Contents lists available at ScienceDirect

Neurocomputing

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

Self-organizing multiobjective optimization based on decomposition with neighborhood ensemble

Hu Zhang [a], Xiujie Zhang [b], Xiao-Zhi Gao [c], Shenmin Song [a,*]

[a] Center for Control Theory and Guidance Technology, Harbin Institute of Technology, Harbin 150001, China
[b] Academy of Fundamental and Interdisciplinary Sciences, Harbin Institute of Technology, Harbin 150001, China
[c] Department of Electrical Engineering and Automation, Aalto University School of Electrical Engineering, Aalto, Finland

ARTICLE INFO

Article history:
Received 2 April 2015
Received in revised form
11 August 2015
Accepted 14 August 2015
Communicated by K. Chan
Available online 6 November 2015

Keywords:
Multiobjective optimization
Evolutionary algorithm
Reproduction
Self-organizing map
Neighborhood ensemble

ABSTRACT

Currently, most of the multiobjective evolutionary algorithms (MOEAs) directly adopt the reproduction operators designed for the single-objective optimization. Since these operators do not consider the characteristics of multiobjective optimization problems (MOPs), they cannot always perform well in the MOEAs. Inspired by this case, this paper presents a self-organizing reproduction mechanism based on the regularity property of MOPs, and proposes a self-organizing multiobjective evolutionary algorithm based on decomposition with neighborhood ensemble. In the new reproduction, a self-organizing map approach is firstly employed to discover the population distribution structure, and to build a mating pool for each solution. Thereafter, reproductions are only allowed among the solutions within the same mating pools. In order to establish the mating pools, an ensemble of multiple neuron neighborhood sizes is also introduced. The probability of choosing different neighborhood sizes is updated based on their performance on producing new solutions over the last certain generations. Comprehensive experiments denote that the proposed algorithm is efficient and competitive. The contributions of the new reproduction mechanism and neighborhood ensemble are also experimentally validated.

1. Introduction

In engineering, there are many optimization problems with more than one objective, which are called *multiobjective optimization problems* (*MOPs*). The formulation of a MOP could be summarized as follows:

$$\min \quad F(x) = (f_1(x), f_2(x) \ldots, f_m(x))$$
$$\text{s.t.} \quad x = (x_1, x_2, \ldots, x_n) \in \Omega \qquad (1)$$

where Ω is the decision(variable) space, and a solution $x = (x_1, x_2, \ldots, x_n) \in \Omega$ is a vector of decision variables. $F: \Omega \to R^m$ is a vector of m objective functions, and R^m is called the objective space. Usually, the m objectives of a MOP are conflicted so that there exists no single solution which is able to optimize all the objectives simultaneously. Therefore, different from the optimization problem with one objective (single-objective optimization problem, SOP) whose solution is an optimal point, the MOP yields a set of tradeoff solutions (*Pareto optimal solutions*), named as *Pareto set* (*PS*) in the decision space and *Pareto front* (*PF*) in the objective space. Moreover, for a continuous MOP, the distribution of Pareto optimal solutions also shows a high degree of *regularity property* [1]. According to the Karush–Kuhn–Tucker condition, under mild conditions, for a m-objective continuous MOP, its PS (PF) structure in the decision (objective) space is a $(m-1)$-dimensional piecewise continuous manifold. Specifically, if $m=2$, the PS (PF) structure is piecewise continuous curves, and if $m=3$, the PS (PF) structure is a piecewise continuous surface.

There are two types of methods to deal with MOPs, i.e., traditional analytic algorithms and *evolutionary algorithms* (*EAs*). Since the EA optimization technique is population-based and not problem-dependent, it is able to approximate the PF (PS) of a MOP in a single run, which makes *multiobjective evolutionary algorithm* (*MOEA*) naturally become a major approach to address complicated MOPs. MOEA has become one of the most hot research directions in the EA field. Due to that the population size is limited, in general, the task of a MOEA for a MOP is to obtain a set of solutions whose objective points approach to the PF as close as

Abbreviation: MOP, Multiobjective optimization problems; SOP, Single-objective optimization problem; PS, Pareto set; PF, Pareto front; EA, Evolutionary algorithm; MOEA, Multiobjective evolutionary algorithm; SOM, Self-organizing map; SMOEA/D, Self-organizing multiobjective evolutionary algorithm based on decomposition with neighborhood ensemble; SRM, Self-organizing reproduction mechanism; LLE, Local linear embedding; MEDA, Multiobjective estimation of distribution algorithm; NNS, Neuron neighborhood size; VNS, Variable neighborhood search

* Corresponding author.
E-mail addresses: jxzhanghu@126.com (H. Zhang), xiujiezhang@hit.edu.cn (X. Zhang), xiao-zhi.gao@aalto.fi (X.-Z. Gao), songshenmin@hit.edu.cn (S. Song).

http://dx.doi.org/10.1016/j.neucom.2015.08.092
0925-2312/© 2015 Elsevier B.V. All rights reserved.

possible (*convergence*) and distribute along the PF as diverse as possible (*diversity*).

Since Schaffer's pioneer work in 1985 [2], over the last three decades, MOEAs have experienced a rapid development [3], and various efficient MOEAs have been proposed (e.g., [4–7]). There are two major components in a MOEA, reproduction and environmental selection. Reproduction, including mating selection, crossover, mutation and so on, produces new trial solutions based on the current population. Environmental selection selects promising solutions into the next generation. In the past, most of the research works on MOEA are given on the second component, and various selection mechanisms are designed. In summary, the most widely used selection approaches can be classified into three categories [8]:

- *Dominance based selection*: this type of selection approaches first utilizes the Pareto dominance relation as the primary mechanism to maintain the convergence of population, and then the other strategies, such as crowding distance [9] and *K*-nearest neighbor method [10], are added to preserve the population diversity. The typical dominance based MOEAs includes NSGA-II [9], SPEA2 [10], NPGA [11], MOGA [12], PAES [13], PESA [14], PESA-II [15].
- *Indicator based selection*: this type of selection approaches employs a performance indicator to guide the selection of the promising solutions, and the hypervolume indicator [16] is most adopted. The typical application paradigms are IBEA [17], SMS-EMOA [18] and HyPE [19].
- *Decomposition based selection*: this type of selection approaches decomposes a MOP into a number of single objective optimization subproblems [20], or several simple multiobjective subproblems [21], and then optimize them in a collaborative manner. The selection is determined by the objective functions of the subproblems. The representatives are MOGLS [22], C-MOGA [23] MOEA/D [20,24,25] and MOEA/D-M2M [21].

Compared with the environmental selection, research on the reproduction of MOEA is relatively sparse. The reason is that many popular MOEAs directly use those reproduction operators originally designed for SOPs. The successful applications of these operators make the researchers ignore the importance of developing specific reproduction operators for MOEAs [26]. As a matter of fact, since the topology of optimal solutions of a SOP is a point, and the distribution of optimal solutions of a MOP commonly presents a regular manifold structure, the essential topological difference leads that directly applying the reproduction operators for SOP in MOEAs cannot always ensure them to perform well. Some scholars have realized this issue and verified that the MOEAs, which directly adopt the reproduction operators of SOPs, perform poorly while tackling rotated and complicated MOPs [26–28]. In order to improve the performance of MOEAs, designing specific reproduction operators based on the topologies of Pareto optimal solutions of MOPs is necessary. At present, more focus is being put on this topic [28]. This paper aims to contribute to this research line.

Inspired by the regularity property of MOPs, based on *self-organizing map* (*SOM*) approach, we propose a *self-organizing multiobjective evolutionary algorithm based on decomposition with neighborhood ensemble* (*SMOEA/D*). SOM is an unsupervised artificial neural network developed by Kohonen [29,30]. It implements to exact the distinguishable features of input data by learning, and to map these features into neurons, such that the topological information of input data are preserved in a low dimensional network. Similar input samples are mapped into nearby neurons. The major contributions of this paper are summarized as follows.

- A *self-organizing reproduction mechanism* (*SRM*) is designed in SMOEA/D to produce new solutions. In the mechanism, a SOM is applied to discover the topological structure of population distribution at first. Afterwards, according to the obtained structure information, a mating pool is established for each solution. Finally, only the solutions within the same mating pool are allowed us to interact to produce new solutions.
- An ensemble of neuron neighborhoods with multiple sizes is designed in SMOEA/D to construct the mating pools. The probability of selection of neighborhood sizes is adjusted at each generation, based on their performance on generating new solutions over the last certain generations.
- Systematical experiments on multiple types of test instances have been conducted to do the performance and parameter analysis of SMOEA/D.

The reminder of this paper is organized as follows: Section 2 introduces some background about regularity property based MOEAs, procedures of SOM and its applications in MOEAs. Section 3 detailedly introduces the proposed algorithm framework and its components. Section 4 depicts the test instances, performance metrics, experimental results and comparison analysis to validate our contributions. Section 5 performs further discussions on the proposed algorithm, novel reproduction mechanism, neighborhood ensemble and parameter sensitivity. Conclusion is given in Section 6.

2. Background

2.1. Regularity property based MOEAs

As discussed in [26,31,32], the regularity property of MOPs has been utilized in the mathematical programming methods to approximate the PSs or PFs of MOPs [1,33,34]. However, it has not been fully exploited in the context of MOEAs. At present, the number of contributions on designing reproduction operators for MOEAs based on the characteristic of MOPs is still insignificant, and most of these contributions are based on the framework of *multiobjective estimation of distribution algorithm* (*MEDA*).

Regularity model-based multiobjective estimation of distribution algorithm (RM-MEDA) proposed in 2008 is the first work that explicitly applies the regularity property of MOPs in reproduction of a MOEA [26]. The RM-MEDA firstly partitions the population into *K* disjoint clusters. Then a promising area in the decision space is modeled by a probability distribution for each cluster. Finally, the trial solutions are sampled from the models. In 2009, through generalizing the idea of RM-MEDA, Zhang et al. designed a probabilistic model based MOEA [35] to deal with a class of MOPs with different dimensionalities of the PF and PS manifolds. Meanwhile, Yang et al. adopted a local linear embedding (LLE) to build the distribution model, introduced an immune inspired sparse individual clone algorithm to enhance local search ability, and presented a hybrid MEDA [36]. In 2010, Mo et al. [37] just selected some good solutions to build the probability distribution model, and proposed an elitist strategy based RM-MEDA. Since the fixed number of probability models is not beneficial to address complicated MOPs, in 2012, Wang et al. incorporated a reducing redundant cluster into RM-MEDA to dynamically modify the number of models during the evolution [38]. In 2014, to enhance the quality of solutions at the early evolutionary stage, Zhang et al. hybrid the LLE based approach and the traditional genetic operations to produce offspring at that stage in RM-MEDA [39]. Moreover, in order to generate desirable solutions when the population has no obvious regularity, Li et al. added a local learning strategy to produce new solutions by sampling from the neighborhood of elitist solutions to improve the RM-MEDA [40].

In addition to the MEDAs, the regularity property also has been utilized in the other MOEA categories. In 2012, Qi et al. [41] developed a regularity based multiobjective immune algorithm with Baldwinian learning, which employed a Baldwinian learning operator to build piecewise linear probability models and to produce a new generation. In 2013, through embedding a clustering approach in a traditional MOEA to gradually discover the PS structures, and selecting the neighboring solutions to recombine according to the structure information, we developed a clustering based MOEA [42]. In 2014, By using a principal curve algorithm and a Laplacian eigenmaps algorithm to obtain the approximation of the PS manifold, and interpolating new candidate solutions that obey the geometric property of PS, Li and Kwong proposed a general framework for evolutionary multiobjective optimization via manifold learning [31]. Based on the MOEA/D, through applying the Baldwinian learning operator to learn a piecewise distribution model for the current population, and producing an offspring along the descent direction which combines the evolving history of the parent individual and the environment information provided by the learned distribution model, Ma et al. proposed a MOEA/D with Baldwinian learning [43].

2.2. SOM and its applications in MOEAs

As mentioned above, SOM is an unsupervised artificial neural network. It produces a low-dimensional, discretized representation of the input space constructed by input data, which is called a map [29]. The topology of a 2-dimensional SOM is shown in Fig. 1. The SOM is consisted of an input and an output layers. Each layer contains a number of neurons, each neuron associates with a weight vector. The SOM procedures can be divided into two stages, learning and clustering. At first, the input samples are presented to input layer iteratively. SOM extracts their significant features and adjusts the weight value of each neuron for recognizing the features in future. After that, SOM classifies the input samples according into the weights, and maps them to the neurons. Similar samples are mapped into the neighboring neurons so that the topology of the input samples are kept [44,45] in the network.

Currently, there are various types of SOM based MOEAs existed. According to the usages of SOM, these MOEAs could be roughly divided into two categories. The first one includes the algorithms that directly employ the SOM to generate new solutions. The second category is characterized by using the SOM to guide search of the MOEAs.

In the first category, in 2002, Büche et al. [46] first utilized the SOM as a recombination operator to interpolate the parent population. The neural network orients along the Pareto optimal set through learning and interpolating neighboring neurons, and helps sharing of information, to improve the convergence speed. The proposed MOEA in [46] was applied to optimize airfoils, and a good result was obtained [47]. In 2009, Hakimi-Asiabar et al. [48] took the SOM to improve the NSGA-II. The just used the elite solutions of the current population to train the neurons. The training rule is developed based on the learning rule supported by the variable neighborhood search (VNS). The trained neural weight values determines a set of new solutions. In order to derive optimal operating policies for a three-objective multi-reservoir system, Hakimi-Asiabar et al. [49] also presented an improved self-learning genetic algorithm. This algorithm used the SOM and VNS to add a memory to the genetic algorithm to promote the local search accuracy further. Although aforementioned MOEAs has been improved by the SOM, the MOP's regularity property has not been considered when inbreeding new solutions. Therefore, in [50], Zhan et al. developed a model based MOEA by replacing the local principal component analysis with the SOM in RM-MEDA to learn the manifold structures of the PSs. The new solutions are produced by adding noisy vectors to uniformly extend the neural weight values of the SOM.

In the second category, Norouzi et al. [44] also applied the SOM to strength the NSGA-II when optimizing the water distribution networks. But different from [48], this algorithm applies the SOM to guide search. It first evolves the NSGA-II for certain generations to conduct the exploration. Then the exploitation is conducted repeatedly as follow. A SOM operation is called to cluster and divide the population into several subpopulations. Each subpopulation completes genetic operations and evolutions separately for certain generations. Then, the subpopulations are combined, and the whole population evolves for one generation.

Previous summary shows that there are few works about employing the SOM for PS detections of MOPs and search guidance of MOEAs. Thus we propose to adopt the SOM to gradually discover population distribution structure, design a SRM based on the structure for guiding search, and develop a SMOEA/D in this paper. Hopefully, our work will provide a novel perspective for promoting MOEAs based on the regularity property of MOPs.

3. Algorithm

3.1. Framework

Fig. 2 presents the schematic of SMOEA/D. Algorithm 1 shows the framework. It requires the following initial settings:

- N: population size;
- T: maximum evolutionary generation;
- G: history length;
- $V=\{v^1,\dots,v^K\}$: ensemble of different neuron neighborhood sizes (NNSs);
- τ_0: initial learning rate.

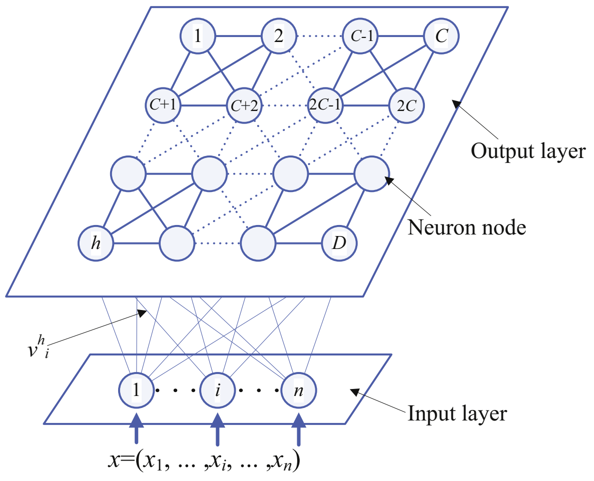

Fig. 1. Topology of a 2-dimensional SOM.

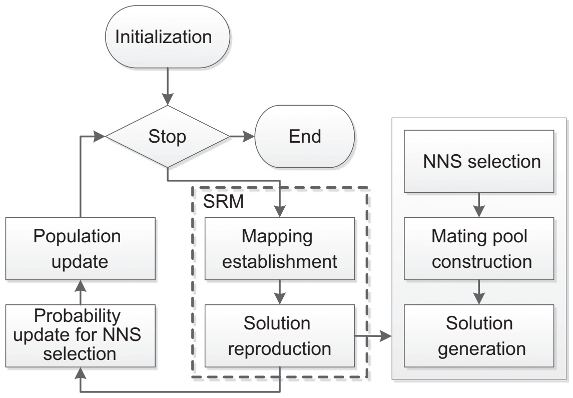

Fig. 2. The schematic of SMOEA/D.

In addition, at the tth generation, $\beta^{t,k}$ represents the probability of generating solutions with the kth NNS. Moreover, L denotes the mapping of the solutions to the neurons and $cn^{t,k}$ indicates the number of new solutions generated with v^k. Furthermore, $rn^{t,k}$ denotes the number of solutions that can be improved by the new ones produced with v^k. The number of neurons is D and σ_0 is the initial radius of domain for updating weights of neurons. The algorithm can be simply summarized into three operations:

Algorithm 1. SMOEA/D.

```
1  Initialize the population $P=\{x^1,x^2,\dots,x^N\}$, weights of neurons $W=\{w^1,\dots,w^D\}$, estimated ideal objective vector $z^*$, training set $S=P$,
   probability of selecting the $k$th NNS, $\beta^{1,k}, k=1,\dots,K$, distances matrix of the neuron set, $M$. Set an archive $A=P$.
2  Set $CN=\varnothing, RN=\varnothing$.
3  for $t=1$ to $T$ do
4      Map the solutions to neurons :
       $[W,L]=Mapping(P,N,S,M,W,\sigma_0,\tau_0,t,T)$.
5      Set $cn^{t,k}=0, rn^{t,k}=0, k=1,\dots,K$.
6      Randomly select $K$ indexes $idx^k$ from $\{1,\dots,N\}, k=1,\dots,K$.
7      Set $z'=z^*, A'=A$.
8      for $q=1$ to $N, x^q\in P$ do
9          Determine a NNS $v^{pos}$ for $x^q$ based on $idx^k$ and $\beta^{t,k}, k=1,\dots,K$.
10         Set the mating pool for $x^q$ : $B=MatingPool(q,L,M,v^{pos},N,P)$
11         Generate a trial solution : $y=Reproduction(x^q,B)$.
12         Update the archive and ideal point :
           $[A,z^*]=Update(y,A,z^*,N,q)$.
13         Count the number of solutions in $A'$ that could be improved by $y$ : $r=Count(y,A',z',N)$.
           Set $cn^{t,pos}=cn^{t,pos}+1, rn^{t,pos}=rn^{t,pos}+r$.
14
15     end
16     Set$CN=CN\cup\{cn^{t,1},\dots,cn^{t,K}\}, RN=RN\cup\{rn^{t,1},\dots,rn^{t,K}\}$.
17     Update the probability : $\{\beta^{t+1,1},\dots,\beta^{t+1,K}\}=Probability(t,CN,RN,G)$.
       Update the training set : $S=A\backslash P$.
18     Remove the duplicated solutions in $S$.
19     Update the population : $P=A$.
20
21 end
22 return $P$.
```

(1) *Solution reproduction:* In the operation, a SOM is firstly applied to learn the new added solutions in current population (Lines 18 and 19), and to implement mapping of the solutions to the neurons (Line 4). Thereafter, for each solution, a NNS is determined (Lines 6 and 9), and a mating pool is constructed based on the NNS and the mapping relationship obtained previously (Line 10). Then, a new solution is generated based on the mating pool (Line 11). The whole solution generation procedures are referred to as SRM. The approach of determining the NNS for x^q in Line 9 is as follows. If $q=idx^k, k\in\{1,\dots,K\}$, set $v^{pos}=v^k$, which aims to ensure that each NNS generates at least one solution. Otherwise, a roulette wheel method is employed to select a v^{pos} from V based on $\{\beta^{t,1},\dots,\beta^{t,K}\}$.

(2) *Solution update:* The newly produced solutions are utilized to update the external archive, ideal point and population, and to promote the convergence of the algorithm (Lines 12 and 20).

(3) *Neighborhood ensemble:* An ensemble consisted of neuron neighborhoods with multiple NNSs are utilized in the algorithm. The probability of selecting different NNSs is adaptively adjusted according to their performance on generating solutions over the last certain generations. (Lines 5 and 13–17).

The details of these three operations are given in the next sections.

3.2. Solution reproduction

Solution reproduction is achieved by the SRM, and it is consisted of three components: mapping relationship building, mating pool construction, and solution generation. Following gives the details of these components.

3.2.1. Mapping relationship building

The basic principle of building mapping relationship is that, a SOM is utilized to discover the population distribution structure, and then the associations between solutions and neurons are established based on the structure information. Algorithm 2 shows the procedures, which can be divided into two parts, i.e., learning of structure information, and building of mapping relationship.

- *Learning of structure information*: In this part, SOM learns the new added solutions in current population. While learning a solution, it sequently updates training parameters, identifies winning neuron, and adjusts weight values of the neurons around the winning neuron (Lines 2–9).
- *Building of mapping relationship*: After learning, for all the solutions in the population, based on the updated neural weight values, the mapping relationships are established by finding out their closest neurons in the output layer (Lines 10–20). It is expected that each neuron associates with at least one but not more than two solutions, which aims to ensure each neuron to

associate with the similar number of solutions.

Three points on the operator of mapping relationship building should be commented.

- At each generation, the operator is performed only once. The evolution and learning are conducted alternately.
- During the whole evolution, all the solutions to be learned are treated as a whole to adjust their training parameters in Line 3.
- At each generation, the initial values of neuron weights are inherited from the last generation in Line 7. That is to say, during the whole process, the weights are updated successively.

Algorithm 2. Mapping$(P, N, S, M, W, \sigma_0, \tau_0, t, T)$.

```
1 Let $U = \{1, 2, \ldots, D\}$ denote the neuron set.
2 for $i = 1$ to $|S|, x^i \in A$ do
3     Update training parameters : $\sigma = \sigma_0 \times (1 - \frac{(t-1)\times N + i}{N \times T})$, $\tau = \tau_0 \times (1 - \frac{(t-1)\times N + i}{N \times T})$ .
4     Find the closest neuron $h$ for $x^i$ :
5     $h = \arg\min_{1 \le j \le D} dis(x^i, w^j), w^j \in W$.
6     Update weights of neurons :
7     for $k \in \{k | M(h,k) < \sigma\}, w^k \in W$
8         $w^k = w^k + \tau \times \exp(-M(h,k)) \times (x^i - w^k)$.
      end
9 end
10 Randomly sample $N - D$ different neurons from $U$ to build a set $Q$.
11 Set $U' = U$.
12 for $i = 1$ to $N, x^i \in P$ do
13     Find the closest neuron $h$ for $x^i$ from $U'$ according to Line 4.
14     if $h \in Q$ then
15         $Q = Q \backslash \{h\}$. else
16         $U' = U' \backslash \{h\}$.
17     end
1819   Build mapping : $L(i) = h$.
20 end
21 return $W, L$.
```

According to above three points, we can see that the whole evolutionary process of SMOEA/D can also be regarded as an entire learning process of SOM. The task of this SOM is to learn the PS structure of a MOP hidden in the dynamic data generated by EA. The SOM and EA implement a seamless integration.

It should be noted that, in Algorithm 2, a 2-dimensional square SOM shown in Fig. 1 is employed. In the output layer, there are $C = \lfloor \sqrt{N} \rfloor$ neurons in each row and column, and the number of neurons is $D = C \times C$. The distance matrix M is shown in Eq. (2), and its element Δ_{ij} indicates the topological distance between two neurons i and j in the output layer. A number (i.e., D) of different solutions from P are randomly selected to build an initial weight set of neurons, W. The initial radius of domain for updating weights of neurons in SOM is set as $\sigma_0 = \sqrt{2C^2}/2$.

$$M = \begin{bmatrix} \Delta_{11} & \Delta_{12} & \ldots & \Delta_{1D} \\ \Delta_{21} & \Delta_{22} & \ldots & \Delta_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ \Delta_{D1} & \Delta_{D2} & \ldots & \Delta_{DD} \end{bmatrix} \quad (2)$$

3.2.2. *Mating pool construction*

The second step of SRM is to construct a mating pool for a solution, and the implementation is presented in Algorithm 3. For a solution x^q, its corresponding neuron h is first identified according to the mapping relationship obtained by *Mapping* operator (Line 2). Afterwards, v^{pos} neighboring neurons to the h are formed into a neuron neighborhood H based on the distance matrix M (Lines 3 and 4). Finally, all the solutions associated to the neurons in H are gathered as a mating pool of x^q (Lines 5–8). Since the similar individuals are mapped into the nearby neurons in SOM, gathering the solutions associated to the neighboring neurons implements high similarity of the solutions within a mating pool. As stated before, each neuron associates with one or two solutions, thus under a specific NNS, different solutions may have different sizes of mating pools. However, the differences among these sizes are not quite significant.

Algorithm 3. MatingPool(q, L, M, v^{pos}, N, P).

```
1 Set $B = \emptyset$.
2 Find out the neuron $h$ mapped by $x^q$: $h = L(q)$.
3 Sort $h$th row of $M$ in ascending order and keep the sorted indices in $I$.
4 Find out $v^{pos}$ neighboring neurons: $H = \{I(2), \ldots, I(v^{pos}+1)\}$.
5 for $i = 1$ to $v^{pos}$ do
6     Build mating pool for $x^q$ :
7     $\begin{cases} B = B \cup \{x^{k1}\} & \text{if } L(k1) = H(i) \\ B = B \cup \{x^{k1}, x^{k2}\} & \text{if } L(k1,k2) = H(i) \end{cases}$ ,
      where $x^{k1}, x^{k2} \in P, k1, k2 \in \{1, \ldots, N\}$
8 end
9 return $B$.
```

3.2.3. *Solution generation*

The last step of SRM is to produce a new solution y from current solution x^q and its mating pool B. The solution generation operators adopted in MOEA/D-DE [51], which have been validated to be effective for the MOPs with complicated PSs, are directly applied in the SRM. The details are presented in Algorithm 4. Here a differential evolution crossover operator [52] is firstly used to generate a new solution (Line 2). After that, a repair mechanism is adopted to make sure the solution be feasible (Line 3). Next, a polynomial mutation [2] operator is utilized to enhance the diversity of solutions (Line 4). Finally, a repair operation is also performed on the mutated solutions (Line 5). It is noted that the other solution generation operators can also be utilized here.

Algorithm 4. Reproduction(x^q, B).

```
1 Take a binary tournament selection approach to select two parents $x^1$ and $x^2$ from $B$, which are different from each other and $x^q$.
2 Generate a trial solution: $y' = x^q + F \times (x^1 - x^2)$, where $F$ is a control parameter.
```

```
3 Repair the solution: $y_i'' = \begin{cases} a_i & \text{if } y_i' < a_i \\ b_i & \text{else if } y_i' > b_i \\ y_i' & \text{otherwise} \end{cases}$, where $a_i, b_i$ are
  the lower and upper boundaries of variables $x_i$, $i = 1, 2, \dots, n$.
4 Mutate the solution: $y_i''' = \begin{cases} y_i'' + \delta_i \times (b_i - a_i) & \text{if } rand() < p_m \\ y_i'' & \text{otherwise} \end{cases}$
  with
  $\delta_i = \begin{cases} \left[2r + (1-2r)\left(\frac{b_i - y_i''}{b_i - a_i}\right)^{\eta_m + 1}\right]^{1/(\eta_m+1)} - 1 & \text{if } r < 0.5 \\ 1 - \left[2 - 2r + (2r-1)\left(\frac{y_i'' - a_i}{b_i - a_i}\right)^{\eta_m + 1}\right]^{1/(\eta_m+1)} & \text{otherwise} \end{cases}$,
  where $p_m$ and $\eta_m$ are the probability and distribution index
  of mutation, respectively. $r = rand(), i = 1, 2, \dots, n$.
5 Repair the
  solution:$y_i = \begin{cases} y_i'' - rand() \times (y_i'' - a_i) & \text{if } y_i''' < a_i \\ y_i'' + rand() \times (b_i - y_i'') & \text{else if } y_i''' > b_i \\ y_i''' & \text{otherwise} \end{cases}$, for
  $i = 1, 2, \dots, n$.
6 return $y = (y_1, y_2, \dots, y_n)$
```

3.3. *Solution update*

To update the existing solutions with the new generated y and promote convergence, through modifying the work in [28], a decomposition based environmental selection with greedy strategy is employed in SMOEA/D. The details are shown in Algorithm 5. In the approach, based on a set of weight vectors $\Lambda = \{\lambda^1, \dots, \lambda^N\}$, a Tchebycheff decomposition approach [51] is used to construct N subproblems (Line 3). While updating, y is firstly utilized to update z^* (Line 1). Afterwards, for the ith subproblem, $i = 1, 2, \dots, N$, the function values $g(y)$, $g(x^i)$ and improvable degree $de(i) = g(y) - g(x^i)$ are calculated (Lines 2–5). Finally, de is sorted in descending order, and two solutions with highest de are replaced by y. This operation means that y replaces two solutions with the largest improvement room, and it shows a greedy property (Lines 6–9).

Algorithm 5. Update(y, A, z^*, N, q).

```
1 Update ideal point $z^*$: If $z_j^* > f_j(y)$, set $z_j^* = f_j(y), j = 1, 2, \dots, m$.
2 for $i=1$ to $N, \lambda^i = (\lambda_1^i, \lambda_2^i, \dots, \lambda_m^i) \in \Lambda, x^i \in A$ do
3     Calculate objective value of subproblem $i$ for $y$ and $x^i$ :
      $g(y|\lambda^i, z^*) = \max_{1 \le j \le m}\{\lambda_j^i * |f_j(y) - z_j^*|\}$,
      $g(x^i|\lambda^i, z^*) = \max_{1 \le j \le m}\{\lambda_j^i * |f_j(x^i) - z_j^*|\}$.
4     Calculate improvable degree : $de(i) = g(y|\lambda^i, z^*) - g(x^i|\lambda^i, z^*)$.
5 end
6 Sort $de$ in descending order and keep the sorted indices in $I$.
7 if $de(I(k)) > 0, k = 1, 2$ then
8     Replace $x^{I(k)} \in A : x^{I(k)} = y$.
9 end
10 Return $A, z^*$
```

It needs to be commented that z^* is initialized by $z_j^* = \min\{f_j(x) | x \in P_0\}, j = 1, 2, \dots, m$, where P_0 is the initial population. Let $\omega^i = (\omega_1^i, \dots, \omega_m^i), i = 1, 2, \dots, N$, be a set of evenly distributed vectors that satisfy $\sum_{j=1}^m \omega_j^i = 1$, and $\omega_j^i > 0$.[1] Moreover, λ_j^i is defined as

$$\lambda_j^i = \frac{\frac{1}{\omega_j^i}}{\sum_{k=1}^m \frac{1}{\omega_k^i}} \tag{3}$$

where $j = 1, 2, \dots, m$, and $i = 1, 2, \dots, N$. In the original MOEA/D [20] and MOEA/D-DE [51], the weights ω^i are directly employed to build the subproblems. However, recent studies present that the subproblems built by λ^i in Eq. (3) can lead to better distributed final population [53,54]. Thus, the Tchebycheff decomposition with λ^i is adopted in SMOEA/D, which is the essential difference with the solution update mechanism designed in [28]. More details on generating weight vectors ω^i, $i = 1, 2, \dots, N$ refer to [20].

Algorithm 6. Probability(t, CN, RN, G).

```
1 Calculate the utility of $v^k$, for $k = 1, \dots, K$:
  $u^{t,k} = \begin{cases} \frac{\sum_{i=1}^t rn^{i,k}}{\sum_{i=1}^t cn^{i,k}} & \text{if } t < G \\ \frac{\sum_{i=t-G+1}^t rn^{i,k}}{\sum_{i=t-G+1}^t cn^{i,k}} & \text{otherwise} \end{cases}$.
2 Calculate the probability, for $k = 1, \dots, K$:
  $\beta^{t+1,k} = \frac{u^{t,k} + \epsilon}{\sum_{j=1}^K u^{t,j} + \epsilon}$.
3 return $\beta^{t+1,k}, k = 1, \dots, K$.
```

3.4. *Neighborhood ensemble*

It is well-known that the neighborhood is a crucial component in the MOEA/D. In the original MOEA/D and most of its variants [20,51,43], there exists only one neighborhood size. In fact, different subproblems require different neighborhood sizes. For certain problems, using various neighborhoods at different stages may improve the algorithm performance. When some solutions are trapped in a locally optimal region, a large neighborhood is required to increase diversity for helping the solutions to escape the trapped region. However, if the globally optimal region has been found, a small neighborhood will be helpful for local exploitation [55]. To improve the performance, we design an ensemble consisted of neuron neighborhoods with K different NNSs in SMOEA/D. Algorithm 6 shows the update on the probability of selecting different NNSs.

At the tth generation, the utility of generating solutions using $v^k, k = 1, \dots, K$, over the last G generations is firstly calculated (Line 1). Afterwards, the probability $\beta^{t+1,k}$ of selecting the kth NNS for the next generation is updated based on the obtained utility (Line 2). In the algorithm, $\epsilon = 10^{-10}$ is used to keep the division legal. The calculations of CN, RN refer to Lines 5, 13, 14 and 16 in Algorithm 1. It should be noted that, while getting RN, the number of solutions in A' which could be replaced by the new solution y needs to be counted. The approach is given in Algorithm 7.

Algorithm 7. Count(y, A', z', N).

```
1 for $i=1$ to $N, \lambda^i = (\lambda_1^i, \lambda_2^i, \dots, \lambda_m^i) \in \Lambda, x^i \in A'$ do
```

[1] If $\omega_j^i = 0$, it is reset to $\omega_j^i = 10^{-5}$ to make the Eq. (3) legal.

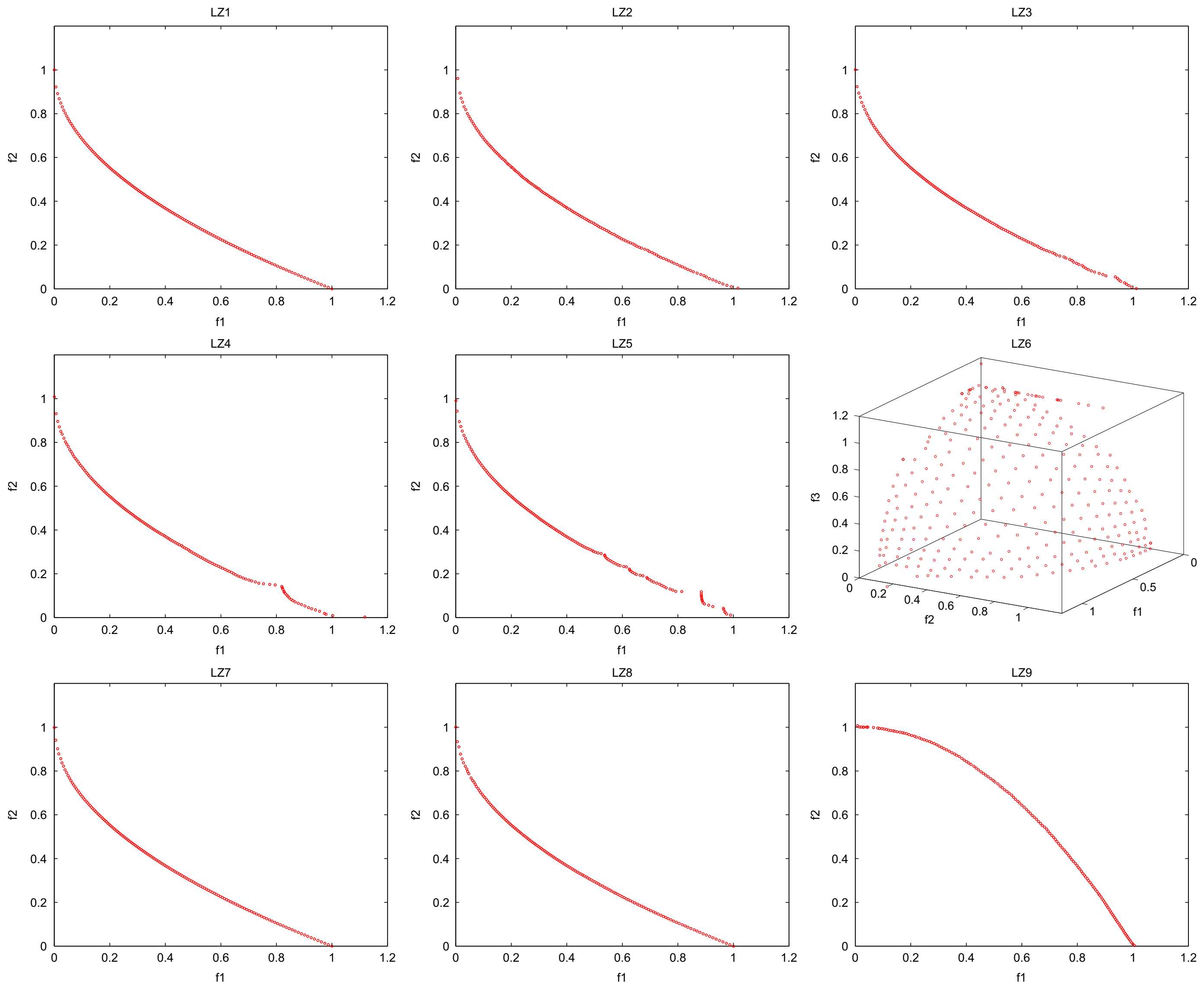

Fig. 3. Final approximations in the objective space with the lowest *IGD* values obtained by SMOEA/D in 30 runs on LZ1–LZ9.

```
2     Calculate objective value of subproblem $i$ for $y$ and $x^i$ :
      $g(y|\lambda^i,z')=\max_{1\le j\le m}\{\lambda^i_j*|f_j(y)-z'_j|\}$,
      $g(x^i|\lambda^i,z')=\max_{1\le j\le m}\{\lambda^i_j*|f_j(x^i)-z'_j|\}$.
3     Calculate improvable degree : $de(i)=g(y|\lambda^i,z')-g(x^i|\lambda^i,z')$.
4 end
5 Set $r=0$.
6 If $de(i)>0$, Set $r=r+1, i=1,2,\ldots,N$.
7 return $r$.
```

3.5. Computational complexity analysis

SMOEA/D includes a SOM operator and adopts a new update strategy, which have higher computational complexities. At each generation, the *Mapping* operation (Line 4 in Algorithm 1) requires *O(nND)* computations, where *n* is the variable dimension, *N* is the population size, and *D* is the number of neurons. *Update* operation (Line 4 in Algorithm 1) requires $O(mN^2)$ computations, where *m* is the number of objectives. Other operations have smaller complexity. Since *n* is usually much larger than *m*, and *D* is similar or equal to *N*, thus the complexity of SMOEA/D at each generation is $O(nND)$.

4. Experiment study

4.1. Test instances and performance metrics

To better present the performance of SMOEA/D, we utilize the problems LZ1–LZ9 with complicated PS shapes, which were designed when proposing the popular MOEA/D-DE [51], as the test instances. In LZ problems, LZ6 is a triobjective MOP, and the others are with two objectives. LZ6 and LZ9 have concave PFs, and the others have convex ones. LZ7 and LZ8 are the multi-peaks problems which have multiple local optimal PFs.

When tackling a MOP, a set of solutions with good convergence and diversity are desirable. Two widely used metrics, *inverted generational distance* (*IGD*) [26] and *hypervolume* (*HV*) [16], for measuring both convergence and diversity are employed to assess the performance of SMOEA/D.

Let P^* be a set of uniformly distributed Pareto optimal points in the theoretical PF, and *P* be an obtained nondominated front

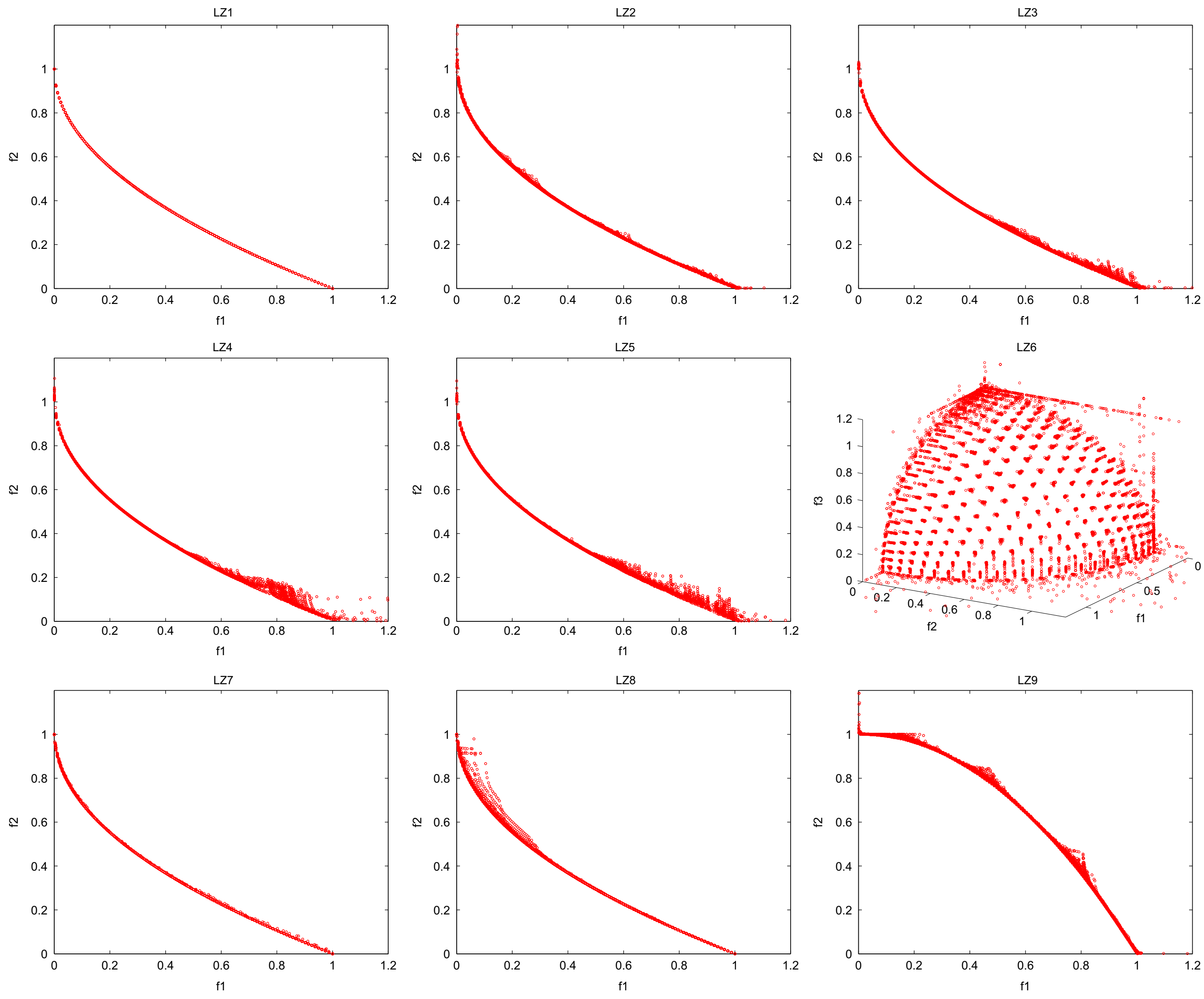

Fig. 4. Plots of the final approximations in the objective space in 30 runs on LZ1–LZ9.

of a MOP. The *IGD* metric is defined as

$$IGD(P^*,P)=\frac{\sum_{x\in P^*}d(x,P)}{|P^*|},$$

where $d(x,P)$ is the minimal distance between x and any point in P, and $|P^*|$ is the cardinality of P^*. Apparently, P^* should be known when using this metric. A lower *IGD* value is desirable.

The definition of *HV* metric is as follows:

$$HV(P,r)=VOL\left(\bigcup_{x\in P}[f_1(x),r_1]\times\ldots[f_m(x),r_m]\right),$$

where $r=(r_1,\ldots,r_m)$ denotes a reference point in the objective space that is dominated by any Pareto optimal points, and $VOL(\cdot)$ is the Lebesgue measure. *HV* metric measures the size of the objective space dominated by the solutions in P and bounded by r. While calculating this metric, $r=(1,1)$ is set for all the LZ problems except LZ6, and $r=(1,1,1)$ is given for LZ6. Solutions with larger *HV* value are desirable.

In addition, in the following experiments, to get statistically sound conclusions, we used each algorithm to independently run each instance for 30 times, and take the statistical metric values to do the comparison. In the comparison table, the best mean metric values are highlighted in bold face with gray background. The Wilcoxon's rank sum test at a 5% significance level is also performed to compare the significance of differences between the mean values of two algorithms.

4.2. Experimental settings

Since SMOEA/D is a decomposition based MOEA based on the regularity property, we choose a typical regularity model based MOEA, RM-MEDA [26], and a popular decomposition based MOEA, MOEA/D-DE [51], as the comparison algorithms. Since the Λ used in SMOEA/D can naturally improve the performance of the algorithms designed based on the MOEA/D, the Λ is also adopted in MOEA/D-DE to ensure a fair comparison. All the algorithms are implemented in MATLAB with their experimental settings are as follows.

- *Public parameters:*
  - population size N: 150 (300) for biobjective (triobjective) MOPs;
  - variable dimension n: 30 for LZ1–LZ5, LZ9; 10 for LZ6–LZ8;

Table 1
Statistical results (mean(std.)) of RM-MEDA, MOEA/D-DE and SMOEA/D, over 30 independent runs for LZ test instances in terms of the *IGD* and *HV* metrics. Wilcoxon's rank sum test at a 0.05 significance level is performed between SMOEA/D and the other two algorithms.

Instances	RM-MEDA	MOEA/D-DE	SMOEA/D
	IGD		
LZ1	3.114E−03(8.804E−05)†	**2.588E−03(7.173E−06)** ≈	2.590E−03(9.054E−06)
LZ2	4.080E−02(8.726E−03)†	8.736E−03(1.539E−02)†	**5.068E−03(6.297E−04)**
LZ3	2.241E−02(9.124E−03)†	2.367E−02(2.917E−02)†	**6.495E−03(1.088E−03)**
LZ4	2.794E−02(1.032E−02)†	2.424E−02(2.432E−02) ≈	**9.057E−03(2.284E−03)**
LZ5	2.049E−02(5.703E−03)†	1.720E−02(1.102E−02)†	**1.097E−02(2.717E−03)**
LZ6	4.925E−02(2.926E−03)†	3.423E−02(2.146E−03) ≈	**3.385E−02(1.781E−03)**
LZ7	7.501E−02(6.943E−02)†	**2.632E−03(9.379E−05)**§	2.814E−03(7.408E−04)
LZ8	2.111E−01(1.273E−01)†	4.107E−03(7.727E−03)†	**3.823E−03(2.188E−03)**
LZ9	4.419E−02(7.330E−03)†	7.205E−03(2.070E−03) ≈	**6.802E−03(1.652E−03)**
†	9	4	
§	0	1	
≈	0	4	
	HV		
LZ1	6.615E−01(2.124E−04)†	**6.630E−01(3.570E−05)** ≈	6.629E−01(4.175E−05)
LZ2	5.975E−01(1.529E−02)†	6.550E−01(7.829E−03)†	**6.577E−01(9.771E−04)**
LZ3	6.403E−01(4.769E−03)†	6.434E−01(2.265E−02) ≈	**6.564E−01(1.539E−03)**
LZ4	6.322E−01(4.758E−03)†	6.412E−01(2.013E−02) ≈	**6.534E−01(2.172E−03)**
LZ5	6.416E−01(3.422E−03)†	6.449E−01(8.520E−03)†	**6.506E−01(3.753E−03)**
LZ6	3.831E−01(5.920E−03)†	4.198E−01(5.731E−03)†	4.247E−01(**3.310E−03**)
LZ7	5.537E−01(9.620E−02)†	**6.628E−01(2.319E−04)**§	6.621E−01(1.205E−03)
LZ8	3.466E−01(1.511E−01)†	**6.607E−01(8.572E−03)**§	6.592E−01(5.601E−03)
LZ9	2.674E−01(9.178E−03)†	**3.218E−01(2.323E−03)** ≈	3.217E−01(2.789E−03)
†	9	3	
§	0	2	
≈	0	4	

Note: "†", "§", and " ≈ " denote that the performance of SMOEA/D is better than, worse than, and similar to that of comparison algorithm, respectively. The data in bold face with gray background in the table are the best mean metric values obtained by three algorithms for each instance.

 - ○ maximal evolutionary generation T: 1000.
- *Parameters of SMOEA/D:*
  - ○ ensemble of NNSs $\{v^1, \ldots, v^4\}$: {5,10,20,40};
  - ○ initial probability of selecting NNSs $\{\beta^{0,1}, \ldots, \beta^{0,4}\}$: {0.25,0.25, 0.25,0.25};
  - ○ initial learning rate τ_0: 0.8;
  - ○ DE control parameter: $F = 0.5$;
  - ○ polynomial mutation control parameters: mutation probability $p_m = 1/n$, and distribution index $\eta_m = 20$;
  - ○ History length: $G = 10$.
- *Parameters of RM-MEDA:*
  The parameters are set the same as those in [26],
  - ○ number of clusters in Local PCA algorithm: 5;
  - ○ maximal iteration number: 50;
  - ○ extension rate of sampling: 0.25.
- *Parameters of MOEA/D-DE:*
  The parameters are set the same as those in [51],
  - ○ size of the neighborhood of each weight vector: 20;
  - ○ probability of selecting parent solutions from neighborhood: 0.9;
  - ○ maximal number of solutions replaced by each child solution: 2;
  - ○ control parameters for DE and polynomial mutation operators are the same as those in SMOEA/D.

4.3. Experimental results

Results of using SMOEA/D to deal with the test instances are presented in Figs. 3 and 4. Plots in Fig. 3 are the best approximations in the objective space. Plots in Fig. 4 are all the approximations of PFs in 30 runs. From Fig. 3, we can see that the obtained best solutions of LZ1–LZ2 and LZ7–LZ9 approximate the PFs well. For LZ3–LZ5, most parts of the PFs are approximated successfully, and only a small number of non-approximated parts are remained. With respect to LZ6, most of the PF surface have been covered, and only a few points on the approximation are missing. Fig. 4 shows that, for each instance, through repeatedly runs, the final solutions obtained by SMOEA/D are able to approximate its whole PF. According to Figs. 3 and 4, it is easy to find that, even though the PS shapes of LZ1–LZ9 are complicated, which brings difficulties for being solved, SMOEA/D still managed to tackle them effectively.

4.4. Comparison analysis

We compare SMOEA/D with RM-MEDA and MOEA/D-DE to reveal its performance further. The statistical results obtained by three algorithms on nine test instances are given in Table 1. It is clear that among the 18 best metric values, SMOEA/D and MOEA/D-DE yields 12 and 6 ones, respectively. Moreover, RM-MEDA cannot obtain any one of them. Specifically, compared with RM-MEDA, SMOEA/D has more advantages in solving all the instances with respect to both metrics. Compared with MOEA/D-DE, SMOEA/D yields better *IGD* values for LZ2–LZ6 and LZ8–LZ9, and achieves more superior *HV* values for LZ2–LZ6. All the better metric values have statistical significance. It can be concluded from Table 1 that SMOEA/D has the best convergence quality among the three algorithms.

To evaluate the convergence speed, we plot the evolution of mean *IGD* metric values of RM-MEDA, MOEA/D-DE and SMOEA/D versus generations in Fig. 5. The plots show that, compared with the other two algorithms, SMOEA/D converges fastest for all the instances except LZ7 and LZ8. Especially for LZ2–LZ6, SMOEA/D is able to obtain the lowest mean *IGD* metric values after evolving a small number of generations. With respect to LZ7 and LZ8, SMOEA/D performs worse than MOEA/D-DE at the early stage; but at the late stage, it starts to show better performance.

According to above visual comparison and statistical analysis, we can claim that SMOEA/D is more efficient to deal with the MOPs with complicated PS shapes. In addition, it also indirectly denotes the design of SRM and neighborhood ensemble contributes to improving the MOEA.

5. Discussions

5.1. More instances with simple PSs

To test the performance of SMOEA/D on addressing the MOPs with simple PS shapes further, we have also selected ZDT [56] and DTLZ [57] test instances, which have simple PF and PS shapes, to conduct the comparison experiments. The variable dimensions are set as 30 for ZDT1–ZDT2, 10 for ZDT4 and ZDT6, 7 for DTLZ1, and 12 for DTLZ2–DTLZ4. The parameter settings of RM-MEDA, MOEA/D-DE and SMOEA/D are the same as those in Section 4.2, except that the maximum generation is set as 500. Statistical results are presented in Table 2. From the table, we can notice that, SMOEA/D yields nine out of the sixteen best metric values. But RM-MEDA and MOEA/D-DE achieves only one and six ones. With respect to the *IGD* metric, SMOEA/D outperforms RM-MEDA on all the instances except DTLZ4, and it is superior to MOEA/D-DE on ZDT4, ZDT6, DTLZ1, DTLZ3 and DLTZ4. For the *HV* metric, SMOEA/D performs better than RM-MEDA on all the instances, and it also shows more advantages than MOEA/D-DE in tackling ZDT4, ZDT6 and DTLZ1-DTLZ3. It needs to be noted that the reference points for ZDT and DTLZ instances are set as [1,1] and [1,1,1], respectively, while calculating the *HV* values. The experimental results indicate that compared with RM-MEDA and MOEA/D-DE, SMOEA/D can also deal with the MOPs with simple PS shapes more effectively.

5.2. WFG test instances

It is known that WFG [58] test instances are difficult and have complex PF shapes. To reveal the performance of SMOEA/D on these types of problems, we employ RM-MEDA, MOEA/D-DE and SMOEA/D to address the WFG instances with two objectives and 30 variables. The experimental settings are the same as those in Section 5.1. In addition, when calculating the *HV* metric, the reference points are set as [3,5] for all the instances. The experimental results are shown in Table 3. The table shows that SMOEA/

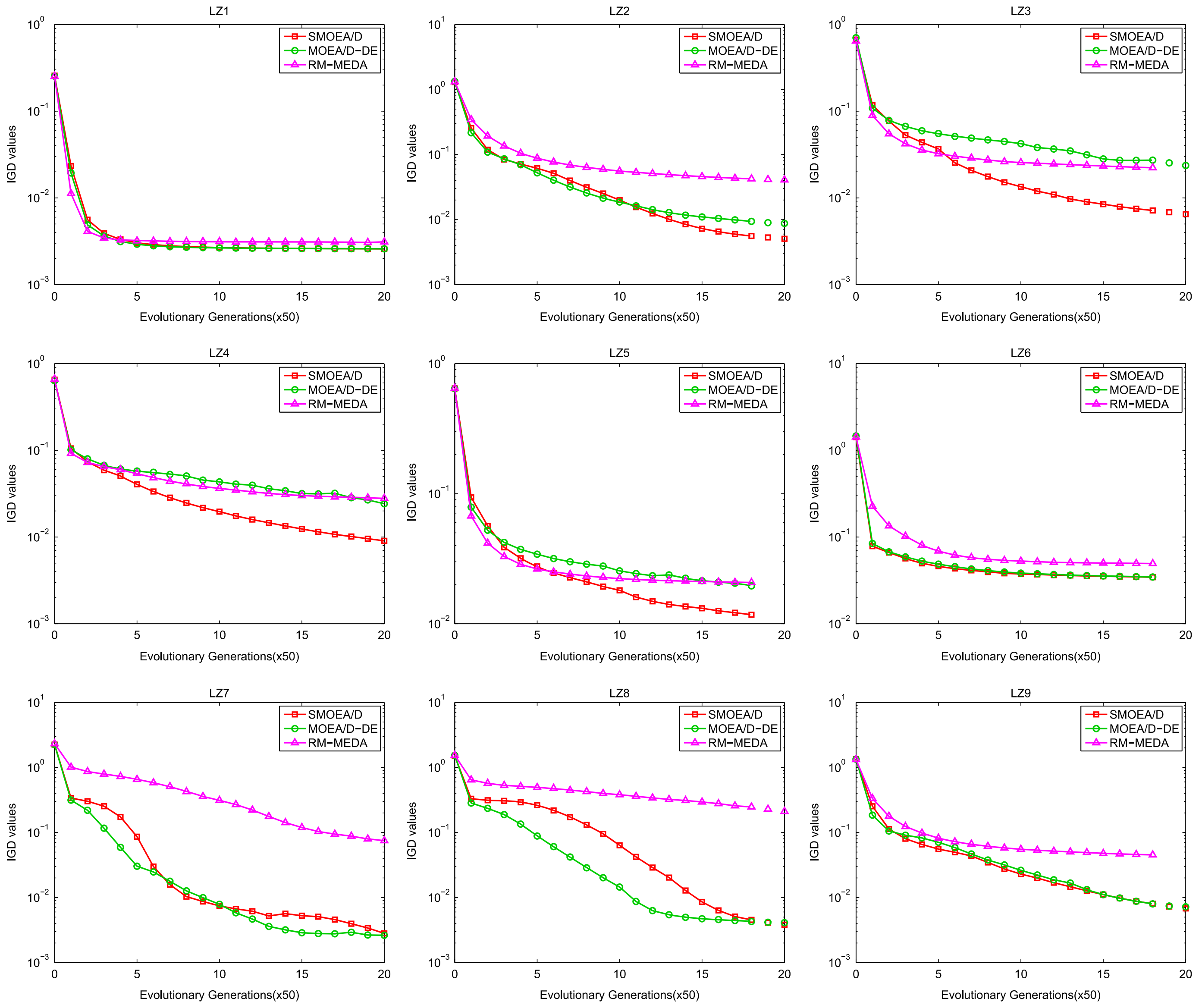

Fig. 5. Evolution of the mean *IGD* metric values versus the generations.

Table 2
Statistical results (mean(std.)) of RM-MEDA, MOEA/D-DE and SMOEA/D, over 30 independent runs for ZDT and DTLZ test instances in terms of the *IGD* and *HV* metrics. Wilcoxon's rank sum test at a 0.05 significance level is performed between SMOEA/D and the other two algorithms.

Instances	RM-MEDA	MOEA/D-DE	SMOEA/D
	IGD		
ZDT1	7.474E−03(1.123E−03)†	**3.212E−03(4.265E−04)**§	3.773E−03(5.215E−04)
ZDT2	4.801E−02(1.594E−01)†	**2.994E−03(1.694E−04)**§	3.813E−03(7.923E−04)
ZDT4	1.346E+01(3.074E+00)†	3.061E−03(3.355E−04)†	**2.751E−03(5.283E−05)**
ZDT6	2.075E−01(3.259E−02)†	2.908E−03(5.592E−04)†	**2.059E−03(4.496E−07)**
DTLZ1	7.600E+00(1.983E+00)†	1.119E−02(3.751E−04)†	**1.077E−02(3.768E−05)**
DTLZ2	4.070E−02(7.701E−04)†	**2.993E−02(1.528E−04)** ≈	2.999E−02(2.096E−04)
DTLZ3	1.633E+02(2.169E+01)†	3.357E−01(8.265E−01) ≈	**3.646E−02(9.751E−03)**
DTLZ4	**1.903E−02(7.180E−04)**§	1.995E−02(4.175E−03)†	1.973E−02(5.945E−03)
†	7	4	
§	1	2	
≈	0	2	
	HV		
ZDT1	6.540E−01(1.692E−03)†	**6.613E−01(9.493E−04)**§	6.600E−01(9.592E−04)
ZDT2	3.009E−01(8.187E−02)†	**3.281E−01(4.986E−04)**§	3.264E−01(1.573E−03)
ZDT4	0.000E+00(0.000E+00)†	6.620E−01(7.828E−04)†	**6.626E−01(2.037E−04)**
ZDT6	1.120E−01(2.503E−02)†	3.217E−01(9.774E−04)†	**3.237E−01(1.264E−07)**
DTLZ1	0.000E+00(0.000E+00)†	9.746E−01(3.333E−03)†	**9.762E−01(2.936E−05)**
DTLZ2	4.090E−01(1.872E−03)†	4.391E−01(2.966E−04)†	**4.396E−01(2.488E−04)**
DTLZ3	0.000E+00(0.000E+00)†	3.552E−01(1.601E−01)†	**4.269E−01(1.769E−02)**
DTLZ4	4.155E−01(1.839E−03)†	**4.338E−01(2.035E−02)**§	4.321E−01(2.854E−02)
†	8	5	
§	0	3	
≈	0	0	

Note: "†", "§", and " ≈ " denote that the performance of SMOEA/D is better than, worse than, and similar to that of comparison algorithm, respectively. The data in bold face with gray background in the table are the best mean metric values obtained by three algorithms for each instance.

Table 3
Statistical results (mean(std.)) of RM-MEDA, MOEA/D-DE and SMOEA/D, over 30 independent runs for WFG test instances in terms of the *IGD* and *HV* metrics. Wilcoxon's rank sum test at a 0.05 significance level is performed between SMOEA/D and the Other Two Algorithms.

Instances	RM-MEDA	MOEA/D-DE	SMOEA/D
	IGD		
WFG1	**1.165E+00(6.707E−03)** ≈	1.189E+00(8.687E−03)†	1.168E+00(1.008E−02)
WFG2	**2.170E−02(4.282E−03)**§	2.758E−02(1.016E−03)§	2.854E−02(1.433E−03)
WFG3	2.354E−02(4.937E−03)†	1.317E−02(5.009E−04)†	**1.279E−02(5.479E−04)**
WFG4	9.077E−02(2.064E−03)†	8.352E−02(6.897E−03)†	**5.269E−02(7.898E−03)**
WFG5	8.761E−02(7.130E−03)†	6.635E−02(2.210E−04)†	**6.604E−02(4.881E−05)**
WFG6	**3.198E−01(3.230E−03)**§	3.731E−01(1.693E−02)§	3.784E−01(1.186E−04)
WFG7	2.669E−02(5.798E−03)†	1.383E−02(4.532E−04)†	**1.304E−02(3.220E−04)**
WFG8	1.449E−01(1.380E−02)†	**3.521E−02(8.968E−03)** ≈	3.700E−02(7.700E−03)
WFG9	**2.050E−01(2.726E−03)**§	2.122E−01(1.166E−02)§	2.697E−01(1.298E−02)
†	5	5	
§	3	3	
≈	1	1	
	HV		
WFG1	5.637E+00(4.454E−02) ≈	5.545E+00(3.836E−02)†	**5.645E+00(4.245E−02)**
WFG2	1.132E+01(3.228E−02)†	**1.143E+01(3.330E−03)**§	1.142E+01(5.643E−03)
WFG3	1.084E+01(3.256E−02)†	1.091E+01(4.587E−03)†	**1.092E+01(4.833E−03)**
WFG4	8.134E+00(1.605E−02)†	8.274E+00(3.858E−02)†	**8.419E+00(4.257E−02)**
WFG5	8.125E+00(5.650E−02)†	8.196E+00(5.886E−02)†	**8.290E+00(1.517E−03)**
WFG6	**6.416E+00(1.756E−02)**§	6.150E+00(8.637E−02)§	6.123E+00(4.763E−04)
WFG7	8.544E+00(3.841E−02)†	8.647E+00(4.120E−03)†	**8.657E+00(3.189E−03)**
WFG8	7.782E+00(8.538E−02)†	8.491E+00(5.717E−02) ≈	**8.498E+00(4.298E−02)**
WFG9	**6.431E+00(1.978E−02)**§	6.332E+00(6.745E−02)§	6.126E+00(6.520E−02)
†	6	5	
§	2	3	
≈	1	1	

Note: "†", "§", and " ≈ " denote that the performance of SMOEA/D is better than, worse than, and similar to that of comparison algorithm, respectively. The data in bold face with gray background in the table are the best mean metric values obtained by three algorithms for each instance.

D obtains 10 out of 18 best metric values, but RM-MEDA and MOEA/D-DE only yield six and two ones, respectively. Specifically, compared with RM-MEDA, SMOEA/D performs better on 12 metric values, and 11 out of them have the statistical significance. Compared with MOEA/D-DE, SMOEA/D achieves 11 superior metric values, out of which 10 ones are significantly different. It can be concluded from the experimental results that SMOEA/D outperforms RM-MEDA and MOEA/D-DE on WFG test instances.

5.3. SRM analysis

To verify the contribution of the SRM operator, we develop a comparison algorithm, called as VARIANT1, by removing the design of the neighborhoods from SMOEA/D, i.e., there exist no SOM and neighborhood ensemble in VARIANT1. VARIANT1 is also employed to independently compute each test instance for 30 times. The box plots of *IGD* metric values for the instances obtained by two algorithms are shown in Fig. 6. The box plots indicate that the medians of *IGD* metric values for all the instances gained by SMOEA/D are significantly lower than those obtained by VARIANT1. Moreover, the interquartile ranges of *IGD* values for all the instances except LZ5 yielded by SMOEA/D are also shorter than those obtained by VARIANT1. An explicit conclusion could be drawn that SRM is able to dramatically improve the performance of a MOEA. The reason for this improvement is that the information of PS structures of MOPs is successfully discovered by the SOM, and applying the information to lead similar solutions to do recombination can generate high-quality new solutions. The conclusion of this experiment also validates that, the idea of utilizing a machine learning approach to identify the structures of solutions in population and designing specific reproduction operators for the MOEAs based on the structure information are promising.

5.4. Neighborhood ensemble analysis

In order to discover the contribution of the neighborhood ensemble, we design four variant algorithms, named as VARIANT2, VARIANT3, VARIANT4 and VARIANT5, by replacing the ensemble in SMOEA/D with four fixed NNSs, to perform the comparison analysis. The four NNSs are the ones used in the ensemble, i.e., NNSs = 5, 10, 20, 40. In addition to the mating selection from the neighborhood, it also chooses parents from the whole population with a low probability to enhance the diversity in each variant. The probability is set as 0.1, which is the same with that in MOEA/D-DE. Statistical results of the four variants on LZ test instances are presented in Table 4.

From the table, we can see that, VARIANT2 and VARIANT3, which have lesser NNSs, have the better performance among the variant algorithms. However, these two both perform poorer than SMOEA/D, since SMOEA/D yields four best (two second best) *IGD* values and five best (one second best) *HV* values. These results prove that the neighborhood ensemble successfully contributes to the performance of SMOEA/D.

To further reveal the update process of the probability for selecting different NNSs in the ensemble, we plot the mean variation curves of the probability during the whole evolution over 30 runs on LZ3, LZ6 and LZ9 in Fig. 7. It is easy to find that, with the

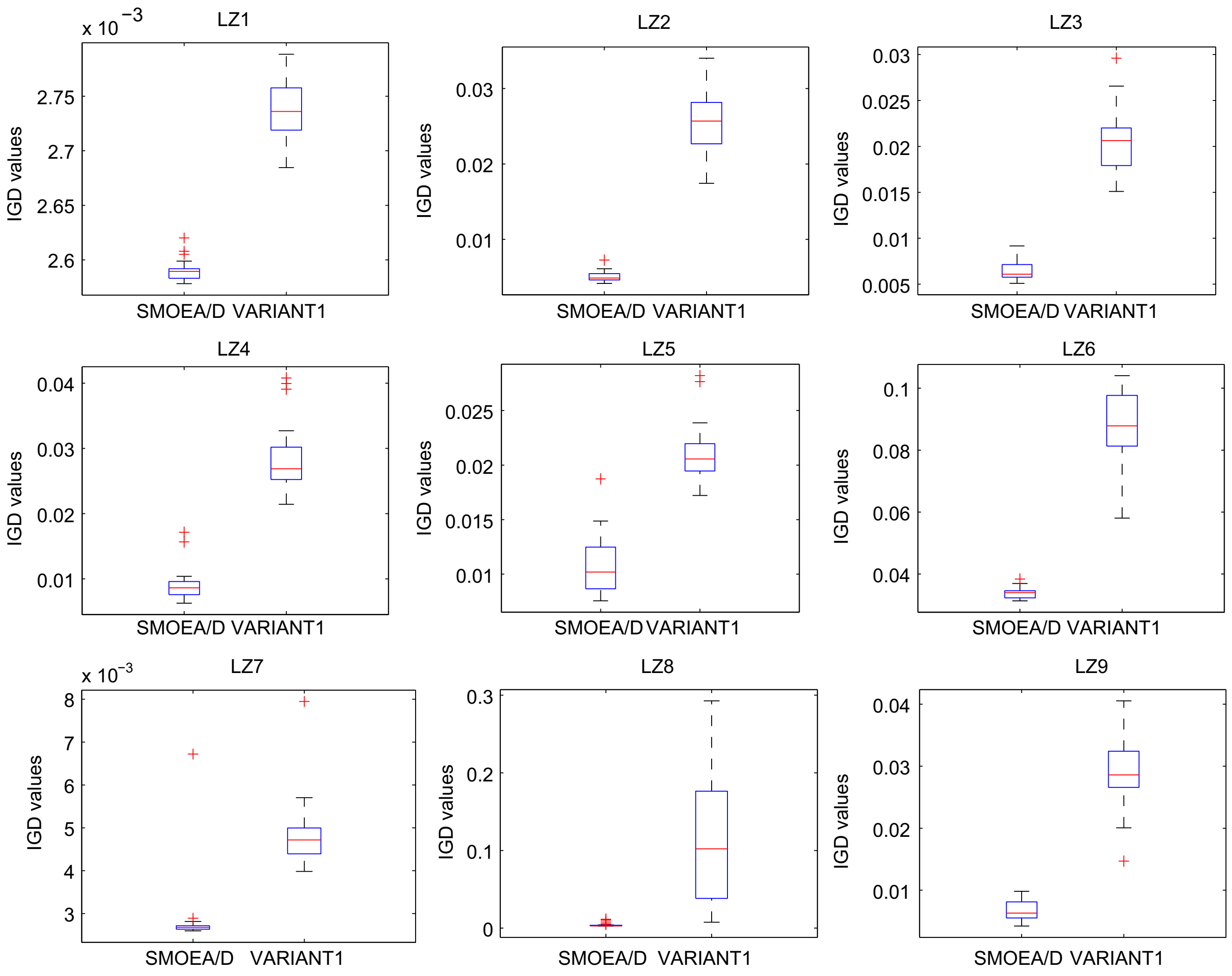

Fig. 6. The box plots of *IGD* metric values for nine instances obtained by SMOEA/D and VARIANT1.

increase of generations, the probability curves of lesser NNSs (NNSs = 5, 10) rise gradually, and those of larger NNSs (NNSs = 20, 40) decrease successively. The reason might be that, the similarity of the solutions in a small mating pool is much higher, and inbreeding within this types of mating pools does better in producing high-quality offsprings. With the convergence of the algorithm, the performance of other types of NNSs is worsen, and only the small ones can still generate superior new solutions, so that the probability of selecting small NNSs is risen. Plots in Fig. 7 also indirectly prove that our design on neighborhood ensemble is quite proper.

5.5. *Parameter sensitivity analysis*

In SMOEA/D, there are three parameters introduced by the SOM, i.e., the SOM size, the initial radius of domain for updating weights of neurons (σ_0), and the initial learning rate of SOM (τ_0). In addition, the history length G is also brought for neighborhood ensemble. The performance sensitivity of SMOEA/D to these four parameters is discussed in the following sections. All the experimental parameters are set as the same as those in Section 4.2 except that only representative LZ6 and LZ8–LZ9 are selected for σ_0, τ_0 and G analysis.

5.5.1. *SOM size*

In SMOEA/D, a square SOM size is adopted, i.e., the neurons in the output layer distribute in a square grid. To present the influence of SOM size on the SMOEA/D, we set another three rectangle SOM sizes and establish three variants of SMOEA/D, i.e., SMOEA/D-OL1, SMOEA/D-OL2 and SMOEA/D-OL3. Table 5 gives the details of the SOM sizes and the experimental results of SMOEA/D and its variants on LZ instances. According to the statistical significance test, we can find that these four algorithms do not have significant different performance on most of the instances. This might be that, in the SMOEA/D framework, it only requires the SOM to discover the neighboring relationships among the solutions, but not to get the precision and entire population structure. This requirement is so simple that the SOMs with different sizes all can meet the requirement of the SMOEA/D framework. However, from the metric values, it can be seen that the square SOM performs the best for the LZ problems. In consideration of that the square SOM has better performance and is convenient to be adopted, it is employed in SMOEA/D.

5.5.2. *Initial learning rate of SOM*

In the case of $G=10$ and $\sigma_0 = \sqrt{2C^2}/2$, the error bars of the *IGD* metric values obtained by SMOEA/D with different initial learning rate τ_0 over 30 independent runs are presented in Fig. 8. It can be seen from the figure that when $\tau_0 = 0.8$, SMOEA/D can perform better for all the three instances, which indicates that SMOEA/D is not quite sensitive to the setting of τ_0. However, the best τ_0 is problem-dependent.

5.5.3. *Initial neighborhood radius of SOM*

In previous experiments, the initial radius of domain for updating the neuron weights of SOM is set as $\sigma_0 = \sqrt{2C^2}/2$. To analyze the sensitivity of σ_0, we define a coefficient ξ, i.e., $\sigma_0 = \xi \times \sqrt{2C^2}$, to set different σ_0. While $G=10$ and $\tau_0 = 0.8$, the error bars of *IGD* metric values versus different ξ obtained by SMOEA/D over 30 runs are plotted in Fig. 9. The figure indicates that when $\xi = 0.5$, SMOEA/D performs the best on LZ8–LZ9. When $\xi = 0.8$, SMOEA/D does the best in LZ6 and LZ8. These results denote that ξ is not a quite sensitive parameter. But, the best setting is also problem-dependent.

Table 4
Statistical results (mean(std.)) of SMOEA/D and its four variants, over 30 independent runs for LZ test instances in terms of the *IGD* and *HV* metrics. Wilcoxon's rank sum test at a 0.05 significance level is performed between SMOEA/D and the other four algorithms.

Instances	VARIANT2 (*NNS* = 5)	VARIANT3 (*NNS* = 10)	VARIANT4 (*NNS* = 20)	VARIANT5 (*NNS* = 40)	SMOEA/D
	IGD				
LZ1	2.590E−03(7.706E−06) ≈	2.592E−03(6.550E−06)†	2.601E−03(7.941E−06)†	2.627E−03(1.037E−05)†	**2.590E−03(9.054E−06)**
LZ2	5.694E−03(6.386E−04)†	5.150E−03(8.653E−04) ≈	5.649E−03(9.878E−04)†	7.544E−03(1.808E−03)†	**5.068E−03(6.297E−04)**
LZ3	6.137E−03(9.981E−04) ≈	**5.998E−03(6.334E−04)** ≈	7.092E−03(9.330E−04)†	9.276E−03(1.078E−03)†	6.495E−03(1.088E−03)
LZ4	**9.030E−03(2.574E−03)** ≈	9.961E−03(2.915E−03) ≈	1.146E−02(2.054E−03)†	1.496E−02(2.575E−03)†	9.057E−03(2.284E−03)
LZ5	**1.077E−02(2.465E−03)** ≈	1.086E−02(2.509E−03) ≈	1.166E−02(1.660E−03) ≈	1.365E−02(1.940E−03)†	1.097E−02(2.717E−03)
LZ6	**3.340E−02(1.620E−03)** ≈	3.406E−02(2.845E−03) ≈	3.470E−02(2.524E−03) ≈	3.892E−02(3.659E−03)†	3.385E−02(1.781E−03)
LZ7	2.717E−03(7.850E−05)§	**2.697E−03(6.178E−05)** ≈	2.718E−03(4.696E−05)§	2.949E−03(1.154E−04)†	2.814E−03(7.408E−04)
LZ8	6.417E−03(5.440E−03)†	8.840E−03(1.550E−02) ≈	4.470E−03(3.016E−03)†	5.173E−03(5.286E−03)†	**3.823E−03(2.188E−03)**
LZ9	8.493E−03(2.503E−03)†	7.264E−03(1.860E−03) ≈	7.641E−03(2.350E−03) ≈	1.248E−02(3.347E−03)†	**6.802E−03(1.652E−03)**
†	3	1	5	9	
§	1	0	1	0	
≈	5	8	3	0	
	HV				
LZ1	6.629E−01(4.471E−05) ≈	6.629E−01(3.709E−05) ≈	6.629E−01(4.075E−05)†	6.627E−01(4.812E−05)†	**6.629E−01(4.175E−05)**
LZ2	6.567E−01(8.630E−04)†	6.575E−01(1.331E−03) ≈	6.567E−01(1.486E−03)†	6.536E−01(2.943E−03)†	**6.577E−01(9.771E−04)**
LZ3	6.570E−01(1.428E−03) ≈	**6.571E−01(8.946E−04)** ≈	6.556E−01(1.092E−03)†	6.523E−01(1.513E−03)†	6.564E−01(1.539E−03)
LZ4	**6.537E−01(2.738E−03)** ≈	6.523E−01(2.906E−03) ≈	6.498E−01(2.337E−03)†	6.447E−01(2.970E−03)†	6.534E−01(2.172E−03)
LZ5	**6.511E−01(3.093E−03)** ≈	6.509E−01(3.279E−03) ≈	6.496E−01(2.624E−03) ≈	6.468E−01(2.470E−03)†	6.506E−01(3.753E−03)
LZ6	4.240E−01(2.893E−03) ≈	4.235E−01(4.192E−03) ≈	4.226E−01(4.652E−03) ≈	4.141E−01(6.627E−03)†	**4.247E−01(3.310E−03)**
LZ7	6.622E−01(2.148E−04) ≈	**6.622E−01(2.464E−04)** ≈	6.621E−01(1.618E−04)§	6.614E−01(3.338E−04)†	6.621E−01(1.205E−03)
LZ8	6.534E−01(1.329E−02) ≈	6.502E−01(2.800E−02) ≈	6.574E−01(7.130E−03)†	6.567E−01(9.840E−03)†	**6.592E−01(5.601E−03)**
LZ9	3.187E−01(4.671E−03)†	3.208E−01(3.308E−03) ≈	3.199E−01(4.212E−03) ≈	3.105E−01(6.281E−03)†	**3.217E−01(2.789E−03)**
†	2	0	5	9	
§	0	0	1	0	
≈	7	9	3	0	

Note: "†", "§", and " ≈ " denote that the performance of SMOEA/D is better than, worse than, and similar to that of comparison algorithm, respectively. The data in bold face with gray background (light gray background), in the table are the best (second best), mean metric values obtained by five algorithms for each instance.

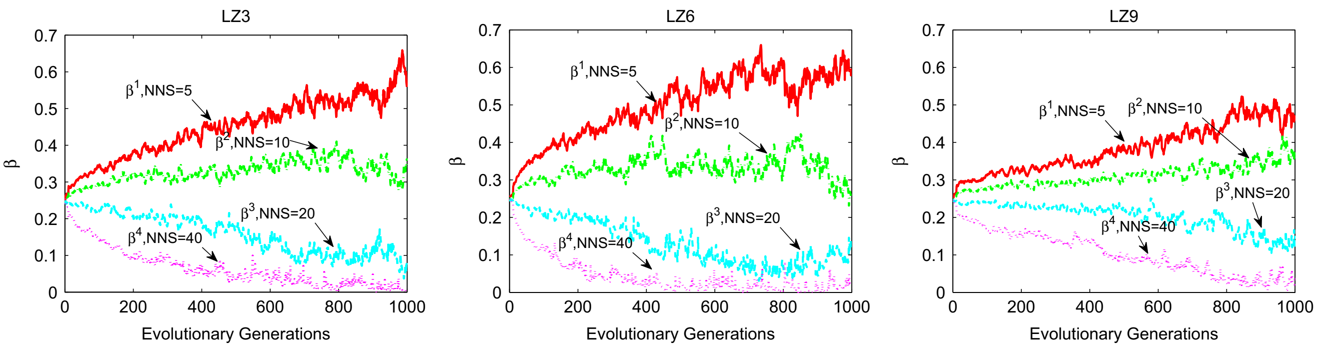

Fig. 7. Variation curves of β versus the generations.

Table 5
Statistical results (mean(std.)) of SMOEA/D and its three variants with different SOM sizes, over 30 independent runs for LZ test instances in terms of the *IGD* and *HV* metrics. Wilcoxon's rank sum test at a 0.05 significance level is performed between SMOEA/D and the other three algorithms.

Instances	SMOEA/D-OL1 Biobjective :2 × 75; Triobjective :2 × 150	SMOEA/D-OL2 Biobjective :5 × 30; Triobjective :5 × 60	SMOEA/D-OL3 Biobjective :10 × 15; triobjective :10 × 30	SMOEA/D Biobjective :12 × 12; Triobjective :17 × 17
	IGD			
LZ1	**2.586E−03(1.764E−05)**§	2.592E−03(8.354E−06)≈	**2.589E−03(6.600E−06)**≈	2.590E−03(9.054E−06)
LZ2	5.473E−03(8.002E−04)†	5.529E−03(1.007E−03)†	**5.004E−03(6.440E−04)**≈	**5.068E−03(6.297E−04)**
LZ3	8.873E−03(1.168E−02)≈	7.263E−03(1.503E−03)†	**7.174E−03(1.520E−03)**†	**6.495E−03(1.088E−03)**
LZ4	1.318E−02(2.483E−02)≈	**9.375E−03(2.006E−03)**≈	1.017E−02(2.849E−03)≈	**9.057E−03(2.284E−03)**
LZ5	**1.136E−02(3.253E−03)**≈	1.184E−02(2.613E−03)≈	1.174E−02(3.535E−03)≈	**1.097E−02(2.717E−03)**
LZ6	**3.315E−02(1.660E−03)**≈	3.371E−02(2.144E−03)≈	**3.357E−02(1.741E−03)**≈	3.385E−02(1.781E−03)
LZ7	**2.670E−03(8.298E−05)**≈	2.731E−03(9.875E−05)§	**2.692E−03(9.049E−05)**≈	2.814E−03(7.408E−04)
LZ8	**6.066E−03(1.057E−02)**≈	1.226E−02(1.457E−02)†	1.181E−02(1.894E−02)≈	**3.823E−03(2.188E−03)**
LZ9	8.899E−03(2.531E−03)†	8.697E−03(2.978E−03)†	**7.939E−03(2.206E−03)**≈	**6.802E−03(1.652E−03)**
†	2	4	1	
§	1	0	0	
≈	6	4	8	
	HV			
LZ1	**6.630E−01(7.622E−05)**§	6.629E−01(5.387E−05)≈	6.629E−01(3.183E−05)≈	**6.629E−01(4.175E−05)**
LZ2	6.570E−01(1.240E−03)≈	6.569E−01(1.601E−03)†	**6.577E−01(1.117E−03)**≈	**6.577E−01(9.771E−04)**
LZ3	6.544E−01(1.014E−02)≈	6.555E−01(1.870E−03)†	**6.557E−01(1.966E−03)**≈	**6.564E−01(1.539E−03)**
LZ4	6.514E−01(1.649E−02)†	**6.532E−01(2.145E−03)**≈	6.526E−01(2.691E−03)≈	**6.534E−01(2.172E−03)**
LZ5	**6.507E−01(3.340E−03)**≈	6.496E−01(3.609E−03)≈	6.498E−01(3.925E−03)≈	**6.506E−01(3.753E−03)**
LZ6	**4.258E−01(3.152E−03)**≈	4.251E−01(4.275E−03)≈	**4.252E−01(3.616E−03)**≈	4.247E−01(3.310E−03)
LZ7	**6.625E−01(2.236E−04)**§	6.622E−01(3.295E−04)≈	**6.623E−01(2.299E−04)**≈	6.621E−01(1.205E−03)
LZ8	**6.566E−01(1.224E−02)**≈	6.442E−01(2.006E−02)†	6.456E−01(3.433E−02)≈	**6.592E−01(5.601E−03)**
LZ9	3.187E−01(4.127E−03)†	3.186E−01(4.524E−03)†	**3.197E−01(3.804E−03)**†	**3.217E−01(2.789E−03)**
†	2	4	1	
§	2	0	0	
≈	5	5	8	

Note: "†", "§", and "≈" denote that the performance of SMOEA/D is better than, worse than, and similar to that of comparison algorithm, respectively. The data in bold face with gray background (light gray background), in the table are the best (second best), mean metric values obtained by five algorithms for each instance.

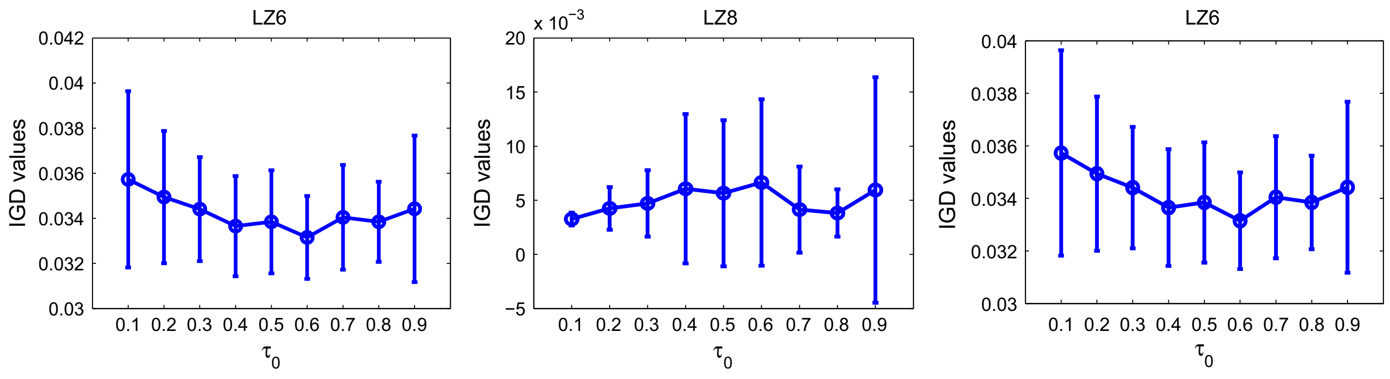

Fig. 8. The error bars of *IGD* metric values versus different τ_0 on LZ6 and LZ8–LZ9.

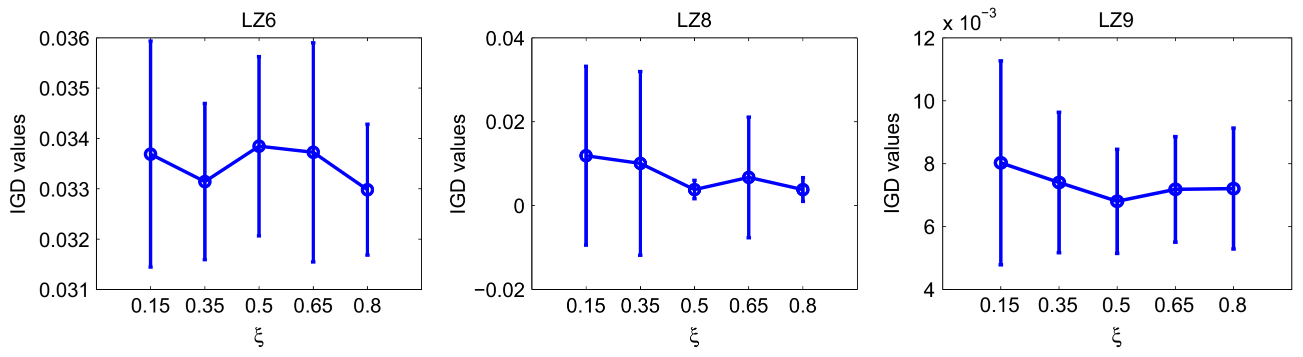

Fig. 9. The error bars of *IGD* metric values versus different ξ on LZ6 and LZ8–LZ9.

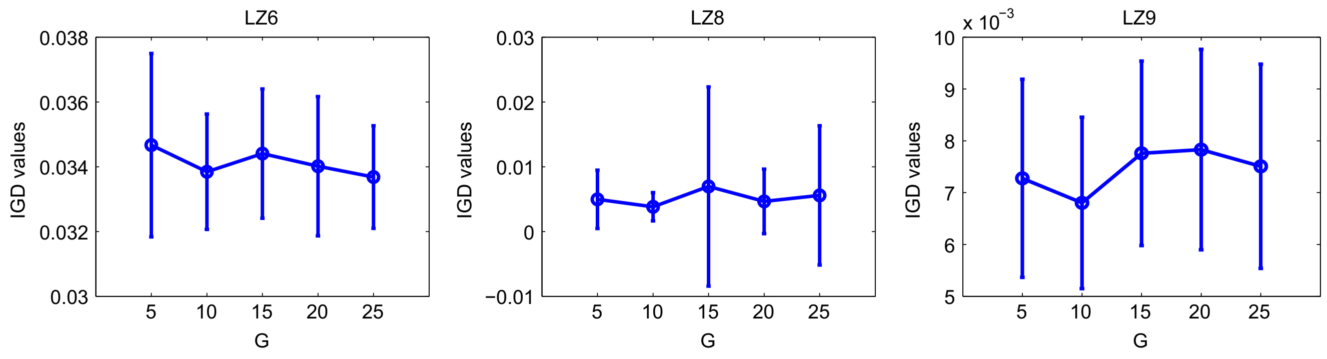

Fig. 10. The error bars of *IGD* metric values versus different *G* on LZ6 and LZ8–LZ9.

5.5.4. *History length*

Fig. 10 shows the error bars of *IGD* metric values versus different history length *G* over 30 independent runs of SMOEA/D with $\tau_0 = 0.8$ and $\sigma_0 = \sqrt{2C^2}/2$. From the plots, we can observe that SMOEA/D does well in solving the test instances when $G = 10$, which denotes that SMOEA/D is also insensitive to the history length *G*.

6. Conclusion and future work

This paper proposes a self-organizing multiobjective evolutionary algorithm based on decomposition with neighborhood ensemble (SMOEA/D). The distinct characteristic of SMOEA/D is that a self-organizing reproduction mechanism (SRM) is designed to produce new solutions. In the SRM, a self-organizing map (SOM) approach is firstly used to discover the distribution structure of the population. Thereafter, based on the structure information, a mating pool consisted of neighboring solutions is established for each solution. Finally, a new solution is generated through interacting within a current solution and its mating pool. In addition to the SRM, SMOEA/D also utilizes an ensemble of neuron neighborhoods with different sizes to establish the mating pools. At each generation, the probability of selecting different neuron neighborhoods is updated according to their performance on generating solutions over the last certain generations.

SMOEA/D is employed to nine test instances with complicated PSs and compared with two state of the art MOEAs: RM-MEDA and MOEA/D-DE. The experimental results indicate that SMOEA/D is efficient to solve the instances, and significantly outperforms the comparison algorithms. In addition, further discussions show that SMOEA/D also performs well on solving instances with simple PS shapes and complex PF shapes. SRM and neighborhood ensemble explicitly contributes to the performance of SMOEA/D. Moreover, SMOEA/D is not quite sensitive to some of the parameters.

Along the research line of this paper, two possible research issues in the future are listed as follows.

- Different machine learning approaches (such as dimensionality reduction and manifold learning) could be used to mine the PS structures of MOPs and to construct various reproduction operators for MOEAs based on the characteristic of MOPs.
- The idea of this paper could be generalized to the other kinds of optimization algorithms, such as particle swarm optimization and estimation of distribution algorithm.

Acknowledgments

The authors would like to thank Aimin Zhou for his helpful comments and suggestions on the original manuscripts. This work was supported by the National Basic Research Program of China (Grant no. 2012CB821205), the Foundation for Creative Research Groups of the National Natural Science Foundation of China (Grant no. 61021002), China National Instrumentation Program (Grant no. 2012YQ180132-01), the National Natural Science Foundation of China (Grant nos. 51275274, and 61174037), and the Innovation Funds of China Academy of Space Technology (Grant no. CAST20120602).

References

[1] C. Hillermeier, Nonlinear Multiobjective Optimization—A Generalized Homotopy Approach, Birkhauser Verlag AG, Basel, Switzerland, 2001.
[2] J.D. Schaffer, Multiple objective optimization with vector evaluated genetic algorithms, in: Proceedings of the 1st International Conference on Genetic Algorithms, Lawrence Erlbaum Associates Inc., New Jersey, USA, 1985, pp. 93–100.
[3] 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.
[4] M. Pilát, R. Neruda, Aggregate meta-models for evolutionary multiobjective and many-objective optimization, Neurocomputing 116 (2013) 392–402.
[5] D. Gong, X. Ji, J. Sun, X. Sun, Interactive evolutionary algorithms with decision-makers preferences for solving interval multi-objective optimization problems, Neurocomputing 137 (2014) 241–251.

[6] L.M. Almeida, T.B. Ludermir, A multi-objective memetic and hybrid methodology for optimizing the parameters and performance of artificial neural networks, Neurocomputing 73 (79) (2010) 1438–1450.
[7] S. Sengupta, S. Das, M. Nasir, A.V. Vasilakos, W. Pedrycz, An evolutionary multiobjective sleep-scheduling scheme for differentiated coverage in wireless sensor networks, IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev. 42 (6) (2012) 1093–1102.
[8] A. Zhou, Q. Zhang, G. Zhang, Approximation model guided selection for evolutionary multiobjective optimization, in: Proceedings of the 7th International Conference on Evolutionary Multi-Criterion Optimization (EMO 2013), Springer, Berlin, Heidelberg, 2013, pp. 398–412.
[9] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[10] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: improving the strength Pareto evolutionary algorithm, in: Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001), International Centre for Numerical Methods in Engineering (CIMNE), Athens, 2002, pp. 95–100.
[11] J. Horn, N. Nafpliotis, D.E. Goldberg, A niched Pareto genetic algorithm for multiobjective optimization, in: Proceedings of the 1st IEEE Conference on Evolutionary Computation, IEEE, 1994, pp. 82–87.
[12] C.M. Fonseca, P.J. Fleming, et al., Genetic algorithms for multiobjective optimization: formulation, discussion and generalization, in: Proceedings of the 5th Conference on Genetic Algorithms, Morgan Kaufmann Publisher, San Francisco, CA, USA, 1993, pp. 416–423.
[13] J.D. Knowles, D.W. Corne, Approximating the nondominated front using the Pareto archived evolution strategy, Evol. Comput. 8 (2) (2000) 149–172.
[14] D.W. Corne, J.D. Knowles, M.J. Oates, The Pareto envelope-based selection algorithm for multiobjective optimization, in: Proceedings of the 6th Workshop on Parallel Problem Solving from Nature (PPSN VI), Springer, Berlin, Heidelberg, 2000, pp. 839–848.
[15] D. Corne, N. Jerram, J. Knowles, M. Oates, PESA-II: region-based selection in evolutionary multiobjective optimization, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001), Morgan Kaufmann Publishers, Inc., San Francisco, CA, USA, 2001, pp. 283–290.
[16] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, IEEE Trans. Evol. Comput. 3 (4) (1999) 257–271.
[17] E. Zitzler, S. Künzli, Indicator-based selection in multiobjective search, in: Proceedings of the 8th International Conference on Parallel Problem Solving from Nature (PPSN VIII), Springer, Berlin, Heidelberg, 2004, pp. 832–842.
[18] N. Beume, B. Naujoks, M. Emmerich, SMS-EMOA: multiobjective selection based on dominated hypervolume, Eur. J. Oper. Res. 181 (3) (2007) 1653–1669.
[19] J. Bader, E. Zitzler, HypE: an algorithm for fast hypervolume-based many-objective optimization, Evol. Comput. 19 (1) (2011) 45–76.
[20] Q. Zhang, H. Li, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731.
[21] H.-L. Liu, F. Gu, Q. Zhang, Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems, IEEE Trans. Evol. Comput. 18 (3) (2014) 450–455.
[22] H. Ishibuchi, T. Murata, A multi-objective genetic local search algorithm and its application to flowshop scheduling, IEEE Trans. Syst. Man Cybern. Part C: Appl. Rev. 28 (3) (1998) 392–403.
[23] T. Murata, H. Ishibuchi, M. Gen, Specification of genetic search directions in cellular multi-objective genetic algorithms, in: Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001), Springer, Berlin, Heidelberg, 2001, pp. 82–95.
[24] X. Ma, F. Liu, Y. Qi, M. Gong, M. Yin, L. Li, L. Jiao, J. Wu, MOEA/D with opposition-based learning for multiobjective optimization problem, Neurocomputing 146 (1) (2014) 48–64.
[25] S.M. Venske, R.A. Gonalves, M.R. Delgado, ADEMO/D: multiobjective optimization by an adaptive differential evolution algorithm, Neurocomputing 127 (0) (2014) 65–77.
[26] Q. Zhang, A. Zhou, Y. Jin, RM-MEDA: a regularity model based multiobjective estimation of distribution algorithm, IEEE Trans. Evol. Comput. 12 (1) (2008) 41–63.
[27] A. Iorio, X. Li, Rotated problems and rotationally invariant crossover in evolutionary multi-objective optimization, Int. J. Comput. Intell. Appl. 7 (2) (2008) 149–186.
[28] A. Zhou, Q. Zhang, G. Zhang, Multiobjective evolutionary algorithm based on mixture Gaussian models, J. Softw. 25 (5) (2014) 913–928.
[29] T. Kohonen, The self-organizing map, Neurocomputing 78 (9) (1998) 1–6.
[30] G.J. Bowden, G.C. Dandy, H.R. Maier, Input determination for neural network models in water resources applications. Part 1: background and methodology, J. Hydrol. 301 (1) (2005) 75–92.
[31] K. Li, S. Kwong, A general framework for evolutionary multiobjective optimization via manifold learning, Neurocomputing 146 (2014) 65–74.
[32] Y. Jin, B. Sendhoff, Connectedness, regularity and the success of local search in evolutionary multi-objective optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2003), IEEE, 2003, pp. 1910–1917.
[33] M. Ehrgott, Multicriteria Optimization, Springer, Berlin, Heidelberg, 2005.
[34] O. Schtze, S. Mostaghim, M. Dellnitz, J. Teich, Covering Pareto sets by multilevel evolutionary subdivision techniques, in: Proceedings of the 2nd International Conference on Evolutionary Multi-Criterion Optimization (EMO 2003), Springer, Heidelberg, Berlin, 2003, pp. 118–132.
[35] A. Zhou, Q. Zhang, Y. Jin, Approximating the set of Pareto-optimal solutions in both the decision and objective spaces by an estimation of distribution algorithm, IEEE Trans. Evol. Comput. 13 (5) (2009) 1167–1189.
[36] D. Yang, L. Jiao, M. Gong, H. Feng, Hybrid multiobjective estimation of distribution algorithm by local linear embedding and an immune inspired algorithm, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2009), IEEE, 2009, pp. 463–470.
[37] L. Mo, G. Dai, J. Zhu, The RM-MEDA based on elitist strategy, in: Proceedings of the 5th International Conference on Advances in Computation and Intelligence (ISICA 2010), Springer, Berlin, Heidelberg, 2010, pp. 229–239.
[38] Y. Wang, J. Xiang, Z. Cai, A regularity model-based multiobjective estimation of distribution algorithm with reducing redundant cluster operator, Appl. Soft Comput. 12 (11) (2012) 3526–3538.
[39] Y. Zhang, G. Dai, L. Peng, M. Wang, HMOEDA_LLE: a hybrid multi-objective estimation of distribution algorithm combining locally linear embedding, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2014), IEEE, 2014, pp. 707–714.
[40] Y. Li, X. Xu, P. Li, L. Jiao, Improved RM-MEDA with local learning, Soft Comput. 18 (7) (2013) 1–15.
[41] Y. Qi, F. Liu, M. Liu, M. Gong, L. Jiao, Multi-objective immune algorithm with Baldwinian learning, Appl. Soft Comput. 12 (8) (2012) 2654–2674.
[42] H. Zhang, S. Song, A. Zhou, X.-Z. Gao, A clustering based multiobjective evolutionary algorithm, in: Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2014), IEEE, 2014, pp. 723–730.
[43] X. Ma, F. Liu, Y. Qi, L. Li, L. Jiao, M. Liu, J. Wu, MOEA/D with Baldwinian learning inspired by the regularity property of continuous multiobjective problem, Neurocomputing 145 (2014) 336–352.
[44] K. Norouzi, G. Rakhshandehroo, A self organizing map based hybrid multi-objective optimization of water distribution networks, IJST, Trans. Civil Environ. Eng. 35 (C1) (2011) 105–119.
[45] Q. Cai, H. He, H. Man, Imbalanced evolving self-organizing learning, Neurocomputing 133 (8) (2014) 258–270.
[46] D. Büche, M. Milano, P. Koumoutsakos, Self-organizing maps for multi-objective optimization, in: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2002), Springer, Berlin, Heidelberg, 2002, pp. 152–155.
[47] D. Büche, G. Guidati, P. Stoll, P. Koumoutsakos, Self-organizing maps for Pareto optimization of airfoils, Springer, Berlin, Heidelberg (2002), p. 122–131.
[48] M. Hakimi-Asiabar, S.H. Ghodsypour, R. Kerachian, Multi-objective genetic local search algorithm using kohonens neural map, Comput. Ind. Eng. 56 (4) (2009) 1566–1576.
[49] M. Hakimi-Asiabar, S.H. Ghodsypour, R. Kerachian, Deriving operating policies for multi-objective reservoir systems: application of self-learning genetic algorithm, Appl. Soft Comput. 10 (4) (2010) 1151–1163.
[50] W. Zhan, H. Liu, G. Dai, Low earth orbit regional satellite constellation design via self organization feature maps, Int. J. Adv. Comput. Technol. 4 (13) (2012).
[51] 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.
[52] K. Price, R.M. Storn, J.A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Springer, Berlin, Heidelberg, 2006.
[53] I. Giagkiozis, R. Purshouse, P. Fleming, Generalized decomposition and cross entropy methods for many-objective optimization, Inf. Sci. 282 (2014) 363–387.
[54] I. Giagkiozis, R. Purshouse, P. Fleming, Generalized decomposition, in: Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, Springer, Berilin, Heidelberg, 2013, pp. 428–442.
[55] S.-Z. Zhao, P. Suganthan, Q. Zhang, Decomposition-based multiobjective evolutionary algorithm with an ensemble of neighborhood sizes, IEEE Trans. Evol. Comput. 16 (3) (2012) 442–446.
[56] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: empirical results, Evol. Comput. 8 (2) (2000) 173–195.
[57] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable test problems for evolutionary multiobjective optimization, in: Proceedings of the 2002 IEEE Congress on Evolutionary Computation (CEC 2002), IEEE, 2002, pp. 825–830.
[58] S. Huband, L. Barone, L. While, P. Hingston, A scalable multi-objective test problem toolkit, in: Proceedings of the 3rd International Conference on Evolutionary Multi-Criterion Optimization (EMO 2005), Springer, Berlin, Heidelberg, 2005, pp. 280–295.

Hu Zhang received the M.Sc. degree in mechanical design and theory from China Three Gorges University, Hubei Province, China, in 2012. He is currently working toward the Ph.D. degree in the Center for Control Theory and Guidance Technology, Harbin Institute of Technology, Harbin, China. His current research interests include evolutionary computation, statistical machine learning, and their applications.

Xiujie Zhang received the M.Sc. degree in intelligence engineering from Chiba University, Chiba, Japan, in 2006. She received her Ph.D. degree in Control Theory and Application from Harbin Institute of Technology, Harbin, China, in 2013. Her current research interests include evolutionary computation and their applications.

Xiao-Zhi Gao earned his D.Sc. (Tech.) degree from the Helsinki University of Technology (now Aalto University), Finland in 1999. Since January 2004, he has been appointed as a Docent (Soft Computing Methods and Applications) at the same university. He has published over 280 papers in refereed journals and international conferences. His current research interests are nature-inspired computing methods with applications in optimization, data mining, and industrial electronics.

Shenmin Song received his Ph.D. degree in Control Theory and Application from Harbin Institute of Technology, Harbin, China, in 1996. He carried out post-doctoral research at Tokyo University from 2000 to 2002. He is currently a professor at the School of Astronautics, Harbin Institute of Technology. His main research interests include spacecraft guidance and control, intelligent optimization and control, nonlinear theory and application. He has published research papers at national and international journals, conference proceedings as well as two books.