Information Sciences 182 (2012) 220–242

Contents lists available at SciVerse ScienceDirect

Information Sciences

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

Achieving balance between proximity and diversity in multi-objective evolutionary algorithm

Ke Li [a], Sam Kwong [a,*], Jingjing Cao [a], Miqing Li [b], Jinhua Zheng [b], Ruimin Shen [b]

[a] Department of Computer Science, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong
[b] College of Information and Engineering, Xiangtan University, China

ARTICLE INFO

Article history:
Received 3 November 2010
Received in revised form 5 July 2011
Accepted 21 August 2011
Available online 10 September 2011

Keywords:
Multi-objective evolutionary optimization
Minimum spanning tree
Fitness assignment
Hypervolume indicator
Population maintenance

ABSTRACT

Currently, an alternative framework using the hypervolume indicator to guide the search for elite solutions of a multi-objective problem is studied in the evolutionary multi-objective optimization community very actively, comparing to the traditional Pareto dominance based approach. In this paper, we present a dynamic neighborhood multi-objective evolutionary algorithm based on hypervolume indicator (DNMOEA/HI), which benefits from both Pareto dominance and hypervolume indicator based frameworks. DNMOEA/HI is featured by the employment of hypervolume indicator as a truncation operator to prune the exceeded population, while a well-designed density estimator (i.e., tree neighborhood density) is combined with the Pareto strength value to perform fitness assignment. Moreover, a novel algorithm is proposed to directly evaluate the hypervolume contribution of a single individual. The performance of DNMOEA/HI is verified on a comprehensive benchmark suite, in comparison with six other multi-objective evolutionary algorithms. Experimental results demonstrate the efficiency of our proposed algorithm. Solutions obtained by DNMOEA/HI well approach the Pareto optimal front and are evenly distributed over the front, simultaneously.

1. Introduction

Many real-world optimization problems are required to optimize several competing criteria or objectives simultaneously. These problems are so-called multi-objective optimization problems (MOPs). Contrasting to the single-objective optimization problems (SOPs), which aim at finding a single global optimal point, solutions to MOPs are a set of trade-offs (i.e., Pareto optimal front). However, instead of obtaining infinite number of Pareto optimal solutions, which is a time-consuming task, decision makers prefer to search for a set of representative solutions which are as close as possible to the true Pareto optimal front, in the meanwhile, being uniformly distributed along the whole front. In other words, the optimality of solutions are ensured by the closeness to the Pareto optimal front, while an ideal and thorough exploration of the search space is guaranteed by the uniform distribution of solutions.

Evolutionary algorithms (EAs) seem to be well suited for dealing with MOPs due to their population-based property which achieves an approximation of the Pareto set in a single run. Since the pioneering work of Schaffer [1], a large variety of multi-objective evolutionary algorithms (MOEAs) have been proposed and applied to various problem domains [2,3]. Later, Goldberg [4] suggested a non-dominated sorting procedure, which opened a new avenue and generated an overwhelming interest in MOEAs. After then, this suggestion was initially implemented in three contemporaneous state-of-the-art MOEAs [5–7], separately. Srinivas and Deb [7] proposed NSGA [7] which used non-dominated sorting

* Corresponding author.
E-mail address: cssamk@cityu.edu.hk (S. Kwong).

0020-0255/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2011.08.027

to filter out non-dominated solutions and a niching strategy to maintain the population diversity. Hereafter, Deb et al. improved the non-dominated sorting strategy and replaced the original diversity maintenance method with a crowding distance mechanism, proposed the state-of-the-art NSGA-II [8]. In MOGA proposed by Fonseca and Fleming [5], all individuals in the whole population are ranked based on their dominance relationship and the selection procedure was just guided by these rank values. Horn, Nafpliotis and Goldberg proposed NPGA [6], which used Pareto domination tournaments instead of the non-dominated sorting and ranking selection method in solving multi-objective optimization problems. Combining with the Pareto strength based fitness assignment strategy and a clustering procedure to reduce the non-dominated set, Zitzler et al. proposed SPEA [9]. An improved version of SPEA (i.e., SPEA2 [10]) with an enhanced fitness assignment strategy and truncation mechanism was designed by Zitzler et al. three years later. Knowles and Corne proposed a meta-heuristic PAES [11], which was based on a simple (1+1)-evolution strategy. During the same period, Corne and Knowles proposed PESA [12], in which selection and diversity maintenance were controlled by a hyper-grid based scheme. Instead of assigning selective fitnesses to individuals, Corne and Knowles designed PESA-II [13] two years later, whose selection units were replaced by hyper-boxes in objective space that were occupied by at least one individual in the current approximation to the Pareto optimal front. Different from the previous Pareto-based approaches, Hughes [14] proposed to use multiple single objective optimisers to perform a parallel search of multiple conventional target vector based optimizations. Laumanns et al. proposed ε-dominance concept [15] which is a modification of the original Pareto dominance. The underlying principle of ε-dominance is that two solutions are not allowed to be non-dominated to each other, if the difference between them is less than properly chosen value ε_i in the ith objective. Qu et al. proposed a MOEA based on Summation of normalized objective values and diversified selection (SNOV-DS) [16]. Shi et al. proposed a novel data structure named dominance tree [17] to store and handle the population in MOEAs. Chan et al. proposed to adapt the jumping gene paradigm to the framework of MOEA in [18] and Nawaz Ripon et al. adapted this same scheme for solving continuous MOPs [19]. In order to alleviate the lost of solutions lying at the extreme parts of the Pareto front caused by the use of ϵ-dominance and better control the appropriate value of ϵ, Hernández-Díaz et al. proposed three different schemes whose use depends on the users preferences in [20]. Recently, Bader and Zitzler [21] proposed an indicator-based MOEA which used Monte Carlo sampling to approximately calculate the hypervolume indicator, and used this indicator to guide the evolution. More detailed development of current research literature could be found in a recently good survey [22]. It is worth noting that an alternative evolution mechanism to the aforementioned genetic operators, being known as particle swarm optimization (PSO), which is inspired by the motion of bird flocks, is attracting more and more researchers to graft it for solving MOPs. Wang et al. proposed a new optimal criterion based on preference order scheme to identify the best compromise in multi-objective particle swarm optimization (MOPSO) [23]. Zhao et al. proposed a novel two local bests based MOPSO (2LB-MOPSO) [24] which focused the search around small regions in the parameter space and avoided the case that pbest is far away from the gbest.

Recently, a performance metric called hypervolume indicator [9] achieves fabulous success in the EMO community. The following three features motivate the prosperity and development of it.

1. Hypervolume indicator is a comprehensive performance metric. The convergence towards the Pareto optimal front as well as the representative distribution of solutions along this front could be evaluated simultaneously by this single indicator.
2. Hypervolume indicator is the only performance metric known to be strictly monotonic with respect to Pareto dominance [25]. In other words, if a solutions set S is better than the other set S', the corresponding hypervolume indicator of S is larger than S'.
3. Hypervolume indicator guarantees that the Pareto optimal set achieves the maximum indicator value [26].

Hypervolume indicator not only plays as an off-line indicator to evaluate the quality of the solutions set, but also, more importantly, it could be incorporated into the framework of MOEA to guide the optimization process. Some precursor studies on hypervolume based MOEA have been developed in the literature [27–30,21]. However, there are still two major drawbacks existing in the current research activities.

1. Most of the fitness assignment strategies in current hypervolume based MOEAs are merely based on the hypervolume contribution of a single solution. This might be too restricted so that some other interesting information in the current population might not be fully explored.
2. The computational overhead to exactly evaluate the hypervolume contribution of a solution is extremely high. In [29,30], the difference of the hypervolume indicator values between the original set and the set excludes a solution is used as the hypervolume contribution of this solution. That is to say, we have to calculate the hypervolume indicator twice when evaluating the hypervolume contribution of a single solution. It obviously further burden the already high computational overhead.

Pareto dominance based algorithms have been studied in the EMO community for the last two decades. It benefits from the straightforward way to conform to the intrinsic properties of MOPs. On the other hand, indicator based algorithms with hypervolume metric have strong theoretical background and high search ability. The motivation of this work is to assimilate

the advantages from both prototypes. Generally speaking, the main contributions of this work could be summarized as follows:

1. A novel method, which is based on minimum spanning tree, to estimate the density information of individuals in the current population.
2. A slicing based method to exactly evaluate the hypervolume contribution of a single solution.
3. The truncation procedure is modeled as a single-objective optimization process, which aims at maximizing the total hypervolume measure of the current external archive.

This paper is organized as follows: Section 2 briefly reviews the density estimation method and hypervolume related research. Section 3 elaborates the design details of our proposed algorithm. Section 4 presents experimental studies. Section 5 concludes our work and discusses some possible future directions.

2. Related work

Before detailed discussion of our proposed approach, we would like to give a brief review on the density estimation techniques and the hypervolume related research at first.

2.1. Brief review of research on density estimation

One of the essentials of MOEA is to distribute solutions as diversified as possible along the trade-off front during the evolutionary process. Designing effective diversity preservation methods is always an active research field in multi-objective optimization domain. Although different types of diversity preservation methods have been proposed, the underlying principles are similar. Almost all of them aim at assessing the density information of a solution in either decision or objective space. One of the most important techniques to assess density is sharing, which was originally proposed by Goldberg and Richardson [31]. In this approach, the density information was estimated by a sharing function which considered the distances among neighboring individuals. Although the fitness sharing technique has been widely used in many studies [7,6], it was restricted by the difficulty of determining the appropriate niche sizes. In view of this drawback, Fonseca and Fleming [5] proposed a method to automatically tune the niche size based on the distribution of population at each generation. Another important technique for density estimation is the use of grid [11]. This approach aims at partitioning the feature space into a series of grids, and then the crowded information in a particular grid is used as the density value of the solution reside in it. Crowding is also a popular density estimator in MOEAs. It was proposed by Deb et al. [8] in his NSGA-II to estimate the density of a solution through the perimeter of the rectangular surrounds it. Zitzler et al. [9,10] applied a clustering method to maintain the size of external archive. Farhang-Mehr and Azarm [32] employed an entropy-based metric to assess the diversity of solutions, and at the same time the entropy metric was used as a quality assessment to compare the performance of two MOEAs.

2.2. Brief review of research on hypervolume indicator

Hypervolume indicator was firstly proposed by Zitzler and Thiele [9]. Originally, it was employed as a performance metric to give a quantitative measure to the outcome of a MOEA. Primarily, the underlying principle of algorithms to calculate hypervolume indicator was known as the hypervolume by slicing objectives. Most of these algorithms use a recursive procedure to decrement the number of objectives recursively. Representative algorithms belong to this category are approach provided by Zitzler and Thiele [9], method proposed by Knowles and Corne [33], and heuristics proposed by While et al. [34]. The worst-case runtime complexity of this kind of algorithms is $O(N^{n-1})$, which is exponential in the number of objectives. Here N is the number of individuals in the population and n is the number of objectives. Recently, some advanced algorithms for improving the efficiency of the hypervolume indicator calculation have been proposed due to the wide applications of this indicator. Fonseca et al. [35] proposed a dimension-sweep method which had a worst-case runtime complexity of $O(N^{n-2}logN)$. Beume [36] considered the hypervolume measure as a special case of Klee's measure problem, and proposed a method which owned a worst-case runtime complexity of order $O(NlogN + N^{n/2})$. Yang et al. [37] proposed an algorithm which had a worst-case runtime complexity of $O((n/2)^N)$. Most recently, Bringmann and Friedrich have done a series of significant work on improving the efficiency of the calculation of hypervolume indicator [38–40]. They proved the calculation of hypervolume is $\sharp P$-complete, which meant that no polynomial time algorithm exists since this conclusion implied $NP = P$ [38].

The application of hypervolume indicator into the optimization process was firstly introduced by Knowles et al. [27,33]. Huband et al. [28] proposed an algorithm which replaced the original density estimator in the environmental selection of SPEA2 with the hypervolume measure. Zitzler et al., proposed a general framework of indicator based evolutionary algorithm (IBEA) [29] which aggregated the hypervolume indicator into the fitness function. Emmerich et al. [30] combined non-dominated sorting and hypervolume indicator together to design a steady state $(\mu + 1)$-MOEA (SMS-MOEA). Bradstreet [41] proposed a specific algorithm to quickly determine a solution's hypervolume contribution. All these algorithms introduced so far

aim at calculating the hypervolume indicator exactly. The main drawback of this kind of algorithms is the extremely high computational overhead, especially for high dimensional problems, although some advanced hypervolume indicator calculation techniques have been proposed. As discussed in [21], the exact hypervolume contributions of solutions are not that crucial in environmental selection. Instead, the ranks of these measures are more interesting. Consequently, another emerging trend of the hypervolume related research is to find an ideal representation to approximate the hypervolume contribution. Bader and Zitzler [21] proposed a fast algorithm that used Monte Carlo simulation to approximate the hypervolume indicator. Similar to this work, Voß et al. [42] also employed the Monte Carlo algorithm to approximate the hypervolume contribution, and proposed a steady state MO-CMO-ES. Different from the previous work, Ishibuchi et al. [43] used the achievement scalarizing functions to approximate the hypervolume contribution.

3. Dynamic neighborhood MOEA based on hypervolume indicator

Multi-objective optimization problems contain a number of conflicting objectives to be optimized. Using EAs to solve MOPs must address the following two issues.

1. How to carry out fitness assignment? It is important to evaluate the qualities of solutions impartially, so that the solutions could be guaranteed to search towards the Pareto optimal.
2. How to preserve the diversity of the obtained solutions set with a guaranteed convergence? Such kind of methods could avoid the premature convergence as well as obtain uniformly distributed solutions.

3.1. Fitness assignment

Traditionally, the fitness assignment scheme is composed of two important ingredients. One is Pareto dominance relationship, which divides individuals into dominated and non-dominated categories. And the latter ones are always preferable to the formers when referring to fitness values. The other ingredient is usually represented by the density information of an individual. As briefly reviewed in Section 2.1, many density estimation techniques have been proposed in the literature up till now. However, none of these density estimators are not far away from perfect or are problem restricted. Here, we would like to analyze a pathological example (as given in Fig. 1) of the fitness assignment approach that used in the state-of-the-art SPEA2 at first, and then elaborate our proposed strategy. In SPEA2, the density estimator is formulated as $D(i) = 1/(2 + \sigma_i^k)$ where σ_i^k means the Euclidean distance between the individual i and its kth nearest neighbor. Here set $k = 3$ and the number of individuals is 6. Intuitively, it is not difficult to find that a is the 3rd nearest neighbor to both of b and c. And the Euclidean distance between c and a is larger than that of b and a, that is to say $\sigma_c^k > \sigma_b^k$. Accordingly, we could conclude $D(c) < D(b)$, which means that c locates in a less crowded area than b. Consequently, based on the fitness calculation formula defined in [10], the fitness value of c ought to be better than that of b, in case that c and b are non-dominated to each other. Therefore, c would have a larger chance to be selected in the environmental selection procedure, compared to b. Neverthelss, as illustrated in Fig. 1 intuitively, the neighborhood of c is more crowded than that of b. Thus the fitness value of c ought to be inferior to that of b.

In this section, we propose a robust density estimator called Tree Neighborhood Density (TND), which could enforce a higher level of spread between the non-dominated solutions. For the sake of a clearer discussion, some definitions and terminologies are firstly given as follows.

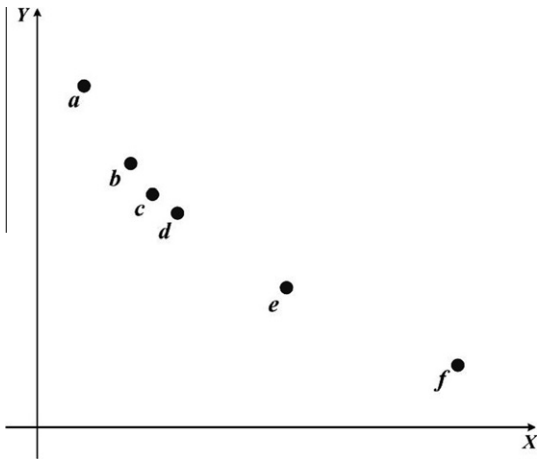

Fig. 1. A pathological example of the density estimator in SPEA2.

Definition 1 (*Tree crowding density*). Let T be a minimum spanning tree connecting all the individuals of population P. For any individual i in P, let d_i be the degree of i in T, i.e., the number of edges of T connected to i; and let these edges be $\{l_{i,1},\ l_{i,2},\ldots,l_{i,d_i}\}$. The tree crowding density of i is estimated as:

$$T_{crowd}(i) = \sum_{j=1}^{d_i} l_{i,j}/d_i. \tag{1}$$

Definition 2 (*Tree neighborhood*). Let $r_i = \max\{l_{i,1}, l_{i,2},\ldots,l_{i,d_i}\}$. A circle centered in i, with radius r_i, is defined as the tree neighborhood of i.

Definition 3 (*Membership of individual on the tree neighborhood*). Let the Euclidean distance between individuals i and j be denoted as $dist_{i,j}$. The individual j is considered as a member of the tree neighborhood of i if and only if $dist_{i,j} \leqslant r_i$ (denoted as $i \triangleright_T j$).

Based on these definitions, the calculation procedure for the Tree Neighborhood Density (TND) is implemented as follows:

Step 1. The Euclidean distance between each individual of the population P and the other $N-1$ (N is the population size) ones is calculated.
Step 2. A minimal spanning tree T connecting all individuals is generated.
Step 3. The tree crowding density for each individual i in T is evaluated, and the corresponding tree neighborhood is generated.
Step 4. Count the number of individuals that are the member of the tree neighborhood of each individual i, denoted as $count(i) = \|U\|$, where $U = \{j | j \in P,\ i \triangleright_T j\}$ and $\|\cdot\|$ means the cardinality of the set.
Step 5. Then, the tree neighborhood density of individual i is calculated as:

$$TND(i) = \frac{\sum_{j\in U}(1/Tcrowd_j)}{count(i)}. \tag{2}$$

Step 6. Finally, the TND values of all individuals are normalized:

$$n\mathrm{TND}(i) = \frac{\mathrm{TND}(i) - \mathrm{TND}_{min}}{\mathrm{TND}_{max} - \mathrm{TND}_{min}}. \tag{3}$$

where nTND(i) is the normalized TND of individual i, and TND_{max} and TND_{min} indicate, respectively, the maximum and minimum TND in the current population.

It is worth noting that the smaller TND the better. The underlying motivation for the proposal of TND could be explained as follows. The whole set of solutions in the population could be regarded as a connected graph, with the Euclidean minimum spanning tree of this graph being an optimized structure that reflects the distribution of the solutions of the current population in the search space. Then, for a given individual, the corresponding neighborhood could be defined by the other individuals connected to it, and finally, the crowdedness of this neighborhood could be used to represent its density.

In order to illustrate the calculation process of TND, let us consider the previous pathological example discussed in Fig. 1. Assuming $S = \{(3,10),(5,7),(6,6),(8,5),(11,4),(15,3)\}$, a to f represent the corresponding points in S. We plot the distribution of these points in Fig. 2. Edges connecting each point constitute the minimal spanning tree of S. Dashed line indicated circles A to F correspond to the tree neighborhood of each point in S, according to definition 2. Then, we could easily obtain the tree crowding densities of a to f as: $T_{crowd}(a) = 3.6$, $T_{crowd}(b) = 2.5$, $T_{crowd}(c) = 1.8$, $T_{crowd}(d) = 2.7$, $T_{crowd}(e) = 3.65$, $T_{crowd}(f) = 4.1$, based on definition 1. According to definition 3 and step 4 in the calculation procedure of TND, here we only consider points b and c as an example, we could obtain that $count(b) = 4$, $count(c) = 3$. Next, the tree neighborhood density values of b and c could be calculated as: $TND(b) = 0.401$, and $TND(c) = 0.443$. Thus we get $TND(c) > TND(b)$, which satisfied our original assumption that c locates in a more crowded area than that of b.

Based on the aforementioned discussion, the fitness value of each individual i is calculated as the combination of its normalized TND (i.e., nTND(i)) and strength value:

$$F(i) = R(i) + n\mathrm{TND}(i), \tag{4}$$

where $R(i) = \sum S(j),\ S(j) = \|\{i | i \in E + Q \wedge j \succ i\}\|$ [10]. It is worth noting that only the nTND measure is normalized between 0 and 1. Hence, evolution proceeds by firstly minimizing $R(i)$, i.e., approaching the Pareto optimal front; and then, as soon as some non-dominated solutions (i.e., with $R(i) = 0$) are found, nTND becomes significant in the fitness evaluation, and a higher spread between the non-dominated solutions is promoted. The whole procedure of fitness assignment is illustrated as follows:

Step 1. Merge the evolutionary population E and external archive Q together into a hybrid population P;
Step 2. Calculate the strength value $R(i)$ of each individual i in P;

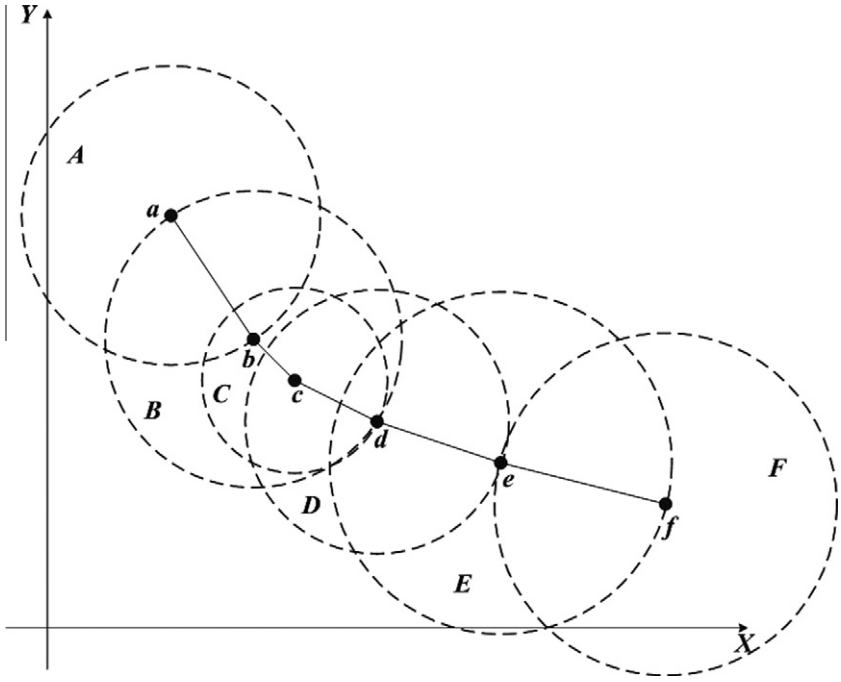

Fig. 2. An example for calculating tree neighborhood density.

Step 3. The corresponding normalized tree neighborhood density nNTD(i) of each individual i in P could be obtained according to the aforementioned calculation procedure of TND;
Step 4. The corresponding fitness value of each individual i in P could be calculated according to formula (4).

Different from the fitness assignment strategy proposed by Li et al. [44,45] that used tree crowding density directly to represent the fitness value, here we use a more sophisticated scheme to estimate the density in each individual's tree neighborhood. Most of the existing methods for setting neighborhood aim at partitioning the search space into several fixed regions. Then the density of each individual is estimated according to circumstances of its fixed own region. The dynamic neighborhood density estimation method proposed here is based on the relationship between an individual and its adjacent neighbors. The sizes of the neighborhoods are changed with the variation of the densities of individuals. Thus, the up-to-date density information in the current population could be tracked timely.

3.2. Calculation method for hypervolume indicator of the individual dominance area

After the discussion of fitness assignment, the other important design issue of MOEA, i.e., diversity maintenance, is going to be elaborated in this subsection. As introduced in Section 1, the hypervolume indicator is employed to guide the diversity preservation in our approach. Before further discussion, some preliminary definitions related to hypervolume indicator are provided at first.

Definition 4 (*Individual dominance area*). Given a solutions set S, individual $i \in S$, the area which is exclusively dominated by i is called the individual dominance area of i.

Definition 5 (*Hypervolume indicator of set S*). Let the reference point is denoted as $Ref = (r_1, r_2, \ldots, r_k)$, the hypervolume indicator of S (denoted as $Hv(S)$) is defined as the volume of the hypercube restricted by all points in S and Ref.

$$Hv(S) = Leb\left(\bigcup_{\vec{x}\in S}[f_1(\vec{x}_1), r_1] \times [f_2(\vec{x}_2), r_2] \times \cdots \times [f_1(\vec{x}_k), r_k]\right), \tag{5}$$

where k is the number of dimension, $Leb(S)$ indicates the Lebesgue measure of S, and $[f_1(\vec{x}_1), r_1] \times [f_2(\vec{x}_2), r_2] \times \cdots \times [f_1(\vec{x}_k), r_k]$ represents the hypercube formed by points which are dominated by $\vec{x}$ but Ref.

As shown in Fig. 3, the hypervolume indicator of the solutions set $S = \{P_1, P_2, P_3\}$ is exactly the area of the polygon $ACBEFGHI$, where H represents the reference point. In addition, the individual dominance area of P_2 is the rectangular area $BCDE$. Intuitively, the hypervolume contribution of an individual is just the hypervolume indicator value of its individual dominance area.

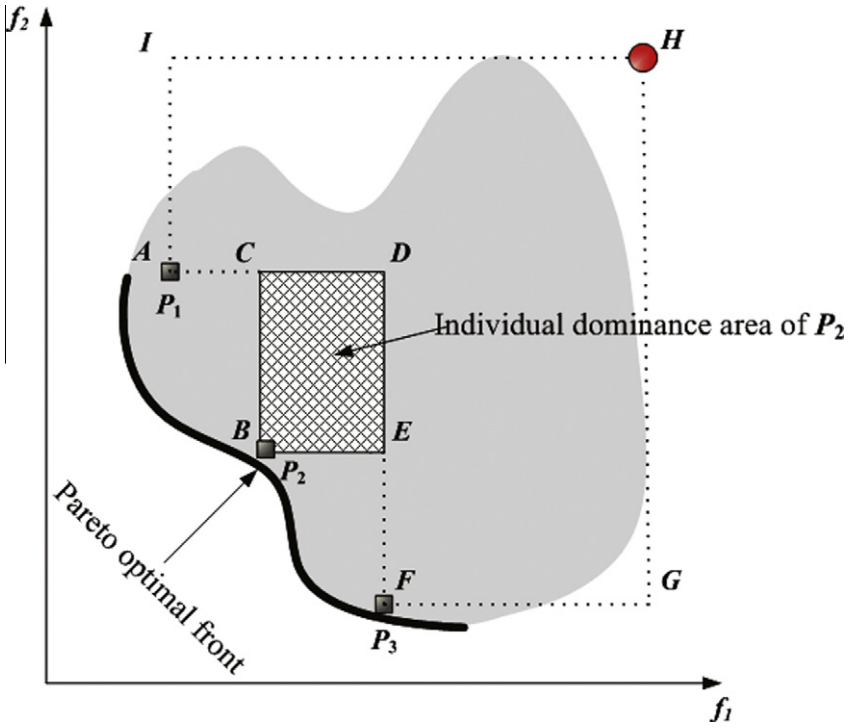

Fig. 3. Hypervolume indicator in two-dimensional case.

3.3. Two-dimensional case

It is relatively easy to calculate the hypervolume indicator of individual dominance area in the two-dimensional case. At first, individuals in the population are sorted by ascending order based on their objective values at the first dimension. Then the corresponding hypervolume indicator of the individual dominance area for an individual p_i (denoted as $ExcHv(p_i)$) could be calculated as:

$$ExcHv(p_i) = |obj_1(p_{i+1}) - obj_1(p_i)| * |obj_2(p_{i-1}) - obj_2(p_i)|, \tag{6}$$

where $2 \leqslant i \leqslant N-1$, N is the population size, and $obj_i(j)$ indicates the objective value of the jth individual at the ith dimension. As for the two extreme solutions (also noted as boundary solutions, i.e., p_1 and p_N), we set $ExcHv(p_1) = ExcHv(p_N) = \infty$. This way the boundary solutions could be reserved constantly, as well as the spread of the solutions set is ensured. The pseudo code of this procedure is shown in Algorithm 1.

Algorithm 1: Calculating the hypervolume indicator of the individual dominance area in two-dimensional case

```
1 begin
2   if index == N ∨ index == 1 then
3     ExcHv = INF;
4   else
5     ExcHv = |obj_1(p_{index-1}) − obj_1(p_index)| * |obj_2(p_{index+1}) − obj_2(p_index)|
6   end
7   return ExcHv
8 end
```

3.4. Three-dimensional case

Instead of the simple rectangular area as in the two-dimensional case, the individual dominance area in three-dimensional case is always a complex cube which requires an even more sophisticated technique to compute its hypervolume indicator. In order to illustrate this difficulty, consider a complex cube, which is hard to evaluate its hypervolume directly, as shown in Fig. 4. Inspired from the slicing objectives algorithm proposed in [34], here we also consider slicing this out-of-shape cube into a series of less complicated slices, in order to reduce the computational difficulty. However, different

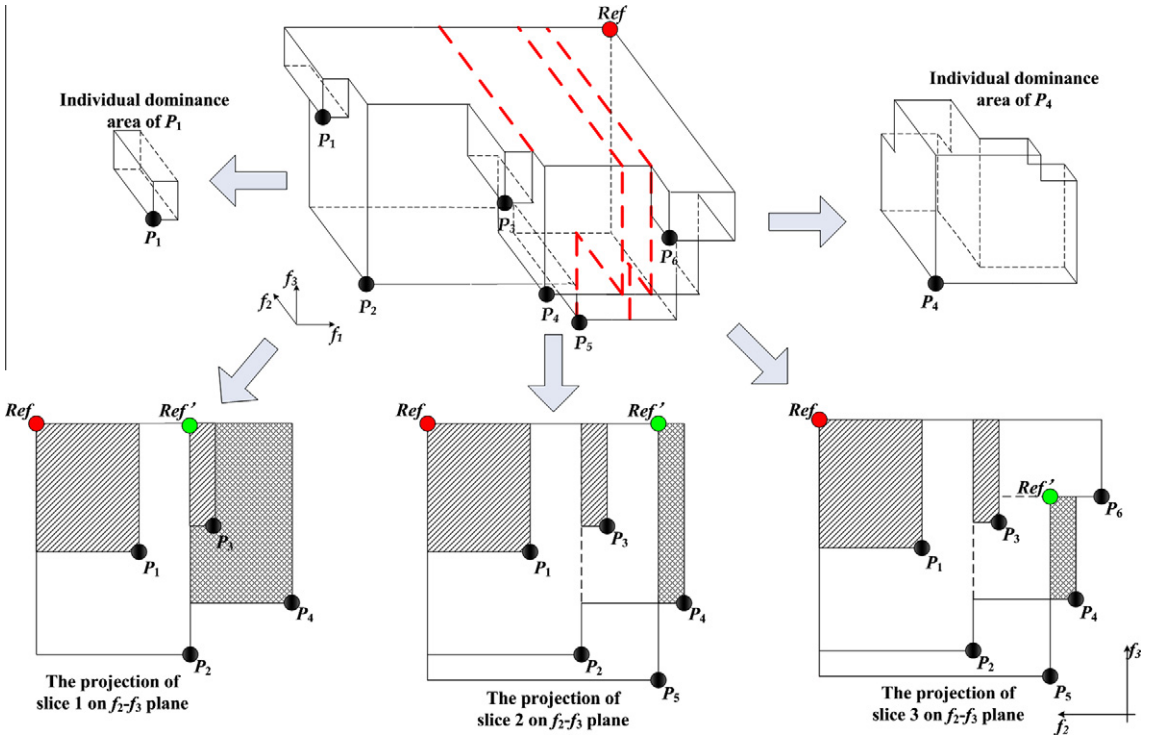

Fig. 4. An example for calculating the hypervolume indicator of individual dominance area in three-dimensional case.

from the method presented in [34] which aimed at slicing the total cube, our proposed method only interests in the part which is organized by the current significant points. Afterwards, each slice is projected onto a two-dimensional plane which is more convenient to determine its corresponding individual dominance area. Finally, the total hypervolume indicator is cumulated by the hypervolume indicator of these slices. In the following paragraph, several relevant definitions and terminologies are given preliminarily, for the sake of further discussion.

Definition 6 (*Significant point (Invalid point)*). Regarding to a point p, points that are non-dominated with p are called significant, otherwise they are called invalid.

Significant points are used to confirm the boundary while the invalid points are used to confirm the shape, when determining the individual dominance area. Take P_2 as an example, as shown in Fig. 5, P_1 and P_3 are significant while P_4 and P_5 are invalid. Particularly, P_1 and P_3, respectively, are used to determine the upper and lower bounds of the individual dominance

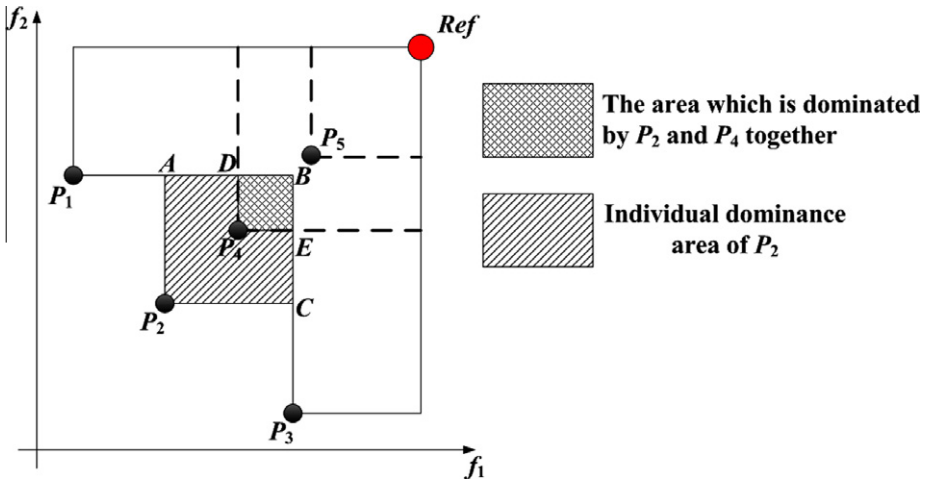

Fig. 5. An example to determine the individual dominance area.

area of P_2. The overlapped area P_4DBE formed by the shared dominance area of P_4 and P_2 need to be eliminated at last. Consequently, the individual dominance area of P_2 is the polygon area P_2ADP_4EC.

Definition 7 (*Least upper bound point*). Given a solutions set S on an M dimensional space. $\forall p \in S$, $p : (f_1(p), f_2(p), \ldots, f_M(p))$, $\exists q \in S$, $q : (f_1(q), f_2(q), \ldots, f_M(q))$, q is denoted as the least upper bound point of p in S on the ith dimension, if and only if the following condition fulfilled:

$$f_i(q) = min\{f_i(u) | f_i(u) > f_i(p), u \in S\}. \tag{7}$$

If p is the last point of the solutions set S whose points are sorted by ascending order based on the values at the ith dimension, we set the least upper bound point of p is the reference point *Ref*.

Definition 8 (*Depth of a slice (denoted as $depth_i$)*). Supposing that the ith dimension is chosen to be sliced, let the current investigated point be $p : (f_1(p), f_2(p), \ldots, f_M(p))$, and the least upper bound point of p be $q : (f_1(q), f_2(q), \ldots, f_M(q))$. Then the depth of the corresponding slice is defined as the absolute value of the difference between p and q on the ith dimension, namely: $depth_i = |f_i(q) - f_i(p)|$.

Before the computation process, individuals are sorted by ascending order based on their values at the first dimension. Let p be the current investigating individual, then the others could be classified into the following four categories:

1. *Prior dominated individuals*: Individuals that are prior to p and dominated by it. This kind of individuals are stored in set *PD*.
2. *Prior non-dominated individuals*: Individuals that are prior to p but non-dominated with it. These individuals are stored in set *PL*.
3. *Posterior non-dominated individuals*: Individuals posterior to p and non-dominated with it. We store this kind of individuals in set *NL*.
4. *Fellow individuals*: Individuals that own the same values with p at the first dimension. We store this kind of individuals in set *SC*.

After the above taxonomy procedure, individuals in the population respect to p could be categorized into the following four circumstances to process:

1. p is the first individual of the population after sorting, namely both *PD* and *PL* are null.
2. There are only dominated individuals prior to p, namely *PL* is null, but *PD*.
3. There are only non-dominated individuals prior to p, namely *PD* is null, but *PL*.
4. Both dominated and non-dominated individuals exist prior to p, namely neither *PD* nor *PL* is null.

We could find that circumstance 1 is similar to circumstance 3 while circumstance 2 is analogous to circumstance 4. For the sake of simplicity, we only detail the implementation of circumstance 3 and circumstance 4 whose pseudo-codes are shown, respectively, in Algorithms 3 and 4. In addition, the general framework for calculating the hypervolume indicator of the individual dominance area is given in Algorithm 2.

Algorithm 2: Main framework for calculating the hypervolume indicator of the individual dominance area

```
1  begin
2      Filtering out four kinds of individuals from the population and store them into sets PD, PL, NL and SC
       separately;
3      if PD == ∅ ∧ PL == ∅ then
4          ExcHv = SubProc1(S, N);
5      else if PD ≠ ∅ ∧ PL == ∅ then
6          ExcHv = SubProc2(S, N);
7      else if PD == ∅ ∧ PL ≠ ∅ then
8          ExcHv = SubProc3(S, N);
9      else
10         ExcHv = SubProc4(S, N);
11     end
12     return ExcHv;
13 end
```

Generally speaking, the hypervolume indicator of the individual dominance area for an specific individual could be calculated based on the following five steps after the aforementioned pretreatment.

Step 1. Determine the depth of the current slice.
Step 2. Confirm the shape of the current slice.
Step 3. Relocate the position of reference point and set it as the current temporary reference point.
Step 4. Initialize a set R as an empty set. If PD is null, turns to Step 5. Otherwise we store the invalid points that locate in the area restricted by the temporary reference point and p into the set R, after then turns to Step 5.
Step 5. If R is null, we calculate the hypervolume indicator of the individual dominance area of p based on the shape of the current slice. Otherwise, we subtract the hypervolume indicator of R from that of the individual dominance area of p, then the subtracted indicator value is used as the hypervolume indicator of the individual dominance area of p for the current slice. If all points in NL have been processed, returns the current cumulated hypervolume indicator and exit, otherwise turns to Step 1.

The underlying principle of this method is slicing the cube which is formed by points that are significant to p. There is no significant point posterior to p if set NL is null. That is to say only points that are prior to p and its fellow individuals need to be considered (only one slice exists in this case, like the top left cube, i.e., the individual dominance area of P_1, shown in Fig. 4). Otherwise, points that are posterior to p also need to be processed one by one (multiple slices exist in this case). Firstly, the least upper bound point of p at the first dimension is determined according to Definition 7. Then the depth of the current slice could be obtained based on Definition 8. Subsequently, four different categories of points are filtered out, thus the shape of the current slice (i.e., the individual dominance area of p in the current slice) could be determined correspondingly. After the determination of the least upper bound point of p on the other two dimensions, the current temporary reference point is relocated as $Ref' : (upper_1(p), upper_2(p), upper_3(p))$. Next, points that locate in the area restricted by Ref' and p are regarded as invalid points and stored into a specific set R. The hypervolume indicator of the individual dominance area of p in the current slice could be obtained by subtracting the hypervolume indicator of R from that of the current slice. Afterwards, cumulate this hypervolume into the total hypervolume indicator of the individual dominance area of p. If all points in NL have been processed, this procedure terminates. It is worth noting that the main difference between circumstance 3 and circumstance 4 is whether it is necessary to consider the set R, so as to circumstance 1 and circumstance 2.

Algorithm 3: Calculating the hypervolume indicator of the individual dominance area for circumstance 3, i.e., $SubProc3()$

```
1  begin
2      ExcHv(p) = 0;
3      if NL == ∅ then
4          The least upper bound point of p at the first dimension is u_1;
5          depth = |obj_1(p) − u_1|;
6          Relocate the position of current reference point Ref';
7          CurArea = |(obj_2(p) − obj_2(Ref'))| * |(obj_3(p) − obj_3(Ref'))|;
8          ExcHv(p) = depth * CurArea;
9
10     else
11         Include p into set NL;
12         for i=1 to ||NL|| do
13             The least upper bound point of NL[i] at the first dimension is u_1;
14             SliceDepth = |obj_1(NL[i]) − u_1|;
15             Relocate the position of current reference point Ref';
16             CurArea = |(obj_2(NL[i]) − obj_2(Ref'))| * |(obj_3(NL[i]) − obj_3(Ref'))|;
17             ExcHv(p)+ = SliceDepth * CurArea;
18         end
19     end
20     return ExcHv(p);
21 end
```

To illustrate how this computation process works, consider the previous example given in Fig. 4. Assuming that these six points belong to a set $S = \{(1,8,7), (2,6,3), (4,5,8), (5,2,5), (7,3,2), (10,1,9)\}$ and the reference point is $Ref = (10,10,10)$. Here we only detail the calculation procedure for P_4, analogous process could be found for the others. Before computation, the

pretreatment is carried out to classify these points into their corresponding categories. Particularly, $PD = \{P_1, P_3\}$, $PL = \{P_2\}$ and $NL = \{P_5, P_6\}$. Intuitively, the individual dominance area of P_4 is extracted as the cube shown in the top right of Fig. 4. This cube could be chopped into three slices and then projected onto f_2–f_3 plane as the three polygons that shown in the bottom of Fig. 4. *Ref* indicates the original reference point and *Ref′* represents the temporary one. Correspondingly, the individual dominance area of P_4 in each slice is marked in reticular lines. For example, the hypervolume indicator of the individual dominance area for P_4 in slice 1 could be obtained by subtracting the hypervolume indicator of the dominance area for P_3. The other two slices could be processed analogously.

Algorithm 4: Calculating the hypervolume indicator of the individual dominance area for circumstance 4, i.e., $SubProc4()$

```
1  begin
2      if NL == ∅ then
3          The least upper bound point of p at the first dimension is u_1;
4          depth = |obj_1(p) − u_1|;
5          Relocate the position of current reference point Ref′;
6          Select qualified individuals from PD to fill R;
7          Calculate the hypervolume indicator of R, denoted as CutArea;
8          TotalArea = |(obj_2(p) − obj_2(Ref′))| ∗ |(obj_3(p) − obj_3(Ref′))|;
9          CurArea = TotalArea − CutArea;
10         ExcHv(p) = depth ∗ CurArea;
11
12     else
13         for i=1 to ||NL|| do
14             The least upper bound point of NL[i] at the first dimension is u_1;
15             SliceDepth = |obj_1(NL[i]) − u_1|;
16             Relocate the position of current reference point Ref′;
17             Select qualified individuals from PD to fill R;
18             Calculate the hypervolume indicator of R, denoted as CutArea;
19             TotalArea = |(obj_2(NL[i]) − obj_2(Ref′))| ∗ |(obj_3(NL[i]) − obj_3(Ref′))|;
20             CurArea = TotalArea − CutArea;
21             ExcHv(p)+ = SliceDepth ∗ CurArea;
22         end
23     end
24     return ExcHv(p);
25 end
```

3.5. Truncation procedure

In MOEAs, the Pareto optimal set might be enormous, or even infinite. Thus it is impossible and unnecessary to archive all the non-dominated solutions. Alternatively, it would be more reasonable to maintain a representative non-dominated subset. Similar to SPEA2, here we also use an external archive to preserve elite individuals that are selected from the evolutionary population. A truncation procedure is invoked if the cardinality of the archive exceeds its pre-defined size. Different from the other state-of-the-art truncation methods, we use the hypervolume indicator of individual dominance area to guide the truncation procedure. The specific implementation of this truncation procedure is given as follows.

Step 1. Calculate the hypervolume indicator of the individual dominance area for each individual in the external archive Q;
Step 2. The individual that contributes the least hypervolume indicator is eliminated from Q;
Step 3. $|Q| = |Q| - 1$, the total procedure terminates if $|Q| = N$, otherwise turns back to Step 1.

As discussed in [26], the necessary and sufficient condition for a solutions set to achieve Pareto optimal is the maximization of its corresponding hypervolume indicator. Here the truncation procedure, which is based on the criterion that maximizes the hypervolume indicator of the current population, is a local greedy strategy. It ensures the search towards the Pareto optimal and overcomes degeneration. Particularly, after the elimination of the least contributed individual, the hypervolume indicator of the individual dominance area for any other remaining individual in the current external archive has to be updated correspondingly.

3.6. Framework of DNMOEA/HI

We have already discussed the two key design issues of our proposed algorithm, namely fitness assignment strategy and truncation procedure. Here the main framework of DNMOEA/HI is given as follows.

Step 1. Set the sizes of the evolutionary popualtion E and the external archive Q be N. Thus,the size of the intermediate population P is $2N$. Initialize generation count $gen = 1$.
Step 2. Randomly initialize E and Q as E_{gen} and Q_{gen}.
Step 3. Calculate the fitness values of all individuals in E_{gen} and Q_{gen}.
Step 4. Merge E_{gen} and Q_{gen} together into P_{gen}, then select out all non-dominated individuals and copy them into Q_{gen+1}. If $|Q_{gen+1}| < N$, the best remaining individuals in P_{gen} are copied to Q_{gen+1} till it is filled. This could be implemented by sorting the P_{gen} based on ascending order of their fitness values. If $|Q_{gen+1}| > N$, truncation procedure is evoked to prune the external archive Q_{gen+1} to the predefined size. If the termination criteria are met, outputs all individuals in Q_{gen+1}. Otherwise turns to Step 5.
Step 5. Carry out a tournament selection upon Q_{gen+1}, and implement evolutionary operations (i.e., crossover and mutation) on the selected individuals. The evolved individuals are stored into E_{gen+1} for the next generation, set $gen = gen + 1$, then turns to Step 3.

3.7. Time complexity analysis

Let the population size is N, the time complexity of the calculation procedure for the tree neighborhood density is determined by the time cost for generating minimum spanning tree. Here Prim algorithm [46] is used to generate the minimum spanning tree, thus the time complexity is $O(N^2)$. In fitness assignment procedure, the time complexity to compute the strength value $R(i)$ is the same as [10], namely $O(N^2)$. Therefore, the whole fitness assignment procedure cost $O(N^2) + O(N^2) = O(N^2)$ time. The time complexity of Algorithm 1 is determined by the sorting algorithm. We use quick sort here, which costs $O(NlogN)$. In Algorithm 2, finding out four kinds of individuals which meet the point taxonomy proposed in Section 3.4 costs $O(N^2)$ time. And the time cost to calculate the hypervolume indicator of the individual dominance area for in Algorithms 3 and 4 is $O(N^2)$. Consequently, the time complexity for calculating the hypervolume indicator of the individual dominance area for the whole population is $O(N^3)$. In summary, DNMOEA/HI evolves one generation costs $O(N^3)$ time.

4. Experiments

In order to validate the efficiency of our proposed DNMOEA/HI, we compare the performance of DNMOEA/HI with six other MOEAs (i.e., NSGA-II [8], SPEA2 [10], IBEA [29], SMS-EMOA [30], AbYSS [47], MOEA/D-DE [48]). Particularly, NSGA-II and SPEA2 are two state-of-the-art MOEAs whose performances have been verified in many studies. IBEA and SMS-EMOA are two hypervolume based MOEAs as our DNMOEA/HI. Different from the traditional MOEAs, AbYSS adapts the well-known scatter search template that designed for solving single-objective optimization to the multi-objective optimization domain. MOEA/D-DE is a strong and efficient MOEA based on the idea of decomposition which is popular in the traditional mathematical programming.

In the following paragraph, a brief introduction on the benchmark problems and performance metrics are given at first. Next, our experiments to compare the performances of these MOEAs are described and analyzed. After that, the scalability of DNMOEA/HI to the number of objectives is verified. At last, we further compare their performance on some dynamic test problems.

4.1. Test problem

In this section, we introduce the different sets of both bi- and tri-objective benchmark problems used in this work. These test problems have been widely used in the multi-objective optimization domain. ZDT-1, ZDT-2, ZDT-3, ZDT-4 and ZDT-6,

Table 1
Properties of WFG test suite.

Problem	Separability	Modality	Bias	Geometry
WFG-1	Separable	Uni	Polynomial, flat	Convex, mixed
WFG-2	Non-separable	f_1 uni, f_2 multi	no bias	convex, disconnected
WFG-3	Non-separable	Uni	No bias	Linear, degenerate
WFG-4	Non-separable	Multi	No bias	Concave
WFG-5	Separable	Deceptive	No bias	Concave
WFG-6	Non-separable	Uni	No bias	Concave
WFG-7	Separable	Uni	Dependent	Concave
WFG-8	Non-separable	Uni	Dependent	Concave
WFG-9	Non-separable	Multi, deceptive	Dependent	Concave

which were defined in [49], are selected as the bi-objective test problems. Furthermore, seven bi-objective unconstrained test problems (i.e., UF-1 to UF-7), which were proposed in CEC 2009 competition [50], are also included in this benchmark. As for the tri-objective benchmark, it is composed of DTLZ-1 to DTLZ-7 [51], tri-objective version of WFG-1 to WFG-9 [52] and UF-8 to UF-10 defined in [50]. More detailed properties of the WFG Toolkit problems are given in Table 1.

4.2. Performance metrics

Two critical issues are normally taken into account when evaluating the performances of MOEAs on the test problems. On the one hand, the distance between the obtained solutions set to the true Pareto optimal front should be as minimal as possible. On the other hand, the spread of solutions found should be maximized and the distribution should be as uniform as possible. In order to meet these two issues, the exact location and distribution of the true Pareto optimal front is necessary to be known as a priori. Fortunately, the true Pareto optimal fronts of all the benchmark problems utilized in this paper have been already known.

The performance metrics used in our numerical experiments aim at evaluating the following three aspects of the obtained solutions.

1. Evaluate the closeness between the obtained solutions set and the true Pareto optimal front.
2. Evaluate the diversity of the obtained solutions.
3. Evaluate the comprehensive performance of the obtained solutions set on both the convergence and diversity.

The following three performance metrics are chosen to evaluate the aforementioned aspects of MOEAs respectively.

1. *Generation Distance* (*GD*): This metric was firstly proposed by Veldhuizen and Lamont [53] to measure the distance between the Pareto approximated front and the Pareto optimal front. It is formulated as:

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n}, \tag{8}$$

   where n is the number of solutions in the Pareto approximated front and d_i indicates the Euclidean distance between each of these solutions and its nearest neighbor in the Pareto optimal front in the objective space. It is clear that lower *GD* value means better convergence of the obtained solutions. Particularly, $GD = 0$ means that all Pareto approximated solutions are located on the Pareto optimal front. In order to obtain a reliable result, obtained solutions are normalized before calculating this metric.
2. *Uniformity Assessment* (*UA*): This is a robust metric introduced by Li et al. [54] to calculate the ratio of distances between the individuals and their neighbors. UA takes the value between zero and one (one represents the best). The larger value it obtains, the better uniformity it achieves. Readers could have a more detailed description in [54].
3. *Hypervolume Indicator* (*HV*): The definition of *HV* has been given in Definition 5, and a larger *HV* value is preferable when comparing the performances of different solutions set. Here the objective function values have been normalized when evaluating this metric because of the different scaling of objectives. Without loss of generality, the reference points used for assessments are given in Table 2.

4.3. Parameters settings of MOEAs

Here the decision variables are real coded. In other words, each gene in a chromosome (or an individual) indicates the value of the corresponding decision variable. As for evolutionary operators, the classical simulated binary crossover (SBX) [55] is used for crossover operation. The crossover rate $p_c = 0.9$ and the crossover index $\eta_c = 20$. The polynomial mutation [56] is employed with mutation rate $p_m = 1/nreal$ (*nreal* is the dimension of the decision space) and the mutation index $\eta_m = 20$. The population sizes are consistent for all benchmark problems. The sizes of two reference sets in AbYSS are set as 10. More detailed parameter settings are given in Table 3.

Table 2
Reference points settings for different test problems.

Test problems	Reference points
ZDT test suite	(2.0,2.0)
DTLZ-1	(1.0,1.0,1.0)
DLTZ-2 to DTLZ-6	(2.0,2.0,2.0)
DTLZ-7	(2.0,2.0,7.0)
WFG test suite	(3.0,5.0,7.0)
UF-1 to UF-7	(2.0,2.0)
UF-8 to UF-10	(2.0,2.0,2.0)

Table 3
Parameter settings for different test problems.

Test problems	Population size	No. of generation
ZDT test suite	100	200
DTLZ test suite	100	300
WFG test suite	100	300
UF-1 to UF-7	100	300
UF-8 to UF-10	100	500

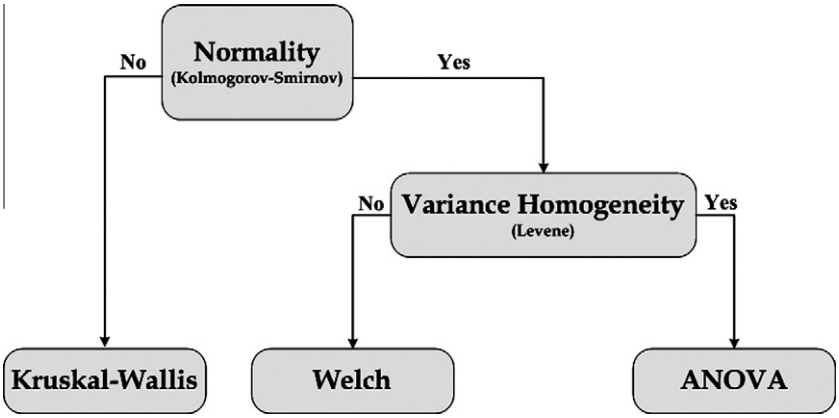

Fig. 6. The general structure of the statistical analysis used in this work.

Fifty independent simulation runs with randomly generated populations are performed for each benchmark problem. The best experimental results are highlighted with boldface and dark background in corresponding tables. As discussed in [57], all these candidate MOEAs are stochastic algorithms, the use of statistical tests to improve the significant of the evaluation process for the comparison of different algorithms has become a popular technique in the field of computational intelligence. The general structure of statistical analysis is given in Fig. 6. First of all, the Kolmogorov–Smirnov test was performed to check whether the results follow a Gaussian distribution or not. If the results are not Gaussian distributed, then a non-parametric Kruskal–Wallis test is used to compare the median of each algorithm. On the other hand, the homogeneity of the variances is checked by the use of Levene test if the results are Gaussian distributed. If variances of the results are equal, the ANOVA test is invoked to verify the confidence. Otherwise, the Welch test is performed to accomplish this task. In the statistical test, 95% confidence level is adopted to compare the significance between two competing algorithms, with † indicating that DNMOEA/HI is significantly better than all its competitors in the corresponding table, while − representing that the best competitor significantly outperforms DNMOEA/HI.

4.4. Experimental environment

DNMOEA/HI is coded in ANSI C. As for the other MOEAs, we use jMetal framework [58] with SUN Java JDK/JRE 1.6.0.2. All numerical experiments are conducted on Intel Core 2 Duo CPU P8400 @ 2.26 GHz, 2 GB RAM computer, with Microsoft Windows 7 operating system.

4.5. Measure of convergence

As is mentioned in Section 4.2, the convergence of each MOEA is measured by GD. The experimental results of all candidate MOEAs on this metric are tabulated in Table 4. The last column of Table 4 gives the ratio of the corresponding MOEA win on all 31 benchmark problems. Generally speaking, our proposed DNMOEA/HI outperforms on 20 out of 31 benchmark problems for GD metric, performing significantly better in 17 of them. The second best candidate algorithm is another hypervolume based MOEA, i.e., IBEA, which wins on 8 out of 31 benchmark problems. Furthermore, both NSGA-II and SPEA2 are outperformed on all benchmark problems, while each of the other three candidates wins on only 1 benchmark problem. Specifically, as for bi-objective benchmark problems, DNMOEA/HI only be outperformed on 3 out 12 test cases, i.e., ZDT-1, UF-5 and UF-7. Comparing to two state-of-the-art MOEAs, i.e., NSGA-II and SPEA2, MOEA/D-DE and AbYSS show competitive performances on ZDT test suites. Referring to the other 19 tri-objective benchmark problems, our proposed DNMOEA/HI wins on 11 of them. Especially for some difficult benchmark problems, e.g., DTLZ-1, DTLZ-3, WFG-1, WFG-9, the superiority of DNMOEA/HI is obvious. It is worth noting that MOEA/D-DE shows very encouraging performance on WFG-8.

Table 4
Performance results on GD metric for seven MOEAs on the benchmark problems.

	NSGA-II	SPEA2	IBEA	SMS-EMOA	MOEA/D-DE	AbYSS	DNMOEA/HI	S
ZDT 1	1.807E-4(6.75E-5)	1.627E-4(5.97E-5)	**8.998E-6(1.31E-5)**	1.736E-5(2.51E-5)	3.665E-5(5.93E-5)	1.056E-4(2.13E-5)	1.423E-5(2.93E-5)	-
ZDT 2	1.529E-4(2.99E-5)	1.569E-4(5.22E-5)	1.035E-5(7.63E-6)	2.098E-5(6.59E-6)	1.495E-5(5.66E-6)	5.395E-5(6.54E-5)	**8.996E-6(6.31E-7)**	†
ZDT 3	9.672E-5(4.70E-5)	9.720E-5(4.70E-5)	3.490E-5(1.54E-5)	5.880E-5(1.05E-5)	5.960E-5(1.98E-5)	5.970E-5(1.22E-5)	**3.334E-5(3.10E-6)**	†
ZDT 4	3.690E-4(1.79E-4)	5.802E-4(4.61E-4)	1.862E-4(4.30E-4)	2.345E-4(1.31E-4)	3.111E-5(1.11E-5)	1.029E-3(1.31E-4)	**1.068E-5(3.54E-6)**	†
ZDT 6	7.255E-4(7.63E-5)	1.118E-3(9.76E-4)	1.896e-4(1.93E-5)	1.946E-4(4.25E-5)	3.653E-6(4.67E-5)	7.489E-5(5.35E-5)	**3.571E-6(1.74E-7)**	†
DTLZ 1	6.743E-2(2.25E-1)	1.384E-1(2.47E-1)	1.838E-4(2.76E-4)	2.783E-4(2.15E-4)	2.487E-4(4.35E-5)	3.382E-4(6.21E-5)	**1.811E-4(3.11E-4)**	†
DTLZ 2	1.330E-3(1.72E-4)	1.246E-3(2.22E-4)	**4.634E-4(1.55E-4)**	5.012E-4(4.37E-4)	5.644E-4(3.72E-5)	5.668E-4(3.51E-5)	4.666E-4(3.71E-4)	-
DTLZ 3	9.603E-1(1.09E+0)	4.700E-1(7.85E-1)	5.012E-4(1.81E-3)	5.445E-4(2.23E-3)	5.601E-4(8.21E-5)	1.155E-1(9.84E-2)	**5.010E-4(1.09E-3)**	
DTLZ 4	8.362E-3(6.09E-3)	7.557E-3(6.18E-3)	**4.231E-4(6.44E-4)**	4.613E-4(4.11E-4)	4.635E-4(7.23E-5)	5.873E-4(9.32E-4)	4.598E-4(4.63E-4)	-
DTLZ 5	2.168E-4(7.22E-5)	1.930E-4(5.42E-5)	**1.173E-5(4.15E-6)**	1.318E-5(2.03E-5)	1.355E-5(1.01E-6)	1.867E-5(4.06E-5)	1.276E-5(1.81E-5)	-
DTLZ 6	8.784E-2(1.08E-2)	9.044E-2(8.83E-3)	1.428E-5(4.13E-4)	6.514E-5(4.17E-4)	4.363E-6(5.61E-7)	4.714E-2(5.66E-3)	**4.315E-6(3.55E-7)**	†
DTLZ 7	3.702E-3(1.08E-3)	3.874E-3(1.49E-3)	**7.634E-4(1.24E-4)**	1.141E-3(2.33E-4)	8.370E-4(1.03E-4)	8.000E-4(2.14E-4)	7.956E-4(2.18E-4)	-
WFG 1	3.745E-2(7.63E-3)	3.610E-2(8.97E-3)	6.810E-3(6.01E-4)	6.927E-3(5.94E-4)	3.025E-2(2.05E-3)	2.559E-2(3.94E-3)	**6.792E-3(4.38E-4)**	†
WFG 2	1.234E-2(1.36E-3)	1.267E-2(1.06E-3)	1.130E-2(1.01E-3)	1.024E-2(1.46E-3)	1.167E-2(2.06E-3)	1.079E-2(1.98E-3)	**9.982E-3(6.58E-4)**	†
WFG 3	2.656E-2(1.56E-3)	3.058E-2(1.17E-3)	**2.235E-2(1.65E-3)**	2.336E-2(2.01E-3)	2.912E-2(2.25E-3)	2.682E-2(3.03E-3)	2.946E-2(2.69E-3)	-
WFG 4	1.812E-2(1.48E-3)	1.885E-2(8.29E-4)	3.367E-2(6.74E-3)	2.588E-2(6.19E-3)	1.653E-2(5.29E-3)	1.826E-2(4.20E-3)	**1.507E-2(5.35E-3)**	†
WFG 5	1.431E-2(1.76E-3)	1.687E-2(7.30E-4)	1.707E-2(6.29E-3)	1.522E-2(5.24E-4)	1.331E-2(4.21E-3)	**1.220E-2(5.12E-3)**	1.613E-2(4.39E-4)	-
WFG 6	8.685E-3(4.58E-4)	8.597E-3(4.87E-4)	8.094E-3(5.22E-4)	7.121E-3(4.35E-4)	6.041E-3(3.02E-4)	8.852E-3(5.31E-4)	**5.486E-3(3.52E-4)**	†
WFG 7	4.145E-3(2.32E-4)	3.830E-3(2.13E-4)	8.878E-3(2.16E-4)	6.792E-3(2.68E-4)	6.709E-3(5.03E-4)	4.068E-3(4.66E-4)	**3.782E-3(3.05E-4)**	†
WFG 8	7.578E-3(1.20E-3)	7.920E-3(4.92E-4)	1.685E-2(4.36E-3)	1.081E-2(3.08E-3)	**3.853E-3(4.16E-4)**	8.335E-3(9.36E-4)	7.704E-3(2.55E-4)	=
WFG 9	1.081E-2(6.77E-4)	1.079E-2(5.11E-4)	1.766E-2(3.06E-3)	2.087E-2(5.06E-3)	1.639E-2(4.22E-3)	1.200E-2(3.62E-3)	**1.039E-2(2.27E-3)**	†
UF 1	1.574E-3(2.63E-3)	1.035E-3(1.99E-3)	4.490E-4(3.63E-4)	4.570E-4(2.62E-4)	1.023E-2(3.52E-3)	2.300E-3(4.09E-3)	**4.261E-4(1.95E-4)**	†
UF 2	2.300E-3(5.80E-4)	1.800E-3(4.45E-4)	3.462E-3(5.15E-4)	4.185E-3(4.47E-4)	1.169E-2(2.23E-3)	4.690E-3(3.15E-3)	**1.360E-3(7.65E-4)**	†
UF 3	1.691E-2(6.16E-3)	1.088E-2(5.45E-3)	1.322E-3(5.32E-4)	1.229E-2(4.35E-3)	1.598E-3(3.49E-4)	2.397E-2(4.32E-3)	**1.128E-3(6.04E-3)**	†
UF 4	6.093E-3(1.93E-4)	6.045E-3(2.00E-4)	8.857E-3(3.17E-4)	1.081E-2(4.75E-3)	1.073E-2(5.13E-3)	1.196E-2(4.63E-3)	**5.730E-3(2.23E-4)**	†
UF 5	4.575E-2(1.81E-2)	4.207E-2(1.66E-2)	1.192E-1(2.17E-2)	**2.883E-2(1.55E-2)**	8.601E-2(2.42E-2)	1.630E-1(1.83E-2)	4.215E-2(2.26E-2)	-
UF 6	1.774E-2(1.02E-2)	1.357E-2(7.58E-3)	1.712E-3(6.74E-4)	4.184E-3(7.01E-4)	2.201E-2(5.17E-3)	1.100E-2(3.15E-3)	**1.372E-3(8.27E-3)**	†
UF 7	1.051E-3(8.62E-4)	1.016E-3(1.10E-3)	**4.460E-4(2.22E-5)**	4.480E-4(1.85E-5)	7.199E-3(2.35E-4)	2.073E-3(4.37E-3)	4.474E-4(2.89E-3)	
UF 8	1.258E-1(4.21E-2)	1.512E-1(3.25E-2)	**1.132E-2(3.35E-3)**	3.382E-2(5.68E-3)	2.125E-2(4.03E-3)	4.535E-2(2.35E-3)	1.536E-2(3.33E-3)	-
UF 9	3.447E-1(1.34E-1)	3.634E-1(2.25E-2)	3.246E-2(2.53E-3)	8.321E-1(2.65E-1)	1.086E-2(3.28E-3)	5.692E-1(3.04E-2)	**1.068E-2(2.36E-3)**	
UF 10	1.039E-1(5.69E-2)	1.012E-1(3.86E-2)	1.170E-2(3.37E-3)	1.010E-2(4.03E-3)	1.036E-2(3.53E-3)	1.126E-1(6.57E-2)	**1.001E-2(1.31E-3)**	
Ratio	**0/31**	**0/31**	**8/31**	**1/31**	**1/31**	**1/31**	**20/31**	

Table 5
Performance results on UA metric for seven MOEAs on the benchmark problems.

	NSGA-II	SPEA2	IBEA	SMS-EMOA	MOEA/D-DE	AbYSS	DNMOEA/HI	S
ZDT 1	7.206E-3(5.92E-4)	7.815E-1(2.03E-2)	6.197E-1(3.25E-2)	8.809E-1(2.80E-2)	**9.690E-1(3.55E-2)**	8.727E-1(1.80E-2)	8.080E-1(1.62E-2)	-
ZDT 2	7.402E-3(6.57E-4)	7.775E-1(2.09E-2)	6.166E-1(3.21E-2)	8.706E-1(2.14E-2)	**9.861E-1(5.34E-3)**	8.925E-1(2.65E-3)	8.219E-1(1.89E-2)	-
ZDT 3	7.911E-3(7.19E-4)	7.264E-1(1.89E-2)	4.114E-1(3.28E-2)	**8.057E-1(3.51E-2)**	7.175E-1(4.46E-2)	7.872E-1(3.05E-2)	7.660E-1(1.77E-2)	-
ZDT 4	7.539E-3(7.62E-4)	7.719E-1(3.76E-2)	2.863E-1(6.85E-2)	8.190E-1(5.31E-2)	**9.971E-1(7.00E-2)**	8.504E-1(5.36E-2)	8.055E-1(1.85E-2)	-
ZDT 6	5.614E-3(4.75E-4)	7.385E-1(2.44E-2)	4.461E-1(4.01e-2)	6.924E-1(2.49E-2)	**9.912E-1(4.33E-2)**	8.735E-1(2.75E-2)	8.199E-1(2.34E-2)	-
DTLZ 1	3.679E-1(1.66E+0)	7.786E-1(6.97E-2)	1.500E-1(3.92E-2)	**8.341E-1(5.61E-2)**	5.772E-1(5.69E-2)	3.672E-1(2.36E-2)	6.997E-1(3.37E-2)	-
DTLZ 2	5.769E-2(5.00E-3)	**8.208E-1(1.68E-2)**	5.708E-1(3.12E-2)	7.369E-1(5.56E-2)	5.619E-1(6.12E-2)	3.792E-1(5.32E-2)	6.933E-1(5.38E-2)	-
DTLZ 3	3.935E+0(6.54E+0)	**8.329E-1(2.61E-2)**	4.432E-1(2.12E-3)	6.975E-1(1.33E-2)	5.573E-1(3.29E-2)	1.974E-1(2.71E-2)	6.910E-1(1.63E-2)	-
DTLZ 4	8.297E-2(4.01E-2)	**7.187E-1(5.43E-2)**	5.408E-1(8.92E-2)	7.586E-1(4.19E-2)	2.698E-1(2.33E-2)	4.638E-1(5.33E-2)	5.794E-1(5.96E-2)	-
DTLZ 5	9.809E-3(7.31E-4)	7.766E-1(2.04E-2)	4.992E-1(4.82E-2)	6.512E-1(1.08E-2)	1.933E-1(2.38E-2)	**8.860E-1(3.21E-2)**	6.779E-1(2.62E-2)	-
DTLZ 6	9.399E-2(2.84E-2)	**6.755E-1(3.55E-2)**	3.070E-1(1.38E-1)	6.358E-1(3.73E-2)	2.043E-1(2.51E-2)	3.074E-1(3.43E-2)	5.853E-1(3.07E-2)	–
DTLZ 7	7.150E-2(9.56E-3)	**7.420E-1(2.36E-2)**	4.640E-1(4.62E-2)	7.137E-1(2.27E-2)	3.383E-1(5.63E-2)	4.776E-1(3.67E-2)	4.505E-1(3.30E-2)	–
WFG 1	4.405E-1(3.80E-2)	7.879E-1(2.12E-2)	4.983E-1(5.69E-2)	4.648E-1(4.18E-2)	4.064E-1(6.67E-2)	3.907E-1(7.54E-2)	**8.060E-1(3.15E-2)**	†
WFG 2	4.134E-1(4.08E-2)	7.834E-1(1.74E-2)	5.121E-1(2.94E-2)	5.512E-1(3.41E-2)	4.684E-1(2.81E-2)	4.076E-1(3.11E-2)	**8.249E-1(2.15E-2)**	†
WFG 3	4.129E-1(4.08E-2)	7.774E-1(1.89E-2)	5.819E-1(2.03E-2)	7.085E-1(1.29E-2)	4.525E-1(3.12E-2)	3.906E-1(2.95E-2)	**7.878E-1(3.10E-2)**	†
WFG 4	4.368E-1(3.54E-2)	7.917E-1(1.77E-2)	5.958E-1(3.51E-2)	6.834E-1(1.53E-2)	4.202E-1(2.03E-2)	4.713E-1(4.31E-2)	**8.028E-1(2.18E-2)**	†
WFG 5	4.170E-1(4.18E-2)	7.991E-1(2.58E-2)	6.035E-1(6.32E-2)	6.571E-1(3.31E-2)	6.281E-1(6.04E-2)	3.854E-1(3.74E-2)	**8.132E-1(1.95E-2)**	†
WFG 6	3.976E-1(4.21E-2)	**8.144E-1(1.89E-2)**	5.067E-1(3.24E-2)	6.444E-1(4.05E-2)	5.162E-1(4.11E-2)	4.155E-1(3.19E-2)	7.924E-1(2.39E-2)	-
WFG 7	3.893E-1(4.10E-2)	8.122E-1(1.83E-2)	5.291E-1(3.42E-2)	6.633E-1(4.07E-2)	5.485E-1(3.56E-2)	3.312E-1(2.01E-2)	**8.153E-1(2.08E-2)**	†
WFG 8	4.012E-1(4.35E-2)	8.123E-1(1.46E-2)	3.009E-1(3.03E-2)	5.386E-1(6.11E-2)	4.571E-1(5.03E-2)	3.723E-1(3.02E-2)	**8.358E-1(1.53E-2)**	†
WFG 9	4.226E-1(4.43E-2)	8.072E-1(1.74E-2)	5.296E-1(2.03E-2)	5.827E-1(3.14E-2)	6.122E-1(3.08E-2)	5.173E-1(4.25E-2)	**8.399E-1(5.30E-2)**	†
UF 1	2.805E-1(6.62E-2)	3.857E-1(1.21E-1)	1.946E-1(2.65E-2)	3.601E-1(6.09E-2)	**5.073E-1(6.53E-2)**	3.193E-1(2.34E-2)	4.836E-1(3.73E-2)	-
UF 2	3.880E-1(4.18E-2)	6.795E-1(2.77E-2)	2.213E-1(3.35E-2)	6.481E-1(2.36E-2)	**8.700E-1(3.31E-2)**	5.235E-1(4.17E-2)	7.696E-1(3.74E-2)	-
UF 3	1.268E-1(3.44E-2)	9.704E-2(4.02E-2)	1.942E-1(3.41E-2)	1.314E-1(4.37E-2)	5.506E-1(2.65E-2)	2.761E-2(3.41E-2)	**5.709E-1(2.71E-2)**	†
UF 4	3.930E-1(3.99E-2)	6.168E-1(5.23E-2)	1.731E-1(4.39E-2)	2.913E-1(3.57E-2)	5.290E-1(4.62E-2)	2.192E-1(3.73E-2)	**6.573E-1(5.31E-2)**	†
UF 5	1.314E-1(3.58E-2)	1.088E-1(4.18E-2)	9.574E-2(3.83E-3)	2.284E-1(4.81E-2)	**2.978E-1(3.93E-2)**	8.855E-2(2.53E-3)	1.093E-1(2.71E-2)	-
UF 6	1.409E-1(3.45E-2)	1.141E-1(4.06E-2)	2.513E-1(3.19E-2)	3.257E-1(1.13E-2)	**4.205E-1(3.42E-2)**	1.085E-1(2.53E-2)	3.913E-1(2.54E-2)	-
UF 7	3.159E-1(1.08E-1)	5.054E-1(2.44E-1)	3.374E-1(1.87E-1)	**8.031E-1(2.04E-1)**	6.968E-1(3.39E-1)	7.394E-1(2.52E-1)	7.255E-1(2.11E-1)	-
UF 8	3.322E-1(4.17E-2)	**4.892E-1(5.36E-2)**	2.139E-1(3.68E-2)	3.790E-1(4.25E-2)	2.729E-1(3.25E-2)	4.375E-1(2.83E-2)	3.256E-1(4.21E-2)	-
UF 9	3.516E-1(4.90E-2)	4.362E-1(2.52E-2)	1.384E-1(1.82E-2)	4.037E-1(2.32E-2)	3.942E-1(2.63E-2)	4.583E-1(3.05E-2)	**4.612E-1(2.08E-2)**	†
UF 10	2.907E-1(5.95E-2)	3.536E-1(3.32E-2)	1.757E-1(2.63E-2)	1.224E-1(3.05E-2)	3.343E-1(4.39E-2)	**4.215E-1(3.16E-2)**	3.242E-1(3.12E-2)	-
Ratio	**0/31**	**7/31**	**0/31**	**3/31**	**8/31**	**2/31**	**11/31**	

Table 6
Performance results on HV metric for seven MOEAs on the benchmark problems.

	NSGA-II	SPEA2	IBEA	SMS-EMOA	MOEA/D-DE	AbYSS	DNMOEA/HI	S
ZDT 1	3.65882(3.99E-4)	3.65880(6.78E-4)	3.65927(6.43E-4)	3.66165(2.52E-4)	3.66121(1.96E-4)	3.66017(3.44E-4)	**3.66193(2.55E-5)**	†
ZDT 2	3.32438(6.07E-4)	3.32400(1.01E-3)	3.32686(5.78E-4)	3.32774(5.93E-4)	3.32802(1.56E-4)	3.32846(2.35E-4)	**3.32851(3.39E-5)**	†
ZDT 3	4.81271(4.02E-4)	4.79666(7.18E-2)	4.80681(2.09E-1)	4.81509(6.15E-2)	4.81151(7.18E-2)	4.44818(6.98E-2)	**4.81541(6.93E-2)**	†
ZDT 4	3.65136(5.52E-3)	3.64597(1.19E-2)	2.27936(1.54E-1)	3.65614(6.78E-2)	3.66063(1.19E-2)	3.63770(2.21E-2)	**3.66199(5.11E-4)**	†
ZDT 6	3.01962(2.66E-3)	3.01553(3.44E-3)	3.03436(1.99E-3)	3.03568(1.79E-3)	3.04165(3.18E-3)	3.04024(2.56E-3)	**3.04180(1.81E-5)**	†
DTLZ 1	0.94723(6.01E-2)	0.96723(2.81E-2)	0.56871(7.15E-3)	0.97379(4.90E-3)	0.96628(4.99E-3)	0.96841(4.99E-3)	**0.98071(2.81E-3)**	†
DTLZ 2	7.32607(2.55E-2)	**7.38261(1.24E-2)**	5.58817(4.31E-1)	7.34118(1.22E-1)	7.36494(2.98E-2)	7.37287(3.16E-2)	7.37907(1.40E-1)	-
DTLZ 3	0.49633(1.29E+0)	2.04757(2.67E+0)	1.33153(1.65E-3)	7.38064(2.01E-2)	7.35891(1.38E-2)	2.44554(1.87E-2)	**7.38119(1.75E-2)**	
DTLZ 4	6.87040(6.30E-1)	6.96379(6.18E-1)	6.25939(0.80E-1)	**7.32931(1.13E-1)**	7.30586(7.03E-1)	7.38787(6.98E-1)	6.92153(3.51E-1)	-
DTLZ 5	6.09980(1.12E-3)	6.10378(2.94E-3)	4.12704(5.54E-2)	6.12113(1.06E-3)	6.08966(3.65E-3)	6.10388(2.34E-3)	**6.21163(1.97E-3)**	†
DTLZ 6	3.73857(2.98E-1)	3.69282(2.61E-1)	4.12705(5.55E-2)	5.90107(1.10E-1)	5.98955(3.14E-2)	4.98941(4.05E-2)	**5.99975(2.77E-1)**	
DTLZ 7	13.0627 (9.65E-2)	13.2700(6.03E-1)	13.52223(5.69E+0)	13.86821(4.20E-1)	13.1946(8.71E-1)	13.1556(7.51E-1)	**13.86986(7.43E-2)**	
WFG 1	45.6018(1.53E+0)	45.9082(1.75E+0)	46.0862(1.67E+0)	45.9781(2.01E+0)	45.8843(1.02E+0)	45.8957(1.98E+0)	**46.8413 (1.10E+0)**	†
WFG 2	94.0148(6.17E+0)	93.1287(7.05E+0)	98.8121(5.71E+0)	98.0168(3.57E+0)	96.4872(4.56E+0)	74.0923(6.22E+0)	**99.0107(4.26E+0)**	†
WFG 3	73.5298(5.90E-1)	71.9774(6.15E-1)	73.1395(6.14E+0)	74.2080(7.65E+0)	74.3918(5.94E+0)	74.4230(6.12E+0)	**75.1583(5.62E+0)**	†
WFG 4	59.4254(7.83E-1)	60.9057(5.24E-1)	54.9582(4.91E-1)	64.9283(6.23E-1)	58.2394(5.23E-1)	57.4921(4.17E-1)	**69.2890(2.36E-1)**	†
WFG 5	68.3348(6.28E-1)	69.0810(6.08E-1)	66.2481(6.26E-1)	67.4021(1.24E-1)	60.1503(2.35E-1)	60.2592(4.22E-1)	**69.2054(3.23E-1)**	†
WFG 6	68.5098(8.41E-1)	69.9045(6.61E-1)	67.4568(5.42E-1)	68.3456(3.23E-1)	63.4836(2.65E-1)	61.6789(3.21E-1)	**70.6170(3.52E-1)**	†
WFG 7	71.9666(7.36E-1)	73.2849(5.00E-1)	69.1251(4.35E-1)	71.6879(6.35E-1)	66.6482(7.36E-1)	66.6823(2.99E-1)	**74.0404(3.21E-1)**	†
WFG 8	64.2618(5.97E-1)	65.8147(6.08E-1)	61.6853(5.26E-1)	64.6823(3.69E-1)	61.6812(6.32E-1)	60.2389(3.39E-1)	**65.9464(3.80E-1)**	
WFG 9	64.9680(8.07E-1)	66.5953(1.23E+0)	64.0982(9.03E-1)	65.1072(4.66E-1)	65.1582(6.05E-1)	63.9282(1.09E-1)	**67.6778(2.03E-1)**	†
UF 1	3.35669(9.21E-2)	3.30421(1.22E-1)	3.49962(1.09E-1)	3.50375(8.92E-2)	3.52113(7.09E-2)	3.32599(6.28E-2)	**3.52787(6.76E-2)**	
UF 2	3.54245(4.47E-2)	3.51218(7.04E-2)	3.58077(6.81E-2)	3.57903(5.10E-2)	**3.60054(4.65E-2)**	3.40573(3.33E-2)	3.59892(4.67E-2)	
UF 3	2.79321(7.18E-2)	2.79109(1.08E-1)	2.53357(2.15E-1)	2.69006(3.51E-1)	3.64404(2.73E-1)	2.76121(3.06E-1)	**3.65019(1.18E-1)**	†
UF 4	3.18026(5.49E-3)	3.17989(5.72E-3)	3.18614(4.36E-3)	3.11154(4.16E-3)	3.07413(3.34E-3)	3.11977(3.39E-3)	**3.19579(6.39E-3)**	†
UF 5	2.01807(3.19E-1)	1.97166(3.28E-1)	1.95487(3.39E-1)	2.62261(4.08E-1)	1.51522(4.16E-1)	1.86062(3.99E-1)	**2.63152(3.70E-1)**	†
UF 6	2.66042(2.56E-1)	2.54486(2.89E-1)	2.87853(3.31E-1)	2.85088(2.15E-1)	**3.14898(5.11E-1)**	3.07124(4.62E-1)	3.06183(2.69E-1)	-
UF 7	3.02964(3.85E-1)	2.97103(3.98E-1)	3.40653(2.86E-1)	3.39590(3.28E-1)	3.41992(3.68E-1)	3.39231(1.86E-1)	**3.49232(4.11E-1)**	†
UF 8	4.83286(1.93E+0)	4.93252(1.02E+0)	5.40399(1.27E+0)	5.60966(2.10E+0)	5.31305(1.98E+0)	5.03347(2.67E+0)	**5.68381(1.23E+0)**	†
UF 9	3.66801(1.47E+0)	3.83525(3.01E+0)	**4.54679(1.95E+0)**	4.38204(2.01E+0)	3.80608(1.09E+0)	3.89451(1.59E+0)	4.46823(1.02E+0)	
UF 10	3.08289(1.04E+0)	3.53651(2.62E+0)	4.35181(1.11E+0)	4.46131(1.25E+0)	4.11757(1.61E+0)	1.94042(1.28E+0)	**4.53682(1.36E+0)**	†
Ratio	**0/31**	**1/31**	**1/31**	**1/31**	**2/31**	**0/31**	**26/31**	

4.6. Measure of diversity

Table 5 gives the experimental results on the uniformity measure for all MOEAs. As shown in Table 5, we could easily find that the superiority of DNMOEA/HI is not as significant as that on GD metric. Followed by MOEA/D-DE and SPEA2, which win on 7 and 8 out of all benchmark problems respectively, DNMOEA/HI achieves significant better results on 11 out of the 31 benchmark problems. AbYSS and SMS-EMOA achieve best results on 2 and 3 test cases, respectively. It is worth noting that IBEA shows competitive performance on GD metric as discussed in the previous subsection. But as for UA metric, it has been outperformed on all test cases. This could be explained by the simple environmental selection mechanism of IBEA, which might not be sufficient for searching well-distributed solutions. Particularly, DNMOEA/HI is outperformed on all test problems in ZDT and DTLZ test suites. MOEA/D-DE takes the first position on almost all the ZDT test problems, except ZDT-3. As for DTLZ test suite, SPEA2 takes the lead on most of the benchmark problems (5 out of 7). However, the rank situation is changed when referring to WFG test suite. DNMOEA/HI achieves the best UA metrics on almost all WFG test problems except WFG-6. Lastly, MOEA/D-DE and DNMOEA/HI show better performances on UF test suite and the former is slightly better.

4.7. Measure of comprehensive property

The proximity and diversity of a solutions set could be evaluated simultaneously by the single hypervolume indicator, which illustrates the comprehensive performance of the corresponding algorithm. As shown in Table 6, it is encouraging to see that DNMOEA/HI takes the first position on 26 out of 31 benchmark problems, performing significantly better on 21 of them. It is worth noting that our proposed DNMOEA/HI takes the leading position on all ZDT and WFG test suites. The second best algorithm is MEOA/D-DE, which only outperforms on 2 out of all test cases. The other two hypervolume based MOEA, i.e., IBEA and SMS-EMOA only win on 1 test problem respectively. The success of our proposed DNMOEA/HI on HV metric might owe a great deal to the advanced fitness assignment scheme, which is based on dynamic neighborhood, and the hypervolume guided truncation procedure.

Table 7
Average CPU time for seven MOEAs on the benchmark problems.

	NSGA-II	SPEA2	IBEA	SMS-EMOA	MOEA/D-DE	AbYSS	DNMOEA/HI
ZDT Test Suite	2.83	8.95	7.49	106.32	1.80	1.49	9.40
DTLZ Test Suite	3.04	15.66	9.45	784.15	1.81	1.76	63.01
WFG Test Suite	3.44	26.87	10.00	1756.85	2.06	2.80	187.36
UF Test Suite	3.97	3.38	11.84	541.42	2.36	1.64	19.31

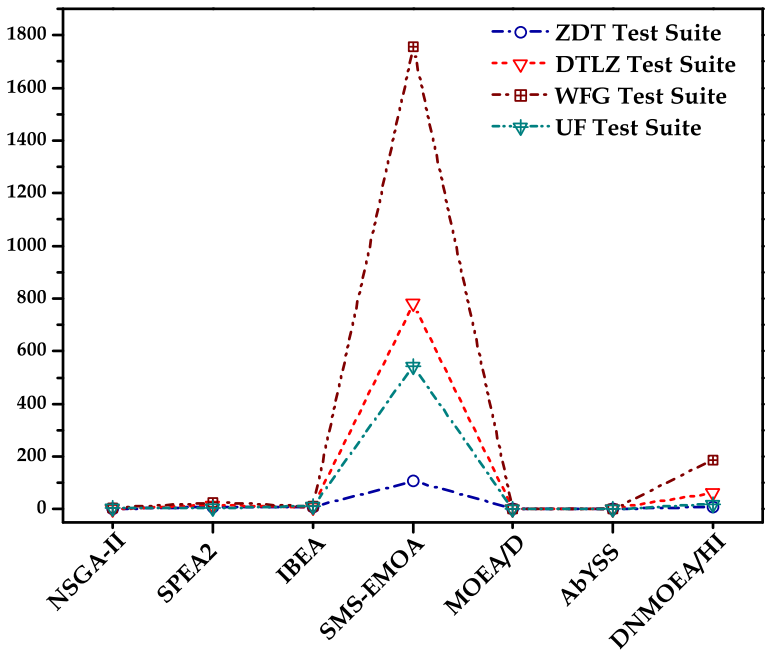

Fig. 7. Average computation time for each MOEA.

Table 8
Performance results on HV metric for seven MOEAs on the many-objective benchmark problems.

	NSGA-II	SPEA2	IBEA	SMS-EMOA	MOEA/D-DE	AbYSS	DNMOEA/HI	S
DTLZ-2_4D	13.4917(2.43E-2)	13.5321(6.92E-1)	13.8824(2.59E-2)	14.1082(3.11E-2)	13.8148(2.06E-2)	13.7098(3.17E-2)	**14.1768(2.17E-2)**	†
DTLZ-2_5D	20.6832(5.67E-2)	20.8186(6.10E-2)	28.5726(5.66E-2)	**29.0517(4.28E-2)**	27.9065(3.56E-2)	25.7418(4.25E-2)	28.9142(5.04E-2)	
DTLZ-2_6D	14.5036(4.13E+0)	12.8659(3.98E+0)	56.1001(1.52E+0)	60.4728(2.51E+0)	51.5831(3.88E+0)	44.0291(4.75E+0)	**60.9248(2.81E+0)**	†
DTLZ-5_4D	8.43250(6.46E-2)	8.38759(5.71E-1)	8.67409(5.12E-1)	**8.69105(4.20E-1)**	8.6129(3.79E-2)	8.53279(4.33E-2)	8.68592(3.82E-1)	
DTLZ-5_5D	17.4434(1.43E+0)	13.7611(2.52E+0)	19.2404(1.21E+0)	19.8135(1.57E+0)	19.6124(2.36E+0)	16.3084(2.04E+0)	**20.7281(1.82E+0)**	†
DTLZ-5_6D	32.9651(1.05E+0)	8.78663(2.12E+0)	37.4109(3.57E+0)	38.8541(4.04E+0)	38.1051(3.12E+0)	36.5459(3.05E+0)	**39.0746(3.12E+0)**	†

Table 9
Performance results on HV metric for seven MOEAs on the dynamic benchmark problems.

	NSGA-II	SPEA2	IBEA	SMS-EMOA	MOEA/D-DE	AbYSS	DNMOEA/HI
FDA-1	0.69222(3.25E-3)	0.69816(3.06E-3)	0.70256(3.02E-3)	0.71563(2.43E-3)	0.68759(3.10E-3)	**0.71527(2.98E-3)**	0.71033(9.39E-3)
FDA-2	0.78994(8.68E-3)	0.79799(5.97E-3)	0.78154(5.42E-3)	0.79921(6.49E-3)	0.79305(6.98E-3)	0.79827(6.34E-3)	**0.80121(4.98E-3)**
FDA-3	0.66041(6.12E-3)	0.67588(4.36E-3)	0.67051(4.51E-3)	0.65167(3.93E-3)	0.67484(5.35E-3)	0.67329(6.23E-3)	**0.67802(3.11E-3)**

4.8. Computational time comparison

We record the CPU time used by each algorithm on these four different test suites. The average time for solving each test problem are tabulated in Table 7, with the unit of second. Fig. 7 gives an intuitive way to illustrate the CPU time comparisons of these MOEAs. Inferred from Table 7 MOEA/D-DE and AbYSS are two MOEAs that cost least computational time. This could be explained by the decomposition based property of MOEA/D-DE and the small reference set sizes used in AbYSS. As for the three hypervolume based MOEAs, it is clear to see that SMS-EMOA is the most time consuming MOEA overall. It is not difficult to understand this high computational cost mostly come from the high computational overhead for hypervolume calculation. Hypervolume indicator has to be calculated twice when evaluating the hypervolume contribution of a single solution. This obviously increases the already high computational overhead. Although the CPU time of our proposed DNMOEA/HI ranks second in this comparison, it is much less than that of SMS-EMOA. This owes to the efficient algorithm proposed in Section 3.3 and Section 3.4 which only needs to calculate hypervolume indicator one time when evaluating the hypervolume contribution. It is worth noting that the CPU time of the other hypervolume based MOEA (i.e., IBEA) is considerably low. It is even less than that of the Pareto based SPEA2 in most of the cases. This might be explained by the simple fitness-based environmental selection procedure used in IBEA. However, the quality of solutions obtained by IBEA is not satisfied enough, especially on the diversity metric that discussed in Section 4.6.

4.9. Studies on the scalability of DNMOEA/HI with different number of objectives

As discussed in [59], most of the current MOEAs are well capable for solving bio- and tri-objective problems. The performances of them would gradually deteriorate with the increase of the number of objectives. However, many real world applications involve more than three objectives to be optimized. For example, many customer satisfactions (such as cost, life time of the battery, affective design, weight, functions, etc.) are required to be optimized simultaneously in new mobile phone design. Thus, the scalability of DNMOEA/HI with different number of objectives is investigated in this section.

The previously proposed approaches for calculating the hypervolume indicator of individual dominance area are designed for bio- and tri-objective problems. Nevertheless, it is not difficult to adapt this approach to tackle problems with more than three objectives, by the employment of a recursive procedure. First of all, four different kinds of individuals are found for the current investigating individual p. After this taxonomy procedure, respect to p, the population is divided into four different circumstances to tackle separately. Similar to [33], the hypervolume indicator of individual dominance area could be computed by recursively projecting the set of points (i.e., the corresponding sets of p) into lower dimensions and calculating the corresponding hypervolume measure of them.

In this experiment, DTLZ-2 and DTLZ-5 of the DTLZ test family are chosen as the benchmark problems. These two test problems are all scalable to arbitrary number of objectives and have been widely used in many-objective optimization studies. Four to six objectives for each test problem are studied here. The performance of each MOEA is evaluated by hypervolume indicator which is also scalable to arbitrary number of objectives.

Through the experimental results shown in Table 8, we could find that DNMOEA/HI is the most competitive algorithm among all these MOEAs. It achieves significantly best results in 4 out of 6 test cases. It is noteworthy that the performance of hypervolume based MOEAs are preferable to the other Pareto based MOEAs in most of these test cases. Except in DTLZ-5 with 5 and 6 objectives, the performances of MOEA/D-DE are better than that of IBEA. The performances of two state-of-the-art Pareto based MOEAs (i.e., NSGA-II and SPEA2) are always worse than the other competitors, especially SPEA2, which performs the worst of all. The other two MOEAs, i.e., MOEA/D-DE and AbYSS, which are based on decomposition and scatter search respectively, also show competitive performances on these high dimensional test problems. Traditionally, the underlying principle of the selection module in MOEA is establishing a complete order among individuals by mapping the objective vector to a ranking criterion. One of the selection operators is based on Pareto dominance and the other operator is diversity preservation. But many solutions would become non-dominated with each other and the search based on Pareto dominance deteriorates, when the number of objectives increases. This phenomenon is referred as the curse of dimensionality for many-objective optimization problems [59]. The selection module of our proposed DNMOEA/HI is not only composed of Pareto dominance and density information, but also the hypervolume contribution of each individual is considered. This enhanced selection module brings plenty of selection pressure even in high dimensional search space.

4.10. Studies on dynamic optimization test cases

The previous experimental results have already demonstrated the effectiveness of our proposed DNMOEA/HI for solving static benchmark problems. However, many real-world optimization problems involve objective functions, constraint

functions and problem parameters that are dynamically changed with time. That is to say the optimization task would be changed with the variation of time and the optimization problem would be different from that at the starting time. Consequently, it is also interesting to study the performance of our proposed approach when tackling the dynamic multi-objective optimization problems. In this subsection, three artificial dynamic multi-objective optimization problems (i.e., FDA-1, FDA-2 and FDA-3) proposed in [60] are selected as the benchmark problems. The dynamics of these selected benchmark problems is controlled by the following time function

$$t = \frac{1}{n_T}\left\lfloor \frac{\tau}{\tau_T} \right\rfloor, \tag{9}$$

where n_T specifies the number of different steps in t while τ_T means the frequencies of environmental changes, and τ is the generation counter. The same as in [60], $\tau_T = 5$ and $n_T = 10$. More detailed description about these benchmark problems could be found in [60].

The population size is fixed to 100 for all benchmark problems. Each MOEA evolves 250 generations and fifty independent simulation runs are performed for each benchmark problem. Different from the static optimization problems, the landscapes and optima vary with time in dynamic optimization problems. Thus, instead of merely obtain the final optimal front, decision makers are more interested in whether the algorithm could track the instant optimal right before the next landscape variation is triggered. Similar to [61], we use the average time hypervolume indicator to evaluate the performance of a MOEA

$$HV_{avg} = \frac{1}{\tau}\sum_{g=1}^{\tau} \sigma * HV, \tag{10}$$

where HV is the hypervolume indicator of the solutions set obtained at the gth generation. If g could be exactly divisible by τ_T, $\sigma = 1$; otherwise $\sigma = 0$. The larger HV_{avg} the better. The reference points are fixed to $(2.0, 2.0)$ for all benchmark problems. Considering the experimental results shown in Table 9, the performance of our proposed DNMOEA/HI clearly outperforms the other competitors, showing better performances on 2 out of 3 benchmark problems.

5. Conclusions

Approaching the Pareto optimal front as close as possible and distribute the solutions over the approximated Pareto front as uniform as possible are two common goals when designing MOEAs. In this paper, we define the membership of individual on the tree neighborhood to describe the adjacency between individuals. The fitness value of each individual is composed of the tree neighborhood density and its Pareto strength value. In addition, a novel algorithm is proposed to evaluate the hypervolume contribution of each individual efficiently. Compared to the approach used in SMS-EMOA, which needs to calculate the hypervolume indicator twice when evaluating the hypervolume contribution of a single solution, our proposed approach could accomplish this task directly and efficiently.

In the numerical experiments, the performance of our proposed DNMOEA/HI is compared with six other MOEAs, including the state-of-the-art Pareto based NSGA-II and SPEA2, hypervolume based IBEA and SMS-EMOA, and two new variants MOEA/D-DE and AbYSS. The quality of the solutions set is evaluated on three different aspects: convergence, diversity and comprehensive performance. Experimental results demonstrate the efficiency of DNMOEA/HI, especially on hypervolume metric. Furthermore, the scalability of DNMOEA/HI with different number of objectives is also studied. The performance of DNMOEA/HI is competitive even in high dimensional search space. At last, we also applied our DNMOEA/HI to solve some dynamic multi-objective optimization problems. Experimental results demonstrate that DNMOEA/HI is also capable to solve problems with dynamically changed landscapes and optima.

Although competitive performance obtained by our proposed DNMOEA/HI, the computational overhead of our algorithm is still high, as discussed in Section 4.8. Thus, a promising direction of future research is to develop more efficient and faster algorithm to evaluate the hypervolume indicator of individual dominance area. On the other hand, as declared in [21], the exact hypervolume contribution might not be crucial for the environmental selection in some cases. Instead, well approximated values are sufficient for this purpose. Furthermore, the computational overhead of approximating algorithms would be much less than that of exact calculation. So it is also a promising direction for future research on designing efficient approximating algorithm to estimate hypervolume with low error rate. In addition, both the density estimation and hypervolume methods introduced in this paper are required for a real parameter space which could be normalized. However, many real-world application problems do not have commensurable space. In other words, different objectives might have different units, take a two-objective optimization problem as an example, suppose the unit of the first objective is cost like dollar and the other one is distance like meter. Thus the normalization is not allowed in this case. Recently, some diversity promotion techniques have been proposed to work in the non-commensurable space, like target vector approaches [62]. Thus it is also another possible future direction to design density estimation method which is workable in this case.

Acknowledgements

The authors thank the anonymous reviewers for their valuable comments which greatly helped them to improve the contents of this paper. The second author acknowledges support from City University Grant No. 9610025 and City University Strategic Grant No. 7002680.

References

[1] J.D. Schaffer, Multiple objective optimization with vector evaluated genetic algorithms, in: Proceedings of the First International Conference on Genetic Algorithms, L. Erlbaum Associates Inc., Hillsdale, NJ, USA, 1985, pp. 93–100.
[2] K. Deb, Multi-objective Optimization Using Evolutionary Algorithms Multi-objective Optimization Using Evolutionary Algorithms, John Wiley & Sons, 2001.
[3] Coello, G.B. Lamont, D.A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems, second ed., Genetic and Evolutionary Computation, Springer, New York, 2007.
[4] D.E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Company, Reading, Massachusetts, 1989.
[5] C. Fonseca, P. Fleming, Multiobjective genetic algorithms made easy: selection sharing and mating restriction, in: First International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications, 1995, pp. 45–52.
[6] J. Horn, N. Nafpliotis, D.E. Goldberg, A niched Pareto genetic algorithm for multiobjective optimization, in: Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, 1994, pp. 82–87.
[7] N. Srinivas, K. Deb, Multiobjective optimization using nondominated sorting in genetic algorithms, Evolutionary Computation 2 (3) (1994) 221–248.
[8] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197.
[9] E. Zitzler, L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach, IEEE Transactions on Evolutionary Computation 3 (4) (1999) 257–271.
[10] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization, in: K. Giannakoglou, et al. (Eds.), Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001), International Center for Numerical Methods in Engineering (CIMNE), 2002, pp. 95–100.
[11] J.D. Knowles, D.W. Corne, Approximating the nondominated front using the pareto archived evolution strategy, Evolutionary Computation 8 (2000) 149–172.
[12] D. Corne, J. Knowles, M. Oates, The Pareto envelope-based selection algorithm for multiobjective optimization, in: M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. Merelo, H.-P. Schwefel (Eds.), Parallel Problem Solving from Nature PPSN VI, vol. 1917, Springer, Berlin, Heidelberg, 2000, pp. 839–848.
[13] D.W. Corne, N.R. Jerram, J.D. Knowles, M.J. Oates, PESA-II: Region-based selection in evolutionary multiobjective optimization, in: Proceedings of the Third Genetic and Evolutionary Computation Conference, GECCO'01, Morgan Kaufman Publishers, 2001, pp. 283–290.
[14] E. Hughes, Multiple single objective Pareto sampling, in: Proceedings of the 2003 Congress on Evolutionary Computation, vol. 4, 2003, pp. 2678–2684.
[15] M. Laumanns, L. Thiele, K. Deb, E. Zitzler, Combining convergence and diversity in evolutionary multiobjective optimization, Evolutionary Computation 10 (3) (2002) 263–282.
[16] B.Y. Qu, P.N. Suganthan, Multi-objective evolutionary algorithms based on the summation of normalized objectives and diversified selection, Information Sciences 180 (2010) 3170–3181.
[17] C. Shi, Z. Yan, K. Lü, Z. Shi, B. Wang, A dominance tree and its application in evolutionary multi-objective optimization, Information Sciences 179 (2009) 3540–3560.
[18] T. Chan, K. Man, K. Tang, S. Kwong, A jumping gene paradigm for evolutionary multiobjective optimization, IEEE Transactions on Evolutionary Computation 12 (2) (2008) 143–159.
[19] K.S.N. Ripon, S. Kwong, K. Man, A real-coding jumping gene genetic algorithm (RJGGA) for multiobjective optimization, Information Sciences 177 (2) (2007) 632–654.
[20] A.G. Hernández-Díaz, L.V. Santana-Quintero, C.A. Coello Coello, J. Molina, R. Caballero, Improving the efficiency of ϵ-dominance based grids, Information Sciences 181 (15) (2011) 3101–3129.
[21] J. Bader, E. Zitzler, HypE: an algorithm for fast hypervolume-based many-objective optimization, Evolutionary Computation 19 (1) (2011) 45–76.
[22] 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 and Evolutionary Computation 1 (1) (2011) 32–49.
[23] Y. Wang, Y. Yang, Particle swarm optimization with preference order ranking for multi-objective optimization, Information Sciences 179 (2009) 1944–1959.
[24] S.Z. Zhao, P.N. Suganthan, Two-lbests based multi-objective particle swarm optimizer, Engineering Optimization 11 (7) (2010) 1–17.
[25] E. Zitzler, L. Thiele, M. Laumanns, C. Fonseca, V. da Fonseca, Performance assessment of multiobjective optimizers: an analysis and review, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 117–132.
[26] M. Fleischer, The measure of Pareto optima applications to multi-objective metaheuristics, in: Proceedings of the Second International Conference on Evolutionary Multi-criterion Optimization, EMO'03, Springer-Verlag, Berlin, Heidelberg, 2003, pp. 519–533.
[27] J.D. Knowles, D.W. Corne, M. Fleischer, Bounded archiving using the Lebesgue measure, in: Proceedings of the 2003 IEEE Congress on Evolutionary Computation, CEC'03, IEEE Press, 2003, pp. 2490–2497.
[28] S. Huband, P. Hingston, L. While, L. Barone, An evolution strategy with probabilistic mutation for multi-objective optimisation, in: Proceedings of the 2003 IEEE Congress on Evolutionary Computation, CEC'03, vol. 4, 2284–2291, 2003.
[29] E. Zitzler, S. Knzli, Indicator-based selection in multiobjective search, in: in Proceedings of the Eighth International Conference on Parallel Problem Solving from Nature (PPSN VIII), Springer, 2004, pp. 832–842.
[30] N. Beume, B. Naujoks, M. Emmerich, SMS-EMOA: multiobjective selection based on dominated hypervolume, European Journal of Operational Research 181 (3) (2007) 1653–1669.
[31] D.E. Goldberg, J. Richardson, Genetic algorithms with sharing for multimodal function optimization, in: J. Grefenstette (Ed.), Proceedings of the Second International Conference on Genetic Algorithms, Lawrence Erlbaum Associates, Mahwah, New Jersey, 1987, pp. 41–49.
[32] A. Farhang-Mehr, S. Azarm, Diversity assessment of Pareto optimal solution sets: an entropy approach, in: Proceedings of the 2002 Congress on Evolutionary Computation, CEC '02, vol. 1, IEEE Press, 2002, pp. 825–830.
[33] J. Knowles, D. Corne, Properties of an adaptive archiving algorithm for storing nondominated vectors, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 100–116.
[34] L. While, P. Hingston, L. Barone, S. Huband, A faster algorithm for calculating hypervolume, IEEE Transactions on Evolutionary Computation 10 (1) (2006) 29–38.
[35] C. Fonseca, L. Paquete, M. Lopez-Ibanez, An improved dimension-sweep algorithm for the hypervolume indicator, in: Proceedings of the 2006 IEEE Congress on Evolutionary Computation, CEC'06, 2006, pp. 1157–1163.
[36] N. Beume, S-metric calculation by considering dominated hypervolume as klee's measure problem, Evolutionary Computation 17 (2009) 477–492. ISSN 1063-6560.
[37] Q. Yang, S. Ding, Novel algorithm to calculate hypervolume indicator of Pareto approximation set, in: D.-S. Huang, L. Heutte, M. Loog (Eds.), A dvanced Intelligent Computing Theories and Applications. With Aspects of Contemporary Intelligent Computing Techniques, Communications in Computer and Information Science, vol. 2, Springer, Berlin, Heidelberg, 2007, pp. 235–244.
[38] 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.
[39] K. Bringmann, T. Friedrich, Do not be greedy when calculating hypervolume contributions, in: Proceedings of the 10th ACM SIGEVO Workshop on Foundations of Genetic Algorithms, FOGA '09, ACM, New York, NY, USA, 2009, pp. 103–112.

[40] K. Bringmann, T. Friedrich, Approximating the least hypervolume contributor: NP-Hard in general, but fast in practice, in: M. Ehrgott, C. Fonseca, X. Gandibleux, J.-K. Hao, M. Sevaux (Eds.), Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, vol. 5467, Springer, Berlin, Heidelberg, 2009, pp. 6–20.
[41] W.L.B.L. Bradstreet, A fast incremental hypervolume algorithm, IEEE Transactions on Evolutionary Computation 12 (6) (2008) 714–723.
[42] T. Voß, T. Friedrich, K. Bringmann, C. Igel, Scaling up indicator-based MOEAs by approximating the least hypervolume contributor: a preliminary study, in: Proceedings of the 12th annual conference companion on Genetic and evolutionary computation, GECCO'10, ACM, New York, NY, USA, 2010, pp. 1975–1978.
[43] 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.
[44] M. Li, J. Zheng, G. Xiao, An efficient mufti-objective evolutionary algorithm based on minimum spanning tree, in: Proceedings of 2008 IEEE Congress on Evolutionary Computation, CEC'08, 2008, pp. 617–624.
[45] M. Li, J. Zheng, J. Wu, Improving NSGA-II algorithm based on minimum spanning tree, in: X. Li, M. Kirley, M. Zhang, D. Green, V. Ciesielski, H. Abbass, Z. Michalewicz, T. Hendtlass, K. Deb, K. Tan, J. Branke, Y. Shi (Eds.), Simulated Evolution and Learning, Lecture Notes in Computer Science, vol. 5361, Springer, Berlin, Heidelberg, 2008, pp. 170–179.
[46] R.C. Prim, Shortest connection networks and some generalizations, Bell System Technology Journal 36 (1957) 1389–1401.
[47] A.J. Nebro, F. Luna, E. Alba, B. Dorronsoro, J.J. Durillo, A. Beham, AbYSS: adapting scatter search to multiobjective optimization, IEEE Transactions on Evolutionary Computation 12 (4) (2008) 439–457.
[48] H. Li, Q. Zhang, Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II, IEEE Transactions on Evolutionary Computation 13 (2) (2009) 284–302.
[49] E. Zitzler, K. Deb, L. Thiele, Comparison of multiobjective evolutionary algorithms: empirical results, Evolutionary Computation 8 (2000) 173–195.
[50] Q. Zhang, A. Zhou, S. Zhao, P. Suganthan, W. Liu, S. Tiwari, Multiobjective optimization test instances for the CEC 2009 special session and competition, University of Essex and Nanyang Technological University, Tech. Rep. CES-487.
[51] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable test problems for evolutionary multiobjective optimization, in: L. Jain, X. Wu, A. Abraham, L. Jain, R. Goldberg (Eds.), Evolutionary Multiobjective Optimization, Advanced Information and Knowledge Processing, Springer, Berlin, Heidelberg, 2005, pp. 105–145.
[52] S. Huband, P. Hingston, L. Barone, L. While, A review of multiobjective test problems and a scalable test problem toolkit, IEEE Transactions on Evolutionary Computation 10 (5) (2006) 477–506.
[53] D.A.V. Veldhuizen, G.B. Lamont, Evolutionary Computation and Convergence to a Pareto Front, Stanford University, California, Morgan Kaufmann, 1998.
[54] M. Li, J. Zheng, G. Xiao, Uniformity assessment for evolutionary multi-objective optimization, in: Proceedings of 2008 IEEE Congress on Evolutionary Computation, CEC'08, 2008, pp. 625–632.
[55] K. Deb, R.B. Agrawal, Simulated binary crossover for continuous search space, Complex System 9 (2) (1985) 115–148.
[56] K. Deb, M. Goyal, A combined genetic adaptive search (GeneAS) for engineering design, Computer Science and Informatics 26 (4) (1996) 30–45.
[57] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm and Evolutionary Computation 1 (1) (2011) 3–18. ISSN: 2210-6502.
[58] J. Durillo, A. Nebro, E. Alba, The jMetal framework for multi-objective optimization: design and architecture, in: Proceedings of the 2010 IEEE Congress on Evolutionary Computation, CEC'10, 2010, pp. 4138–4325.
[59] T. Wagner, N. Beume, B. Naujoks, Pareto-, aggregation-, and indicator-based methods in many-objective optimization, in: S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, T. Murata (Eds.), Evolutionary Multi-Criterion Optimization, Lecture Notes in Computer Science, vol. 4403, Springer, Berlin, Heidelberg, 2007, pp. 742–756.
[60] M. Farina, K. Deb, P. Amato, Dynamic multiobjective optimization problems: test cases, approximations, and applications, IEEE Transactions on Evolutionary Computation 8 (5) (2004) 425–442.
[61] C.-K. Goh, K.C. Tan, A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization, IEEE Transactions on Evolutionary Computation 13 (1) (2009) 103–127.
[62] R.C. Purshouse, C. Jalbă, P.J. Fleming, Preference-driven co-evolutionary algorithms show promise for many-objective optimisation, in: Proceedings of the Sixth International Conference on Evolutionary Multi-criterion Optimization, EMO'11, Springer-Verlag, 2011, pp. 136–150.