MOEA/HD: A Multiobjective Evolutionary Algorithm Based on Hierarchical Decomposition

Hang Xu, *Student Member, IEEE*, Wenhua Zeng, Defu Zhang, and Xiangxiang Zeng, *Senior Member, IEEE*

***Abstract*—Recently, numerous multiobjective evolutionary algorithms (MOEAs) have been proposed to solve the multiobjective optimization problems (MOPs). One of the most widely studied MOEAs is that based on decomposition (MOEA/D), which decomposes an MOP into a series of scalar optimization subproblems, via a set of uniformly distributed weight vectors. MOEA/D shows excellent performance on most mild MOPs, but may face difficulties on ill MOPs, with complex Pareto fronts, which are pointed, long tailed, disconnected, or degenerate. That is because the weight vectors used in decomposition are all preset and invariant. To overcome it, a new MOEA based on hierarchical decomposition (MOEA/HD) is proposed in this paper. In MOEA/HD, subproblems are layered into different hierarchies, and the search directions of lower-hierarchy subproblems are adaptively adjusted, according to the higher-hierarchy search results. In the experiments, MOEA/HD is compared with four state-of-the-art MOEAs, in terms of two widely used performance metrics. According to the empirical results, MOEA/HD shows promising performance on all the test problems.**

***Index Terms*—Decomposition, hierarchy, multiobjective evolutionary algorithm (MOEA), search direction adjustment.**

I. Introduction

MANY real-world problems [1]–[9] are multiobjective optimization problems (MOPs) [10]–[13], which are often solved by multiobjective evolutionary algorithms (MOEAs) [14]–[21]. Although other meta-heuristics [22]–[30] can also handle MOPs, the evolutionary algorithm (EA) [31]–[38] naturally constitutes the Pareto-optimal set (PS) [39] with a population of optimal solutions. For decades, numerous MOEAs have been proposed, and the one based on decomposition (MOEA/D) [40], [41], proposed by Zhang *et al*., may be the most widely studied. In MOEA/D, an MOP is decomposed into a series of scalar optimization subproblems by a set of uniformly distributed weight vectors and a global ideal point. The decomposition method brings excellent population diversity on most mild MOPs, but may fail in tackling complex Pareto fronts (PFs) [42], that are pointed or long tailed, disconnected, and degenerate, because all the weight vectors used in decomposition are fixed.

To overcome this, many follow-up studies [43]–[50] have been recently proposed. For example, MOEA/D-AWA [51] adjusts part of the weight vectors in the later evolution stage, according to the crowdedness of a constantly maintained archive; NSGA-III [52], [53] combines decomposition with nondominated sorting, and adaptively normalizes the objective vectors of solutions; MOEA/D-STM [54] dynamically matches each subproblem with a single appropriate solution, according to the preferences of both subproblem and solution; MOEA/D-TPN [55] adopts a two-phase strategy, where the crowdedness found in the first phase is used to distribute the computational resources in the second phase; and MOEA/D-PaS [56] uses an L_p scalarizing method, where the value of p is adaptively estimated for each subproblem.

In this paper, we propose a new MOEA framework based on hierarchical decomposition (MOEA/HD), which hierarchically constructs and modifies subproblems. In fact, the idea of hierarchy comes from the work distribution in a company, where the board of directors commands the general manager, the general manager commands the department managers, and each department manager commands their staff. It indicates how a complicated task can be simplified and finished through the hierarchical structure. Thus, we apply this idea to the proposed algorithm, which aims at uniformly distributing the population in objective space. Generally, we have made two major contributions as follows.

First, a subproblem layering method is proposed, which layers subproblems into different hierarchies. A superior-subordinate relationship is built between subproblems, during layering. Basically, a lower-hierarchy subproblem is related with two higher-hierarchy subproblems, called superior subproblems. The superior subproblems are used to guide the search directions of the lower-hierarchy subproblems, just like the superior members guide the distributed work of the subordinate members in a company.

Second, a search direction adjustment method is proposed, which adaptively adjusts the search directions of the lower-hierarchy subproblems, according to the higher-hierarchy search results. Basically, the search direction of a

Manuscript received July 16, 2017; revised October 16, 2017 and November 21, 2017; accepted November 28, 2017. Date of publication December 25, 2017; date of current version January 15, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61202011 and Grant 61272152, in part by the Natural Science Foundation of the Higher Education Institutions of Fujian Province under Grant JZ160400, in part by the President Fund of Xiamen University under Grant 20720170054, and in part by the European Project under Grant FP7 CFET 612146. This paper was recommended by Associate Editor Q. Zhao. *(Corresponding author: Xiangxiang Zeng.)*

H. Xu and W. Zeng are with the School of Software, Xiamen University, Xiamen 361005, China (e-mail: hangxu520@hotmail.com; whzeng@xmu.edu.cn).

D. Zhang and X. Zeng are with the School of Information Science and Engineering, Xiamen University, Xiamen 361005, China (e-mail: dfzhang@xmu.edu.cn; xzeng@xmu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2017.2779450

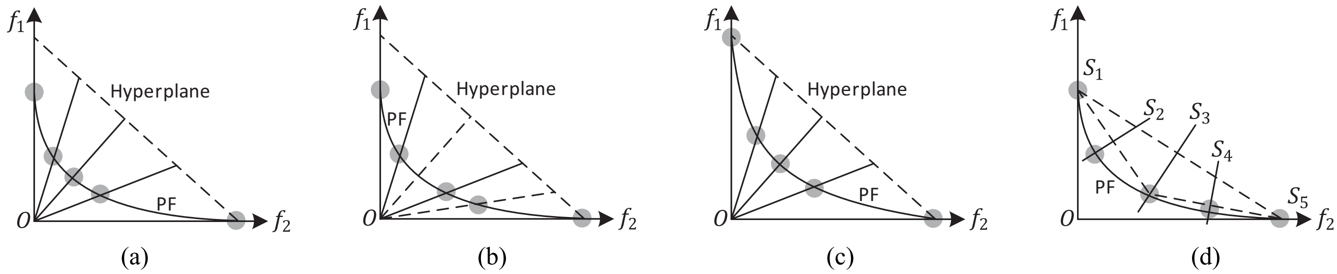


Fig. 1. (a) Weight vector and optimal solution distributions of MOEA/D. (b) Weight vector adjustment mechanism of MOEA/D-AWA. (c) Normalization effect of NSGA-III. (d) Hierarchical decomposition mechanism of the proposed algorithm.

lower-hierarchy subproblem is set to the perpendicular bisector between the current solutions of two superior subproblems. The adjustment of search directions includes the adjustment of both weight vectors and reference points.

The remainder of this paper is organized as follows. First, some preliminaries are introduced in Section II. Second, the proposed algorithm is elaborated in Section III. Third, the experimental study is provided in Section IV. Finally, the conclusion and future work are shown in Section V.

II. Preliminaries

A. Related Works

The original weight vector construction of MOEA/D is based on the method of Das and Dennis [57], which uniformly distributes the weight vectors on a hyperplane in objective space. A simple bi-objective example is shown in Fig. 1(a), where each line denotes the weight vector of a subproblem, the dotted line denotes a hyperplane, and the curve denotes a PF. Thus, in mild condition, each intersection between the lines and curve is supposed to be an optimal solution. However, the distribution of solutions on the PF is not uniform, although all the weight vectors are uniformly distributed on the hyperplane. That is because the shape of the PF, which is pointed and long tailed, is not similar to the shape of the ideal hyperplane. Many researchers are working on this problem, and have proposed some mechanisms as follows.

Qi *et al.* [51] proposed an MOEA based on decomposition and adaptive weight vector adjustment, MOEA/D-AWA. In MOEA/D-AWA, only part of the weight vectors are adaptively adjusted, according to the crowdedness analysis of the archived solutions in objective space. The adjustment mechanism is briefly sketched in Fig. 1(b), where the PF shape is the same as that in Fig. 1(a). At first, the weight vector distribution in Fig. 1(b) is the same as that in Fig. 1(a). Subsequently, the weight vector with the greatest crowdedness (i.e., the one denoted by the dotted line in the middle) is removed. Meanwhile, a new weight vector (i.e., the one denoted by the other dotted line below) is inserted into the sparsest region. This process happens every few generations in the later evolution stage, and is supposed to solve the problem of sharp peak and long tail. However, it requires many generations of time to take significant effect.

Deb and Jain [52] proposed an MOEA based on reference point (i.e., decomposition) and nondominated sorting, NSGA-III. In NSGA-III, nondominated sorting makes solutions converge toward the PF, and decomposition helps maintain the population distribution. In NSGA-III, another mechanism to improve the population diversity is the adaptive normalization method. The effect of this method is briefly sketched in Fig. 1(c), where the weight vector distribution is the same as that in Fig. 1(a). At first, the PF shape in Fig. 1(c) is the same as that in Fig. 1(a). Subsequently, the PF shape is stretched to fit the ideal hyperplane, because of the normalization effect on all the solutions in objective space. Normalization solves the scaled problem, and makes the population distribution more symmetrical. However, the distribution uniformity is not significantly improved, and the same problem of sharp peak and long tail still remains.

B. Major Motivation

Both MOEA/D-AWA and NSGA-III find good ways to improve the population diversity, but they still have some flaws as mentioned above. Therefore, we propose a hierarchical decomposition mechanism, without crowdedness analysis, archive, normalization or uniformly distributed weight vectors. In fact, the idea of hierarchy derives from the work distribution in a company, where the board of directors commands the general manager, the general manager commands the department managers, and each department manager commands their staff. It is indicated that a complex task can be simplified and finished by the hierarchical structure. Thus, we apply this idea to subproblem decomposition, in order to make the population uniformly distributed. A simple example of how hierarchical decomposition works is shown in Fig. 1(d).

In Fig. 1(d), S_1, S_2, S_3, S_4, and S_5 are five subproblems. S_1 and S_5 are extreme subproblems, whose weight vectors are the basis vectors of objective space, i.e., f_1 and f_2. We first find the optimal solutions of S_1 and S_5, and the perpendicular bisector between them becomes the search direction of S_3. We then find the optimal solution of S_3, and the perpendicular bisector between the optimal solutions of S_1 and S_3 becomes the search direction of S_2. Similarly, the search direction of S_4 is the perpendicular bisector between the optimal solutions of S_3 and S_5. In this way, the optimal solutions of all five subproblems are uniformly distributed, according to the perpendicular bisector principle. As a result, the problem of sharp peak and long tail is well solved.

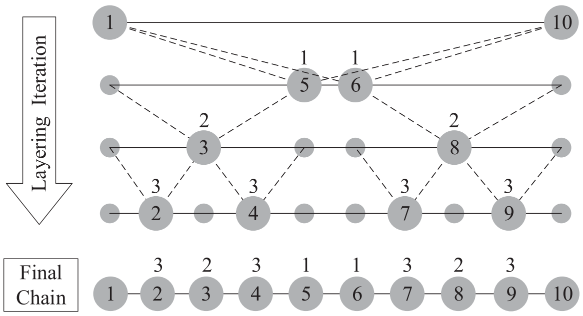


Fig. 2. Layering ten subproblems in a chain.

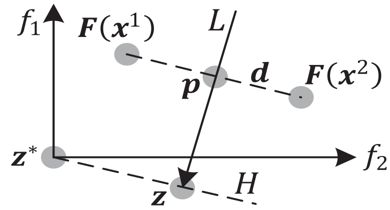


Fig. 3. Search direction adjustment method.

Algorithm 1 Layer_Chain(C)

Input: chain of subproblems, C
Output: chain of layered subproblems, C

while not all subproblems in C are layered **do**
  if it is the first iteration **then**
    Find median subproblems between the first and last subproblems of C.
    Set the latter ones as the superior subproblems of the former ones.
  else
    Find out all the layered subproblems in C.
    for each two sequent layered subproblems **do**
      Find median subproblems between them.
      Set the latter ones as the superior subproblems of the former ones.
    end for
  end if
end while

III. Proposed Algorithm

A. Application to Bi-Objective Space

In bi-objective space, all subproblems are stored in a linear data structure, called chain. We use chain to represent the hierarchical topology structure of subproblems. In a chain, subproblems are layered into different hierarchies, and each lower-hierarchy subproblem is related with two superior subproblems. An example of how to layer a chain with ten subproblems is shown in Fig. 2, where each circle denotes a subproblem, the number inside the circle denotes the index of the subproblem in the chain, the number above the circle denotes the hierarchy of the subproblem (the larger the number, the lower the hierarchy), and each dashed line denotes a superior-subordinate relationship.

For the sake of clearness, we use $S_1, S_2, \ldots, S_{10}$ to denote all the subproblems in Fig. 2, sequentially. In the first iteration, we find both ends of the chain, i.e., S_1 and S_{10}. We do not assign any hierarchy number to them, because they are already in the highest hierarchy of the chain, by default. In the second iteration, we find two median subproblems between S_1 and S_{10}, i.e., S_5 and S_6. S_1 and S_{10} then become the common superior subproblems of both S_5 and S_6. The median indexes can be either singular or dual, according to the specific indexes of two superior subproblems. The hierarchy number of S_5 and S_6 is set to one, which is the second highest hierarchy. In the next iteration, we keep looking for the median subproblems between each pair of sequent and layered subproblems, such as S_3 between S_1 and S_5. The hierarchy number, increased by one in each iteration, is assigned to each median subproblem found. The hierarchy level decreases, as the hierarchy number increases. Layer by layer, after all the subproblems are layered, the superior-subordinate relationship is built. Moreover, the pseudocode of the above layering process is shown in Algorithm 1, where a layered subproblem means it has already been associated with other subproblems in terms of the superior-subordinate relationship.

The search direction adjustment for a target subproblem is based on the current solutions of two superior subproblems. An example of the search direction adjustment method is shown in Fig. 3. In Fig. 3, we first find the current solutions of two superior subproblems in objective space, i.e., $\boldsymbol{F}(\boldsymbol{x}^1)$ and $\boldsymbol{F}(\boldsymbol{x}^2)$, and connect them as a segment, i.e., $\boldsymbol{d}$ in (1). We then find the partition point on $\boldsymbol{d}$, i.e., $\boldsymbol{p}$ in (3), and make a perpendicular line L across $\boldsymbol{p}$. Finally, we make another line H, which is both parallel to $\boldsymbol{d}$ and passing through the global ideal point $\boldsymbol{z}^*$. As a result, the intersection point between L and H is the adjusted reference point $\boldsymbol{z}$, and the direction vector of L is the adjusted weight vector $\boldsymbol{w}$. Moreover, the calculation of $\boldsymbol{w}$ and $\boldsymbol{z}$ is formalized as follows:

$$\boldsymbol{d} = \boldsymbol{F}\left(\boldsymbol{x}^2\right) - \boldsymbol{F}\left(\boldsymbol{x}^1\right) \tag{1}$$

$$ratio = (i-j)/(k-j) \tag{2}$$

$$\boldsymbol{p} = \boldsymbol{F}\left(\boldsymbol{x}^1\right) + \boldsymbol{d} \cdot ratio \tag{3}$$

$$\boldsymbol{z} = \boldsymbol{z}^* + \left((\boldsymbol{p} \cdot \boldsymbol{d} - \boldsymbol{z}^* \cdot \boldsymbol{d})/(\boldsymbol{d} \cdot \boldsymbol{d})\right) \cdot \boldsymbol{d} \tag{4}$$

$$\boldsymbol{w} = \boldsymbol{p} - \boldsymbol{z} \tag{5}$$

where *ratio* is the partition ratio of $\boldsymbol{p}$ on $\boldsymbol{d}$, i is the index of the target subproblem in the chain, and j and k ($j < k$) are the indexes of two superior subproblems in the chain. Here, a dot between two vector variables refers to the dot product. Besides, the extreme subproblems need no adjustment for search directions, because they have no superior subproblem.

B. Extension to Tri-Objective Space

In tri-objective space, a single chain is not enough to approximate the PF. Therefore, we need to connect a series of chains together as a net. Since there are three extreme subproblems, we decide to build a triangle-similar topology structure. As shown in Fig. 4, squares denote the extreme subproblems, and dots denote the other subproblems. In the topology, three

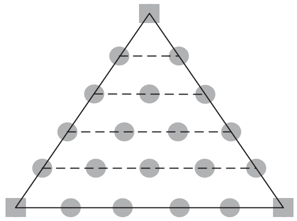


Fig. 4. Example of the triangle topology structure.

Algorithm 2 Modify_Topo(N, *Topo*)

Input: population size, N; topology structure, *Topo*
Output: modified topology structure, *Topo*

1: Get all the adaptive chains in *Topo*, from top to bottom: $AC = (AC_1, \ldots, AC_K)$.
2: **for** $i \leftarrow 1$ **to** K **do**
3: $ED(i) \leftarrow$ calculate the Euclidean distance, in objective space, between the current solutions of the first and last subproblems in AC_i.
4: **end for**
5: **if** $Sum(ED) = 0$ **then**
6: Change *Topo* to a single chain, and stop.
7: **end if**
8: **for** $i \leftarrow 1$ **to** K **do**
9: Adjust the subproblem number of AC_i: $SN_i = 1 + Max(1, Round((ED(i)/Sum(ED)) \cdot (N - 1 - K)))$.
10: **end for**
11: Calculate the total subproblem number difference before and after the adjustment: $\Delta = N - 1 - Sum(SN)$.
12: Remedy the quantity difference by adding or deleting $|\Delta|$ subproblems in AC.
13: Replace all the adaptive chains in *Topo* with AC.

outside chains, denoted by solid lines, are called boundary chains, since they are used to approximate the PF boundaries; all the inside chains, denoted by dashed lines, are called interior chains, since they are used to approximate the interior of PF. The hierarchy of the boundary chains is higher than that of the interior chains. Nevertheless, the layering method of each chain still remains the same. It is defined that the lengths of three boundary chains are all equal, denoted as L, and the length of each interior chain changes from two to $(L - 1)$, from top to bottom.

The triangle topology fits most tri-objective MOPs, but some PFs have irregular, or degenerate shapes. Thus, to improve the generality, we propose a method to adaptively modify the triangle topology. In the modified topology, the total number of chains or subproblems still remains the same, but the lengths of certain chains are adaptively changed. These adaptive chains include all the interior chains, and a boundary chain at the bottom. Basically, the modification principle is to positively correlate the topological length (i.e., number of subproblems in a chain) with the realistic length (i.e., Euclidean distance between both ends of the chain in objective space). Moreover, the detailed modification process is shown in Algorithm 2.

In Algorithm 2, a topology structure (*Topo*), made up of chains, is input. At first, we get all the adaptive chains (line 1),

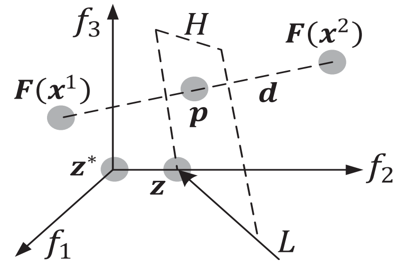


Fig. 5. Modified search direction adjustment method.

and calculate their realistic lengths (lines 2–4). If each length is zero, leaving no space to insert any subproblem in each adaptive chain, then we consider this to be degenerate, and change *Topo* to a single chain structure. In this case, all the subproblems are stored in the chain, where two extreme subproblems at the bottom of *Topo* form one end, and the remaining extreme subproblem forms the other end. Otherwise, the algorithm proceeds, and the subproblem number of each adaptive chain is adjusted by the realistic lengths (lines 8–10). Basically, the larger the realistic length, the larger the subproblem number. Subsequently, the total subproblem number difference is calculated (line 11), and is remedied by fine-tuning the subproblem numbers of some adaptive chains (line 12). Finally, all the original adaptive chains in *Topo* are modified by the adjusted adaptive chains (line 13).

The search direction adjustment method, introduced in Fig. 3, is also applicable in tri-objective space. However, we want to pull the search directions of the subproblems in each boundary chain onto a corresponding base plane, in order to search for the outmost PF boundaries. As shown in Fig. 5, there are three base planes: $f_1z^*f_2$, $f_1z^*f_3$, and $f_2z^*f_3$. Assume the boundary chain at the bottom corresponds to $f_1z^*f_2$, then the other two boundary chains on both sides correspond to $f_1z^*f_3$ and $f_2z^*f_3$. Fig. 5 shows how the modified search direction adjustment method works on a target subproblem in the boundary chain at the bottom.

In Fig. 5, $\boldsymbol{p}$, $\boldsymbol{d}$, $\boldsymbol{F}(\boldsymbol{x}^1)$, $\boldsymbol{F}(\boldsymbol{x}^2)$, and z^* are found in the same way as in Fig. 3; whereas H is the perpendicular plane of $\boldsymbol{d}$ and L is the intersection line between H and the base plane $f_1z^*f_2$. Thus, the direction vector of L becomes the adjusted weight vector $\boldsymbol{w}$. As for the adjusted reference point z, it can be the intersection point between L and the base line z^*f_2. Since $z(1) = z^*(1)$ and $z(3) = z^*(3)$, the value of $z(2)$ is calculated by the following equation:

$$(z - \boldsymbol{p}) \cdot \boldsymbol{d} = 0. \tag{6}$$

In fact, z can also be the intersection point between L and the base line z^*f_1, which does not affect the search effect. In this case, $z(2) = z^*(2)$ and $z(3) = z^*(3)$, and the value of $z(1)$ is also calculated by (6).

C. General Framework of MOEA/HD

The general framework of the proposed algorithm is shown in Algorithm 3, similar to the framework of MOEA/D-DE [58]. In the initialization procedure (lines 1–4), some

Algorithm 3 MOEA/HD(M, N)

Input: objective number, M; population size, N
Output: population, *Pop*

1: Initialize a population with N solutions, randomly sampled from decision space: $Pop = (\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N)$.
2: Initialize the global ideal point: $\boldsymbol{z}^* = (z_1^*, \ldots, z_M^*)$, where z_i^* is the best value found so far, for the ith objective.
3: Construct either a chain, introduced in Section III-A, or a triangle topology, introduced in Section III-B: *Topo*.
4: Layer each chain in *Topo*, by Algorithm 1.
5: **while** termination criterion is not fulfilled **do**
6: Get N pairs of solutions: *PS*, where each pair is randomly selected from either *Pop* or the neighborhood, with a certain probability.
7: Reproduce a set of N offspring solutions: *OS*, by the crossover and mutation of *PS*.
8: Update $\boldsymbol{z}^*$ with each solution in *OS*.
9: Get a union population: $UP = Pop \cup OS$.
10: Delete the last few nondominated fronts of *UP*, but make sure $Size(UP)$ is still larger than N.
11: Update *Pop* by Algorithm 4.
12: **if** *Topo* contains more than one chain **then**
13: Modify *Topo* by Algorithm 2, for each twenty generations, and then layer each chain in *Topo*.
14: **end if**
15: **end while**

essential components are successively initialized, and the hierarchical relationship is built after layering. In the mating procedure (line 6), either global (*Pop*) or local (neighborhood) mating is chosen to select the parents (*PS*). The neighborhood of each solution consists of a preset number of nearest solutions. In the reproduction procedure (line 7), the offspring (*OS*) are made by *PS*. Then, *OS* is used to update the global ideal point (line 8), and to make a union population (line 9). In the updating procedure (lines 10 and 11), we first delete some solutions with poor convergence, by the nondominated sorting method [31], and then update *Pop* by Algorithm 4. Finally, for every 20 generations, *Topo* is modified by Algorithm 2, and the hierarchical relationship is rebuilt (lines 12–14).

In Algorithm 4, at first, we update the extreme subproblems (line 1); subsequently, if *Topo* contains only one chain, we update all the other subproblems in the chain (lines 2 and 3); otherwise, we first update the other subproblems in the boundary chains (lines 4 and 5), and then update the remaining subproblems in the interior chains (lines 6–11). Finally, *Pop* is updated with the current solutions of all the subproblems in *Topo* (line 12). Generally, updating a subproblem means updating both search direction and solution. The search direction of each extreme subproblem is fixed and preset, i.e., $\boldsymbol{w}^i = (w_1^i, \ldots, w_M^i)$, where $w_i^i = 1$ and $w_j^i = 0, j \neq i$. The search directions of all the other subproblems are dynamically constructed and adjusted, according to the adjustment methods introduced in Sections III-A and III-B. The search direction adjustment method shown in Fig. 3 is used in lines 3 and 9 of Algorithm 4, while that shown in Fig. 5 is used in line 5. As for the updating of solutions, we select the solution (denoted by $\boldsymbol{x}$) in *UP*, with the best fitness value for a target subproblem, formally calculated as follows:

Algorithm 4 Update_Pop($Pop, Topo, \boldsymbol{z}^*, UP$)

Input: population, *Pop*; topology structure, *Topo*; global ideal point, $\boldsymbol{z}^*$; union population, *UP*
Output: updated population, *Pop*

1: For each extreme subproblem in *Topo*, select the updated solution from *UP*.
2: **if** *Topo* is a single chain **then**
3: Adjust the search directions of all the other subproblems in *Topo*, by the proposed search direction adjustment method, introduced in Section III-A, and then select their updated solutions from *UP*.
4: **else**
5: Adjust the search directions of all the other subproblems in the boundary chains of *Topo*, by the modified search direction adjustment method, introduced in Section III-B, and then select their updated solutions from *UP*.
6: **for** every other subproblem in the interior chains **do**
7: Get the objective vectors for the current solutions of two superior subproblems: $\boldsymbol{v}^1 = (v_1^1, \ldots, v_M^1)$ and $\boldsymbol{v}^2 = (v_1^2, \ldots, v_M^2)$.
8: Transform $\boldsymbol{z}^*$: $z^*(3) = Min(v_3^1, v_3^2)$.
9: Adjust the search direction by the method in Section III-A, and then select the updated solution from *UP*.
10: **end for**
11: **end if**
12: Replace all the original solutions in *Pop*, with the current solutions of all the subproblems in *Topo*.

$$d_1 = |(\boldsymbol{F}(\boldsymbol{x}) - z) \cdot \boldsymbol{w}/\|\boldsymbol{w}\|| \tag{7}$$

$$d_2 = \|\boldsymbol{F}(\boldsymbol{x}) - z - \boldsymbol{w} \cdot d_1/\|\boldsymbol{w}\|\| \tag{8}$$

where $\boldsymbol{F}(\boldsymbol{x})$ is the objective vector of $\boldsymbol{x}$, and $\boldsymbol{w}$ and $\boldsymbol{z}$ are, respectively, the weight vector and reference point of the target subproblem. In fact, (7) and (8) are the so-called PBI [40] distances, but only d_2 in (8) is used as the fitness value, in this algorithm. That is because the convergence ability is mainly ensured by the nondominated sorting method used in line 10 of Algorithm 3, while Algorithm 4 is mainly used to maintain the population diversity.

The computational complexity of Algorithm 3 is analyzed as follows. The updating of $\boldsymbol{z}^*$ costs $\mathcal{O}(MN)$, and the layering of chains costs $\mathcal{O}(N \log N)$, in the worst case. The mating procedure costs $\mathcal{O}(TN^2)$, where T is the number of nearest neighbors. The reproduction procedure costs $\mathcal{O}(DN)$, where D is the number of decision variables. The nondominated sorting method costs $\mathcal{O}(MN^2)$, in the worst case. Moreover, Algorithm 4 costs $\mathcal{O}(DN^2)$, and Algorithm 2 costs $\mathcal{O}(N)$, in the worst case. Therefore, the general computational complexity is not greater than $\mathcal{O}(TN^2)$, $\mathcal{O}(MN^2)$, or $\mathcal{O}(DN^2)$, which is acceptable and not time-consuming.

TABLE I
NUMBER OF DECISION VARIABLES

Test Problem	Decision Number
WFG1, WFG2, WFG4	$K = M - 1, L = 10$
DTLZ1, D-DTLZ1	$M + 4$
C-DTLZ2, DTLZ5, DTLZ6	$M + 9$
DTLZ7	$M + 19$
JY1, JY3, ZDT3	30
GLT1, GLT2, GLT3, LGZ1	10

IV. EXPERIMENTAL STUDY

A. Benchmark Problems and Performance Metrics

Sixteen test problems, with various PF geometry features, are applied. For two objectives, there are: WFG1 [59]; ZDT3 [60]; JY1 and JY3, i.e., F1 and F3 in [55]; GLT1, GLT2 and GLT3, i.e., F1, F2, and F3 in [49]; LGZ1, i.e., MOP1 in [43]. For three objectives, there are: WFG2 and WFG4 [61]; DTLZ1 and DTLZ5–DTLZ7 [62]; C-DTLZ2, i.e., a convex version of DTLZ2 in [52]; D-DTLZ1, i.e., a disconnected version of DTLZ1, proposed in this paper. D-DTLZ1 is all the same with DTLZ1, except

$$f_3(\boldsymbol{x}) = (2 - x_1 - \text{sign}(\cos(2\pi x_1)))(1 + g(\boldsymbol{x}))/2 \qquad (9)$$

where $\boldsymbol{x}$ is the decision vector of a solution. The number of decision variables for each test problem is given in Table I, where M, K, and L denote the numbers of objectives, position-related variables and distance-related variables, respectively.

Two widely used performance metrics are adopted: 1) inverted generational distance (IGD) [63] and 2) hypervolume (HV) [64]. Both IGD and HV can evaluate the convergence and diversity performances of an algorithm, on the basis of proper reference points. For IGD, about 2000 reference points are uniformly sampled from the PF of each test problem; whereas, for HV, the reference point is set to the worst objective vector calculated by the reference points of IGD. A smaller IGD value indicates better performance; whereas, HV has just the opposite effect. Moreover, when calculating HV, each objective vector is normalized between zero and one, in each objective direction. Thus, we are actually using the normalized HV (NHV) in this paper.

B. Comparison Algorithms and Parameter Settings

Four state-of-the-art algorithms, MOEA/D-AWA, NSGA-III, MOEA/D-DRA [41], and NSGA-II [31], are chosen, for the sake of comparison with MOEA/HD. The reasons for our choice are explained as follows. First, MOEA/HD is based on the framework of MOEA/D-DE, and MOEA/D-DRA is an advanced version of MOEA/D-DE, with dynamic computational resource allocation. Second, both MOEA/HD and MOEA/D-AWA adaptively adjust the weight vectors; but, the former uses a top-down adjustment mechanism (i.e., adjusting all the weight vectors hierarchically in one generation), whereas the latter uses a bottom-up adjustment mechanism (i.e., adjusting only part of the weight vectors locally in one generation). Third, both MOEA/HD and NSGA-II utilize the nondominated sorting method to delete some poor-convergence solutions, but their diversity maintaining mechanisms are totaly different. Fourth, both MOEA/HD and NSGA-III combines decomposition with nondominated sorting; however, NSGA-III uses fixed weight vectors, and an adaptive normalization method.

For a fair comparison, we have referred to [65, Eq. (4)] and [66, Eq. (2)], when designing the weight vector for the Tchebychev decomposition approach used in MOEA/D-DRA. As a result, the modified weight vector used in MOEA/D-DRA, which is similar to the transformed weight vector used in MOEA/D-AWA, is actually the reciprocal of the conventional weight vector [66].

All the above algorithms are coded in MATLAB [67], and run independently for 30 times. The stopping criterion is based on the number of function evaluations, which is set to 100 000 for bi-objective problems and 200 000 for tri-objective problems. The population size is set to 50 for two objectives, and 105 for three objectives. The probability to choose global mating is set to 0.1, and the neighborhood size is set to one-tenth of the population size. Three operators are used in reproduction: 1) simulated binary crossover (SBX) [68]; 2) polynomial mutation (PM) [69]; and 3) differential evolution (DE) [70]. The distribution indexes of both SBX and PM are set to 20. The probability of SBX is set to 1, while that of PM is set to $1/D$. F and CR of DE are set to 0.5 and 1, respectively. In the experiments, only GLT1–GLT3 and LGZ1 adopt both DE and PM, whereas all the other test problems adopt both SBX and PM.

C. Algorithm Performances on Bi-Objective Problems

The statistical results of each MOEA on bi-objective test problems are given in Table II, where MOEA/HD performs best on all the problems. Moreover, the population distributions on some characteristic problems, with the best IGD values of each MOEA, are shown in Fig. 6. Detailed analyses of each problem are as follows.

WFG1 is a challenging problem, with an irregular, scaled and mixed PF, including both convex and concave geometries. As shown in Fig. 6(a), only MOEA/HD successfully approximates the head of the PF, whereas all the other MOEAs lose much diversity in this area.

JY1, GLT2, and GLT3 all have a typical PF feature of sharp peak and long tail, and GLT2 is also a scaled problem. As shown in Fig. 6(b), the extremely convex PF of JY1 makes the population distributions of MOEA/D-AWA, NSGA-III, and MOEA/D-DRA too sparse on both ends, and too crowded in the middle. Because of the flexible crowding distance mechanism, NSGA-II maintains generally good diversity in such extreme conditions. However, the population distribution of MOEA/HD is much more uniform than that of NSGA-II.

JY3, GLT1, and ZDT3 all test the capability of each MOEA for tackling disconnected PFs, and JY3 also has a feature of sharp peak and long tail. As shown in Fig. 6(c), the circumstance of JY3 is similar to that in Fig. 6(b). Moreover, MOEA/HD maintains a much more uniform population distribution than all the other MOEAs, even in each local PF fragment, such as the second fragment from the right.

TABLE II

MEAN IGD AND NHV VALUES OBTAINED BY EACH MOEA ON BI-OBJECTIVE PROBLEMS, WHERE THE BEST PERFORMANCE ON EACH PROBLEM IS SHOWN IN BOLD AND ANY PERFORMANCE NOT SIGNIFICANTLY DIFFERENT FROM THAT OF MOEA/HD IS SUFFIXED BY †

Algorithm	Metric	WFG1	JY1	GLT2	GLT3	JY3	GLT1	ZDT3	LGZ1
MOEA/HD	IGD	**2.780E-02** ±8.731E-03	**9.450E-03** ±5.868E-06	**5.383E-02** ±6.765E-04	**9.778E-03** ±4.541E-05	**8.238E-03** ±3.600E-06	**3.677E-03** ±2.000E-05	**1.272E-02** ±6.889E-03	**1.864E-01** ±1.107E-01
	NHV	**6.275E-01** ±7.394E-03	**9.483E-01** ±1.320E-06	**7.733E-01** ±5.176E-04	**9.474E-01** ±1.176E-04	**9.160E-01** ±2.388E-06	**3.698E-01** ±1.031E-03	**9.098E-01** ±1.038E-03	**3.715E-01** ±1.806E-01
MOEA/D-AWA	IGD	4.734E-02 ±3.567E-02	3.122E-02 ±4.510E-03	2.828E-01 ±4.476E-02	2.147E-02 ±1.965E-03	1.410E-02 ±2.288E-03	9.447E-02 ±1.781E-01	1.350E-02 ±4.967E-03	2.802E-01 ±7.920E-02
	NHV	6.122E-01 ±2.526E-02	9.482E-01† ±5.327E-04	7.269E-01 ±8.109E-03	9.449E-01 ±2.991E-04	9.156E-01 ±2.749E-04	3.385E-01 ±6.201E-02	9.093E-01 ±1.040E-03	2.358E-01 ±1.428E-01
NSGA-III	IGD	5.444E-02 ±4.570E-02	6.813E-02 ±1.330E-03	1.148E-01 ±1.534E-02	5.293E-02 ±1.230E-02	3.614E-02 ±2.806E-02	6.149E-03 ±5.882E-04	1.875E-02 ±2.038E-02	3.162E-01 ±4.140E-02
	NHV	6.048E-01 ±2.889E-02	9.436E-01 ±2.606E-04	7.693E-01 ±2.487E-03	9.392E-01 ±2.412E-03	9.097E-01 ±7.630E-03	3.586E-01 ±4.761E-03	9.029E-01 ±1.582E-02	1.678E-01 ±6.958E-02
MOEA/D-DRA	IGD	6.232E-02 ±5.361E-02	6.937E-02 ±2.179E-04	3.715E-01 ±3.880E-02	4.109E-02 ±1.275E-02	2.894E-02 ±5.065E-06	1.971E-01 ±2.209E-01	2.133E-02 ±1.027E-04	3.157E-01 ±8.750E-02
	NHV	6.039E-01 ±3.576E-02	9.435E-01 ±2.100E-05	7.227E-01 ±7.551E-03	9.408E-01 ±2.561E-03	9.117E-01 ±6.363E-04	3.027E-01 ±7.719E-02	9.038E-01 ±6.046E-05	1.938E-01 ±1.220E-01
NSGA-II	IGD	6.921E-02 ±4.421E-02	1.167E-02 ±4.256E-04	6.269E-02 ±2.542E-03	1.497E-02 ±5.513E-03	9.027E-03 ±7.135E-04	5.910E-03 ±2.445E-04	1.686E-02 ±1.130E-02	3.169E-01 ±4.391E-02
	NHV	6.077E-01 ±2.314E-02	9.475E-01 ±5.167E-04	7.685E-01 ±1.146E-03	9.449E-01 ±1.088E-03	9.154E-01 ±3.320E-04	3.622E-01 ±2.333E-03	9.089E-01 ±1.986E-03	1.670E-01 ±7.131E-02

TABLE III

MEAN IGD AND NHV VALUES OBTAINED BY EACH MOEA ON TRI-OBJECTIVE PROBLEMS, WHERE THE BEST PERFORMANCE ON EACH PROBLEM IS SHOWN IN BOLD AND ANY PERFORMANCE NOT SIGNIFICANTLY DIFFERENT FROM THAT OF MOEA/HD IS SUFFIXED BY †

Algorithm	Metric	C-DTLZ2	DTLZ5	DTLZ6	D-DTLZ1	WFG2	DTLZ7	DTLZ1	WFG4
MOEA/HD	IGD	**3.103E-02** ±5.887E-05	**3.812E-03** ±9.439E-07	**3.812E-03** ±4.192E-08	**1.465E-02** ±4.203E-05	**1.418E-01** ±4.451E-03	8.486E-02 ±1.462E-03	**1.883E-02** ±5.118E-07	**2.005E-01** ±4.576E-05
	NHV	**9.521E-01** ±2.146E-05	**9.336E-02** ±3.994E-07	**9.336E-02** ±2.630E-08	**7.362E-01** ±6.164E-04	**9.152E-01** ±6.752E-04	1.601E-01 ±2.316E-03	**7.929E-01** ±1.415E-05	4.179E-01 ±5.730E-05
MOEA/D-AWA	IGD	3.589E-02 ±1.889E-03	8.038E-03 ±7.302E-05	8.004E-03 ±6.216E-05	1.737E-02 ±2.892E-04	2.675E-01 ±6.033E-02	9.350E-02 ±4.592E-03	1.888E-02 ±2.434E-05	2.123E-01 ±4.939E-03
	NHV	9.493E-01 ±4.061E-04	9.060E-02 ±1.183E-04	9.067E-02 ±1.159E-04	7.339E-01 ±2.507E-04	9.024E-01 ±7.932E-03	1.591E-01 ±1.811E-03	7.927E-01 ±9.755E-05	4.134E-01 ±3.429E-03
NSGA-III	IGD	4.570E-02 ±2.898E-04	1.202E-02 ±1.771E-03	1.275E-02 ±1.692E-03	1.609E-02 ±4.944E-04	1.891E-01 ±2.790E-03	7.834E-02 ±5.086E-02	1.883E-02† ±3.517E-06	2.018E-01 ±3.565E-05
	NHV	9.464E-01 ±1.104E-04	8.815E-02 ±1.062E-03	8.788E-02 ±1.348E-03	7.190E-01 ±1.209E-03	9.126E-01 ±6.427E-04	**1.668E-01** ±4.476E-03	7.928E-01 ±7.406E-05	**4.183E-01** ±5.093E-05
MOEA/D-DRA	IGD	4.566E-02 ±7.519E-06	1.861E-02 ±1.015E-06	1.861E-02 ±8.686E-07	1.814E-02 ±6.681E-05	2.873E-01 ±6.602E-02	2.204E-01 ±1.185E-01	1.883E-02† ±2.277E-06	2.197E-01 ±1.436E-04
	NHV	9.463E-01 ±3.141E-06	8.593E-02 ±7.568E-07	8.593E-02 ±4.431E-07	7.329E-01 ±1.260E-04	8.944E-01 ±1.366E-02	1.499E-01 ±4.144E-03	7.929E-01† ±4.911E-05	4.083E-01 ±4.389E-05
NSGA-II	IGD	3.618E-02 ±1.517E-03	5.442E-03 ±2.708E-04	5.036E-03 ±2.582E-04	1.620E-02 ±6.564E-04	1.909E-01 ±1.130E-02	**7.168E-02** ±3.828E-03	2.594E-02 ±1.138E-03	2.672E-01 ±8.589E-03
	NHV	9.338E-01 ±4.260E-03	9.262E-02 ±1.698E-04	9.314E-02 ±9.773E-05	6.991E-01 ±6.281E-03	8.952E-01 ±2.720E-03	1.603E-01 ±1.873E-03	7.664E-01 ±5.000E-03	3.619E-01 ±5.615E-03

LGZ1 is difficult for convergence, where an MOEA can easily get trapped in local optima. According to [43], LGZ1 requires an MOEA to maintain good diversity during the entire evolution. However, most MOEAs, including the four comparison algorithms, emphasize more convergence than diversity, and thus may lose diversity for the sake of convergence. By contrast, MOEA/HD makes a good balance between convergence and diversity, connecting and organizing each optimal solution, together as a hierarchically structured chain or triangle topology.

D. Algorithm Performances on Tri-Objective Problems

The statistical results of each MOEA on tri-objective test problems are given in Table III, where MOEA/HD performs

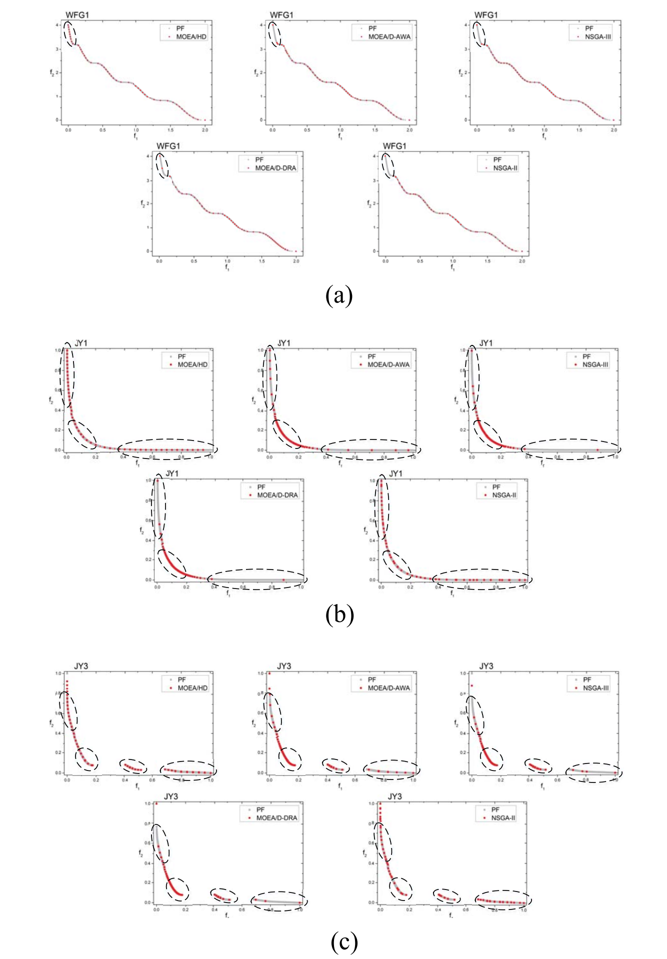


Fig. 6. Population distributions of each MOEA, on three bi-objective test problems, where the significant differences are indicated by dashed circles. Distributions on (a) WFG1, (b) JY1, and (c) JY3.

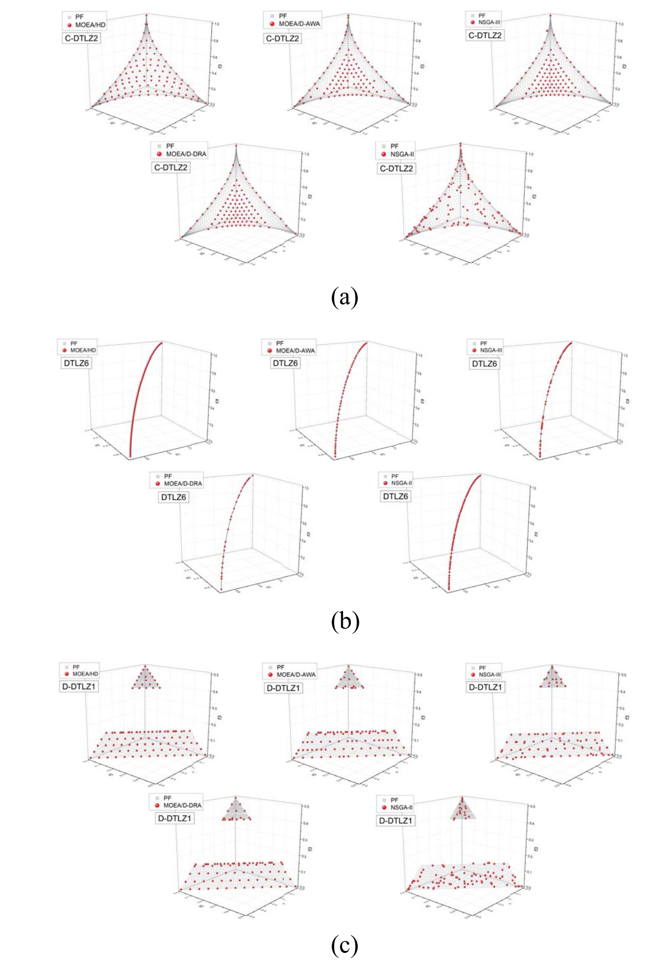


Fig. 7. Population distributions of each MOEA, on three tri-objective test problems. Distributions on (a) C-DTLZ2, (b) DTLZ6, and (c) D-DTLZ1.

best on most of the problems, and shows promising performances on all of the problems. Moreover, the population distributions on some characteristic problems, with the best IGD values of each MOEA, are shown in Fig. 7. Detailed analyses of each problem are as follows.

C-DTLZ2 is a convex version of DTLZ2, with an extremely convex PF. As shown in Fig. 7(a), the circumstance of C-DTLZ2 is similar to that in Fig. 6(b), but is extended to 3-D space. It is obvious that the population distribution of MOEA/HD is much more uniform than that of any other MOEA.

DTLZ5 and DTLZ6 are both degenerate problems, with the same PF shapes, but DTLZ6 is more difficult to converge than DTLZ5. As shown in Fig. 7(b), MOEA/HD has a great population diversity and the best distribution uniformity. By contrast, the performances of both NSGA-III and MOEA/D-DRA are not good. MOEA/D-AWA has a good distribution uniformity, but it still loses some population diversity. On the contrary, NSGA-II has a good population diversity, but its distribution uniformity is not enough.

D-DTLZ1 is a disconnected problem, reconstructed on the basis of DTLZ1. As shown in Fig. 7(c), MOEA/HD shows better performance than all the other MOEAs. NSGA-II finds enough optimal solutions on the PF, but the distribution uniformity is poor. The population diversity and distribution uniformity of the other algorithms still need improvement.

Both WFG2 and DTLZ7 are disconnected, irregular, and scaled problems. MOEA/HD performs best on WFG2, and shows promising performance on DTLZ7. On DTLZ7, MOEA/HD gets third in terms of both IGD and NHV metrics, still better than MOEA/D-AWA and MOEA/D-DRA.

DTLZ1 has a regular triangle PF, and WFG4 has a scaled sphere PF. On DTLZ1, The performances of MOEA/HD, NSGA-III and MOEA/D-DRA are the best and have no significant difference. By contrast, on WFG4, both MOEA/HD and NSGA-III show the best performances in terms of the IGD and NHV metrics, respectively.

V. Conclusion

How to improve the population uniformity for a decomposition-based MOEA has always been a popular research field. In this paper, we proposed an MOEA/HD, which layers subproblems into different hierarchies, and adaptively adjusts the search direction of each lower-hierarchy subproblem, according to the hierarchical structure and the current search results from higher hierarchies. With hierarchical decomposition, subproblems are now connected together as a chain or a triangle topology, during evolution, instead of evolving independently. According to the empirical results, MOEA/HD shows promising performances on both mild and ill test problems with various PF features, in terms of two widely used indicators.

Our future work directions are as follows. First, we want to improve the generality of MOEA/HD by designing a more flexible topology structure. Second, we will study the applicability of MOEA/HD to some suitable real-world problems. Third, we will try to extend MOEA/HD to many-objective space, by improving the layering mechanism.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable suggestions on improving this paper.

References

[1] R. Cheng, T. Rodemann, M. Fischer, M. Olhofer, and Y. Jin, "Evolutionary many-objective optimization of hybrid electric vehicle control: From general optimization to preference articulation," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 1, no. 2, pp. 97–111, Apr. 2017.

[2] R. Roy and J. Mehnen, "Dynamic multi-objective optimisation for machining gradient materials," *CIRP Ann. Manuf. Technol.*, vol. 57, no. 1, pp. 429–432, 2008.

[3] K. Maalawi, *Special Issues on Design Optimization of Wind Turbine Structures*. INTECH Open Access, 2011.

[4] S. Palaniappan, S. Zein-Sabatto, and A. Sekmen, "Dynamic multiobjective optimization of war resource allocation using adaptive genetic algorithms," in *Proc. IEEE SoutheastCon*, Clemson, SC, USA, 2001, pp. 160–165.

[5] A. K. Hutzschenreuter, P. A. N. Bosman, and H. La Poutré, "Evolutionary multiobjective optimization for dynamic hospital resource management," in *Proc. 5th Int. Conf. Evol. Multi Criterion Optim.*, Nantes, France, Apr. 2009, pp. 320–334.

[6] A. Konstantinidis, K. Yang, and Q. Zhang, "An evolutionary algorithm to a multi-objective deployment and power assignment problem in wireless sensor networks," in *Proc. IEEE Glob. Telecommun. Conf.*, New Orleans, LO, USA, Nov. 2008, pp. 1–6.

[7] V. A. Shim, K. C. Tan, and C. Y. Cheong, "A hybrid estimation of distribution algorithm with decomposition for solving the multiobjective multiple traveling salesman problem," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 5, pp. 682–691, Sep. 2012.

[8] A. Konstantinidis, C. Charalambous, A. Zhou, and Q. Zhang, "Multi-objective mobile agent-based sensor network routing using MOEA/D," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, Jul. 2010, pp. 1–8.

[9] Q. Zhang, H. Li, D. Maringer, and E. Tsang, "MOEA/D with NBI-style tchebycheff approach for portfolio management," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, Jul. 2010, pp. 1–8.

[10] A. Zhou *et al.*, "Multiobjective evolutionary algorithms: A survey of the state of the art," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 32–49, 2011.

[11] E. Ziztler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization," in *Evolutionary Methods for Design, Optimization, and Control*. Barcelona, Spain: Int. Center Numer. Methods Eng., 2002, pp. 95–100.

[12] Y. Yuan, H. Xu, B. Wang, and X. Yao, "A new dominance relation-based evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 1, pp. 16–37, Feb. 2016.

[13] N. Srinivas and K. Deb, "Muiltiobjective optimization using nondominated sorting in genetic algorithms," *Evol. Comput.*, vol. 2, no. 3, pp. 221–248, Sep. 1994.

[14] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evol. Comput.*, vol. 19, no. 1, pp. 45–76, Mar. 2011.

[15] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization," *IEEE Trans. Evol. Comput.*, to be published.

[16] Z. He and G. G. Yen, "Many-objective evolutionary algorithms based on coordinated selection strategy," *IEEE Trans. Evol. Comput.*, vol. 21, no. 2, pp. 220–233, Apr. 2017.

[17] W. K. Mashwani, "Hybrid multiobjective evolutionary algorithms: A survey of the stateof-the-art," *Int. J. Comput. Sci. Issue*, vol. 8, no. 3, pp. 374–392, 2011.

[18] H. Ishibuchi, K. Doi, and Y. Nojima, "Reference point specification in MOEA/D for multi-objective and many-objective problems," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, Budapest, Hungary, Oct. 2016, pp. 4015–4020.

[19] W. K. Mashwani, A. Salhi, M. A. Jan, R. A. Khanum, and M. Sulaiman, "Impact analysis of crossovers in a multi-objective evolutionary algorithm," *Sci. Int. J.*, vol. 27, no. 6, pp. 4943–4956, 2015.

[20] H. Wang, L. Jiao, and X. Yao, "Two_Arch2: An improved two-archive algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 524–541, Aug. 2015.

[21] W. K. Mashwani and A. Salhi, "Multiobjective evolutionary algorithm based on multimethod with dynamic resources allocation," *Appl. Soft Comput.*, vol. 39, pp. 292–309, Feb. 2016.

[22] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992.

[23] H. Li and D. Landa-Silva, "An adaptive evolutionary multi-objective approach based on simulated annealing," *Evol. Comput.*, vol. 19, no. 4, pp. 561–595, 2011.

[24] T. Yoshida and H. Mori, "A new multi-objective meta-heuristics for distribution network reconfigurations with SVRs," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, Istanbul, Turkey, Oct. 2010, pp. 1467–1472.

[25] L. Zuo, L. Shu, S. Dong, C. Zhu, and T. Hara, "A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing," *IEEE Access*, vol. 3, pp. 2687–2699, 2015.

[26] K. Chandrasekaran and S. P. Simon, "Multi-objective unit commitment problem with reliability function using fuzzified binary real coded artificial bee colony algorithm," *IET Gener. Transm. Distrib.*, vol. 6, no. 10, pp. 1060–1073, Oct. 2012.

[27] A. Ahuja, S. Das, and A. Pahwa, "An AIS-ACO hybrid approach for multi-objective distribution system reconfiguration," *IEEE Trans. Power Syst.*, vol. 22, no. 3, pp. 1101–1111, Aug. 2007.

[28] Y. Zhang, D.-W. Gong, and J. Cheng, "Multi-objective particle swarm optimization approach for cost-based feature selection in classification," *IEEE/ACM Trans. Comput. Biol. Bioinformat.*, vol. 14, no. 1, pp. 64–75, Jan./Feb. 2017.

[29] Q. Zhang, A. Zhou, and Y. Jin, "RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 41–63, Feb. 2008.

[30] W. K. Mashwani and A. Salhi, "Multiobjective memetic algorithm based on decomposition," *Appl. Soft Comput.*, vol. 21, pp. 221–243, Aug. 2014.

[31] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[32] W. K. Mashwani, "Comprehensive survey of hybrid evolutionary algorithms," *Int. J. Appl. Evol. Comput.*, vol. 4, no. 2, pp. 1–19, 2013.

[33] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.

[34] H. Lu and G. G. Yen, "Rank-density-based multiobjective genetic algorithm and benchmark test function study," *IEEE Trans. Evol. Comput.*, vol. 7, no. 4, pp. 325–343, Aug. 2003.

[35] W. K. Mashwani, A. Salhi, M. Asif, R. Adeeb, and M. Sulaiman, "Enhanced version of multi-algorithm genetically adaptive for multiobjective optimization," *Int. J. Adv. Comput. Sci. Appl.*, vol. 6, no. 12, pp. 279–287, 2015.

[36] C. A. C. Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, vol. 5. New York, NY, USA: Springer, 2007.

[37] W. K. Mashwani, A. Salhi, O. Yeniay, H. Hussian, and M. A. Jan, "Hybrid non-dominated sorting genetic algorithm with adaptive operators selection," *Appl. Soft Comput.*, vol. 56, pp. 1–18, Jul. 2017.

[38] W. K. Mashwani, A. Salhi, O. Yeniay, M. A. Jan, and R. A. Khanum, "Hybrid adaptive evolutionary algorithm based on decomposition," *Appl. Soft Comput.*, vol. 57, pp. 363–378, Aug. 2017.

[39] X. Ma *et al.*, "A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 275–298, Apr. 2016.

[40] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[41] Q. Zhang, W. Liu, and H. Li, "The performance of a new version of MOEA/D on CEC09 unconstrained MOP test instances," in *Proc. IEEE Congr. Evol. Comput.*, Trondheim, Norway, May 2009, pp. 203–208.

[42] H. Wang, Q. Zhang, L. Jiao, and X. Yao, "Regularity model for noisy multiobjective optimization," *IEEE Trans. Cybern.*, vol. 46, no. 9, pp. 1997–2009, Sep. 2016.

[43] H.-L. Liu, F. Gu, and Q. Zhang, "Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 450–455, Jun. 2014.

[44] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An evolutionary many-objective optimization algorithm based on dominance and decomposition," *IEEE Trans. Evol. Comput.*, vol. 19, no. 5, pp. 694–716, Oct. 2015.

[45] Y. Yuan, H. Xu, B. Wang, B. Zhang, and X. Yao, "Balancing convergence and diversity in decomposition-based many-objective optimizers," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 180–198, Apr. 2016.

[46] W. K. Mashwani and A. Salhi, "A decomposition-based hybrid multiobjective evolutionary algorithm with dynamic resource allocation," *Appl. Soft Comput.*, vol. 12, no. 9, pp. 2765–2780, 2012.

[47] W. Khan and Q. Zhang, "MOEA/D-DRA with two crossover operators," in *Proc. U.K. Workshop Comput. Intell.*, Colchester, U.K., Sep. 2010, pp. 1–6.

[48] H. Ishibuchi, Y. Sakane, N. Tsukamoto, and Y. Nojima, "Adaptation of scalarizing functions in MOEA/D: An adaptive scalarizing function-based multiobjective evolutionary algorithm," in *Proc. 5th Int. Conf. Evol. Multi Criterion Optim.*, Nantes, France, Apr. 2009, pp. 438–452.

[49] F. Gu, H.-L. Liu, and K. C. Tan, "A multiobjective evolutionary algorithm using dynamic weight design method," *Int. J. Innov. Comput. Inf. Control*, vol. 8, no. 5, pp. 3677–3688, May 2012.

[50] W. K. Mashwani, A. Salhi, M. Jan, R. Khanum, and M. Suliaman, "Evolutionary algorithms based on decomposition and indicator functions: State-of-the-art survey," *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 2, pp. 583–593, 2016.

[51] Y. Qi *et al.*, "MOEA/D with adaptive weight adjustment," *Evol. Comput.*, vol. 22, no. 2, pp. 231–264, Jun. 2014.

[52] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Aug. 2014.

[53] H. Jain and K. Deb, "An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 602–622, Aug. 2014.

[54] K. Li, Q. Zhang, S. Kwong, M. Li, and R. Wang, "Stable matching-based selection in evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 18, no. 6, pp. 909–923, Dec. 2014.

[55] S. Jiang and S. Yang, "An improved multiobjective optimization evolutionary algorithm based on decomposition for complex Pareto fronts," *IEEE Trans. Cybern.*, vol. 46, no. 2, pp. 421–437, Feb. 2016.

[56] R. Wang, Q. Zhang, and T. Zhang, "Decomposition-based algorithms using Pareto adaptive scalarizing methods," *IEEE Trans. Evol. Comput.*, vol. 20, no. 6, pp. 821–837, Dec. 2016.

[57] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems," *SIAM J. Optim.*, vol. 8, no. 3, pp. 631–657, 1998.

[58] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 284–302, Apr. 2009.

[59] S. Huband, L. Barone, L. While, and P. Hingston, "A scalable multi-objective test problem toolkit," in *Proc. 3rd Int. Conf. Evol. Multi Criterion Optim.*, Guanajuato, Mexico, Mar. 2005, pp. 280–295.

[60] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, Jun. 2000.

[61] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Trans. Evol. Comput.*, vol. 10, no. 5, pp. 477–506, Oct. 2006.

[62] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, *Scalable Test Problems for Evolutionary Multiobjective Optimization*. London, U.K.: Springer, 2005, pp. 105–145.

[63] P. Czyżżak and A. Jaszkiewicz, "Pareto simulated annealing—A metaheuristic technique for multiple-objective combinatorial optimization," *J. Multi Criteria Decis. Anal.*, vol. 7, no. 1, pp. 34–47, 1998.

[64] L. While, P. Hingston, L. Barone, and S. Huband, "A faster algorithm for calculating hypervolume," *IEEE Trans. Evol. Comput.*, vol. 10, no. 1, pp. 29–38, Feb. 2006.

[65] H.-L. Liu, L. Chen, Q. Zhang, and K. Deb, "Adaptively allocating search effort in challenging many-objective optimization problems," *IEEE Trans. Evol. Comput.*, to be published.

[66] F. Gu and Y.-M. Cheung, "Self-organizing map-based weight design for decomposition-based many-objective evolutionary algorithm," *IEEE Trans. Evol. Comput.*, to be published.

[67] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization," *IEEE Comput. Intell. Mag.*, vol. 12, no. 4, pp. 73–87, 2017.

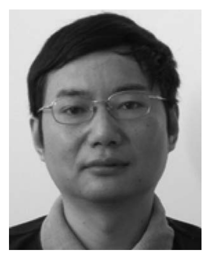
[68] K. Deb and H.-G. Beyer, "Self-adaptive genetic algorithms with simulated binary crossover," *Evol. Comput.*, vol. 9, no. 2, pp. 197–221, Jun. 2001.

[69] K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Comput. Sci. Informat.*, vol. 26, no. 4, pp. 30–45, 1996.

[70] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

Hang Xu (S'15) received the B.S. degree in software engineering from Xiamen University, Xiamen, China, in 2012, where he is currently pursuing the Ph.D. degree in computer science and technology with the School of Software.

His current research interests include computational intelligence and the Internet of Things' technology and application.



Wenhua Zeng received the B.S. degree in automatic control from the Nanjing Institute of Chemical Engineering, Nanjing, China, in 1983, and the M.S. and Ph.D. degrees in automatic control from Zhejiang University, Hangzhou, China, in 1986 and 1989, respectively.

He is currently a Professor with the School of Software, Xiamen University, Xiamen, China. His current research interests include embedded software, intelligent control, evolutionary computation, cloud computing, grid computing, artificial intelligence, and their applications.

Defu Zhang received the B.S. and M.S. degrees in computational mathematics from the Department of Mathematics, Xiangtan University, Xiangtan, China, in 1996 and 1999, respectively, and the Ph.D. degree in computer science and technology from the Huazhong University of Science and Technology, Wuhan, China, in 2002.

He was a Visiting Scholar with the City University of Hong Kong, Hong Kong, from 2008 to 2013. He is currently a Professor and the Associate Head with the Department of Computer Science, Xiamen University, Xiamen, China. His current research interests include computational intelligence, data mining, big data, and combinatorial optimization.

Xiangxiang Zeng (M'11–SM'16) received the B.S. degree in automation from Hunan University, Changsha, China, in 2005, the Ph.D. degree in system engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2011.

From 2010 to 2011, he was with the Natural Computing Group, Seville University, Seville, Spain, for a year. From 2016 to 2017, he was a Visiting Scholar with Oklahmoma State University, Stillwater, OK, USA. He is currently an Associate Professor with the Department of Computer Science, Xiamen University, Xiamen, China. His publications currently report over 1500 citations in Google Scholar with an H-index of 23. His current research interests include computational intelligence, systems biology, and data mining.