of each parameter T_i. Next, sum each column in *SM*(*FM*) as *SR*(*FR*). Finally, the selection probability P_i can be got. (5) Normalize P_i to let the sum equal to 1. Algorithm 4 gives the detail.

5. Experimental Results

In Section 4, the designed algorithm has been described in detail. In this section, we design detailed experiments to evaluate the performance of the proposed HPEA algorithm. HPEA and the comparison algorithms are coded in MATLAB on an Intel Core *i*7 6700 CPU @ 3.4 GHz with 8G RAM. For fairness, all algorithm runs 30 independent times on each instance with same stopping criteria (MaxIter = 200). Noting that, to verify the convergence and diversity of the proposed algorithm, after 30 independent runs, the average results are collected for performance comparison.

The comparison algorithms include MOEA/D (Zhang & Hui, 2007), MOEA/D-M2M (Liu, Gu, & Zhang, 2014), NSGA-III (Deb & Jain, 2014), MO-LR (Wang, Gong, Wu, & Zhang, 2020), and IAIS (Li et al., 2020). And three metrics are used as the performance measures like Hypervolume (HV) (While, Hingston, Barone, & Huband, 2006) value, Generation Distance (GD), and Spread (Deb, Pratap, Agarwal, & Meyarivan, 2002). The equations are given as follows:

Operation sequence	3	2	1	2	1	3	1	3	2
Machine selection	1	3	2	2	2	3	2	2	1

Fig. 2. Encoding representation.

$$HV\left(P,r\right)=\bigcup_{\mathbf{x}\in P}^{P}v\left(\mathbf{x},r\right). \tag{21}$$

$$GD\left(P,P^{*}\right)=\frac{\sqrt{\sum \min dis(x,y)^{2}}_{y\in P x\in P^{*}}}{|P|} \tag{22}$$

$$Spread=\frac{d_l+d_f+\sum_{i=1}^{N-1}\left|d_i-\overline{d}\right|}{d_l+d_f+\left(N-1\right)\overline{d}} \tag{23}$$

HV can measure the comprehensive performance of one algorithm. Regarding these metrics, lower GD and Spread values are better, but a bigger HV value is better.

5.1. Experimental instances

Three benchmarks are selected to verify the convergence and diversity of the proposed HPEA. The first benchmark *Lei*01 and *Lei*02 are obtained from (Lei, 2010) (Lei, 2012). The second benchmark *Remanu* is provided by (Gao et al., 2015b). All the instances are FFJSP with fuzzy processing time. Moreover, we transformed a flexible job shop scheduling problem benchmark *Mk* (Brandimarte, 1993). Refer to each processing time b, two integers a and c were randomly generated in interval $[0, b/2]$. (a,b,c) is a TFN. Therefore, *Mk* benchmark is converted to the FFJSP benchmark *FMk*.

On the first benchmark, each operation can select all machines as a candidate set. There are 5 instances in *Lei*01 and *Lei*02, where N =10, 10, 10, 10, 15, and the total number of operation SH = 40, 40, 50, 50, 80. And M = 10 for all instance.

On the second benchmark, not all machine can be chosen as candidate set because it is an incomplete FFJSP. There are 8 instances in *Remanu*, where N = 5, 8, 10, 10, 15, 15, 20, 20., SH = 23, 64, 81, 100, 171, 185, 308, 355. and M = 4, 8, 6, 10, 8, 10, 10, 15.

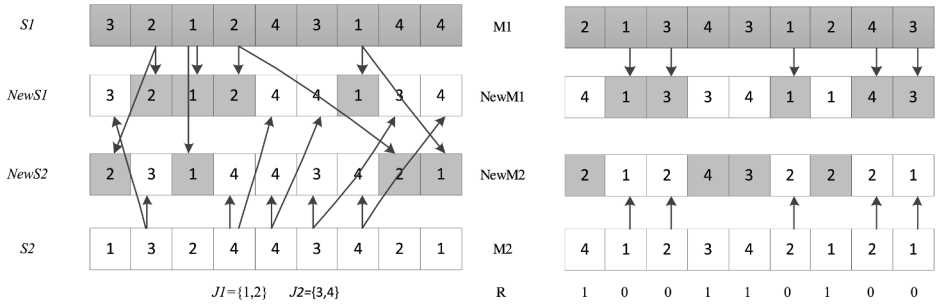
The third benchmark is similar to the second one. An operation can not be processed on all machine. There are 10 instances in *FMk*, where N = 10, 10, 15, 15, 15, 10, 20, 20, 20, 20., SH = 55, 58, 150, 90, 106, 150, 100, 225, 240, 240. and M = 6, 6, 8, 8, 4, 15, 5, 10, 10, 15.

5.2. Experimental parameters

The parameter configuration can impact the performance of the algorithm in solving this problem. The proposed HPEA contains three parameters. The mutation rate R, size of success and failure memory LP and the candidate parameter vector T. A Taguchi approach of design-of-experiment (DOE)(Van Nostrand, 2002). The parameter level is given as follows:

- R = 0.4, 0.5, 0.6, 0.8.
- LP = 45, 50, 55, 60.
- T= {$T1$={3,5,7,8,10}, $T2$={3,5,7,8,10,12}, $T3$={3,5,7,8,10,12,14}, $T4$={3,5,7,8,10,12,14,16,18,20}}.

An orthogonal array $L_{16}(4^4)$ is adopted in this calibration experi-



(a) Example of operation sequence crossover (b) Example of machine selection crossover

Fig. 3. An example of the crossover methods of two vectors.

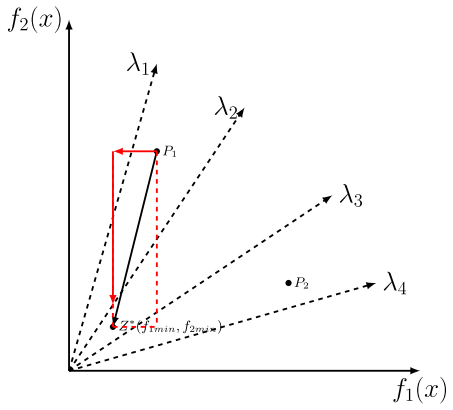


Fig. 4. An Example of calculating Tchebycheff function.

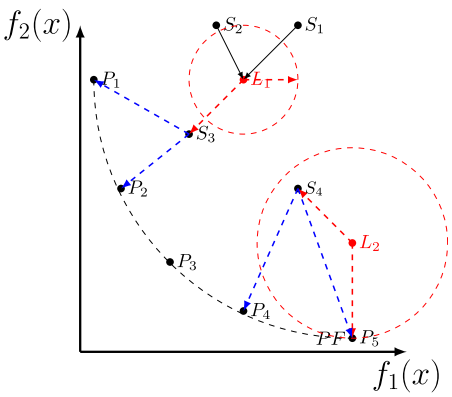


Fig. 5. Illustration of VNS.

ment. For fairness, each parameter runs 30 independent times. The population size *Np* is 100 and the max generation *G* is 200. We collect the average HV value for 30 runs. Fig. 7 shows the main effects plot of three parameters for all metrics. The higher HV metric values is, the better performance is. But for GD and Spread, the lower metrics are, the better performance is. Based on the comprehensive observation, the best configure of parametric value is set as $R = 0.8, LP = 45, T = T2$.

5.3. Effectiveness of each improvement part of HPEA

To verify the effectiveness of each improvement part of the proposed algorithm, we compare three variants of HPEA, where HPEA1 denotes pure MOEA/D, HPEA2 represents HPEA1 embedding with initialization heuristic, and HPEA3 denotes HPEA2 adding VNS strategy. HPEA is HPEA3 embedding with parameter adaption strategy. Table 2 lists metrics values over 30 independent runs on 23 instances. The best mean

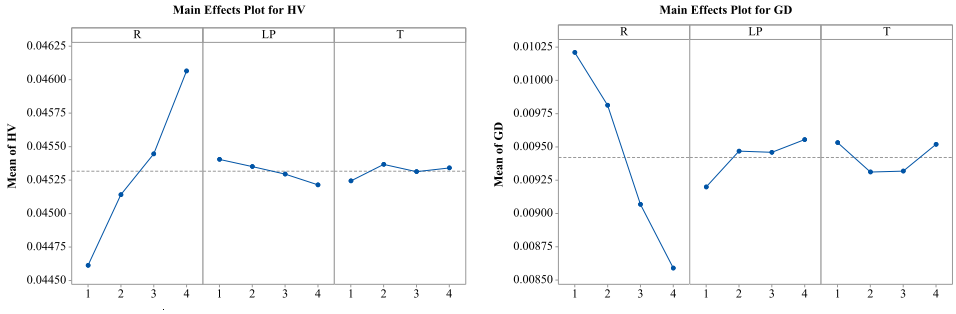
Success Memeory

Index	Parameter 1	Parameter 2	⋯	Parameter n
1	$\mathbf{ns}_{1,\text{G-LP}}$	$\mathbf{ns}_{2,G\text{-}LP}$	⋯	$\mathbf{ns}_{n,G\text{-}LP}$
2	$\mathbf{ns}_{1,G\text{-}LP+1}$	$\mathbf{ns}_{2,G\text{-}LP+1}$	⋯	$\mathbf{ns}_{n,G\text{-}LP+1}$
⋮	⋮	⋮	⋮	⋮
LP	$\mathbf{ns}_{1,G-1}$	$\mathbf{ns}_{2,G-1}$	⋯	$\mathbf{ns}_{n,G-1}$

Failure Memeory

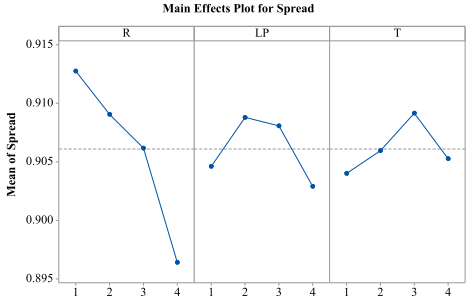
Index	Parameter 1	Parameter 2	⋯	Parameter n
1	$\mathbf{nf}_{1,\text{G-LP}}$	$\mathbf{nf}_{2,G\text{-}LP}$	⋯	$\mathbf{nf}_{n,G\text{-}LP}$
2	$\mathbf{nf}_{1,G\text{-}LP+1}$	$\mathbf{nf}_{2,G\text{-}LP+1}$	⋯	$\mathbf{ns}_{n,G\text{-}LP+1}$
⋮	⋮	⋮	⋮	⋮
LP	$\mathbf{nf}_{1,G-1}$	$\mathbf{nf}_{2,G-1}$	⋯	$\mathbf{nf}_{n,G-1}$

Fig. 6. Success memory and failure memory.



(c) HV

(d) GD



(e) Spread

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

Table 2
Statistical results of all metrics of HPEA and its variants in all instances.

HV

instances	HPEA1		HPEA2		HPEA3		HPEA	
	mean	std	mean	std	mean	std	mean	std
data1	9.42E-02	3.26E-03	9.57E-02	2.73E-03	**9.83E-02**	1.84E-03	9.67E-02	2.48E-03
data2	9.55E-02	3.04E-03	9.63E-02	2.98E-03	**9.87E-02**	2.21E-03	9.70E-02	3.05E-03
data3	6.20E-02	3.24E-03	6.38E-02	3.26E-03	**6.57E-02**	2.58E-03	6.47E-02	2.03E-03
data4	5.92E-02	2.99E-03	6.01E-02	2.83E-03	6.19E-02	2.66E-03	**6.21E-02**	2.47E-03
data5	4.81E-02	2.50E-03	5.09E-02	2.91E-03	**5.36E-02**	1.84E-03	5.34E-02	2.16E-03
FMk01	**5.85E-02**	3.63E-03	5.61E-02	3.27E-03	5.64E-02	3.18E-03	**5.85E-02**	2.64E-03
FMk02	4.04E-02	2.15E-03	3.89E-02	2.48E-03	4.09E-02	2.68E-03	**4.15E-02**	2.07E-03
FMk03	5.67E-02	9.46E-04	5.64E-02	1.13E-03	**5.76E-02**	1.12E-03	5.72E-02	1.19E-03
FMk04	9.39E-02	1.75E-03	9.51E-02	1.64E-03	9.42E-02	2.30E-03	**9.53E-02**	1.81E-03
FMk05	3.72E-02	1.25E-03	**3.76E-02**	1.16E-03	**3.76E-02**	1.11E-03	3.74E-02	8.96E-04
FMk06	4.52E-02	3.24E-03	4.34E-02	3.21E-03	**4.61E-02**	2.82E-03	4.51E-02	2.72E-03
FMk07	**5.81E-02**	1.44E-03	5.71E-02	1.80E-03	5.70E-02	1.38E-03	5.71E-02	1.48E-03
FMk08	1.96E-02	1.11E-03	1.84E-02	1.09E-03	1.95E-02	8.87E-04	**1.97E-02**	8.91E-04
FMk09	3.35E-02	1.75E-03	3.30E-02	2.00E-03	**3.36E-02**	1.64E-03	3.30E-02	1.85E-03
FMk10	3.94E-02	1.90E-03	4.06E-02	2.53E-03	**4.13E-02**	2.30E-03	4.12E-02	2.36E-03
remanu01	**3.70E-02**	1.65E-03	3.69E-02	1.87E-03	3.67E-02	1.95E-03	**3.70E-02**	1.65E-03
remanu02	3.55E-02	3.33E-03	3.66E-02	3.47E-03	3.68E-02	3.06E-03	**3.81E-02**	2.49E-03
remanu03	3.23E-02	2.45E-03	3.24E-02	2.38E-03	3.37E-02	2.74E-03	**3.48E-02**	2.50E-03
remanu04	3.92E-02	3.40E-03	3.93E-02	3.03E-03	4.25E-02	2.71E-03	**4.27E-02**	3.20E-03
remanu05	3.48E-02	2.15E-03	3.66E-02	1.97E-03	**3.84E-02**	1.58E-03	**3.84E-02**	1.78E-03
remanu06	3.77E-02	2.56E-03	4.26E-02	2.16E-03	4.60E-02	2.41E-03	**4.65E-02**	1.98E-03
remanu07	3.16E-02	3.13E-03	4.21E-02	2.36E-03	4.72E-02	2.16E-03	**4.76E-02**	2.35E-03
remanu08	4.12E-02	3.82E-03	6.77E-02	3.76E-03	**7.47E-02**	2.53E-03	7.42E-02	2.74E-03

GD

instances	HPEA1		HPEA2		HPEA3		HPEA	
	mean	std	mean	std	mean	std	mean	std
data1	6.56E-03	2.56E-03	4.62E-03	2.76E-03	**2.92E-03**	7.64E-04	3.96E-03	1.13E-03
data2	4.89E-03	1.83E-03	4.18E-03	1.66E-03	**2.09E-03**	7.42E-04	2.85E-03	1.13E-03
data3	7.60E-03	2.41E-03	8.33E-03	5.24E-03	**4.26E-03**	1.10E-03	4.83E-03	1.26E-03
data4	1.07E-02	4.99E-03	7.14E-03	2.57E-03	**5.09E-03**	1.14E-03	5.97E-03	3.52E-03
data5	1.87E-02	1.13E-02	1.69E-02	9.03E-03	1.25E-02	5.68E-03	**1.21E-02**	5.59E-03
FMk01	9.71E-03	6.05E-03	9.82E-03	5.78E-03	6.59E-03	4.13E-03	**5.25E-03**	3.15E-03
FMk02	7.46E-03	4.43E-03	1.04E-02	4.41E-03	6.10E-03	4.93E-03	**5.77E-03**	3.43E-03
FMk03	1.18E-03	3.95E-04	7.47E-04	2.90E-04	**5.82E-04**	1.72E-04	7.01E-04	3.52E-04
FMk04	1.46E-03	3.91E-04	1.40E-03	6.19E-04	1.27E-03	5.87E-04	**1.24E-03**	4.94E-04
FMk05	7.82E-04	4.53E-04	7.83E-04	2.94E-04	**6.55E-04**	2.63E-04	7.26E-04	2.58E-04
FMk06	1.33E-02	3.56E-03	1.46E-02	4.92E-03	**1.02E-02**	6.01E-03	1.08E-02	4.60E-03
FMk07	1.26E-03	4.83E-04	1.30E-03	4.41E-04	**1.20E-03**	4.35E-04	1.33E-03	5.77E-04
FMk08	4.24E-04	2.63E-04	5.31E-04	3.07E-04	**2.97E-04**	1.93E-04	3.44E-04	2.25E-04
FMk09	2.59E-03	9.61E-04	1.71E-03	1.06E-03	**1.29E-03**	8.31E-04	1.42E-03	5.13E-04
FMk10	4.47E-03	1.29E-03	3.66E-03	1.97E-03	**2.90E-03**	1.52E-03	3.20E-03	1.61E-03
remanu01	4.14E-03	4.87E-03	4.69E-03	4.64E-03	4.35E-03	4.74E-03	**3.09E-03**	4.01E-03
remanu02	3.22E-02	1.43E-02	2.75E-02	1.09E-02	2.34E-02	1.25E-02	**2.21E-02**	1.05E-02
remanu03	1.43E-02	7.47E-03	1.30E-02	7.88E-03	9.94E-03	9.18E-03	**6.07E-03**	3.48E-03
remanu04	9.67E-03	5.14E-03	9.98E-03	4.40E-03	4.93E-03	2.69E-03	**4.39E-03**	2.10E-03
remanu05	8.54E-03	3.13E-03	6.05E-03	2.43E-03	4.67E-03	1.07E-03	**4.52E-03**	1.86E-03
remanu06	3.87E-02	1.05E-02	3.13E-02	7.04E-03	**2.10E-02**	1.37E-02	2.29E-02	1.45E-02
remanu07	3.86E-02	1.14E-02	2.66E-02	8.58E-03	**1.53E-02**	8.67E-03	1.77E-02	9.55E-03
remanu08	5.81E-02	9.46E-03	3.50E-02	1.02E-02	**2.48E-02**	1.14E-02	2.59E-02	1.34E-02

Spread

instances	HPEA1		HPEA2		HPEA3		HPEA	
	mean	std	mean	std	mean	std	mean	std
data1	9.33E-01	1.72E-01	8.46E-01	1.26E-01	**8.29E-01**	1.40E-01	8.31E-01	1.18E-01
data2	9.86E-01	1.28E-01	9.46E-01	1.12E-01	**9.07E-01**	1.41E-01	9.80E-01	1.13E-01
data3	9.16E-01	1.77E-01	9.34E-01	1.63E-01	**9.22E-01**	1.58E-01	9.29E-01	1.88E-01
data4	9.91E-01	1.94E-01	8.82E-01	1.36E-01	9.50E-01	1.81E-01	**9.32E-01**	1.92E-01
data5	9.97E-01	1.92E-01	9.49E-01	1.88E-01	9.28E-01	2.65E-01	**9.09E-01**	2.34E-01
FMk01	8.03E-01	2.03E-01	7.76E-01	1.70E-01	7.31E-01	2.04E-01	**7.25E-01**	2.14E-01
FMk02	**8.88E-01**	2.02E-01	8.94E-01	2.01E-01	9.02E-01	2.03E-01	**8.88E-01**	1.78E-01
FMk03	**9.88E-01**	1.04E-01	1.12E+00	9.30E-02	1.15E+00	8.11E-02	1.13E+00	1.32E-01
FMk04	8.40E-01	8.62E-02	8.45E-01	8.37E-02	**8.16E-01**	7.92E-02	8.61E-01	8.99E-02
FMk05	8.16E-01	8.76E-02	**7.68E-01**	8.18E-02	7.89E-01	1.30E-01	8.14E-01	1.23E-01
FMk06	**7.10E-01**	1.21E-01	7.36E-01	1.33E-01	7.33E-01	1.48E-01	7.18E-01	1.23E-01
FMk07	7.65E-01	1.07E-01	**7.34E-01**	7.06E-02	7.51E-01	9.06E-02	7.44E-01	9.98E-02
FMk08	5.58E-01	2.72E-01	6.25E-01	2.47E-01	4.97E-01	3.03E-01	**4.54E-01**	2.61E-01
FMk09	8.97E-01	1.22E-01	**8.92E-01**	1.03E-01	9.01E-01	1.17E-01	8.95E-01	8.50E-02
FMk10	8.90E-01	1.33E-01	9.67E-01	1.20E-01	**8.88E-01**	1.69E-01	9.08E-01	1.48E-01
remanu01	1.05E+00	3.60E-01	**9.13E-01**	2.73E-01	9.90E-01	3.53E-01	1.11E+00	3.65E-01
remanu02	1.02E+00	1.41E-01	9.89E-01	2.06E-01	1.04E+00	1.97E-01	**9.76E-01**	2.81E-01

(*continued on next page*)

Table 2 (*continued*)

instances	HV							
	HPEA1		HPEA2		HPEA3		HPEA	
	mean	std	mean	std	mean	std	mean	std
remanu03	**9.56E-01**	1.91E-01	9.77E-01	1.89E-01	9.75E-01	1.88E-01	9.91E-01	2.42E-01
remanu04	8.83E-01	1.20E-01	8.96E-01	1.50E-01	**8.68E-01**	1.52E-01	8.76E-01	2.11E-01
remanu05	8.88E-01	1.93E-01	8.84E-01	1.87E-01	**7.85E-01**	1.87E-01	8.74E-01	2.66E-01
remanu06	**9.72E-01**	9.76E-02	9.91E-01	1.25E-01	1.10E+00	2.21E-01	1.09E+00	2.01E-01
remanu07	9.92E-01	1.08E-01	**9.87E-01**	1.19E-01	1.08E+00	1.87E-01	1.12E+00	3.58E-01
remanu08	**9.87E-01**	6.64E-02	1.05E+00	1.39E-01	1.21E+00	5.90E-01	1.13E+00	2.80E-01

Table 3
Statistical values of three metrics among all algorithm.

MOEAs	HV		GD		Spread	
	mean	std	mean	std	mean	std
MOEA/D	1.10E-01	2.68E-03	2.04E-02	5.10E-03	**9.02E-01**	**1.30E-01**
MOEA/D-M2M	8.37E-02	1.00E-02	5.07E-02	1.26E-02	9.91E-01	1.72E-01
NSGA-III	9.25E-02	5.15E-03	3.33E-02	9.46E-03	1.01E+00	1.39E-01
MO-LR	1.01E-01	4.06E-03	2.63E-02	8.04E-03	1.03E+00	1.63E-01
IAIS	1.01E-01	6.02E-03	4.10E-02	1.44E-02	1.04E+00	1.33E-01
HPEA	**1.15E-01**	**2.23E-03**	**1.88E-02**	**4.70E-03**	9.03E-01	1.52E-01

Table 4
Overall ranks through the friedman test of three metrics among algorithm(a level of significant $\alpha = 0.05$).

MOEAs	HV		GD		Spread	
	rank	p-value	rank	p-value	rank	p-value
MOEA/D	1.913	9.16E-18	1.9565	3.82E-17	**1.6957**	2.03E-09
MOEA/D-M2M	5.2609		5.5217		3.5217	
NSGA-III	4.6522		4.1739		4.1304	
MO-LR	3.3913		3.2609		4.7826	
IAIS	4.6087		4.7391		4.4783	
HPEA	**1.1739**		**1.3478**		2.3913	

values are marked in **bold** and gray. The last row is the Friedman ranking result and the P-value is 2.2716E-05 < 0.05.

5.4. Comparison and discussion

To further evaluate the performance of the proposed HPEA algorithm. We selected the following algorithms for comparison: MOEA/D, MOEA/D-M2M, NSGA-III, MO-LR, and IAIS. For each compared algorithm, the related parameter selects the best one discussed in Section 5.2 and the results obtained after 30 independently runs are used to make detailed comparisons.

The parameter is set that the number of neighborhood $T = 10$ and mutation rate $R = 0.8$. For MOEA/D, MOEA/D-M2M, NSGA-III and HPEA the number of weight vectors is equal to population size $Np = 100$. For MOEA/D-M2M size of sub-population $S = 10$. For HPEA $LP = 45, T = T2$. And IAIS's parameter is set according to (Li et al., 2020), but to suit three benchmarks $\omega = 0.5$ and $CRmax = 1 \times 10^{-4}$. To conduct a fair comparison, they use the above-mentioned encoding and decoding mechanism, and fuzzy operators. Table 3 lists statistical results (mean and standard deviation values) on three benchmarks. The optimal results are marked in **bold**. And Table 4 records the Friedman rank test among all algorithms, where confidence level $\alpha = 0.05$.

Discussion: As observed in Table 3, HPEA achieves lower value than its competitors for the GD and HV metrics. Regarding the Spread, HPEA is inferior to MOEA/D but superior to the other algorithms. As observed in Table 4, HPEA ranks the first for HV and GD metrics, but ranks the second for Spread metric. Also, since the p-values for algorithms are smaller than the significance level of 0.05, it can be concluded that at least one of six algorithms has a significant difference effect for three metrics. The success of HPEA lies in its algorithm design. First, a cooperative initialization strategy is proposed to generate one high-quality population. Second, we utilize problem-specific knowledge and then regard this knowledge as a general VNS heuristic. This knowledge-based VNS can guide trial solutions towards Pareto optimal solutions. By contrast, the other MOEAs does not have this local search. Third, a parameter adaption strategy is designed to let algorithm select the best parameter to further improve the performance. (See Table 5).

To visualize the behavior of different algorithms on those

Table 5
Statistical results of all metric of all algorithms in all instances.

HV

instances	MOEA/D		MOEA/D-M2M		NSGA-III		MO-LR		IAIS		HPEA	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
data1	1.22E-01	3.42E-03	1.02E-01	4.23E-03	1.07E-01	4.66E-03	1.12E-01	5.28E-03	1.02E-01	8.84E-03	**1.24E-01**	2.59E-03
data2	1.45E-01	3.06E-03	1.33E-01	3.49E-02	1.30E-01	5.71E-03	1.36E-01	4.01E-03	1.26E-01	9.20E-03	**1.47E-01**	3.12E-03
data3	1.14E-01	3.11E-03	9.60E-02	4.71E-03	9.96E-02	6.11E-03	1.04E-01	4.89E-03	1.07E-01	4.89E-03	**1.18E-01**	1.69E-03
data4	1.12E-01	2.90E-03	9.08E-02	5.15E-03	9.68E-02	6.80E-03	1.03E-01	4.86E-03	1.06E-01	4.85E-03	**1.15E-01**	2.39E-03
data5	1.53E-01	3.47E-03	1.12E-01	6.38E-03	1.29E-01	7.64E-03	1.40E-01	7.33E-03	1.49E-01	6.37E-03	**1.60E-01**	3.36E-03
FMk01	**7.51E-02**	3.46E-03	6.80E-02	3.00E-03	6.70E-02	3.96E-03	7.02E-02	2.43E-03	6.09E-02	6.65E-03	7.50E-02	2.53E-03
FMk02	7.04E-02	1.50E-03	6.22E-02	2.84E-03	6.42E-02	3.77E-03	6.61E-02	2.82E-03	5.86E-02	3.21E-03	**7.13E-02**	1.47E-03
FMk03	7.12E-02	1.15E-03	6.03E-02	2.38E-03	6.20E-02	2.90E-03	6.38E-02	2.21E-03	4.88E-02	1.02E-02	**7.17E-02**	1.50E-03
FMk04	9.71E-02	1.68E-03	8.27E-02	2.88E-03	7.71E-02	6.65E-03	8.33E-02	4.85E-03	6.83E-02	7.27E-03	**9.87E-02**	1.73E-03
FMk05	6.21E-02	8.93E-04	5.62E-02	3.83E-03	5.58E-02	1.73E-03	5.67E-02	1.32E-03	5.41E-02	3.32E-03	**6.24E-02**	6.49E-04
FMk06	**1.35E-01**	5.79E-03	9.82E-02	5.78E-03	1.17E-01	6.64E-03	1.31E-01	5.23E-03	9.67E-02	1.27E-02	1.33E-01	4.86E-03
FMk07	**9.91E-02**	8.29E-04	9.03E-02	2.63E-03	8.99E-02	2.90E-03	9.19E-02	2.00E-03	8.42E-02	5.27E-03	9.87E-02	9.92E-04
FMk08	4.33E-02	9.40E-04	3.64E-02	1.09E-03	3.81E-02	9.22E-04	3.87E-02	1.34E-03	3.82E-02	1.30E-03	**4.34E-02**	7.58E-04
FMk09	9.26E-02	1.68E-03	7.12E-02	3.62E-03	8.38E-02	3.82E-03	9.00E-02	2.37E-03	8.24E-02	5.66E-03	**9.27E-02**	1.75E-03
FMk10	1.69E-01	2.11E-03	1.24E-01	5.35E-03	1.48E-01	5.28E-03	1.61E-01	4.76E-03	1.53E-01	1.37E-02	**1.72E-01**	2.40E-03
remanu01	**1.23E-01**	1.63E-03	1.13E-01	5.33E-03	1.23E-01	1.68E-03	1.22E-01	2.94E-03	1.16E-01	6.73E-03	1.23E-01	1.64E-03
remanu02	7.38E-02	3.28E-03	6.13E-02	4.04E-03	6.88E-02	3.82E-03	7.11E-02	3.73E-03	6.80E-02	5.46E-03	**7.62E-02**	2.46E-03
remanu03	9.37E-02	1.61E-03	9.23E-02	3.21E-02	8.96E-02	2.06E-03	9.19E-02	1.78E-03	8.90E-02	2.64E-03	**9.53E-02**	1.66E-03
remanu04	1.03E-01	2.45E-03	8.26E-02	5.17E-03	9.32E-02	4.72E-03	9.89E-02	2.64E-03	9.61E-02	3.04E-03	**1.06E-01**	2.44E-03
remanu05	1.07E-01	2.56E-03	8.06E-02	4.53E-02	8.54E-02	5.22E-03	9.82E-02	3.47E-03	1.03E-01	2.91E-03	**1.12E-01**	1.86E-03
remanu06	1.36E-01	3.07E-03	7.78E-02	6.53E-03	1.05E-01	7.58E-03	1.23E-01	6.19E-03	1.36E-01	4.78E-03	**1.47E-01**	2.18E-03
remanu07	1.71E-01	5.64E-03	6.44E-02	3.61E-02	1.13E-01	1.01E-02	1.50E-01	8.31E-03	1.76E-01	4.29E-03	**1.95E-01**	3.95E-03
remanu08	1.66E-01	5.53E-03	6.85E-02	6.54E-03	8.34E-02	1.37E-02	1.26E-01	8.60E-03	1.94E-01	5.20E-03	**2.09E-01**	3.38E-03

GD

instances	MOEA/D		MOEA/D-M2M		NSGA-III		MO-LR		IAIS		HPEA	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
data1	6.06E-03	2.40E-03	2.55E-02	9.67E-03	1.20E-02	4.22E-03	1.01E-02	3.56E-03	1.46E-02	1.21E-02	**3.63E-03**	1.09E-03
data2	**3.24E-02**	7.57E-03	6.67E-02	1.28E-02	4.85E-02	1.40E-02	3.55E-02	1.29E-02	8.99E-02	2.43E-02	3.99E-02	6.27E-03
data3	5.67E-03	1.75E-03	4.36E-02	1.53E-02	1.65E-02	7.68E-03	1.11E-02	4.59E-03	1.43E-02	1.20E-02	**3.12E-03**	9.54E-04
data4	8.31E-03	3.57E-03	4.69E-02	2.07E-02	2.12E-02	1.35E-02	1.45E-02	7.31E-03	9.33E-03	8.59E-03	**4.23E-03**	2.74E-03
data5	1.39E-02	8.12E-03	6.07E-02	1.08E-02	3.36E-02	1.23E-02	2.46E-02	9.39E-03	2.35E-02	1.18E-02	**8.82E-03**	3.99E-03
FMk01	8.64E-03	5.33E-03	1.85E-02	1.41E-02	1.17E-02	9.12E-03	1.01E-02	4.65E-03	2.04E-02	1.50E-02	**4.70E-03**	2.78E-03
FMk02	5.61E-03	2.90E-03	3.05E-02	1.34E-02	1.16E-02	1.14E-02	9.07E-03	8.45E-03	2.07E-02	8.40E-03	**4.67E-03**	2.31E-03
FMk03	1.25E-03	4.23E-04	5.61E-03	1.76E-03	4.16E-03	2.08E-03	2.71E-03	1.11E-03	5.29E-03	2.65E-03	**7.25E-04**	3.58E-04
FMk04	1.45E-03	3.81E-04	1.94E-03	9.32E-04	1.39E-03	9.33E-04	1.68E-03	8.16E-04	6.93E-03	3.44E-03	**9.89E-04**	2.58E-04
FMk05	6.75E-04	4.26E-04	1.79E-02	1.02E-02	**4.99E-04**	4.19E-04	8.24E-04	1.50E-03	3.48E-03	1.95E-03	5.64E-04	2.11E-04
FMk06	9.55E-03	2.81E-03	3.96E-02	4.26E-03	1.91E-02	4.61E-03	1.02E-02	5.27E-03	4.73E-02	2.57E-02	**7.39E-03**	3.42E-03
FMk07	**1.05E-03**	4.39E-04	5.85E-03	5.69E-03	3.14E-03	1.70E-03	1.85E-03	9.49E-04	3.43E-03	2.30E-03	1.13E-03	5.24E-04
FMk08	3.53E-04	2.19E-04	1.22E-02	2.16E-02	7.77E-04	2.19E-03	3.37E-04	5.05E-04	2.19E-03	2.34E-03	**2.83E-04**	1.76E-04
FMk09	2.14E-03	8.78E-04	3.63E-02	1.14E-02	7.58E-03	4.40E-03	3.38E-03	1.59E-03	5.67E-03	6.65E-03	**1.01E-03**	3.67E-04
FMk10	3.46E-03	7.70E-04	3.35E-02	1.64E-02	1.13E-02	2.66E-03	7.03E-03	2.87E-03	9.11E-03	1.37E-02	**1.67E-03**	5.38E-04
remanu01	3.04E-03	3.71E-03	2.39E-02	2.65E-02	3.81E-03	7.68E-03	1.13E-02	1.44E-02	6.04E-03	6.42E-03	**2.21E-03**	2.74E-03
remanu02	2.40E-02	1.10E-02	7.19E-02	2.29E-02	4.09E-02	1.71E-02	3.83E-02	2.72E-02	4.63E-02	2.79E-02	**1.59E-02**	8.17E-03
remanu03	**1.18E-01**	2.04E-02	1.28E-01	2.90E-02	1.56E-01	3.13E-02	1.42E-01	3.10E-02	2.19E-01	3.98E-02	1.47E-01	2.74E-02
remanu04	6.41E-03	3.06E-03	5.19E-02	1.48E-02	1.64E-02	1.18E-02	8.63E-03	3.27E-03	1.73E-02	1.05E-02	**2.98E-03**	1.56E-03
remanu05	**1.27E-01**	2.07E-02	1.39E-01	7.77E-03	1.45E-01	1.89E-02	1.42E-01	2.03E-02	2.42E-01	5.00E-02	1.39E-01	1.79E-02
remanu06	2.39E-02	6.34E-03	9.30E-02	5.07E-03	4.40E-02	9.35E-03	2.57E-02	5.83E-03	3.97E-02	1.64E-02	**1.40E-02**	8.85E-03
remanu07	2.65E-02	7.75E-03	1.01E-01	1.04E-02	6.29E-02	1.18E-02	3.51E-02	8.33E-03	4.68E-02	1.39E-02	**1.18E-02**	6.49E-03
remanu08	4.01E-02	6.41E-03	1.12E-01	4.48E-03	9.41E-02	1.83E-02	5.80E-02	9.30E-03	4.92E-02	1.48E-02	**1.74E-02**	9.00E-03

Spread

instances	MOEA/D		MOEA/D-M2M		NSGA-III		MO-LR		IAIS		HPEA	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
data1	9.25E-01	1.70E-01	9.56E-01	1.79E-01	1.03E+00	1.71E-01	1.15E+00	2.98E-01	1.17E+00	2.87E-01	**8.22E-01**	1.16E-01
data2	9.81E-01	7.80E-02	**9.55E-01**	1.05E-01	1.06E+00	1.05E-01	1.06E+00	1.25E-01	1.01E+00	1.05E-01	9.81E-01	5.69E-02
data3	**9.02E-01**	1.72E-01	9.39E-01	2.15E-01	1.09E+00	2.01E-01	1.12E+00	2.22E-01	1.09E+00	2.04E-01	9.03E-01	1.82E-01
data4	9.72E-01	1.96E-01	1.02E+00	2.83E-01	1.12E+00	2.50E-01	1.06E+00	2.37E-01	1.04E+00	1.99E-01	**9.09E-01**	1.88E-01
data5	9.90E-01	1.88E-01	9.69E-01	1.54E-01	1.07E+00	2.19E-01	1.16E+00	2.82E-01	1.06E+00	2.60E-01	**8.98E-01**	2.32E-01
FMk01	7.93E-01	2.03E-01	9.53E-01	2.54E-01	9.49E-01	2.04E-01	9.51E-01	2.39E-01	1.03E+00	1.95E-01	**7.10E-01**	2.13E-01
FMk02	**8.63E-01**	1.94E-01	9.82E-01	2.70E-01	1.06E+00	2.34E-01	1.08E+00	2.70E-01	1.00E+00	8.78E-02	8.67E-01	1.70E-01
FMk03	9.89E-01	9.37E-02	**9.42E-01**	7.40E-02	9.91E-01	6.69E-02	1.01E+00	4.79E-02	1.02E+00	2.02E-01	1.11E+00	1.15E-01
FMk04	**8.83E-01**	6.29E-02	9.28E-01	1.23E-01	9.97E-01	3.77E-02	9.98E-01	4.56E-02	1.14E+00	3.05E-01	8.98E-01	6.48E-02
FMk05	8.37E-01	7.42E-02	1.09E+00	2.17E-01	9.45E-01	7.95E-02	1.02E+00	8.32E-02	1.12E+00	3.41E-01	**8.35E-01**	1.00E-01
FMk06	**7.77E-01**	9.88E-02	8.82E-01	7.10E-02	9.86E-01	1.14E-01	9.65E-01	1.77E-01	9.92E-01	2.71E-02	7.98E-01	9.26E-02
FMk07	7.43E-01	1.03E-01	9.62E-01	1.94E-01	9.31E-01	8.20E-02	9.65E-01	1.06E-01	1.14E+00	2.27E-01	**7.28E-01**	1.01E-01
FMk08	5.50E-01	2.68E-01	1.13E+00	2.95E-01	8.54E-01	1.31E-01	8.71E-01	1.53E-01	1.05E+00	1.76E-01	**4.50E-01**	2.60E-01
FMk09	**8.78E-01**	1.22E-01	9.77E-01	1.82E-01	1.05E+00	1.65E-01	1.00E+00	1.68E-01	1.02E+00	6.24E-02	8.84E-01	8.07E-02
FMk10	**8.73E-01**	1.13E-01	9.90E-01	2.27E-01	1.01E+00	1.32E-01	9.87E-01	1.17E-01	1.00E+00	1.70E-02	8.88E-01	1.36E-01
remanu01	**9.90E-01**	3.58E-01	1.13E+00	2.88E-01	1.10E+00	3.45E-01	1.14E+00	4.29E-01	1.02E+00	2.37E-01	1.04E+00	3.46E-01
remanu02	1.02E+00	1.45E-01	1.10E+00	2.28E-01	1.05E+00	2.42E-01	1.10E+00	2.97E-01	9.74E-01	5.86E-02	**9.65E-01**	2.75E-01

(*continued on next page*)

Table 5 (*continued*)

	HV											
instances	MOEA/D		MOEA/D-M2M		NSGA-III		MO-LR		IAIS		HPEA	
	mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
remanu03	**9.91E-01**	2.18E-02	1.04E+00	1.54E-01	1.01E+00	2.93E-02	1.02E+00	3.77E-02	1.00E+00	1.15E-03	9.95E-01	1.79E-02
remanu04	8.63E-01	1.09E-01	9.48E-01	1.52E-01	9.91E-01	2.03E-01	1.06E+00	1.98E-01	1.00E+00	2.31E-02	**8.56E-01**	2.01E-01
remanu05	**9.88E-01**	1.75E-02	1.01E+00	1.12E-01	9.95E-01	2.19E-02	1.01E+00	3.38E-02	1.00E+00	2.57E-03	9.90E-01	1.89E-02
remanu06	9.67E-01	9.36E-02	**9.30E-01**	8.46E-02	1.02E+00	9.43E-02	1.00E+00	8.46E-02	1.01E+00	2.63E-02	1.08E+00	1.96E-01
remanu07	9.94E-01	4.98E-02	**9.67E-01**	3.91E-02	9.89E-01	3.58E-02	1.01E+00	5.11E-02	1.00E+00	9.13E-03	1.02E+00	6.45E-02
remanu08	**9.86E-01**	6.34E-02	1.00E+00	5.47E-02	9.93E-01	3.03E-02	9.96E-01	4.16E-02	9.95E-01	1.75E-02	1.13E+00	2.76E-01

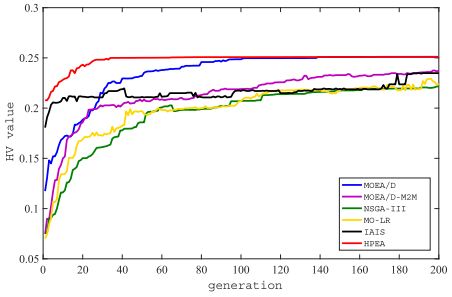


Fig. 8. Comparison of convergence abilities.

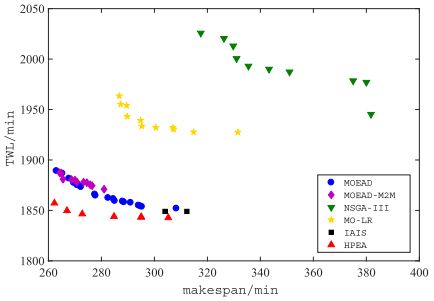


Fig. 9. Pareto Front comparison results.

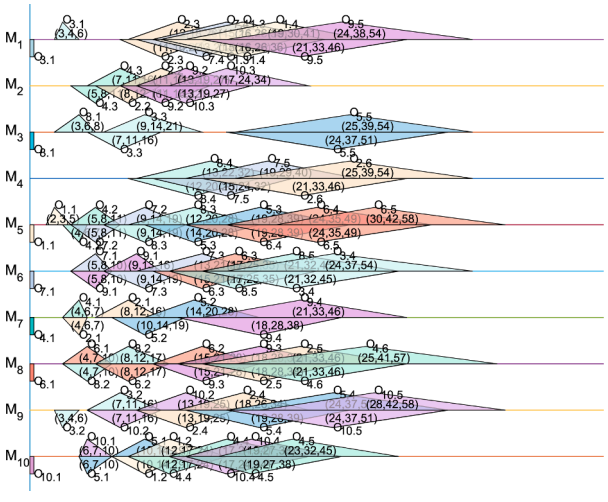


Fig. 10. Gantt chart of solution A with the best C_{max} in data4 instance.

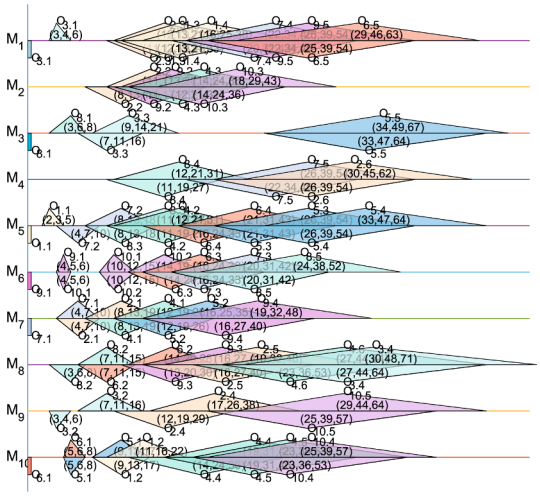


Fig. 11. Gantt chart of solution B with the best *TWL* in data4 instance.

benchmarks, Fig. 8 shows the comparison of convergence curves. It is worth being mentioned that the initial HV value of HPEA and IAIS is more than 0 is because the initialization strategy in Section 4.4 improved the convergence of the initial population.

It can be concluded from Fig. 9 (1) the proposed HPEA algorithm shows better convergence. (2) the proposed initial strategy improves the convergence of population. (3) VNS can improve the convergence. (4) IAIS is unstable because when the population converges, their fitness is close while diversity strategy deletes those solutions and uses the rest of solutions to generate a new solution. That will delete the best solution and reduce the convergence.

Fig. 9 shows the Pareto Front comparison results of all algorithms. Fig. 9 denotes that HPEA's Pareto Front is better than other algorithms because of the proposed strategies such as initial strategy, VNS and parameter adaption.

Fig. 10 shows a Gantt chart of solution A with the best fuzzy makespan of instance data4 $f_1 = (30, 42, 58)$ and $f_2 = (174, 266, 379)$. Fig. 11 shows a Gantt chart of solution B with the best fuzzy total machine workload of instance data4 ($f_1 = (34, 49, 67)$and $f_2 = (172, 261, 373)$).

6. Conclusion

This paper proposes hybrid self-adaptive multi-objective evolutionary algorithm based on decomposition (HPEA) to solve multi-objective flexible job shop scheduling problems with fuzzy processing time. The objective is to minimize the fuzzy maximum completion time and fuzzy total machine workload. To solve multi-objective FFJSP better. An efficient initialization heuristic that combined three different rules is designed to generate an initial population with high convergence

and diversity. Five types of local search methods are fixed to present an efficient variable neighborhood search method to improve the convergence of population and help solution jump out from local optima. Moreover, a parameter adaption strategy is used to make the algorithm adjust the vital parameter automatically, which can promote diversity further. Finally, the proposed HPEA algorithm is compared with four excellent multi-objective optimization algorithms and a novel population-based heuristic for the considered problem. Experimental results show that the proposed HPEA algorithm is superior to other compare algorithms. Both convergence and diversity can be ensured. In conclusion, HPEA can be adaptive to solve multi-objective fuzzy flexible job shop scheduling problems under a high level of uncertainty.

In our future work, we will consider the following tasks: (1) apply the HPEA in different types of realistic applications. Such as distribute flow shop scheduling problems and parallel machine scheduling problems and et al. (2) combine with reinforcement learning algorithm like Q-Learning to automatically choose mutation strategy to make the algorithm more intelligent. (3) Refer to other efficient multi-objective optimization algorithms such as CCMO to design a double population co-evolution algorithm. Each population focuses on different targets like convergence and diversity. (4) Consider other types of fuzzy number such as the type-2 fuzzy number. The proposed algorithm can adapt to other types of applications.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was partly supported by the National Natural Science Foundation of China under Grant No. 62076225.

References

Abdullah, S., & Abdolrazzagh-Nezhad, M. (2014). Fuzzy job-shop scheduling problems: A review. *Information Sciences, 278*, 380–407.

Brandimarte, P. (1993). Routing and scheduling in a flexible job shop by tabu search. *Annals of Operations Research, 41*, 157–183.

Brucker, P., & Schlie, R. (1990). Job-shop scheduling with multi-purpose machines. *Computing, 45*, 369–375.

Chun, W., Na, T., Zhicheng, J., & Yan, W. (2017). Multi-objective evolutionary algorithm to solve fuzzy flexible job shop scheduling problem. *Acta Electronica Sinica, 45*, 2909.

Dai, X., Gong, W., & Gu, Q. (2021). Automated test case generation based on differential evolution with node branch archive. *Computers & Industrial Engineering, 156*, 107290.

Deb, K., & Jain, H. (2014). An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints. *IEEE Transactions on Evolutionary Computation, 18*, 577–601.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation, 6*, 182–197.

Dorfeshan, Y., Tavakkoli-Moghaddam, R., Mousavi, S. M., & Vahedi-Nouri, B. (2020). A new weighted distance-based approximation methodology for flow shop scheduling group decisions under the interval-valued fuzzy processing time. *Applied Soft Computing, 91*, 106248.

Gao, D., Wang, G. G., & Pedrycz, W. (2020). Solving fuzzy job-shop scheduling problem using de algorithm improved by a selection mechanism. *IEEE Transactions on Fuzzy Systems, 28*, 3265–3275.

Gao, K. Z., Suganthan, P. N., Chua, T. J., Chong, C. S., Cai, T. X., & Pan, Q. K. (2015a). A two-stage artificial bee colony algorithm scheduling flexible job-shop scheduling problem with new job insertion. *Expert Systems with Applications, 42*, 7652–7663.

Gao, K. Z., Suganthan, P. N., Pan, Q. K., & Tasgetiren, M. F. (2015b). An effective discrete harmony search algorithm for flexible job shop scheduling problem with fuzzy processing time. *International Journal of Production Research, 53*, 5896–5911.

Gong, W., Liao, Z., Mi, X., Wang, L., & Guo, Y. (2021). Nonlinear equations solving with intelligent optimization algorithms: A survey. *Complex System Modeling and Simulation, 1*, 15–32. https://doi.org/10.1016/j.cie.2021.107290. URL: http://qikan.cqvip.com/Qikan/Article/Detail?id=7105607097.

Gonzalez-Rodriguez, I., Puente, J., Palacios, J. J., & Vela, C. R. (2020). Multi-objective evolutionary algorithm for solving energy-aware fuzzy job shop problems. *Soft Computing*.

J. Palacios, J., Gonzalez-Rodriguez, I., Vela, C.R., & Puente, J. (2017). Robust multiobjective optimisation for fuzzy job shop problems. Applied Soft Computing, 56, 604–616.

Jia, Z., Yan, J., Leung, J. Y. T., Li, K., & Chen, H. (2019). Ant colony optimization algorithm for scheduling jobs with fuzzy processing time on parallel batch machines with different capacities. *Applied Soft Computing, 75*, 548–561.

Lei, D. (2010). A genetic algorithm for flexible job shop scheduling with fuzzy processing time. *International Journal of Production Research, 48*, 2995–3013.

Lei, D. (2012). Co-evolutionary genetic algorithm for fuzzy flexible job shop scheduling. *Applied Soft Computing, 12*, 2237–2245.

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

Lin, J. (2015). A hybrid biogeography-based optimization for the fuzzy flexible job-shop scheduling problem. *Knowledge-Based Systems, 78*, 59–74.

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

Liu, H., Gu, F., & Zhang, Q. (2014). Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems. *IEEE Transactions on Evolutionary Computation, 18*, 450–455.

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

Pan, Z., Lei, D., & Wang, L. (2021). 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). doi:10.1109/TSMC.2021.3120702.

Pavlov, A.A., Misura, E.B., Melnikov, O.V., & Mukha, I.P. (2019). Np-hard scheduling problems in planning process automation in discrete systems of certain classes. In Z. Hu, S. Petoukhov, I. Dychka, & M. He (Eds.), Advances in Computer Science for Engineering and Education (pp. 429–436). Cham.

Qin, A. K., Huang, V. L., & Suganthan, P. N. (2009). Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Transactions on Evolutionary Computation, 13*, 398–417.

Sakawa, M., & Kubota, R. (2000). Fuzzy programming for multiobjective job shop scheduling with fuzzy processing time and fuzzy duedate through genetic algorithms. *European Journal of Operational Research, 120*, 393–407.

Sakawa, M., & Mori, T. (1999). An efficient genetic algorithm for job-shop scheduling problems with fuzzy processing time and fuzzy duedate. *Computers & Industrial Engineering, 36*, 325–341.

Saracoglu, l., & Suer, G.A. (2018). Multi-objective fuzzy flow shop scheduling model in a manufacturing company. Procedia Manufacturing, 17, 214–221.

Shaheed, I. M., Shukor, S. A., & Abdullah, S. (2018). Population initialisation methods for fuzzy job-shop scheduling problems: Issues and future trends. *International Journal on Advanced Science, Engineering and Information Technology, 8*, 1820–1828.

Sun, L., Lin, L., Gen, M., & Li, H. (2019). A hybrid cooperative coevolution algorithm for fuzzy flexible job shop scheduling. *IEEE Transactions on Fuzzy Systems, 27*, 1008–1022.

Van Nostrand, R. C. (2002). Design of experiments using the taguchi approach: 16 steps to product and process improvement. *Technometrics, 44*, 289–289.

Wang, B., Xie, H., Xia, X., & Zhang, X. (2019). 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, 27*, 1075–1084.

Wang, C., Tian, N., Ji, Z., & Wang, Y. (2017). Multi-objective fuzzy flexible job shop scheduling using memetic algorithm. *Journal of Statistical Computation and Simulation, 87*, 2828–2846.

Wang, S., Gong, M., Wu, Y., & Zhang, M. (2020). Multi-objective optimization for location-based and preferences-aware recommendation. *Information Sciences, 513*, 614–626. https://doi.org/10.1016/j.ins.2019.11.028

While, L., Hingston, P., Barone, L., & Huband, S. (2006). A faster algorithm for calculating hypervolume. *IEEE Transactions on Evolutionary Computation, 10*, 29–38.

Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*, 67–82.

Xu, Y., Wang, L., Wang, S.-Y., & Liu, M. (2015). An effective teaching-learning-based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time. *Neurocomputing, 148*, 260–268.

Yuguang, Z., Fan, Y., & Feng, L. (2019). Solving multi-objective fuzzy flexible job shop scheduling problem using mabc algorithm. *Journal of Intelligent and Fuzzy Systems, 36*, 1455–1473.

Zhang, Q., & Hui, L. (2007). Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation, 11*, 712–731.