Swarm and Evolutionary Computation Base Data xxx (2017) 1–17

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation Base Data

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

A decomposition-based multi-objective evolutionary algorithm with quality indicator

Jianping Luo [a,b,*], Yun Yang [a,b], Xia Li [a], Qiqi Liu [a], Minrong Chen [c], Kaizhou Gao [d]

[a] College of Information Engineering, Shenzhen University, Shenzhen 518060, PR China
[b] Shenzhen Key Laboratory of Media Security, Shenzhen 518060, PR China
[c] South China Normal University, Guangzhou 510631, PR China
[d] School of Electrical and Electronic Engineering, Nanyang Technological University, 639798, Singapore

ARTICLE INFO

IndexTerms:
Evolutionary computations
Multi-objective optimization
Algorithm diversity
Decomposition
Indicator-based

ABSTRACT

The issue of integrating preference information into multi-objective optimization is considered, and a multi-objective framework based on decomposition and preference information, called indicator-based MOEA/D (IBMOEA/D), is presented in this study to handle the multi-objective optimization problems more effectively. The proposed algorithm uses a decomposition-based strategy for evolving its working population, where each individual represents a subproblem, and utilizes a binary quality indicator-based selection for maintaining the external population. Information obtained from the quality improvement of individuals is used to determine which subproblem should be invested at each generation by a power law distribution probability. Thus, the indicator-based selection and the decomposition strategy can complement each other. Through the experimental tests on seven many-objective optimization problems and one discrete combinatorial optimization problem, the proposed algorithm is revealed to perform better than several state-of-the-art multi-objective evolutionary algorithms. The effectiveness of the proposed algorithm is also analyzed in detail.

1. Introduction

Optimization of multiple conflicting objectives typically arise in the science and engineering fields. Researchers and practitioners formulate such problems as multi-objective optimization problems (MOPs) to minimize or maximize several conflicting objective functions simultaneously. Multi-objective evolutionary algorithms (MOEAs) are widely used to solve MOPs. As MOPs become more complicated, MOEAs encounter many difficulties. The most serious difficulties come from high-dimensional MOPs and diversity maintenance [1–6]. MOEAs contain two conflicting goals: (i) minimizing the distance between approximate Pareto front and the Pareto-optimal front, and (ii) maximizing the diversity within the approximate Pareto front. Most MOEAs, such as NSGA-II [7] and SPEA2 [8], are Pareto-dominance based MOEAs. Such MOEAs exhibit excellent performance on low-dimensional MOPs, but fail in many-objective optimization problems (MaOPs) (i.e., MOPs with more than three objectives). Generating a good approximate Pareto front in these algorithms is not always easy because the rate of non-dominated solutions increases with the number of objectives, which is difficult for the Pareto-based MOEAs to deal with.

The issue of integrating preference information into multi-objective optimization has been addressed by different researchers to handle the MOPs more effectively [9]. The most well-known MOEA based on this strategy is indicator-based evolutionary algorithm (IBEA) [10]. The main idea of IBEA is to formalize preferences in terms of continuous generalizations of the dominance relation, which leads to a simple algorithmic concept. This formalized preference, usually called indicator, can reflect the quality of the solutions in terms of both convergence and diversity. IBEA has attracted the attention of many researchers, and several new MOEAs based on indicators have been developed [11]. In Ref. [12], a multi-objective shuffled frog leaping algorithm based on an indicator was proposed to solve MaOPs. In the present study, a multi-objective framework based on decomposition and preference information, called indicator-based MOEA/D (IBMOEA/D), is proposed. The hypervolume [13] has frequently been used as an indicator function in IBEAs because it has nice features as a performance measure of solution sets in comparison with other measures [14–16,39]. However, it is difficult to be utilized directly because the computation load increases exponentially with the number of objectives. In Ref. [17], an idea of

* Corresponding author. College of Information Engineering, Shenzhen University, Shenzhen 518060, PR China.
E-mail address: ljp@szu.edu.cn (J. Luo).

https://doi.org/10.1016/j.swevo.2017.11.004
Received 8 May 2017; Received in revised form 28 October 2017; Accepted 14 November 2017
Available online xxxx

approximating the hypervolume using a number of achievement scalarizing functions with uniformly distributed weight vectors was proposed. In Ref. [36], a simple and fast hypervolume indicator-based MOEA (FV-MOEA) is proposed to quickly update the exact hypervolume contributions of different solutions. In Ref. [18], a methodology based on Monte Carlo sampling to estimate the hypervolume contribution of single solutions regarding a specific Pareto set approximation was presented. In this paper, the binary additive ε-indicator [10], which has the properties of simplicity and low computational complexity, is used in IBMOEA/D.

An excellent MOEA based on objective decomposition, called MOEA/D, has recently been proposed by Zhang et al. [19]. Several modified MOEA/D algorithms can be found in the literature [20–22]. In MOEA/D, the MOP is decomposed into a number of scalar optimization subproblems. Different solutions in the current population are associated with different subproblems. The "diversity" among these subproblems will naturally result in diversity in the population. When the decomposition method and the weight vectors are properly selected, such that the optimal solutions to the resultant subproblems are evenly distributed along the *PF*, MOEA/D will have a good chance of search a uniform distribution of Pareto solutions if it optimizes all subproblems well. The original MOEA/D assigns the same computational resources to all the subproblems, however, it may be not the case in certain applications. Suppose that N subproblems are considered in MOEA/D and their weight (or direction) vectors are uniformly distributed to a certain extent. If the MOP is continuous and its *PF* is continuous and convex, then the optimal solutions of these subproblems can constitute a good approximation to the *PF* when the number of weight vectors is sufficiently large. However, in certain cases, such as discrete problems, each subproblem in MOEA/D may have varied contributions to the search process at different search stages [23]. Recently, various combinations of MOEA/D and domination-based techniques have been investigated to address this problem [24–26]. In these combinations, both the decomposition approach and the domination-based approach are used to select good solutions. The main ideas of these studies are that the different subproblems should not be allocated the same amounts of computational resources. The present study attempts to propose a novel mechanism of allocating computational resources to each subproblem dynamically. The main idea is that the more the quality improvement of an individual (subproblem) over several generations is, the more the computational resources assigned to this individual.

Meanwhile, the archive used to save the non-dominated solutions is necessary to improve the final solutions. Many works have been conducted for this research field. Compared with single-objective optimization, where the best solution is always copied into the next population, the incorporation of elitism in MOEAs is substantially more complex. The concept of maintaining an external archive of non-dominated solutions in the evolutionary process is often used. The main objective of the external archive is to keep a historical record of the non-dominated solutions found along the search process. This external archive provides the elitist mechanism for MOEAs. Notably, the archive size is fixed in MOEAs. When the number of non-dominated solutions is more than the maximum size, deleting the inferior non-dominated solutions (i.e., the least useful information), is an important task for the MOEAs. Zitzler et al. [27] showed that elitism helps in achieving better convergence in MOEAs. Among the existing elitist MOEAs, NSGA-II of Deb et al. [7], SPEA2 of Zitzler et al. [8], and Pareto-archived PAES of Knowles et al. [28] have been focused on. In Ref. [23], an external population (EP) management scheme based on NSGA-II selection was also presented. The non-dominated sorting and crowding distance assignment in NSGA-II [7] can effectively select representative solutions of good quality in the case of two or three objectives. However, as discussed previously, NSGA-II selection is ineffective in the case of MaOPs. In the present study, an EP management scheme based on an indicator is used and merged into MOEA/D to store the non-dominated solutions.

Note that our approach differs from literature [11]. Our framework is based on decomposition, and EP based on indicator is used to store the non-dominated solutions in our approach. However, the archive is used to generate new solutions in literature [11]. Also, the proposed algorithm in this paper differs from the literature [23] and [26]. The main differences are as follows: In the External Population (EP), we use an indicator-based approach to evaluate the quality of the individual. It is different from literature [23] in which the method for updating EP is based on NSGA-II selection, and the ways to update the EP are completely different. At the same time, the ways to guide the search of the population are different, we apply a power law distribution probability to select the subproblem to update according to the improvement of the quality of the individual based on the binary additive ε-indicator $I_{\varepsilon+}$ in population, and the EP is not used for guiding. However, in Ref. [23], the EP is used for guiding the search, which is different from our approach.

The major contributions of this work are as follows:

1) A new indicator-based MOEA/D (IBMOEA/D) algorithm, which integrates preference information into multi-objective search, is proposed. The new algorithm is effective to handle MOPs.
2) An archive management scheme based on an indicator is used and merged into MOEA/D to maintain non-dominated solutions found in the evolutionary process. The indicator-based selection, which follows the elitist mechanism, differs from the NSGA-II selection based on non-dominated sorting and crowding distance assignment.
3) The edge effect of the ε-indicator is analyzed in this paper. Meanwhile, we propose an approach to reduce the edge effect.
4) A novel computational resource assignment scheme is developed for each subproblem in the improved MOEA/D. In the original MOEA/D, each subproblem receives the same amount of computational resources. In IBMOEA/D, the likelihood that a subproblem is selected for investment is determined based on its quality evaluated by the indicator.

The remainder of this paper is organized as follows: The second section briefly describes MOPs. The third section describes binary quality indicator. The fourth section introduces our IBMOEA/D algorithm. The fifth section presents the experimental results and further discusses of the algorithm. Finally, the sixth section draws the conclusions.

2. Description of multi-objective problems

An unconstrained m-objective optimization problem can be described by the following equation:

$$\begin{aligned} &\textit{Minimize}\ \ \mathrm{F}(\mathrm{x}) = (f_1(\mathrm{x}), f_2(\mathrm{x}), \ldots, f_m(\mathrm{x}))^T, \\ &\text{Subject to}\ \ \mathrm{x} \in \Omega \end{aligned} \tag{1}$$

where Ω is the decision (variable) space, x is a decision vector, $\mathrm{F}{:}\Omega \rightarrow \mathrm{R}^m$ consists of m real-valued objective functions, and R^m is the objective space. Thus, we aim to determine the points that yield the optimum

values for all m objective functions simultaneously. Pareto dominance is formally defined as follows:

A vector x_A is said to dominate x_B if $x_A, x_B \in \Omega$, $f_i(x_A) \le f_i(x_B)$ $(i \in \{1, 2, ..., m\})$, and there is at least one $j \in \{1, 2, ..., m\}$, such that $f_j(x_A) < f_j(x_B)$, written as $x_A \prec x_B$.
x' is said to be a non-dominated solution, or a Pareto optimal solution, if x' $\in \Omega$ and there are no other solutions that dominate x' in Ω.

The set of all the Pareto optimal solutions is called the *Pareto set* (*PS*). The vectors that correspond to the solutions included in the *PS* are non-dominated.

When all non-dominated solutions are plotted in the objective function space, the non-dominated vectors are collectively known as the *Pareto front (PF)*. Formally,

$$PF := \{(f_1(x), f_2(x), \ldots, f_m(x)) | x \in PS\}$$

Finding an analytical expression that defines the *PF* of a problem is generally impossible. Thus, the most common way to derive the *PF* is to compute a sufficient number of points in the feasible region and then filter out the non-dominated vectors from them.

3. Fitness evaluation based on a binary quality indicator

In general, a quality indicator is a function that maps N Pareto set approximations to a real number. For example, the hypervolume indicator based on the hypervolume concept can be used to evaluate the quality of a set. However, the computational complexity of the hypervolume indicator is very high because the calculation of the hypervolume for a set is a NP-hard problem [29,30]. In this study, we use the binary additive ε-indicator $I_{\varepsilon+}$ [10], which can be used to compare the quality of two Pareto set approximations relative to each other, to guide the search for the evolutionary process of our algorithm. For two sets A and B,$I_{\varepsilon+}(A, B)$ can be defined as follows:

$$I_{\varepsilon+}(A, B) = \underset{\varepsilon}{\operatorname{argmin}}\{\forall y \in B\ \exists x \in A : f_i(x) - \varepsilon \le f_i(y) \ \text{ for } \ i \in \{1, 2, \ldots, m\}\} \quad (2)$$

The binary additive indicator $I_{\varepsilon+}$ gives the minimum distance by which a Pareto set approximation needs to translate or can be translated in each dimension in the objective space, such that another approximation is weakly dominated. $I_{\varepsilon+}$ represents a natural extension of the Pareto dominance relation, and therefore, can be directly used to calculate the fitness of and individual, which is similar to the common Pareto-based fitness assignment schemes. Notably, the $I_{\varepsilon+}$ indicator is compliant with Pareto dominance preserving [10], i.e.,

$$x \prec y \Rightarrow I((\{y\}, \{x\}) > I(\{x\}, \{y\})). \quad (3)$$

Fig. 1 illustrates the two binary quality indicators between two solutions.

Assume that population P is the sample of the decision space Ω. To evaluate the population member, fitness $F_I(x)$ of the population member x is define as follows:

$$F_I(x) = \sum_{y \in P \setminus x} J(\{x\}, \{y\}), \quad (4)$$

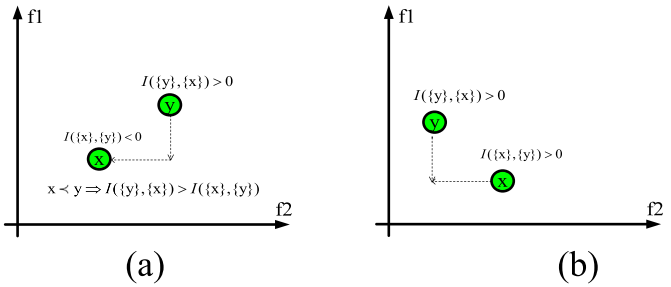

Fig. 1. Illustration of the binary quality indicator $I_{\varepsilon+}$.

$$J(\{x\}, \{y\}) = -\exp(-I(\{y\}, \{x\})/(c \times \nu)), \quad (5)$$

where $c = \max_{x,y \in P} |I(x, y)|$, and ν is a scaling factor and it is set to 0.01 in this study; this value is suitable according to our experiments. This definition of fitness value $F_I(x)$, which is to be maximized, is a measure for the "loss in quality" if x is removed from the population. Thus, we can prove that $x \prec y \Rightarrow F_I(x) > F_I(y)$ [10].

4. IBMOEA/D framework

In this section, we propose a simple yet efficient indicator-based MOEA/D framework (IBMOEA/D). Similar to other MOEA/D variants, the proposed algorithm requires a decomposition approach for converting the problem of approximation of the *PF* into a number of scalar optimization problems. The Tchebycheff approach, weight vector based on a penalized distance measure or weighted sum function approach [19] are applied in our algorithm. For instance, for the Tchebycheff approach, the subproblems can be defined as follows:

$$\text{minimize} \quad g^{te}(x | \lambda^i, z^*) = \max_{1 \le j \le m} \left\{ \lambda_j^i \left| f_j(x) - z_j^* \right| \right\}$$

Subject to. $x \in \Omega$
This framework requires the following parameters:

- N: the number of subproblems.
- $X = (x^1, \ldots, x^N)$: a population of N solutions, where x^i is the current solution to the $i$thsubproblem.
- $p = (p^1, \ldots, p^N)$: the probability that subproblem i should be invested.
- $z^* = (z_1^*, \ldots, z_m^*)$: an estimated ideal point vector.
- $\lambda = (\lambda^1, \ldots, \lambda^N)$: a uniform distribution of N weight vectors.
- T: the size of the neighborhood of each subproblem.
- $B^i \subset \{1, \ldots, N\}$: the index set of the neighboring subproblem of subproblem i, where $|B^i| = T$.
- *gen*: current generation number
- L: over L generations, the probability vector p should be updated.
- A stopping criterion.
- EP: an external population used to store the non-dominated solutions found along the search process.
- M: the number of solutions in EP, and its maximum value is $2N$.
- $R_{Old} = (R_{Old}^1, \ldots, R_{Old}^N)$: the fitness sorting number vector for the subproblems.

Algorithm 1: IBMOEA/D

Step 1: Initialization

a) Initialize λ, and decompose the MOP into N subproblems associated with λ.

b) Compute the Euclidean distance between any two weight vectors and obtain T closest weight vectors to each weight vector. For each $i = 1,...,N$, set $B^i = \{i_1,...,i_T\}$, where $\lambda^{i_1},...,\lambda^{i_T}$ are the T closest weight vectors to λ^i.

c) Compute the included angle between any two weight vectors and obtain the smallest included angle for each weight vector. For each $i = 1,...,N$, A^i is the smallest included angle for λ^i

d) Generate an initial population $\mathrm{X} = (\mathrm{x}^1,...,\mathrm{x}^N)$ randomly. Calculate $F(\mathrm{x}^i)$, where $i \in \{1,...,N\}$.

e) Set $\mathrm{EP} = \mathrm{X}$, $gen = 0$.

f) Initialize the ideal point vector z^*, $z_k^* = \min(f_k(\mathrm{x}^i)),\ i \in \{1,...,N\}$.

g) Initialize the probability vector p, set $p^i = \frac{1}{N},\ i \in \{1,...,N\}$.

h) Initialize the sorting number vector R_{Old}, set $R_{Old}^i = 0,\ i \in \{1,...,N\}$.

Step 2: New solution generation

For $j = 1$ **to** N **do**

a) Select subproblem i for the search based on the probability vector $\mathrm{p} = (p^1, p^2,..., p^N)$.

b) Set the mating pool:

$$Q = \begin{cases} B^i & \text{if } rand() < \delta \\ \{1,...,N\} & \text{otherwise} \end{cases}.$$

c) Randomly select two indexes k and l from Q.

d) Apply the DE operator [5] (for continuous problems) and/or the genetic operators (for discrete problems) on x^k and x^l to generate y for subproblem i.

e) If y is out of the range of Ω, then it is reset using a random solution.

f) Calculate the objective value vector $F(\mathrm{y}) = (f_1(\mathrm{y}),..., f_m(\mathrm{y}))$.

g) Update the idea point z^*, for each $k = 1,...,m$, $z_k^* = \min(z_k^*, f_k(\mathrm{y}))$.

h) Update the solutions, as follows:

Set $r = 0$.

For $k = 1$ **to** $sizeof(Q)$ **do**

(1) Randomly pick an index l from Q.

(2) If $(g^{te}(\mathrm{y} \mid \lambda^l, \mathrm{z}^*) \le g^{te}(\mathrm{x}^l \mid \lambda^l, \mathrm{z}^*))$, then $\mathrm{x}^l = \mathrm{y}$.

(3) Delete l from Q.

(4) $r = r + 1$, *if* $(r > n_c)$, then break.

End For

End For

Step 3: Update the generation number $gen = gen + 1$.

Step 4: Update the EP (see Algorithm 2)

Step 5: Update the probability vector P **(see Algorithm 4)**

Step 6: Termination If the stopping criteria are satisfied, then terminate the algorithm and output EP. Otherwise, go to Step 2.

Algorithm 2: Update the EP

a) Merge X with EP to obtain EP = X ∪ EP.
b) Select all the non-dominated solutions from EP, and store these solutions in S.
c) Clear EP.
d) Set $N' = sizeof(S)$.
If $(N' \le 2N)$, **then** copy S into EP, set $M = N'$ and
Return.
e) Select solutions for each direction vector, store these solutions in EP:
[*M*, S, EP] = Select_DV(S, EP) **(See Algorithm 3)**
f) Set $N' = sizeof(S)$.
g) Calculate $J(\{x^i\},\{x^j\}),\ x^i, x^j \in S,\ i, j \in \{1,...,N'\}$ using Eq. (5).
h) Calculate the fitness value $F_I(x^i),\ x^i \in S$ using Eq. (4).
i) Store the high quality solutions in EP based on indicator:
For l =1 **to** $N'-(2N-M)$ **do**
$d = \arg\min_i \{F_I(x^i),\ x^i \in S\}$.
Set $F_I(x^d)$=0.
For $j = 1$ **to** N' **do**
If $(F_I(x^j) \neq 0)$
then $F_I(x^j) = F_I(x^j) - J(\{x^d\},\{x^j\}), x^j \in S$
End For
End For
For i =1 **to** N' **do**
If $(F_I(x^i) \neq 0)$, then insert x^i into EP, $x^i \in S$
End For
Set $M = 2N$.

Algorithm 3: Select solutions for each direction vector

[*M*, S, EP] = Select_DV(S, EP)

Set the mating pool:
$R = \{1,...,N\}$.
For $j = 1$ **to** N **do**
a) Set $N' = sizeof(S)$
b) Randomly select one index k from R.
c) For $i = 1$ **to** N' **do**
Calculate the included angle $\beta(S(i), \lambda^k)$ between λ^k and the line connecting $S(i)$ to the origin of coordinate.
End For
d) $a \leftarrow \arg\min_{S(i)}(\beta(S(i), \lambda^k))$
e) If ($\beta(a, \lambda^k) < A^k$ **)**
Insert a into EP.
Remove a from S.
End If
f) Remove k from R.
End For
Set M = sizeof(EP)
Return *M*, S, EP

Algorithm 4: Update the probability vector

If ($\mathrm{mod}(gen, L) \neq 0$) //**over** L **generations, the probability vector P should be updated.**

Return ;

Else

a) Calculate the fitness value $F_I(\mathrm{x}^i)$, $\mathrm{x}^i \in \mathrm{X}$.

b) Sort $F_I(\mathrm{x}^i)$, $i = 1,...,N$, in descending order; record the fitness sorting number vector $\mathrm{R}_{New} = (R^1_{New},...,R^N_{New})$, where R^i_{New} is the sorting number for the subproblem i .

c) Record $\mathrm{R}_{Imp} = (R^1_{Imp},...,R^N_{Imp})$, where $R^i_{Imp} = R^i_{New} - R^i_{Old}$, $i = 1,...,N$.

d) Sort R_{Imp} in descending order according to R^i_{Imp} , $i = 1,...,N$.

e) Update the probability vector:

For $r = 1$ **to** N **do**

Derive the subproblem index i of the r th element in R_{Imp} , and update the probability, as follows:

$$p^i = \frac{r^{-\mu}}{1^{-\mu} + 2^{-\mu} + ... + r^{-\mu} + ... + N^{-\mu}}, \quad 1 \le r \le N, \mu \ge 0 \tag{6}$$

End For

f) Save the fitness sorting number vector: $R^i_{Old} = R^i_{New}$, $i = 1,...,N$.

End If

4.1. Maintain the EP

In multi-objective optimization, there is usually no sufficient preference information to guide the evolution process, such that determining whether one non-dominated solution is better than the other solutions is difficult. Therefore, we develop an EP to preserve the diversity and convergence of the non-dominated solutions found along the search process. The EP is maintained based on the $I_{\varepsilon+}$ indicator technique in this study. The detailed operation is shown in Algorithm 2 of IBMOEA/D. Each solution in the EP is assigned an indicator fitness value calculated using Eq. (4). All the solutions in EP are non-dominated solutions, and we prefer the solutions with better quality, i.e., the solutions with the larger fitness F_I value take precedence when stored in the EP.

The set EP is considered for this algorithm to hold the best non-dominated solutions. The maximum size of the EP, i.e., the maximum number of members which can be stored in the EP, is equal to 2*N* in this study, and *N*is the population size. In each generation, the population and EP are merged together; then, all the non-dominated solutions are selected from it and stored in S. If the size of S is less than 2*N*, then all non-dominated solutions are copied into the EP. Otherwise, firstly, the approach of reducing edge effect is applied (See Algorithm 2-(e)). Next, the solution *d* with the least fitness value is discarded from S, and fitness values of all solutions should be re-calculated in the remainder of S. According to the definition of the fitness value, the new fitness value for each solution can be obtained by subtracting the indicator value incurred by *d* (See Algorithm 2-(i)), such that computational complexity is not high. If the size of S is still larger than 2*N*, then the solution with the least fitness value in the remainder of S is discarded continually. The process is continued until the size of S is equal to 2*N*. Finally, all the solutions in S are copied into the EP.

Note that we delete the non-dominated solutions one by one in this approach. If we delete the $N' - N$ solutions simultaneously, then several continuous sections constructed by the non-dominated solutions may be deleted. Fig. 2 illustrates this description. In Fig. 2-(a), solutions $a \sim i$ are all non-dominated solutions, and solutions *c*, *d*, and *e* (section A) are crowded together. Therefore, the fitness values of these three solutions (*c*, *d*, and *e*) are likely to be less than the other solutions. We assume that we need to delete three solutions to comply with the capacity requirement of the EP. If we delete solutions only once, then solutions *c*, *d*, and *e* should be deleted simultaneously. As shown in Fig. 2-(b), the final distribution of the solutions (*PF*) is not uniform. By contrast, if we delete the three solutions one by one, then solution *d* with the least fitness value should be deleted first. At the same time, the fitness of the other solutions should be updated, i.e., $F_I(a) = F_I(a) - J(\{d\},\{a\})$, $F_I(b) = F_I(b) - J(\{d\},\{b\})$, and so on. Evidently, $J(\{d\},\{c\})$ and $J(\{d\},\{e\})$ are less than $J(\{d\},\{f\})$, such that the updated fitness value $F_I(f)$ may be less than $F_I(c)$ and $F_I(e)$. Therefore, solution *f* may be removed in the subsequent deletion. Similarly, solution *h* is deleted last. As shown in Fig. 2-(c), the final *PF* distributes more evenly

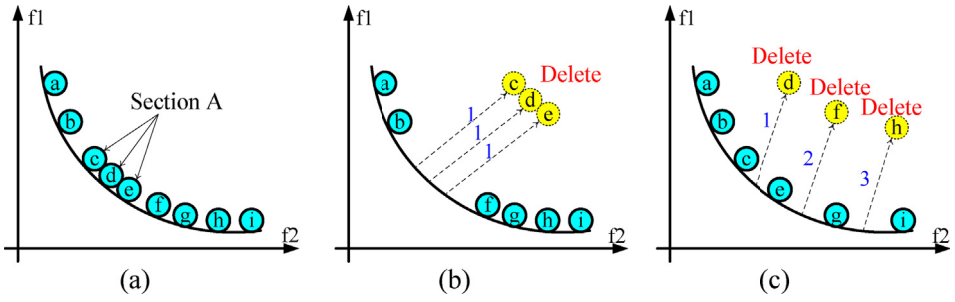

Fig. 2. Illustration of deleting non-dominated solutions.

in this approach.

The large number of objectives of MOPs increases the computational complexity of MOEAs. It is well-known that indicator-based approach is very computationally expensive for MOPs. The hypervolume has frequently been used as an indicator function. However, it is difficult to be applied directly because the computation load increases exponentially with the number of objectives. From Algorithm 2, it can be seen that the fitness computational complexity for all solutions is only $O((N')^2)$, and the computational complexity of selection for the high quality solutions in EP based on the indicator is $O(N^2)$, therefore, the total computational complexity based on the binary additive ε-indicator $I_{\varepsilon+}$ is not high in our algorithm.

4.2. Reduce the edge effect of ε-indicator $I_{\varepsilon+}$

The concept of the binary additive indicator is very concise. However, it can lead the final population to concentrate around the boundary of the front. In order to explain the problem more simple, $J(\{x\},\{y\})$ can be replaced by $I(\{y\},\{x\})$in Eq. (5), so Eq. (4) can be written as:

$$F_I(x) = \sum_{y \in P \backslash x} I(\{y\},\{x\})$$

Therefore, as shown in Fig. 3 (b), the fitness values of the non-dominated points a, b, c, d, e, f, and gare calculated as:

$$\begin{aligned} F_I(a) &= I(b,a) + I(c,a) + I(d,a) + I(e,a) + I(f,a) + I(g,a) \\ &= 1 + 2 + 3 + 4 + 5 + 6 = 21, \end{aligned}$$

$$\begin{aligned} F_I(b) &= I(a,b) + I(c,b) + I(d,b) + I(e,b) + I(f,b) + I(g,b) \\ &= 1 + 1 + 2 + 3 + 4 + 5 = 16, \end{aligned}$$

$F_I(c) = 13,\ F_I(d) = 12,\ F_I(e) = 13,\ F_I(f) = 16,$ and $F_I(g) = 21.$

Obviously, these non-dominated points are equally spaced, and their qualities should be the same. However, the calculated results are not uniform, and the fitness values of the edge points are larger than the fitness values of the internal points. So the edge effect of final population is generated using this binary additive indicator. It typically leads the final population to concentrate around the boundary of the front. Fig. 3 (a) is an example of the edge effect. In order to reduce this effect, similar to reference [38], we use a set of direction vectors to divide the objective space evenly in our algorithm. As shown in Algorithm 3, in each generation, for each direction vector, the nearest non-dominated solution is chosen to store in EP. Note that the distance between the solution and the

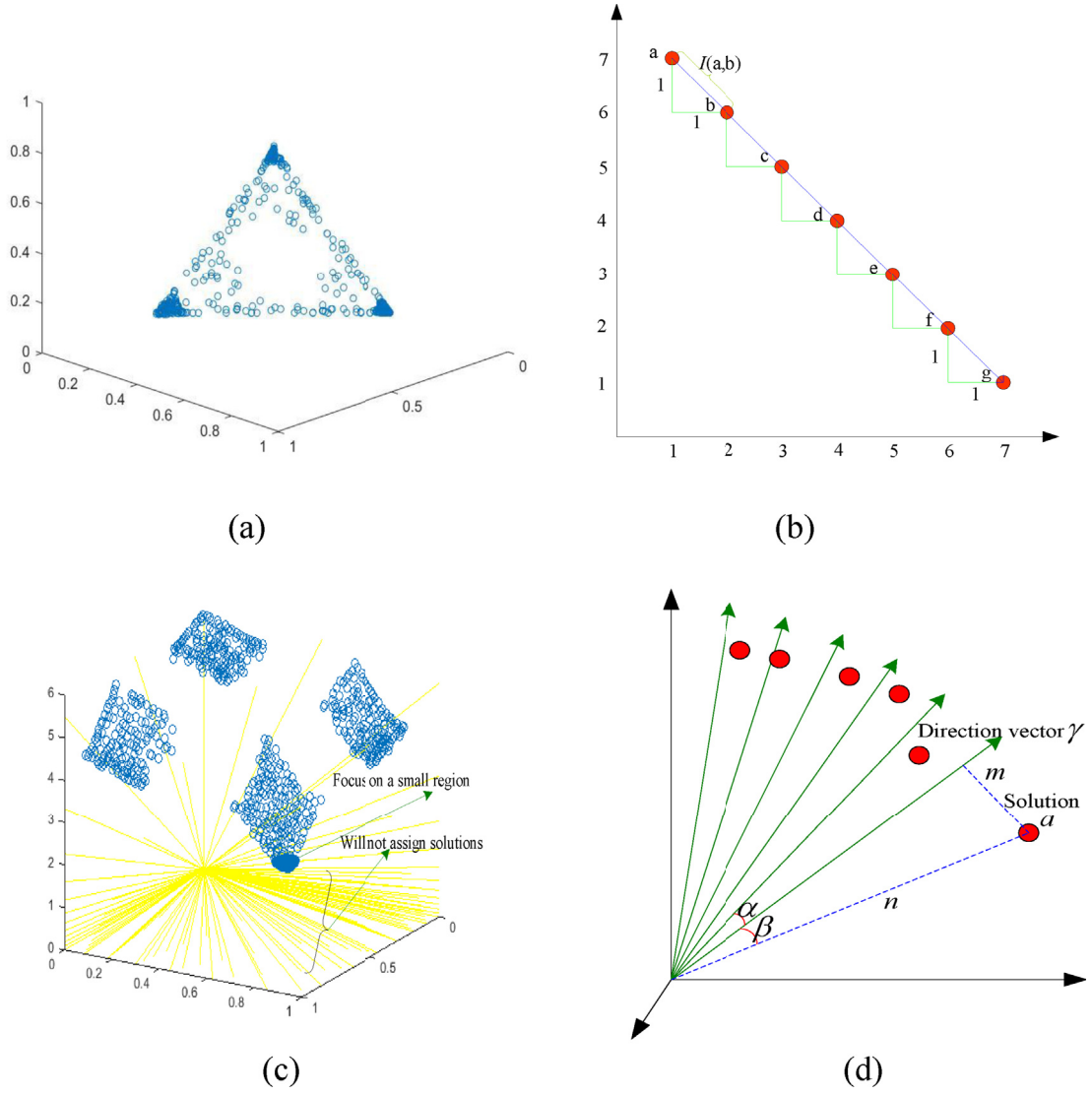

Fig. 3. (a) Example for edge effect. (b) Example of calculation of indicator. (c) Assign solution for each vector. (d) Calculation for included angle.

direction vector cannot exceed the maximum threshold, otherwise no solution will be allocated to this direction vector. We evaluate the distance by calculating the included angle between the direction vector and the line connecting the solution to the origin of coordinate. If the included angle is larger than the smallest included angle between the direction vector and the other direction vectors, the solution will not be assigned to the direction vector. In Fig. 3 (d), the included angle between the direction vector and the line connecting the solution to the origin of coordinate can be calculated as:

$$\beta = \arcsin\left(\frac{m}{n}\right)$$

If $\beta > \alpha$, the solution a will not be assigned to direction vector γ. This mechanism can avoid too many of solutions to be allocated to the invalid direction vectors. In Fig. 3 (c), if each direction vector is assigned a solutions, these solutions which are assigned to the right bottom direction vectors will focus on a very small region because these solutions are closest to those direction vectors. It may not well maintain the diversity of the population. In our algorithm, all these direction vectors are not assigned solutions. Note that the proposed approach differs from reference [38] which uses a predefined set of reference points to ensure diversity in obtained solutions.

4.3. Update the probability vector

Selecting the subproblem to update, i.e., the computational resource assignment for different subproblems, is an important task in the IBMOEA/D framework. In our approach, if a subproblem has been improved more over the last L generations, then it should have a higher probability to be improved over the next few generations. By using the previously presented probability vector, each subproblem has a probability to be selected and the subproblem that has been improved more over the last L generations should be allocated more computational resources again. For each L generation of the evolution process, all the solutions in the population are sorted according to their fitness values in descending order. The sorting number is used to calculate the probability that the corresponding subproblem should be invested. The increment of the sorting number of each subproblem over the last L generations, i.e., $R^i_{Imp} = R^i_{New} - R^i_{Old}$ in Algorithm 4, is used to estimate the improvement of quality for each subproblem in this study. The more the quality improvement of a subproblem is, the higher the probability that the subproblem would be selected to be invested. At the same time, a power law distribution with a fixed value of parameter μ is applied, as shown in Algorithm 4. For Eq. (6), where the rank r ranges from $r = 1$ for the subproblem with the most quality improvement to $r = N$ for the subproblem with the least quality improvement. The power-law exponent μ is an adjustable parameter. For $\mu = 0$, random selections are made, resulting in an even walk through the subproblem space. For $\mu = \infty$, only the extremal solution with the most quality improvement is always selected, such that obtaining an event *PF* is difficult. In this study, in accordance with our experience, the parameters $L$and μ are set as $L = 4$, $\mu = 0.4$, respectively.

5. Experimental studies and discussion

IBMOEA/D is compared with MOEA/D-PBI [19] [38], NSGA-III [38] and EAG-MOEA/D algorithms to analyze the performance of IBMOEA/D and understand its behavior. MOEA/D-PBI is a variant of MOEA/D in which the weight vector is based on a penalized distance measure. The performance of MOEA/D-PBI is better than MOEA/D with Tchebycheff decomposition for the MOPs with more than three objectives [38]. NSGA-III is a recent improvement of its predecessor NSGA-II, which uses predefined reference points to achieve diversity of the solution set. EAG-MOEA/D [23], which is also an improved edition based on MOEA/D, uses a domination-based external archive to guide the dynamic allocation of computational resources among subproblems. The external archive is also used to store the non-dominated solutions generated during evolution. EAG-MOEA/D has very good performance on the discrete combinatorial MOPs. This section conducts the following experimental works:

1) Comparison of IBMOEA/D with MOEA/D-PBI and NSGA-III on the many-objective continuous optimization problems (MaOPs). The DTLZ benchmark problems are considered. (Comparison of IBMOEA/D with EAG-MOEA/D and MOEA/D (MOEAD with Tchebycheff approach) on the MaOPs is in the supplementary materials.)
2) Comparison of IBMOEA/D with MOEA/D (MOEAD with weighted sum function), NSGA-II, and EAG-MOEA/D on the discrete combinatorial MOPs. The MNRP test instances [23] are considered.
3) Investigation of the contributions of the EP and the computational resource allocation strategy in IBMOEA/D.
4) Parametric sensitivity analysis for the proposed algorithm.

In our experiments, all the algorithms are implemented in VC6.0 and each algorithm is run independently 30 times on a test instance.

5.1. Test problems

1) The DTLZ test suit [31] including seven scalable problems DTLZ1-7 is employed to test the performance of the algorithms. This test suite has diverse characteristics such as non-convex, multimodal and disconnected Pareto fronts. All the problems can be scaled to any number of objectives (m) and decision variables ($l = m + k - 1$). In our experiment, we consider 3, 5, 8, 10, and 12 objectives for each problem and k is set to 5, 10 and 20 for DTLZ1, DTLZ2-6, and DTLZ7, respectively.
2) The MNRP test instances [23], which belong to the discrete combinatorial MOPs, have two objectives to optimize. One objective is to minimize the required cost and the other objective is to maximize the total satisfaction score. The total satisfaction score can be changed to minimize and normalize the range of objectives in [0, 1]. The MNRP with U customers and n requirements is denoted as *Cu–U/R–n* in our experiment. Eight randomly generated test instances are used in our studies, which include *Cu–30/R–300, Cu–50/R–200, Cu–50/R–500, Cu–80/R–800, Cu–100/R–1,000, Cu–160/R–1,600, and Cu–200/R–2000.*

5.2. Performance metrics

Several performance metrics in multiobjective optimization have been discussed in literature [37]. In our experiment, the following performance indexes are used.

Hypervolume Indicator (HV-metric) [33]:

We let $y^* = (y_1^*, \ldots, y_m^*)$ be a point in the objective space, which is dominated by any Pareto-optimal objective vectors. Hypervolume is a metric that evaluates the volume of an approximation set related to the reference point y^* in the objective space [33]. The formula is given in Eq. (7), where j is the number of non-dominated solutions in the approximation set, and $v_i$is the Hypervolume of the *i-th* solution related to the reference point. The reference point y^* is set by maximum values of the results of all the comparative algorithms for DTLZ instances, and $y^* = (0, 1)$ for MNRP test instances in this study.

$$Hypevolume = volume\left(\bigcup_{i=1}^{j} v_i\right) \tag{7}$$

Inverted Generational Distance (IGD-metric) [32]:

We let P^* be a set of well representative points along the *PF* (in the objective space). We let A be an approximation to the *PF*, and the average distance from P^* to A is defined as follows:

$$IGD(A, P^*) = \frac{\sum_{v \in P^*} d(v, A)}{|P^*|}, \tag{8}$$

where $d(v,A)$ is the minimum Euclidean distance between v and the points in A. If $|P^*|$ is sufficiently large to represent the PF well, $IGD(A,P^*)$ could measure both the diversity and convergence of A to a certain extent. A must be very close to the PF and must not miss any part of the entire PF to obtain a low value of $IGD(A,P^*)$.

Spread Metric Measurement (SPREAD-metric) [7]:

The Spread metric measures the extent of spread of the obtained non-dominated solutions. Obtaining a set of solutions that spans the entire Pareto-optimal region is desired. The Spread metric can be calculated as follows:

$$Spread = \frac{d_f + d_l + \sum_{i=1}^{N-1} |d_i - \overline{d}|}{d_f + d_l + (N-1)\overline{d}}, \tag{9}$$

where, $d_f$and $d_l$are the Euclidean distances between the extreme solutions and the boundary solutions of the obtained non-dominated set, and the Euclidean distance $d_i$between consecutive solutions in the obtained non-dominated set of solutions, respectively. The parameter $\overline{d}$ is the average of all distances d_i, $i = 1, 2, ..., (N-1)$, assuming that N solutions exist on the best non-dominated front. However, a good distribution would make all distances d_i equal to $\overline{d}$ and would make $d_f = d_l = 0$ (with existence of extreme solutions in the non-dominated set). Consequently, for the most widely and uniformly distributed set of non-dominated solutions, the numerator Spread would be zero.

In our experiments, we select approximately 3000 evenly distributed points in the true Pareto front for DTLZ1-7 with m objectives to calculate IGD and SPREAD. However, the true PF is unknown in the MNRP test instances. Thus, the true Pareto front for the MNRP test instances is the set of all non-dominated solutions obtained by all the algorithms in all runs. These metrics are computed by jMetal [34].

5.3. Parameter settings

The maximum number of function evaluations for all the algorithms is set to 300,000 as the stopping criterion.

The setting of N weight vector $(\lambda^1, ..., \lambda^N)$is related to a positive integer parameter H, which specifies the granularity or resolution of the weight vectors, as in Ref. [19]. Each individual weight takes a value from:

$$\left\{\frac{0}{H}, \frac{1}{H}, \cdots, \frac{H}{H}\right\}$$

Thus, the number of weight vectors can be calculated as:

$$N = C_{H+m-1}^{m-1}, \tag{10}$$

where m is the number of objective. The population size, which is equal to the number of weight vector for all the algorithms, cannot be arbitrarily specified. We set it at approximately 600 in all the algorithms, such that the parameter H can be obtained using Eq. (10).

For the MNRP instances, the weighted sum function [19] is used in MOEA/D, EAG-MOEA/D, and IBMOEA/D. For the DTLZ problems, the penalized distance measure and Tchebycheff approach are used for the algorithms, and the results based on Tchebycheff approach are in the supplementary file. The SBX and polynomial mutation [19] are used to generate the offspring for NSGA-II, MOEA/D, EAG-MOEA/D, MOEA/D-PBI, and IBMOEA/D in the experiments, and the crossover rate and mutation rate are set to 1.0 and $1/l$, respectively, where l is the number of decision variables. The neighborhood sizes are set as 20 for these algorithms. For the DTLZ instances, we do not utilize NSGA-II as a comparing algorithm, because of the poor performance on MaOPs. By contrast, for the MNRP, we employ NSGA-II as a comparing algorithm. The parameter settings are the same as that in MOEA/D. For NSGA-III, the population size and the fitness evaluation number are the same as the other compared algorithms, the other parameters are set following its original publication.

5.4. Comparison experiments on DTLZ problems

We compare IBMOEA/D with MOEA/D-PBI and NSGA-III on the DTLZ benchmark problems to analyze the behavior of our algorithm.

Tables 1–3 list the mean and standard deviation values of the HV, IGD and SPREAD metrics on the DTLZ1-7 problems with 3, 5, 8, 10, and 12 objectives to compare the three algorithms.

Tables 1 and 2 show that IBMOEA/D is almost the top algorithm for all the DTLZ problems. IBMOEA/D has the best HV and IGD values on all the DTLZ1-5 problems, such that IBMOEA/D has a stable performance on these test instances. Note that the three algorithms are very strictly consistent for the IGD and HV performance metrics on most of the instances except DTLZ7 with 12 objectives. For example, for all the DLTZ1-5 problems with different objectives, IBMOEA/D has the best HV and IGD values. NSGA-III performs better than IBMOEA/D for DTLZ 7 with 8 and 10 objectives; however, IBMOEA/D performs better for DTLZ 7 with 3 and 5 objectives. These results prove that the proposed algorithm is effective and efficient to address MaOPs.

Table 3 also shows that IBMOEA./D performs better than MOEA-PBI according to the SPREAD metric on most of the DTLZ instances. At the same time, IBMOEA/D performs better or performs similarly to NSGA-III except DTLZ5-7 with 8, 10 and 12 objectives. It is evident that the our algorithm can get a widely and uniformly distributed set of non-dominated solutions.

Table 1
Mean and standard deviation values of IGD obtained by NSGA-III, MOEA/D-PBI and IBMOEA/D on all the DTLZ1-7 test instances.

IGD	obj	NSGA-III	MOEA/D-PBI	IBMOEA/D
DTLZ1	3	0.00522(0.00030)	0.00515(0.00002)	**0.00442(0.00009)**
	5	0.02011(0.00403)	0.01816(0.00008)	**0.01689(0.00013)**
	8	0.03563(0.00058)	0.03331(0.00022)	**0.03177(0.00056)**
	10	0.04566(0.00066)	0.04065(0.00071)	**0.03896(0.00058)**
	12	0.08825(0.00101)	0.08286(0.00068)	**0.07691(0.00119)**
DTLZ2	3	0.00850(0.00017)	0.00829(0.00000)	**0.00636(0.00007)**
	5	0.02627(0.00126)	0.02360(0.00004)	**0.02242(0.00021)**
	8	0.06180(0.00097)	0.05160(0.00007)	**0.05051(0.00031)**
	10	0.08416(0.00101)	0.07062(0.00009)	**0.06802(0.00022)**
	12	0.16818(0.00081)	0.16276(0.00002)	**0.15441(0.00146)**
DTLZ3	3	0.00918(0.00042)	0.00878(0.00043)	**0.00680(0.00039)**
	5	0.07884(0.01892)	0.02855(0.00286)	**0.02544(0.00129)**
	8	0.09308(0.00880)	0.05864(0.00315)	**0.05531(0.00304)**
	10	0.11181(0.00565)	0.07618(0.00222)	**0.07282(0.00348)**
	12	0.17903(0.00477)	0.16521(0.00196)	**0.15356(0.05070)**
DTLZ4	3	0.00840(0.00009)	0.00829(0.00000)	**0.00634(0.00007)**
	5	0.02451(0.00028)	0.02336(0.00002)	**0.02217(0.00030)**
	8	0.05467(0.00041)	0.05085(0.00003)	**0.04837(0.00031)**
	10	0.07319(0.00048)	0.06974(0.00001)	**0.06721(0.00029)**
	12	0.16283(0.00018)	0.16153(0.00190)	**0.15607(0.00123)**
DTLZ5	3	0.00109(0.00010)	0.00571(0.00000)	**0.00020(0.00000)**
	5	0.03265(0.00198)	0.05231(0.00077)	**0.02448(0.00128)**
	8	0.23357(0.01255)	0.20151(0.00178)	**0.17509(0.00262)**
	10	0.28289(0.01292)	0.22589(0.00079)	**0.18542(0.00559)**
	12	0.15878(0.01616)	0.12771(0.00202)	**0.08794(0.00244)**
DTLZ6	3	**0.00147(0.00015)**	0.10940(0.03543)	0.10386(0.03713)
	5	0.39889(0.03058)	0.11289(0.00435)	**0.02813(0.00419)**
	8	2.72040(0.10276)	0.30538(0.00534)	**0.22887(0.00741)**
	10	3.32349(0.14495)	0.38536(0.00541)	**0.28000(0.00712)**
	12	1.60335(0.11544)	0.18901(0.00253)	**0.09025(0.00379)**
DTLZ7	3	0.01389(0.00026)	0.02713(0.00021)	**0.00721(0.00009)**
	5	0.10905(0.00382)	0.17986(0.00054)	**0.05049(0.00040)**
	8	**0.47816(0.08079)**	1.11369(0.03078)	0.93921(0.03233)
	10	**1.05309(0.16174)**	1.40174(0.00105)	1.18281(0.03294)
	12	3.44614(0.60664)	1.64623(0.01058)	**1.39711(0.04133)**

Table 2
Mean and standard deviation values of HV obtained by NSGA-III, MOEA/D-PBI and IBMOEA/D on all the DTLZ1-7 test instances.

HV	obj	NSGA-III	MOEA/D-PBI	IBMOEA/D
DTLZ1	3	1.14862(0.00089)	1.14884(0.00005)	**1.15128(0.00030)**
	5	1.58703(0.00272)	1.58853(0.00005)	**1.59036(0.00028)**
	8	2.14203(0.00004)	2.14195(0.00005)	**2.14246(0.00004)**
	10	2.59329(0.00003)	2.59244(0.00025)	**2.59331(0.00005)**
	12	3.13819(0.00002)	3.13701(0.00031)	**3.13898(0.00009)**
DTLZ2	3	0.78344(0.00041)	0.78386(0.00000)	**0.79016(0.00008)**
	5	1.35249(0.00045)	1.35515(0.00005)	**1.37095(0.00033)**
	8	2.04333(0.00067)	2.05185(0.00008)	**2.06525(0.00040)**
	10	2.53652(0.00064)	2.54528(0.00007)	**2.55344(0.00038)**
	12	3.09631(0.00040)	3.10035(0.00010)	**3.11361(0.00060)**
DTLZ3	3	0.78207(0.00086)	0.78285(0.00086)	**0.78899(0.00115)**
	5	1.26045(0.06508)	1.34875(0.00403)	**1.36314(0.00562)**
	8	2.01779(0.00838)	2.04571(0.00279)	**2.05673(0.00549)**
	10	2.51522(0.00446)	2.54097(0.00129)	**2.55132(0.00382)**
	12	3.08702(0.00435)	3.09820(0.00122)	**3.10251(0.23389)**
DTLZ4	3	0.78366(0.00021)	0.78386(0.00001)	**0.79021(0.00009)**
	5	1.35420(0.00029)	1.35550(0.00004)	**1.37171(0.00032)**
	8	2.04961(0.00040)	2.05283(0.00005)	**2.06679(0.00044)**
	10	2.54386(0.00035)	2.54632(0.00004)	**2.55413(0.00045)**
	12	3.10024(0.00014)	3.09810(0.00402)	**3.10618(0.00295)**
DTLZ5	3	0.44045(0.00019)	0.43092(0.00001)	**0.44220(0.00001)**
	5	1.44898(0.00578)	1.43866(0.00166)	**1.45264(0.00136)**
	8	1.91962(0.02043)	1.88986(0.00608)	**1.93511(0.00603)**
	10	2.36682(0.02638)	2.29885(0.01776)	**2.36817(0.00881)**
	12	2.81008(0.02891)	2.81704(0.00737)	**2.88424(0.00436)**
DTLZ6	3	**0.43963(0.00032)**	0.32526(0.03370)	0.33424(0.03633)
	5	1.48585(0.00572)	1.53991(0.00264)	**1.56505(0.00156)**
	8	1.83051(0.01810)	2.07940(0.00485)	**2.11648(0.00088)**
	10	2.20722(0.03663)	2.52599(0.00548)	**2.56721(0.00094)**
	12	2.77189(0.02882)	3.04131(0.00491)	**3.09524(0.00223)**
DTLZ7	3	0.44208(0.00024)	0.43405(0.00016)	**0.44579(0.00011)**
	5	0.52757(0.00287)	0.44159(0.00040)	**0.57863(0.00049)**
	8	**0.53693(0.00548)**	0.00055(0.00020)	0.00591(0.00202)
	10	**0.58470(0.01099)**	0.00002(0.00000)	0.00118(0.00058)
	12	**0.68289(0.01320)**	0.00000(0.00000)	0.00025(0.00024)

Table 3
Mean and standard deviation values of SPREAD obtained by NSGA-III, MOEA/D-PBI and IBMOEA/D on all the DTLZ1-7 test instances.

Spread	obj	NSGA-III	MOEA/D-PBI	IBMOEA/D
DTLZ1	3	0.63653(0.01635)	0.63998(0.01779)	**0.62105(0.01182)**
	5	0.41201(0.05818)	0.38528(0.00779)	**0.37109(0.01516)**
	8	0.27332(0.00636)	0.28093(0.00559)	**0.26117(0.00361)**
	10	0.23972(0.00730)	0.24377(0.00707)	**0.22305(0.00401)**
	12	0.21679(0.00842)	0.22643(0.00755)	**0.21301(0.00469)**
DTLZ2	3	0.65129(0.01938)	0.64702(0.01224)	**0.64163(0.01133)**
	5	0.37502(0.00808)	0.37740(0.00841)	**0.37287(0.00767)**
	8	0.23992(0.00732)	0.24544(0.00618)	**0.23206(0.00700)**
	10	0.20281(0.00586)	0.21036(0.00559)	**0.20161(0.00762)**
	12	**0.16539(0.00660)**	0.18110(0.01042)	0.17208(0.01317)
DTLZ3	3	0.65931(0.01358)	0.65137(0.01038)	**0.64228(0.08092)**
	5	0.63189(0.19439)	0.37732(0.00974)	**0.36141(0.00883)**
	8	0.24462(0.00804)	0.24561(0.00750)	**0.24309(0.00751)**
	10	0.20698(0.00450)	0.21063(0.00615)	**0.20223(0.00489)**
	12	0.17503(0.01252)	0.18414(0.01176)	**0.17180(0.01059)**
DTLZ4	3	0.65401(0.01193)	**0.64900(0.00957)**	0.65444(0.01163)
	5	**0.37145(0.00873)**	0.37915(0.00678)	0.37470(0.00674)
	8	0.23841(0.00644)	0.24479(0.00644)	**0.23108(0.00797)**
	10	**0.19744(0.00472)**	0.21184(0.00624)	0.20119(0.00710)
	12	0.16584(0.00646)	0.18997(0.01894)	**0.16054(0.01758)**
DTLZ5	3	0.93051(0.04163)	1.62883(0.03613)	**0.57055(0.12595)**
	5	0.61268(0.01891)	1.55977(0.01389)	**0.51105(0.01030)**
	8	**0.37273(0.01291)**	1.72985(0.01654)	0.45855(0.01493)
	10	**0.30603(0.01132)**	1.36610(0.06621)	0.49266(0.02715)
	12	**0.32357(0.01778)**	1.07057(0.02294)	0.62036(0.04495)
DTLZ6	3	1.18805(0.03521)	1.13201(0.09234)	**0.71938(0.11663)**
	5	**0.46500(0.01698)**	1.08861(0.13580)	0.69689(0.01406)
	8	**0.29551(0.00953)**	0.97538(0.05468)	0.56161(0.01451)
	10	**0.27258(0.00702)**	0.84490(0.06628)	0.52588(0.01329)
	12	**0.27917(0.02234)**	0.82703(0.04800)	0.53840(0.01523)
DTLZ7	3	**0.66177(0.01586)**	0.99980(0.01102)	0.66222(0.01533)
	5	0.52623(0.02020)	0.98717(0.01105)	**0.44769(0.00993)**
	8	**0.51389(0.02389)**	0.98670(0.06060)	0.80455(0.01723)
	10	**0.57764(0.02069)**	0.81242(0.02966)	0.81042(0.02184)
	12	**0.67386(0.07742)**	0.69719(0.02649)	0.82095(0.03078)

Fig. 4 shows the evolution of the average hypervolume values with the numbers of function evaluations in the compared algorithms on DTLZ1-7 with 8 objectives, where the minimum number of function evaluations is 30,000, and the maximum number of function evaluations is 300,000. IBMOEA/D and MOEA/D-PBI have the faster convergence speed and the better convergence precision than NSGA-III on most of the problems. At the same time, IBMOEA/D outperforms MOEA/D-PBI for these instances. In Ref. [35], the authors point out that MOEA/D can produce widely and uniformly distributed solution sets on low-dimensional objectives, however, it performs worse on DTLZ instances with high-dimensional objectives. The reason for it is likely to be due to inappropriate specifications of a priori weight vectors when the number of objectives is large. Literature [35] also claimed that MOEA/D is sensitive to parameters such as the number of weight vectors and the selection or replacement neighbor size. By contrast, in our study, the different subproblems are allocated different computational resources, and the improvement of quality of each subproblem over the last L generations is used to determinate the allocation probability for each subproblem. At the same time, the external population based on an indicator develops the strong searching capability to preserve the distribution and convergence, which is crucial for the algorithm to address MaOPs.

In summary, IBMOEA/D is the best-performing algorithm on the DTLZ problems, except for the DTLZ7 with 8 and 10 objectives, where NSGA-III has a better performance than IBMOEA/D. These confirm that the guides from the preference information based on indicator-selection significantly improve the performance of the algorithm.

5.5. Comparison experiments on MNRP problems

We compare IBMOEA/D with MOEA/D, NSGA-II, and EAG-MOEA/D on eight MNRP instances. IBMOEA/D and EAG-MOEA/D are the top 2 algorithms for all the MNRP test instances. MOEA/D produces widely but not uniformly distributed solutions. NSGA-II produces uniformly but not widely distributed solutions.

Table 4 lists the mean and standard deviation values of the HV, IGD, and SPREAD metrics on eight MNRP instances. IBMOEA/D outperforms MOEA/D and NSGA-II on all eight MNRP test instances in terms of the HV and IGD metrics. At the same time, IBMOEA/D slightly outperforms EAG-MOEA/D in terms of the HV and IGD metrics for all the MNRP test instances, except for *Cu–200/R–2000* in the HV values. However, EAG-MOEA/D has better SPREAD values than IBMOEA/D on all the MNRP test instances, except for *Cu–100/R–1000* and *Cu–160/R–1600*.

The average hypervolume values with the numbers of function evaluations of the four algorithms on the MNRP test instances in 30 independent runs are shown in Fig. 5. IBMOEA/D and EAG-MOEA/D are the top 2 algorithms for the convergence speed and the quality of the final solution sets. EAG-MOEA/D has faster convergence speed than IBMOEA/D on half of the MNRP instances, whereas IBMOEA/D has better convergence precision than EAG-MOEA/D on all the MNRP instances, followed by MOEA/D. Meanwhile, NSGA-II has the worst convergence precision.

In general, IBMOEA/D and EAG-MOEA/D are the two best-performing algorithms on the eight MNRP test instances. The reason for the good performance of EAG-MOEA/D on the MNRP test instances

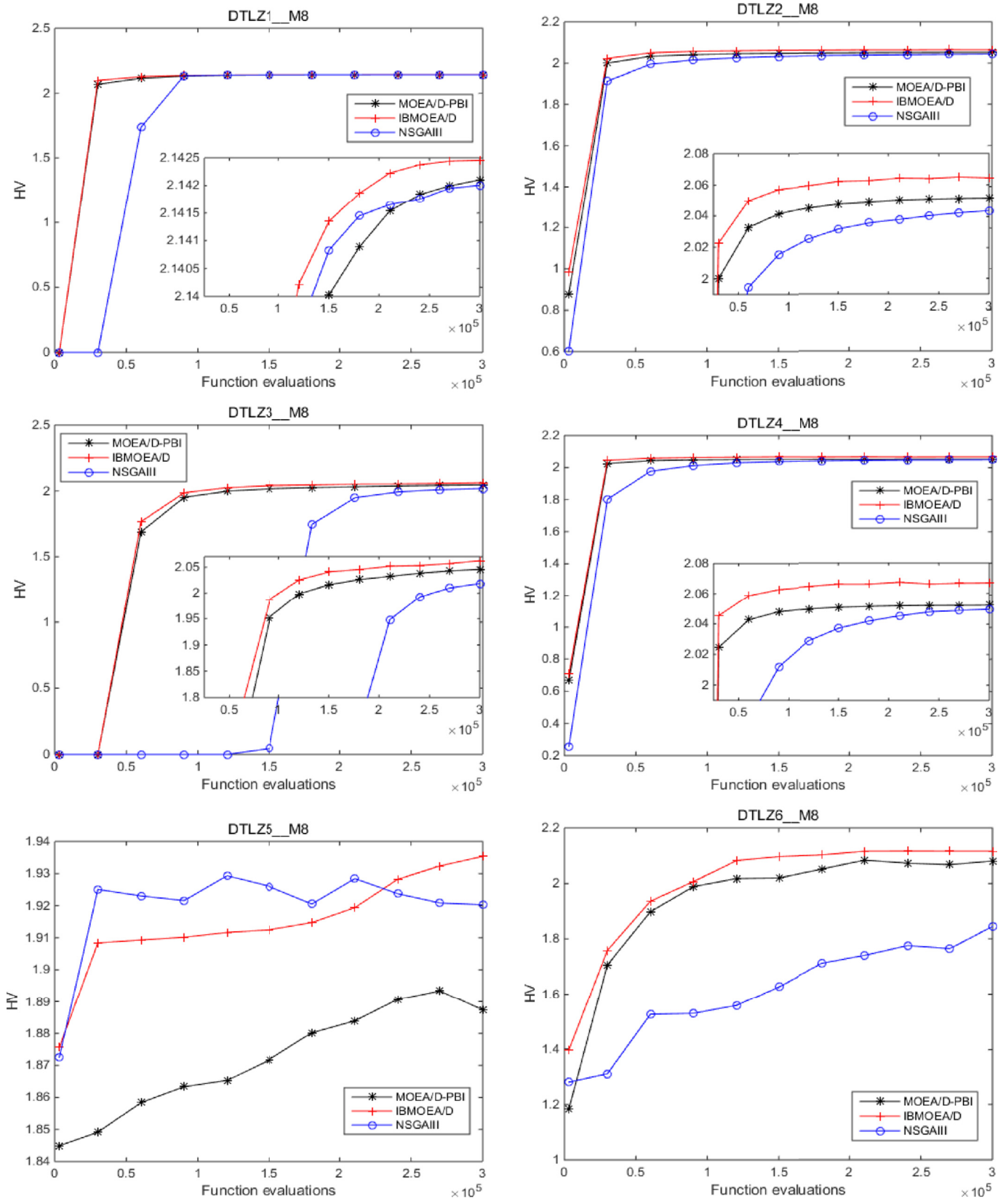

Fig. 4. Convergence graphs in terms of HV obtained by IBMOEA/D, NSGA-III, and MOEA/D-PBI on DTLZ instances.

Table 4
Mean and standard deviation values of HV, IGD and SPREAD obtained by IBMOEA/D, MOEA/D, NSGA-II and EAG-MOEA/D on eight MNRP instances.

Test instance		HV				IGD				SPREAD			
		IBMOEA/D	MOEAD	NSGA-II	EAG-MOEA/D	IBMOEA/D	MOEAD	NSGA-II	EAG-MOEA/D	IBMOEA/D	MOEAD	NSGA-II	EAG-MOEA/D
CU30-R300	mean	0.64875	0.64557	0.56324	0.6486	0.00002	0.00107	0.05581	0.00026	0.5766	1.43784	0.5639	0.2645
	std	0.00000	0.00011	0.00747	0.00001	0.00001	0.00007	0.0046	0.00002	0.00725	0.0036	0.04284	0.01081
CU50-R200	mean	**0.64266**	0.63945	0.59696	0.64264	**0**	0.00032	0.03309	0.00009	0.65362	1.51976	0.44944	**0.35933**
	std	0.00001	0.00008	0.00574	0.00001	0	0.00003	0.00417	0.00002	0.00365	0.00124	0.0288	0.00784
CU50-R500	mean	**0.64407**	0.64143	0.51084	0.64386	**0.00011**	0.00176	0.0844	0.00037	0.37173	1.27017	0.70159	**0.26094**
	std	0.00000	0.00016	0.00788	0.00002	0	0.00011	0.00499	0.00002	0.00459	0.01024	0.03232	0.00866
CU80-R800	mean	**0.64470**	0.64123	0.47008	0.64448	**0.00026**	0.0024	0.11126	0.0005	**0.2426**	1.37065	0.81391	0.27771
	std	0.00001	0.00021	0.00812	0.00002	0.00001	0.00014	0.00561	0.00001	0.00446	0.01105	0.03565	0.00769
CU100-R1000	mean	**0.65420**	0.6503	0.45295	0.654	**0.00036**	0.00266	0.12873	0.00056	0.31329	1.40751	0.87089	**0.28945**
	std	0.00001	0.00032	0.00758	0.00002	0	0.00019	0.00513	0.00001	0.00424	0.01313	0.03415	0.00929
CU120-R1200	mean	**0.64958**	0.64461	0.4355	0.64944	**0.00041**	0.00308	0.13884	0.00059	0.31916	1.48153	0.92056	**0.30872**
	std	0.00002	0.0005	0.00754	0.00002	0.00001	0.00031	0.00492	0.00001	0.00988	0.01925	0.0379	0.01117
CU160-R1600	mean	**0.64428**	0.63787	0.41373	0.64428	**0.00058**	0.00369	0.14812	0.00069	**0.32256**	1.55606	0.97845	0.32419
	std	0.00004	0.00055	0.00557	0.00005	0.00004	0.00023	0.00331	0.00003	0.0081	0.01865	0.02522	0.01106
CU200-R2000	mean	0.64615	0.63843	0.39791	**0.64616**	**0.00081**	0.00427	0.15839	0.00085	0.36145	1.5986	1.02891	**0.34579**
	std	0.00011	0.00073	0.00606	0.00009	0.00006	0.00036	0.00354	0.00004	0.0097	0.02292	0.02654	0.00877

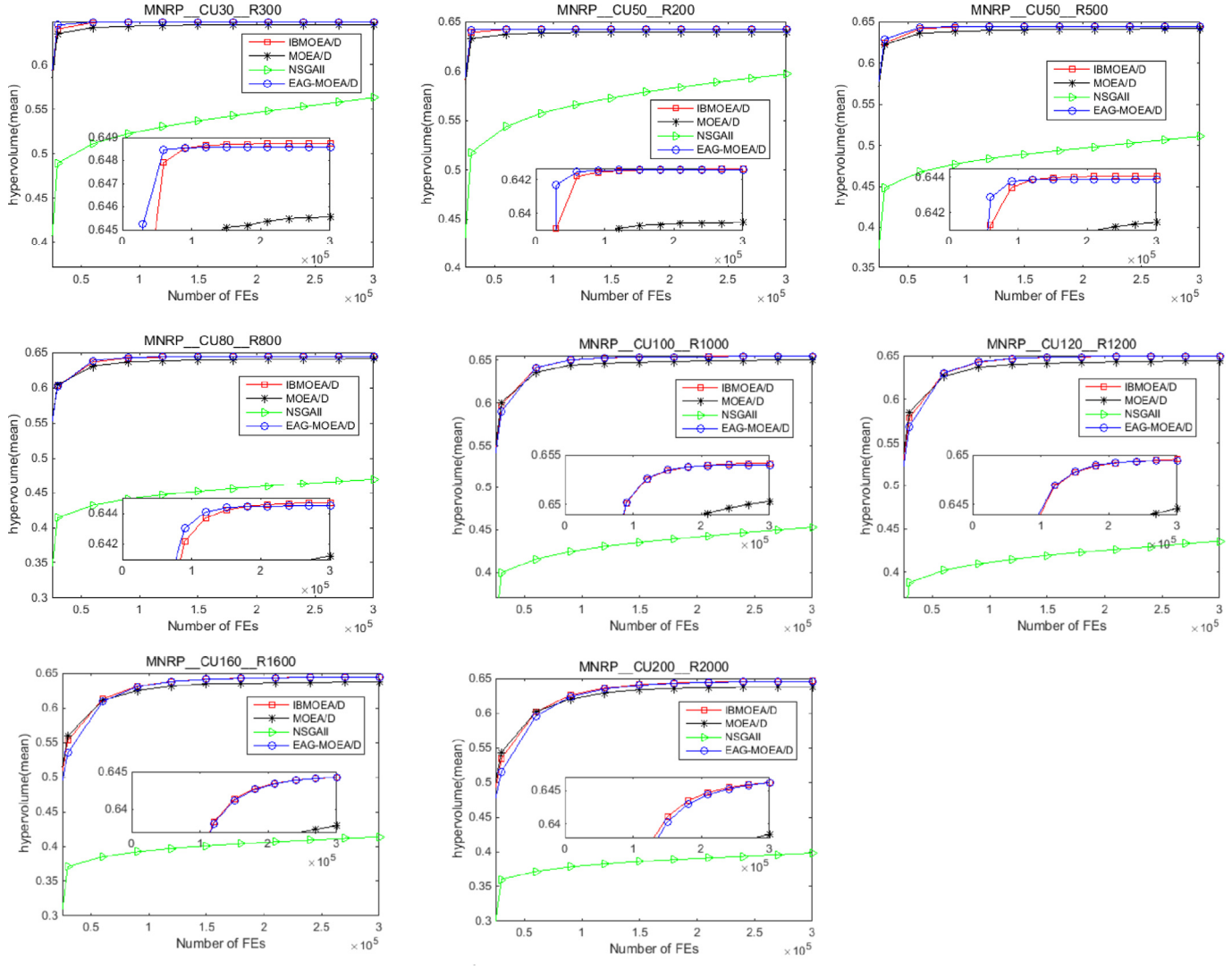

Fig. 5. Convergence graphs in term of hypervolume (mean) obtained by IBMOEA/D, MOEA/D, NSGA-II, and EAG-MOEA/D on MNRP instances.

may be the adoption of the external archive based on the domination-based sorting strategy. The MNRP test problems have only two objectives, and the domination-based sorting is suitable for such problems. From these results, we determined that IBMOEA/D is not only suitable for MaOPs but is also suitable for the discrete combinatorial MOPs.

5.6. Effects of EP (IBMOEA/D versus MOEA/D-RAI)

MOEA/D-RAI, a variant of MOEA/D, allocates computational resources based on indicator selection on each subproblem (as shown in Algorithm 3). On the basis of MOEA/D-RAI, IBMOEA/D uses the EP based on indicator selection to store the non-dominated solutions. The purpose of the comparison between IBMOEA/D and MOEA/D- RAI is to show the effects of EP in IBMOEA/D. In our IBMOEA/D versus MOEA/D-RAI experiment, the parameter settings of IBMOEA/D and MOEA/D-RAI are the same.

Table 5 compares IBMOEA/D and MOEA/D-RAI on the DTLZ and MNRP test instances in terms of the average HV, IGD and SPREAD metrics over 30 independent runs. At the same time, we apply the Wilcoxon nonparametric statistical test on the algorithms. The results are shown in Table 5. In the table, "+," "∼," and "−" denote that IBMOEA/D is better, equal, or worse than MOEA/D-RAI with a 95% significance level. Comparing IBMOEA/D and MOEA/D-RAI according to the IGD and HV metrics, IBMOEA/D performs similarly to or outperforms MOEA/D-RAI on all the problems. On the basis of these results, we can claim that the EP is better or no worse than the working population for the approximation of the true *PF* and has certain advantages for the evolution of algorithms on MaOPs and discrete combinatorial MOPs. Therefore, using the EP based on an indicator to store non-dominated solutions is reasonable.

5.7. Effects of computational resource allocation

5.7.1. IBMOEA/D versus MOEA/D-EP

MOEA/D-EP is a variant of IBMOEA/D in which no dynamic allocation of computational resources exists and EP only serves as the output in each generation (as shown in Algorithm 2). The purpose of this experiment is to assess the effects of the computational resource allocation scheme in IBMOEA/D proposed in this study. Thus, MOEA/D-EP and IBMOEA/D have the same parameter settings on all test instances. Table 6 presents the average values of HV, IGD and SPREAD of the IBMOEA/D and MOEA/D-EP algorithms on the DTLZ and MNRP test instances over 30 independent runs. Their Wilcoxon nonparametric statistical test results are also shown in Table 6.

In terms of HV, the table shows that IBMOEA/D performs better than MOEA/D-EP on all instances, except for DTLZ1 and DTLZ7 with 3

Table 5
Mean and standard deviation values of HV, IGD and SPREAD obtained by IBMOEA/D and MOEA/D-RAI on DTLZ and MNRP instances. the'-', '+', and '∼' denotes the corresponding value is worse, better or similar to that of MOEA/D-RAI according to the Wilcoxon test with the 5% significance level.

Instance		HV			IGD			GSPREAD		
–	obj#	IBMOEA/D	MOEA/D-RAI	T	IBMOEA/D	MOEA/D-RAI	T	IBMOEA/D	MOEA/D-RAI	T
DTLZ1	3	**1.15128(0.00030)**	1.15019(0.00045)	+	**0.00442(0.00009)**	0.00501(0.00004)	+	**0.62105(0.01182)**	0.63884(0.01852)	+
	5	**1.59036(0.00028)**	1.58905(0.00035)	+	**0.01689(0.00013)**	0.01796(0.00009)	+	**0.37109(0.01516)**	0.38296(0.00969)	∼
	8	**2.14246(0.00004)**	2.14220(0.00004)	+	**0.03177(0.00056)**	0.03285(0.00032)	+	**0.26117(0.00361)**	0.27619(0.00446)	+
	10	**2.59331(0.00005)**	2.59276(0.00018)	+	**0.03896(0.00058)**	0.03996(0.00069)	+	**0.22305(0.00401)**	0.24101(0.00690)	+
	12	**3.13898(0.00009)**	3.13749(0.00055)	+	**0.07691(0.00119)**	0.08128(0.00095)	+	**0.21301(0.00469)**	0.22108(0.00665)	+
DTLZ2	3	**0.79016(0.00008)**	0.78367(0.00004)	+	**0.00636(0.00007)**	0.00840(0.00000)	+	**0.64163(0.01133)**	0.64509(0.01356)	∼
	5	**1.37095(0.00033)**	1.36098(0.00003)	+	**0.02242(0.00021)**	0.02309(0.00008)	+	**0.37287(0.00767)**	0.37855(0.00692)	∼
	8	**2.06525(0.00040)**	2.05368(0.00061)	+	**0.05051(0.00031)**	0.05127(0.00016)	+	**0.23206(0.00700)**	0.24688(0.00504)	+
	10	**2.55344(0.00038)**	2.54929(0.00065)	+	**0.06802(0.00022)**	0.06982(0.00015)	+	**0.20161(0.00762)**	0.20952(0.00671)	+
	12	**3.11361(0.00060)**	3.10353(0.00022)	+	**0.15441(0.00146)**	0.15869(0.00092)	+	**0.17208(0.01317)**	0.18025(0.01108)	+
DTLZ3	3	**0.78899(0.00115)**	0.78278(0.00069)	+	**0.00680(0.00039)**	0.00893(0.00096)	+	**0.64228(0.08092)**	0.65188(0.02168)	∼
	5	**1.36314(0.00562)**	1.34670(0.00398)	+	**0.02544(0.00129)**	0.02910(0.00346)	+	**0.36141(0.00883)**	0.37819(0.00847)	+
	8	**2.05673(0.00549)**	2.04889(0.00189)	+	**0.05531(0.00304)**	0.05735(0.00289)	+	**0.24309(0.00751)**	0.24688(0.00845)	∼
	10	**2.55132(0.00382)**	2.54493(0.00290)	+	**0.07282(0.00348)**	0.07547(0.00290)	+	**0.20223(0.00489)**	0.20956(0.00558)	+
	12	**3.10251(0.23389)**	3.09936(0.00202)	+	**0.15356(0.05070)**	0.16002(0.00869)	+	**0.17180(0.01059)**	0.18414(0.01176)	+
DTLZ4	3	**0.79021(0.00009)**	0.78167(0.00002)	+	**0.00634(0.00007)**	0.00882(0.00009)	+	0.65444(0.01163)	**0.65065(0.00887)**	∼
	5	**1.37171(0.00032)**	1.36410(0.00039)	+	**0.02217(0.00030)**	0.02290(0.00018)	+	**0.37470(0.00674)**	0.37988(0.00398)	∼
	8	**2.06679(0.00044)**	2.05190(0.00004)	+	**0.04837(0.00031)**	0.05009(0.00012)	+	**0.23108(0.00797)**	0.24116(0.005984)	+
	10	**2.55413(0.00045)**	2.54980(0.00003)	+	**0.06721(0.00029)**	0.06896(0.00016)	+	**0.20119(0.00710)**	0.21003(0.00515)	+
	12	**3.10618(0.00295)**	3.10025(0.00322)	+	**0.15607(0.00123)**	0.15936(0.00127)	+	**0.16054(0.01758)**	0.17658(0.01764)	+
DTLZ5	3	**0.44220(0.00001)**	0.43377(0.00002)	+	**0.00020(0.00000)**	0.00227(0.00000)	+	**0.57055(0.12595)**	1.62925(0.06619)	+
	5	**1.45264(0.00136)**	1.43506(0.00237)	+	**0.02448(0.00128)**	0.05309(0.00089)	+	**0.51105(0.01030)**	1.56288(0.01084)	+
	8	**1.93511(0.00603)**	1.90224(0.00586)	+	**0.17509(0.00262)**	0.19360(0.00196)	+	**0.45855(0.01493)**	1.52280(0.01593)	+
	10	**2.36817(0.00881)**	2.31663(0.00251)	+	**0.18542(0.00559)**	0.21058(0.00169)	+	**0.49266(0.02715)**	1.08511(0.02205)	+
	12	**2.88424(0.00436)**	2.83657(0.00687)	+	**0.08794(0.00244)**	0.11677(0.00219)	+	**0.62036(0.04495)**	0.92185(0.03365)	+
DTLZ6	3	**0.33424(0.03633)**	0.32447(0.03591)	+	**0.10386(0.03713)**	0.10983(0.03663)	∼	**0.71938(0.11663)**	0.90558(0.099584)	+
	5	**1.56505(0.00156)**	1.53960(0.00118)	+	**0.02813(0.00419)**	0.11355(0.00448)	+	**0.69689(0.01406)**	0.89882(0.07521)	+
	8	**2.11648(0.00088)**	2.09258(0.00366)	+	**0.22887(0.00741)**	0.28547(0.00662)	+	**0.56161(0.01451)**	0.90541(0.02258)	+
	10	**2.56721(0.00094)**	2.53254(0.00460)	+	**0.28000(0.00712)**	0.35567(0.00618)	+	**0.52588(0.01329)**	0.80814(0.03695)	+
	12	**3.09524(0.00223)**	3.05185(0.00393)	+	**0.09025(0.00379)**	0.13568(0.00303)	+	**0.53840(0.01523)**	0.77850(0.03652)	+
DTLZ7	3	**0.44579(0.00011)**	0.43402(0.00009)	+	**0.00721(0.00009)**	0.02884(0.00015)	+	**0.66222(0.01533)**	0.88651(0.01248)	+
	5	**0.57863(0.00049)**	0.46188(0.00051)	+	**0.05049(0.00040)**	0.14841(0.00048)	+	**0.44769(0.00993)**	0.87053(0.00985)	+
	8	**0.00591(0.00202)**	0.00056(0.00121)	+	**0.93921(0.03233)**	1.09558(0.03022)	+	**0.80455(0.01723)**	0.96338(0.04585)	+
	10	**0.00118(0.00058)**	0.00085(0.00001)	+	**1.18281(0.03294)**	1.35269(0.00949)	+	**0.81042(0.02184)**	0.81250(0.02238)	∼
	12	**0.00025(0.00024)**	0.00010(0.00001)	+	**1.39711(0.04133)**	1.55684(0.02808)	+	0.82095(0.03078)	**0.70356(0.02841)**	–
CU30-R300		**0.64875(0.00000)**	0.64566(0.00007)	+	**0.00001(0.00001)**	0.00101(0.00003)	+	**0.57660(0.00725)**	1.43717(0.00187)	+
CU50-R200		**0.64266(0.00001)**	0.63947(0.00003)	+	**0.00000(0.00000)**	0.00031(0.00000)	+	**0.65362(0.00365)**	1.51920(0.00000)	+
CU50-R500		**0.64407(0.00000)**	0.64153(0.00014)	+	**0.00011(0.00000)**	0.00170(0.00010)	+	**0.37173(0.00459)**	1.26536(0.00705)	+
CU80-R800		**0.64470(0.00001)**	0.64124(0.00028)	+	**0.00026(0.00001)**	0.00203(0.00019)	+	**0.24260(0.00446)**	1.36493(0.01366)	+
CU100-R1000		**0.65420(0.00001)**	0.65030(0.00032)	+	**0.00036(0.00000)**	0.00216(0.00019)	+	**0.31329(0.00424)**	1.40688(0.01171)	+
CU120-R1200		**0.64958(0.00002)**	0.64463(0.00036)	+	**0.00041(0.00001)**	0.00306(0.00020)	+	**0.31916(0.00988)**	1.48648(0.01181)	+
CU160-R1600		**0.64428(0.00004)**	0.63763(0.00072)	+	**0.00058(0.00004)**	0.00319(0.00033)	+	**0.32256(0.00810)**	1.56349(0.01681)	+
CU200-R2000		**0.64615(0.00011)**	0.63841(0.00075)	+	**0.00081(0.00006)**	0.00429(0.00036)	+	**0.36145(0.00970)**	1.58149(0.02547)	+

Table 6
Mean and standard deviation values of HV, IGD and SPREAD obtained by IBMOEA/D and MOEA/D-EP on DTLZ and MNRP instances. the'-', '+', and '∼' denotes the corresponding value is worse, better or similar to that of MOEA/D-EP according to the Wilcoxon test with the 5% significance level.

Instance		HV			IGD			GSPREAD		
–	obj#	IBMOEA/D	MOEA/D-EP	T	IBMOEA/D	MOEA/D-EP	T	IBMOEA/D	MOEA/D-EP	T
DTLZ1	3	**1.15128(0.00030)**	1.15058(0.00041)	+	**0.00442(0.00009)**	0.00465(0.00006)	+	0.62105(0.01182)	**0.61560(0.01035)**	∼
	5	**1.59036(0.00028)**	1.58986(0.00032)	+	**0.01689(0.00013)**	0.01708(0.00011)	+	**0.37109(0.01516)**	0.38465(0.001258)	∼
	8	**2.14246(0.00004)**	2.14235(0.00006)	+	**0.03177(0.00056)**	0.03196(0.00044)	∼	0.26117(0.00361)	**0.25028(0.00325)**	–
	10	**2.59331(0.00005)**	2.59301(0.00009)	+	**0.03896(0.00058)**	0.03915(0.00063)	∼	**0.22305(0.00401)**	0.23303(0.00446)	+
	12	**3.13898(0.00009)**	3.13822(0.00013)	+	**0.07691(0.00119)**	0.07733(0.00098)	+	**0.21301(0.00469)**	0.21227(0.00520)	∼
DTLZ2	3	**0.79016(0.00008)**	0.78908(0.00006)	+	**0.00636(0.00007)**	0.00695(0.00002)	+	**0.64163(0.01133)**	0.66690(0.01089)	+
	5	**1.37095(0.00033)**	1.36639(0.00018)	+	**0.02242(0.00021)**	0.02265(0.00016)	+	0.37287(0.00767)	**0.36382(0.00870)**	∼
	8	**2.06525(0.00040)**	2.05928(0.00035)	+	**0.05051(0.00031)**	0.05093(0.00022)	+	**0.23206(0.00700)**	0.24635(0.00615)	+
	10	**2.55344(0.00038)**	2.55118(0.00026)	+	**0.06802(0.00022)**	0.06853(0.00018)	+	0.20161(0.00762)	**0.20098(0.00663)**	∼
	12	**3.11361(0.00060)**	3.10986(0.00031)	+	**0.15441(0.00146)**	0.15566(0.00088)	∼	**0.17208(0.01317)**	0.17662(0.01298)	∼
DTLZ3	3	**0.78899(0.00115)**	0.78880(0.00086)	+	0.00680(0.00039)	**0.00653(0.00085)**	∼	**0.64228(0.08092)**	0.68535(0.04673)	∼
	5	**1.36314(0.00562)**	1.35862(0.00446)	+	**0.02544(0.00129)**	0.02608(0.00126)	+	0.36141(0.00883)	**0.32985(0.00715)**	–
	8	**2.05673(0.00549)**	2.05367(0.00225)	+	**0.05531(0.00304)**	0.05604(0.00208)	∼	0.24309(0.00751)	**0.23520(0.00766)**	–
	10	**2.55132(0.00382)**	2.54860(0.00315)	+	**0.07282(0.00348)**	0.07339(0.00315)	∼	**0.20223(0.00489)**	0.22516(0.00410)	+
	12	**3.10251(0.23389)**	3.10026(0.01062)	∼	**0.15356(0.05070)**	0.15567(0.01098)	∼	**0.17180(0.01059)**	0.19058(0.00693)	+
DTLZ4	3	**0.79021(0.00009)**	0.78893(0.00008)	+	**0.00634(0.00007)**	0.00693(0.00007)	+	0.65444(0.01163)	**0.62911(0.01030)**	–
	5	**1.37171(0.00032)**	1.36995(0.00036)	+	**0.02217(0.00030)**	0.02241(0.00021)	+	0.37470(0.00674)	**0.31208(0.00554)**	–
	8	**2.06679(0.00044)**	2.05996(0.00037)	+	**0.04837(0.00031)**	0.04912(0.00016)	+	0.23108(0.00797)	**0.20684(0.00635)**	–
	10	**2.55413(0.00045)**	2.55313(0.00022)	+	**0.06721(0.00029)**	0.06763(0.00020)	+	**0.20119(0.00710)**	0.22312(0.00688)	+
	12	**3.10618(0.00295)**	3.10511(0.00282)	+	**0.15607(0.00123)**	0.15714(0.00135)	+	**0.16054(0.01758)**	0.20144(0.01689)	+
DTLZ5	3	**0.44220(0.00001)**	0.43890(0.00001)	+	**0.00020(0.00000)**	0.00039(0.00001)	+	**0.57055(0.12595)**	0.85240(0.08893)	+
	5	**1.45264(0.00136)**	1.44892(0.00183)	+	**0.02448(0.00128)**	0.03128(0.00062)	+	**0.51105(0.01030)**	0.66892(0.01296)	+
	8	**1.93511(0.00603)**	1.92671(0.00574)	+	**0.17509(0.00262)**	0.18022(0.00163)	+	**0.45855(0.01493)**	0.53107(0.01281)	+
	10	**2.36817(0.00881)**	2.35225(0.00996)	+	**0.18542(0.00559)**	0.19015(0.00205)	+	**0.49266(0.02715)**	0.45028(0.02666)	+
	12	**2.88424(0.00436)**	2.87250(0.00598)	+	**0.08794(0.00244)**	0.092226(0.00208)	+	**0.62036(0.04495)**	0.68258(0.04198)	+
DTLZ6	3	**0.33424(0.03633)**	0.33019(0.03458)	+	0.10386(0.03713)	**0.10227(0.03458)**	–	**0.71938(0.11663)**	0.82147(0.08961)	+
	5	**1.56505(0.00156)**	1.55428(0.00195)	+	**0.02813(0.00419)**	0.06235(0.00339)	+	**0.69689(0.01406)**	0.70996(0.01854)	∼
	8	**2.11648(0.00088)**	2.10918(0.00188)	+	**0.22887(0.00741)**	0.25225(0.00385)	+	**0.56161(0.01451)**	0.61258(0.01360)	+
	10	**2.56721(0.00094)**	2.55861(0.00286)	+	**0.28000(0.00712)**	0.30258(0.00597)	+	0.52588(0.01329)	**0.50123(0.01256)**	–
	12	**3.09524(0.00223)**	3.08865(0.00350)	+	**0.09025(0.00379)**	0.11865(0.00273)	+	**0.53840(0.01523)**	0.58696(0.01193)	+
DTLZ7	3	0.44579(0.00011)	**0.44595(0.00012)**	∼	**0.00721(0.00009)**	0.01029(0.00016)	+	0.66222(0.01533)	**0.62580(0.01669)**	–
	5	**0.57863(0.00049)**	0.55246(0.00041)	+	**0.05049(0.00040)**	0.07552(0.00060)	+	0.44769(0.00993)	**0.39652(0.00836)**	–
	8	**0.00591(0.00202)**	0.00493(0.00185)	+	**0.93921(0.03233)**	0.97525(0.03110)	+	0.80455(0.01723)	**0.70650(0.01994)**	–
	10	**0.00118(0.00058)**	0.00106(0.00022)	+	**1.18281(0.03294)**	1.22369(0.01892)	+	0.81042(0.02184)	**0.77145(0.01969)**	–
	12	**0.00025(0.00024)**	0.00019(0.00025)	+	**1.39711(0.04133)**	1.43553(0.03010)	+	0.82095(0.03078)	**0.72559(0.03810)**	–
CU30-R300		**0.64875(0.00000)**	0.64566(0.00007)	+	**0.00001(0.00001)**	0.00101(0.00003)	+	**0.57660(0.00725)**	0.61580(0.00835)	+
CU50-R200		**0.64266(0.00001)**	0.63947(0.00003)	+	**0.00000(0.00000)**	0.00031(0.00000)	+	**0.65362(0.00365)**	0.57835(0.00298)	+
CU50-R500		**0.64407(0.00000)**	0.64153(0.00014)	+	**0.00011(0.00000)**	0.00170(0.00010)	+	**0.37173(0.00459)**	0.41481(0.00467)	+
CU80-R800		**0.64470(0.00001)**	0.64124(0.00028)	+	**0.00026(0.00001)**	0.00203(0.00019)	+	**0.24260(0.00446)**	0.36697(0.00511)	+
CU100-R1000		**0.65420(0.00001)**	0.65030(0.00032)	+	**0.00036(0.00000)**	0.00216(0.00019)	+	**0.31329(0.00424)**	0.44258(0.00401)	+
CU120-R1200		**0.64958(0.00002)**	0.64463(0.00036)	+	**0.00041(0.00001)**	0.00306(0.00020)	+	**0.31916(0.00988)**	0.34879(0.00821)	+
CU160-R1600		**0.64428(0.00004)**	0.63763(0.00072)	+	**0.00058(0.00004)**	0.00319(0.00033)	+	**0.32256(0.00810)**	0.35847(0.00905)	+
CU200-R2000		**0.64615(0.00011)**	0.63841(0.00075)	+	**0.00081(0.00006)**	0.00429(0.00036)	+	**0.36145(0.00970)**	0.58410(0.01120)	+

objectives, where the performances of the two algorithms are similar. In terms of IGD, IBMOEA/D performs better than MOEA/D-EP on all instances, except for DTLZ3 with 3 and objectives, DTLZ6 with 3 objectives, and *CU–50/R–200* problems, on which IBMOEA/D performs similarly to MOEA/D-EP. These results show that the resource allocation strategy based on the indicator does help improve the algorithm efficiency.

5.7.2. MOEA/D-GRA versus MOEA/D-RAI

MOEA/D-RAI is a variant of IBMOEA/D in which no EP exists and allocating computational resources is based on indicator selection on each subproblem (as shown in Algorithm 4). MOEA/D-GRA was presented by Zhou. etc. [26], in which a generalized resource allocation (GRA) strategy for decomposition-based MOEAs was developed by using a probability of improvement vector. Therefore, it is interesting to compare these two different approaches since both of them consider the resource allocation in the selection of subproblem for optimization. In our comparison experiments, the parameter settings of both MOEA/D-GRA and MOEA/D-RAI are the same. Table 7 presents the average values of HV of MOEA/D-GRA and MOEA/D-RAI algorithms on the DTLZ instances over 30 independent runs. From these results, we can see that for most of the instances, the performance of the two algorithms is close. It is evident that the use of the improvement of the quality of the solution for guiding resource allocation does have advantages on MOPs. The results are reasonable because the two algorithms basically share the same framework. Thus, given sufficient computational resource, these algorithms should achieve similar final results.

5.8. Performance of IBMOEA/D when the EP size is equal to N

The EP of IBMOEA/D stores $2N$ solutions at each generation. In this section, an EP with a size of N for IBMOEA/D is considered. When the number $2N$ is changed to N in Algorithm 2 and the loop statement "**For** $j = 1$ **to** N" is changed to "**For** $j = 1$ **to** $N/2$" in Algorithm 3, the EP size of IBMOEA/D will be changed to N. IBMOEA/D with an EP size of N is called IBMOEA/D_N in this section. In each generation, IBMOEA/D_N randomly selects $N/2$ direction vectors for storing the nearest non-dominated solutions in EP. IBMOEA/D_N is compared with MOEA/D-PBI and NSGA-III on the DTLZ benchmark problems in the experiment. Table 8 lists the mean and standard deviation values of the HV metrics on the DTLZ1-7 problems with 3, 5, 8, 10, and 12 objectives to compare the

Table 7
Mean and standard deviation values of HV obtained by MOEA/D- RAI and MOEA/D- GRA on DTLZ instances.

Instance			
–	obj#	MOEAD-GRA	MOEA/D-RAI
DTLZ1	3	**1.15120(0.00012)**	1.15019(0.00045)
	5	**1.58966(0.00024)**	1.58905(0.00035)
	8	2.14035(0.00009)	**2.14220(0.00004)**
	10	2.58218(0.00018)	**2.59276(0.00018)**
	12	**3.14125(0.00044)**	3.13749(0.00055)
DTLZ2	3	**0.78993(0.00005)**	0.78367(0.00004)
	5	**1.37154(0.00009)**	1.36098(0.00003)
	8	2.05110(0.00035)	**2.05368(0.00061)**
	10	**2.56108(0.00046)**	2.54929(0.00065)
	12	3.10168(0.00018)	**3.10353(0.00022)**
DTLZ3	3	**0.78995(0.00103)**	0.78278(0.00069)
	5	**1.36407(0.00255)**	1.34670(0.00398)
	8	2.03261(0.00445)	**2.04889(0.00189)**
	10	**2.55216(0.00280)**	2.54493(0.00290)
	12	**3.10208(0.00350)**	3.09936(0.00202)
DTLZ4	3	**0.78830(0.00042)**	0.78167(0.00002)
	5	1.34209(0.00058)	**1.36410(0.00039)**
	8	**2.05485(0.00018)**	2.05190(0.00004)
	10	**2.56619(0.00009)**	2.54980(0.00003)
	12	3.08225 (0.00108)	**3.10025(0.00322)**
DTLZ5	3	0.43126 (0.00011)	**0.43377(0.00002)**
	5	**1.44205(0.00097)**	1.43506(0.00237)
	8	**1.91667(0.00130)**	1.90224(0.00586)
	10	2.30263 (0.00852)	**2.31663(0.00251)**
	12	2.82865 (0.00425)	**2.83657(0.00687)**
DTLZ6	3	**0.32890(0.01280)**	0.32447(0.03591)
	5	**1.54122(0.00108)**	1.53960(0.00118)
	8	2.07652(0.00332)	**2.09258(0.00366)**
	10	2.52701(0.00280)	**2.53254(0.00460)**
	12	**3.07219(0.00107)**	3.05185(0.00393)
DTLZ7	3	**0.45108(0.00018)**	0.43402(0.00009)
	5	0.45560(0.00045)	**0.46188(0.00051)**
	8	**0.00080(0.00065)**	0.00056(0.00121)
	10	**0.000105(0.000012)**	0.00085(0.00001)
	12	0.00005(0.00000)	**0.00010(0.00001)**

Table 8
Mean and standard deviation values of HV obtained by NSGA-III, MOEA/D-PBI and IBMOEA/D_N on all the DTLZ1-7 test instances.

HV	obj	NSGA-III	MOEA/D-PBI	IBMOEA/D_N
DTLZ1	3	1.14862(0.00089)	1.14884(0.00005)	**1.14931(0.00051)**
	5	1.58703(0.00272)	1.58853(0.00005)	**1.58920(0.00043)**
	8	2.14203(0.00004)	2.14195(0.00005)	**2.14209(0.00012)**
	10	**2.59329(0.00003)**	2.59244(0.00025)	2.59308(0.00018)
	12	3.13819(0.00002)	3.13701(0.00031)	**3.13851(0.00022)**
DTLZ2	3	0.78344(0.00041)	0.78386(0.00000)	**0.78659(0.00015)**
	5	1.35249(0.00045)	1.35515(0.00005)	**1.36609(0.00028)**
	8	2.04333(0.00067)	2.05185(0.00008)	**2.06138(0.00086)**
	10	2.53652(0.00064)	2.54528(0.00007)	**2.55012(0.00029)**
	12	3.09631(0.00040)	3.10035(0.00010)	**3.10752(0.00054)**
DTLZ3	3	0.78207(0.00086)	0.78285(0.00086)	**0.78390(0.00095)**
	5	1.26045(0.06508)	1.34875(0.00403)	**1.35288(0.00293)**
	8	2.01779(0.00838)	2.04571(0.00279)	**2.05113(0.00496)**
	10	2.51522(0.00446)	2.54097(0.00129)	**2.54896(0.00309)**
	12	3.08702(0.00435)	3.09820(0.00122)	**3.10121(0.00197)**
DTLZ4	3	0.78366(0.00021)	0.78386(0.00001)	**0.78820(0.00014)**
	5	1.35420(0.00029)	1.35550(0.00004)	**1.37171(0.00032)**
	8	2.04961(0.00040)	2.05283(0.00005)	**2.06225(0.00038)**
	10	2.54386(0.00035)	**2.54632(0.00004)**	2.54609(0.00087)
	12	3.10024(0.00014)	3.09810(0.00402)	**3.10294(0.00101)**
DTLZ5	3	0.44045(0.00019)	0.43092(0.00001)	**0.44119(0.00002)**
	5	**1.44898(0.00578)**	1.43866(0.00166)	1.44696(0.00160)
	8	1.91962(0.02043)	1.88986(0.00608)	**1.92716(0.00525)**
	10	2.36682(0.02638)	2.29885(0.01776)	**2.36701(0.00767)**
	12	2.81008(0.02891)	2.81704(0.00737)	**2.86358(0.00354)**
DTLZ6	3	**0.43963(0.00032)**	0.32526(0.03370)	0.32169(0.02589)
	5	1.48585(0.00572)	1.53991(0.00264)	**1.55395(0.00128)**
	8	1.83051(0.01810)	2.07940(0.00485)	**2.09615(0.00092)**
	10	2.20722(0.03663)	2.52599(0.00548)	**2.55053(0.00108)**
	12	2.77189(0.02882)	3.04131(0.00491)	**3.08913(0.00198)**
DTLZ7	3	0.44208(0.00024)	0.43405(0.00016)	**0.44298(0.00020)**
	5	0.52757(0.00287)	0.44159(0.00040)	**0.55961(0.00088)**
	8	**0.53693(0.00548)**	0.00055(0.00020)	0.00560(0.00135)
	10	**0.58470(0.01099)**	0.00002(0.00000)	0.00112(0.00069)
	12	**0.68289(0.01320)**	0.00000(0.00000)	0.00021(0.00019)

Table 9
Mean and standard deviation values of HV obtained by MOEA/D-PBI* and IBMOEA/D_N * on all the DTLZ1-7 test instances.

HV	obj	MOEA/D-PBI*	IBMOEA/D_N*
DTLZ1	3	**1.14918(0.00004)**	1.14905(0.00048)
	5	1.58892(0.00005)	**1.58893(0.00040)**
	8	**2.14201(0.00006)**	2.14196(0.0009)
	10	2.59250(0.00022)	**2.59291(0.00020)**
	12	3.13745(0.00028)	**3.13810(0.00025)**
DTLZ2	3	0.78412(0.00000)	**0.78552(0.00024)**
	5	1.35532(0.00003)	**1.36588(0.00021)**
	8	2.05238(0.00008)	**2.05831(0.00080)**
	10	2.54619(0.00009)	**2.54933(0.00019)**
	12	3.10122(0.00009)	**3.10690(0.00049)**
DTLZ3	3	0.78302(0.00077)	**0.78369(0.00088)**
	5	1.34889(0.00320)	**1.35263(0.00166)**
	8	2.04568(0.00169)	**2.05092(0.00360)**
	10	2.54127(0.00103)	**2.54803(0.00291)**
	12	**3.09815(0.00109)**	3.09785(0.00208)
DTLZ4	3	0.78411(0.00002)	**0.78867(0.00028)**
	5	1.35575(0.00006)	**1.36118(0.00044)**
	8	2.05295(0.00004)	**2.05860(0.00045)**
	10	**2.54629(0.00008)**	2.54587(0.00092)
	12	3.09853(0.00336)	**3.10215(0.00125)**
DTLZ5	3	0.43086(0.00001)	**0.43625(0.00043)**
	5	1.43915(0.00135)	**1.44327(0.00131)**
	8	1.89013(0.00552)	**1.92028(0.00433)**
	10	2.31058(0.00819)	**2.35016(0.00510)**
	12	2.82296(0.00422)	**2.85409(0.00388)**
DTLZ6	3	**0.32593(0.01107)**	0.32138(0.01296)
	5	1.54250(0.00224)	**1.55220(0.00115)**
	8	2.07956(0.00380)	**2.08505(0.00102)**
	10	2.53180(0.00404)	**2.54933(0.00102)**
	12	3.04028(0.00395)	**3.08766(0.00135)**
DTLZ7	3	0.43439(0.00015)	**0.44166(0.00018)**
	5	0.45204(0.00035)	**0.55019(0.00094)**
	8	0.00058(0.00028)	**0.00492(0.00167)**
	10	0.00003(0.00000)	**0.00095(0.00103)**
	12	0.00000(0.00000)	**0.00018(0.00006)**

three algorithms. It can be seen that, although the EP size becomes smaller, IBMOEA/D_N still performs better or performs similarly to the other two algorithms.

For a fairer comparison, MOEA/D-PBI is equipped with an EP (called MOEA/D-PBI*), where each solution in the population is attempted to store in EP at each generation using the archive management strategy presented in Ref. [40]. At the same time, a.

Variant (IBMOEA/D_N*) of IBMOEA/D_N in which no dynamic allocation of computational resources exists is generated by randomly selecting subproblem for the search at Step 2 (a) in Algorithm 1. IBMOEA/D_N* is compared with MOEA/D-PBI* on the DTLZ benchmark problems in this experiment. The comparison results are shown in Table 9. From Table 9, we can observe that for most instances, the mean values of HV obtained by IBMOEA/D_N* is greater than the mean values of HV obtained by MOEA/D-PBI*. These results prove that the indicator-

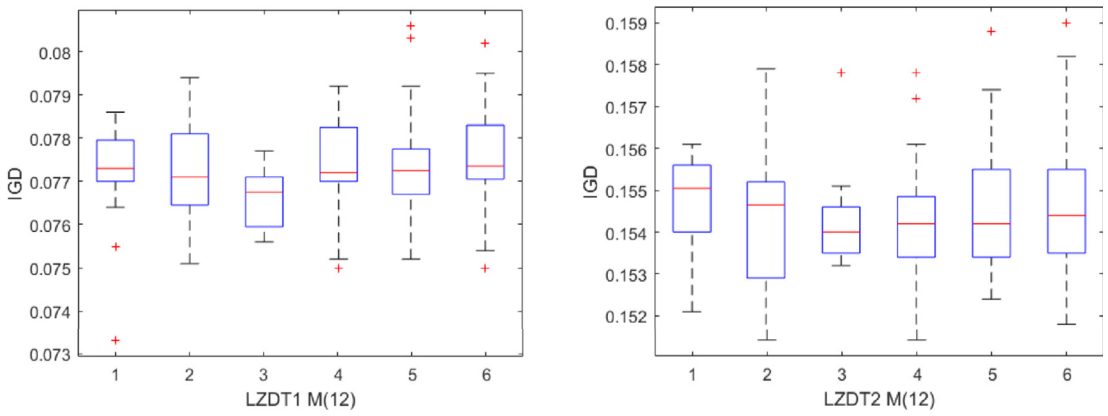

Fig. 6. Sensitivity tests for power law parameter.μ

based EP management strategy presented in this study is effective to handle MOPs.

5.9. Experiments on power law parameter μ

Selecting the subproblem to update, i.e., the computational resource assignment for different subproblems, is an important task in the IBMOEA/D framework. In this study, the subproblem is selected to update according to a power law probability, with which the subproblem that has been improved more over the last generations has a high probability to be selected for exploration. The power law parameter μ is an important adjustable parameter. In this section, we conduct experiments to test the sensitivity of the parameter μ. In the experiments, μ is set to 0.1, 0.2, 0.4, 0.6, 0.8 and 1.0 for DTLZ1 and DTLZ2 with 12 objectives.

The detailed results can be obtained by using boxplots, which represent the distributions of IGD values in the comparisons performed in the experiments. These boxplots are shown in Fig. 6. It shows that μ should not be too small. For $\mu \to 0$, randomly selected subproblem source get force to update, resulting in merely a uniform computational resource assignment in the search space. Fig. 6 also shows that μ is not too large; otherwise, the subproblem that has been improved most is always selected for investment. In this case, the computational system may be trapped in a local optima solution. μ is set to 0.4 for all the instances in this work.

6. Conclusion

In this study, a decomposition-based multi-objective algorithm, called IBMOEA/D, which integrates the preference information based on a binary quality indicator, is presented. In multi-objective optimization, there is usually no sufficient preference information to guide the evolution process, such that determining whether one non-dominated solution is better than the other solutions is difficult. In this improved edition based on MOEA/D, we use an EP based on a binary quality indicator to preserve the diversity of the non-dominated solutions found along the search process. At the same time, information obtained from the improvement of the quality of individuals is used to decide which search regions should be invested at each generation by a power law distribution probability. In this computational resource allocation strategy, the more the quality improvement of a subproblem is, the higher the probability that the subproblem would be selected to be invested. Thus, the indicator-based selection and the decomposition strategy can complement each other. Through extensive comparative experiments on the DTLZ and MNRP problems with varied numbers of objectives, IBMOEA/D is shown to be good at coping with MaOPs and discrete combinatorial MOPs. We have also investigated and discussed the effects of the EP and the computational resource allocation strategy in the search process. In future work, we are interested employing other quality evaluation techniques, e.g., hypervolume approximation by achievement scalarizing functions [17], as evaluation indicator. At the same time, to save the computing resources, we plan to reduce the number of the weight vectors in our algorithm. This requires us to develop a better strategy for decoupling weight vectors from population [41,42] to ensure the diversity of the solutions in objective space.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant Nos. 61301298, 61575126, 61401283 and 61373158, and the Scientific Research and Development Foundation of Shenzhen under Grant Nos. JCYJ20170302145554126.

Appendix A. Supplementary data

Supplementary data related to this article can be found at https://doi.org/10.1016/j.swevo.2017.11.004.

References

[1] K. Sindhya, K. Miettinen, K. Deb, A hybrid framework for evolutionary multi-objective optimization, IEEE Trans. Evol. Comput. 17 (4) (2013) 495–511. AUG.
[2] A. Zhou, B.Y. Qu, H. Li, S.Z. Zhao, P.N. Suganthan, Q. Zhang, Multiobjective evolutionary algorithms: a survey of the state of the art, Swarm Evol. Comput. 1 (1) (2011) 32–49.
[3] H. Wang, X. Yao, Corner sort for pareto-based many-objective optimization, IEEE Trans. Cybern. 44 (3) (2014) 92–102.
[4] Y. Wang, H.X. Li, G.G. Yen, W. Song, MOMMOP: multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems, IEEE Trans. Cybern. 45 (4) (2015) 830–843.
[5] H. Li, Q. Zhang, Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II, IEEE Trans. Evol. Comput. 13 (2) (2009) 284–302.
[6] B. Y. Qu, Y. S. Zhu, Y. C. Jiao, M. Y. Wu, P. N. Suganthan, and J. J. Liang, "A survey on multi-objective evolutionary algorithms for the solution of the environmental/ economic dispatch problems," Swarm Evol. Comput.., https://doi.org/10.1016/j.swevo.2017.06.002.
[7] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (Apr. 2002) 182–197.
[8] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: improving the strength pareto evolutionary algorithm, in: Proc. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, Eurogen, 2001, pp. 1–21.
[9] C.A. CoelloCoello, D.A. Van Veldhuizen, G.B. Lamont, Evolutionary Algorithms for Solving Multi-objective Problems, Kluwer Academic Publishers, New York, 2002.
[10] E. Zitzler, S. Kuńzli, Indicator-based selection in multiobjective search, in: Proceedings of the 8th International Conference. Parallel Problem Solving from Nature-ppsn VIII, Springer, Birmingham, UK, 2004, pp. 832–842.
[11] H. Wang, L. Jiao, X. Yao, Two_Arch2: an improved two-archive algorithm for many-objective optimization, IEEE Trans. Evol. Comput. 19 (4) (2015) 524–541.
[12] X.H. Chen, Solving Many-objective Optimization Problems with Evolutionary Algorithm and its Application, Ph.D. dissertation, Shenzhen University, Shenzhen, 2015.
[13] E. Zitzler, L. Thiele, Multiobjective optimization using evolutionary algorithms – a comparative case study, Lect. Notes Comput. Sci. 1498 (1998) 292–301. Springer, Berlin, September.

[14] J.D. Knowles, D.W. Corne, On metrics for comparing non- dominated sets, in: Proc. Of 2002 IEEE Congress on Evolutionary Computation, 2002, pp. 711–716. Honolulu, May 12-17.
[15] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, V.G. Fonseca, Performance assessment of multiobjective optimizers: an analysis and review, IEEE Trans. Evol. Comput. 2 (7) (April 2003) 117–132.
[16] T. Okabe, Y. Jin, B. Sendhoff, A critical survey of performance indices for multi-objective optimization, in: Proc. Of 2003 IEEE Congress on Evolutionary Computation, 2003, pp. 878–885. Canberra, Australia, December 8-12.
[17] H. Ishibuchi, N. Tsukamoto, Y. Sakane, Y. Nojima, Indicator-based evolutionary algorithm with hypervolume approximation by achievement scalarizing functions, in: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, GECCO'10, ACM, New York, NY, USA, 2010, pp. 527–534.
[18] J. Bader, E. Zitzler, HypE: an algorithm for fast hypervolume-based many-objective optimization, Evol. Comput. 19 (1) (2011) 45–76.
[19] Q. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (2007) 712–731.
[20] L.C. Jiao, H. Wang, R.H. Shang, F. Liu, A co-evolutionary multi-objective optimization algorithm based on direction vectors, Inf. Sci. 228 (2013) 90–112.
[21] W.K. Mashwani, A. Salhi, A decomposition-based hybrid multiobjective evolutionary algorithm with dynamic resource allocation, Appl. Soft Comput. 12 (2012) 2765–2780.
[22] L. Ke, Q. Zhang, R. Battiti, Hybridization of decomposition and local search for multiobjective optimization, IEEE Trans. Cybern 44 (10) (2014) 1808–1820.
[23] X. Cai, Y. Li, Z. Fan, Q. Zhang, An external archive guided multiobjective evolutionary algorithm based on decomposition for combinatorial optimization, IEEE Trans. Evol. Comput. 19 (4) (2015) 508–523.
[24] H. Li, D. Landa-Silva, An adaptive evolutionary multi-objective approach based on simulated annealing, Evol. Comput. 19 (4) (2011) 561–595.
[25] Y. Mei, K. Tang, X. Yao, Decomposition-based memetic algorithm for multiobjective capacitated arc routing problem, IEEE Trans. Evol. Comput. 15 (2) (Apr. 2011) 151–165.
[26] A. Zhou, Q. Zhang, Are all the subproblems equally important? Resource allocation in decomposition based multiobjective evolutionary algorithms, IEEE Trans. Evol. Comput. 20 (1) (2016) 52–64.
[27] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: empirical results, Evol. Comput. 8 (2) (2000) 173–195.
[28] J. Knowles, D. Corne, The Pareto arxchived evolution strategy: a new baseline algorithm for multiobjective optimization, in: Proceedings of the 1999 Congress on Evolutionary Computation, IEEE Press, Piscataway, NJ, 1999.
[29] N. Beume, S-metric calculation by considering dominated hypervolume as klee's measure problem, Evol. Comput. 17 (2009) 477–492.
[30] K. Bringmann, T. Friedrich, Approximating the volume of unions and intersections of high-dimensional geometric objects, in: S.-H. Hong, H. Nagamochi, T. Fukunaga (Eds.), Algorithms and Computation, Lecture Notes in Computer Science vol. 5367, Springer, Berlin, Heidelberg, 2008, pp. 436–447.
[31] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable test problems for evolutionary multiobjective optimization, Evol. Multiobjective Optim. Theor. Adv. Appl. (2005) 105–145.
[32] C.A. Coello Coello, N. Cruz Cortés, Solving multiobjective optimization problems using an artificial immune system, Genet. Program. Evolvable Mach. 6 (2) (June 2005) 163–190.
[33] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach, IEEE Trans. Evol. Comput. 3 (4) (November 1999) 257–271.
[34] J. Durillo, A. Nebro, E. Alba, Thejmetal framework formulti-objective optimization: design and architecture, in: CEC, 2010, pp. 4138–4325.
[35] R. Wang, R. Purshouse, P. Fleming, Preference-inspired co-evolutionary algorithms for many-objective optimization, IEEE Trans. Evol. Comput. 17 (4) (2013) 474–491.
[36] S.W. Jiang, J. Zhang, Y.S. Ong, A.N.S. Zhang, P.S. Tan, A simple and fast hypervolume indicator-based multiobjective evolutionary algorithm, IEEE Trans. Cybern. 45 (10) (2015) 2202–2213.
[37] S. Jiang, Y.S. Ong, J. Zhang, L. Feng, Consistencies or contradictions of performance metrics in multiobjective optimization, IEEE Trans. Cybern. 44 (12) (2014) 2391–2404.
[38] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints, IEEE Trans. Evol. Comput. 18 (4) (2014) 577–601.
[39] S. Rostami, F. Neri, A fast hypervolume driven selection mechanism for many-objective optimisation problems, Swarm Evol. Comput. 34 (1) (2017) 50–67.
[40] M.R. Chen, Y.Z. Lu, A novel elitist multiobjective optimization algorithm: multiobjective extremal optimization, Eur. J. Oper. Res. 188 (3) (2008) 637–651.
[41] S. Jiang, S. Yang, A strength Pareto evolutionary algorithm based reference direction for multiobjective and many-objective optimisation, IEEE Trans. Evol. Comput. 21 (3) (2017) 329–346.
[42] S. Jiang, S. Yang, Y. Wang, X. Liu, Scalarizing functions in decomposition-based multiobjective evolutionary algorithms, IEEE Trans. Evol. Comput. (2017), https://doi.org/10.1109/TEVC.2017.2707980 in press.