A Sorting Based Selection for Evolutionary Multiobjective Optimization

Zhixiang Yang[1], Xinye Cai[1(✉)], and Zhun Fan[2]

[1] College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, Jiangsu, China
xiang052@163.com, xinye@nuaa.edu.cn
[2] Department of Electronic Engineering, Shantou University, Guangdong, China
zfan@stu.edu.cn

Abstract. In this paper, we propose a new solution selection method to balance the convergence and diversity during the evolutionary process for evolutionary multiobjective optimization. The method sorts the solutions based on their ensemble convergence performance, then selects the solutions based on diversity. The selection method is integrated to the framework of decomposition based multiobjective evolutionary algorithms (MOEAs). In order to demonstrate the performance of the algorithm, it is compared with three classical MOEAs and one state-of-art MOEA. The results indicate that our proposed algorithm is very competitive.

Keywords: Multiobjective optimization · Association · Decomposition · Convergence · Diversity

1 Introduction

A multiobjective optimization problem (MOP) can be defined as follows:

$$\begin{aligned} &\text{minimize} \;\; F(x) = (f_1(x), \ldots, f_m(x))^T \\ &\text{subject to} \;\; x \in \Omega \end{aligned} \tag{1}$$

where Ω is the decision space, $F : \Omega \to R^m$ consists of m real-valued objective functions. The attainable objective set is $\{F(x)|x \in \Omega\}$.

Let $u, v \in R^m$, u is said to *dominate* v, denoted by $u \prec v$, if and only if $u_i \leq v_i$ for every $i \in \{1, \ldots, m\}$ and $u_j < v_j$ for at least one index $j \in \{1, \ldots, m\}$[1]. Given a set S in R^m, a solution $x \in S$ can be called non-dominated in S if no other solution in S can dominate it. A solution $x^* \in \Omega$ is *Pareto-optimal* if $F(x^*)$ is non-dominated in the attainable objective set. $F(x^*)$ is then called a *Pareto-optimal (objective) vector*. In other words, any improvement in one objective of a Pareto optimal solution is bound to deteriorate at least another objective.

[1] In the case of maximization, the inequality signs should be reversed.

M. Gong et al. (Eds.): BIC-TA 2015, CCIS 562, pp. 538–549, 2015.
DOI: 10.1007/978-3-662-49014-3_48

The set of all the Pareto-optimal solutions is called the *Pareto set* (*PS*) and the image of (*PS*) on the objective vector space is called the *Pareto front* (*PF*) [11].

Usually, it is desirable to balance between convergence and diversity for obtaining good approximation to the set of Pareto optimal solutions. Over the last two decades, three major evolutionary algorithm paradigms have been developed, i.e., dominance-based MOEAs (e.g., [4,6,18]), indicator-based MOEAs (e.g., [1,2,12,17]) and decomposition-based MOEAs (e.g., [8,14,15]).

Arguably, NSGA-II [6] is the most well-known domination-based MOEA, which uses Pareto dominance relation as the primary selection criterion to promote the convergence and the crowding distance is used as density metric to maintain the diversity.

The indicator based approaches utilize various performance indicators like ϵ-indicator [17], R2 indicator [12] and hypervolume indicator [2] to measure the quality of solutions. Among these indicators, the most commonly used performance indicator is hypervolume, which can measure convergence and diversity simultaneously. However, the hypervolume prefers convex regions to concave ones [1] and its computational complexity is quite huge.

As a representative of the decomposition-based method, the basic idea of MOEA/D [15] is to decompose a MOP into a number of single objective optimization subproblems through aggregation functions and optimizes them simultaneously. The update of solutions is decided by their aggregation function values, and the population diversity is achieved by the wide spread of weight vectors.

Apparently, selection is a major issue in designing MOEAs. In this paper, we propose a sorting-based-selection (SBS) scheme and integrate it into MOEA/D. First, SBS sorts the solutions in the population according to their ensemble convergence performance on all subproblems. Then, the diversity selection is conducted on the sorted population according to each subproblem.

The remainder of this paper is organized as follows. Section 2 provides some background knowledge of this paper. Section 3 explains our motivation of this work. Section 4 details the proposed method, sorting-based-selection. Section 5 mainly describes the proposed MOEA/D-SBS. Experimental settings and performance indicators for MOEAs are detailed in Sect. 6. In Sect. 7, we conduct experiments and present the results of compare our proposed algorithm with three classical MOEAs, NSGA-II, MSOPS-II [7], MOEA/D-DE [9] and one state-of-art MOEA, MOEA/D-STM [10]. Section 8 concludes the paper.

2 Motivation

In this paper, a new solution selection method, SBS, is proposed and used in the framework of MOEA/D to select solutions from the merged population. In SBS, the merged population is sorted based on convergence. The sorting is emphasis on the ensemble performance of each solution on all the subproblems. In order to maintain the diversity, the solutions in the population are associated with the subproblems, first. And then the solutions are selected for each subproblem. Note that the priority relations of the solutions associated with each

subproblem are subject to the ensemble convergence. Consequently, a number of solutions that have a good balance between convergence and diversity are selected in MOEA/D-SBS. Different from MOEA/D, where each subproblem is allowed with one solution, MOEA/D-SBS can be associated with any number of solutions.

3 Sorting-Based-Selection

The new solution selection method, SBS, which selects N solutions from the merged population, is detailed in this section. First, the merged population is sorted based on convergence. Then, the solutions in the sorted population are associated with the subproblems. Finally, N solutions are selected based on diversity depending on the sorted and associated results.

The pseudo-code of SBS is presented in Algorithm 1.

3.1 Convergence Based Sorting

For each subproblem s^j, the aggregation function value, $g^{te}(x|\lambda^j, z^*)$, of each solution x^i in population Z is calculated and stored in $\Delta(i,j)$, as shown in Step 2.1a. Each row of Δ represents a solution and each column represents a subproblem, and λ^j denotes the weight vector with regard to the j-th subproblem. In Step 2.1b, each column of matrix Δ, $\Delta(:, j)$, is sorted in an ascending order and the rank values are kept in $R(:, j)$.

In Step 2.2a, each row of matrix R are sorted in an ascending order of rank values. So, the first column of R will hold the best rank achieved for each solution across the N subproblems, and the N-th column of R will hold the worst rank achieved. Thus the matrix R may be used to rank the population. In Step 2.2b, the solutions in the Z are sorted according to the lexicographical order of the corresponding rows in R.

3.2 Association

In **Step 3**, each solution $x \in Z$ needs to associate with a subproblem. Solution x will be associated with the subproblem whose weight vector has the minimum angle with its objective vector. The acute angle between a solution x and the weight vector λ of a subproblem can be computed as follows:

$$\alpha = \arccos\left(\frac{(F(x) - z^*)^T \lambda}{\|F(x) - z^*\| \|\lambda\|}\right). \tag{2}$$

where $F(x) = (f_1(x), f_2(x), \ldots, f_m(x))^T$ is the objective vector of the solution x, and z^* is the ideal objective vector.

In **Step 3**, each solution in the sorted solution set Z is associated with a subproblem. Furthermore, solutions associated with each subproblem are ordered by ensemble convergence.

Algorithm 1. SBS(Z, z^*, N)

Input:

1. Z: the solution set;
2. z^*: the ideal objective vector z^*;
3. N: the number of subproblems or the maximum size of P.

Output: the population P, index set I

Step 1 Initialization:

a) Set $I = \emptyset$.
b) Set $P = \emptyset$.
c) Set $M = |Z|$.

Step 2 Sorting:
Step 2.1:
For each $j = 1, \ldots, N$, do:

a) For each solution x^i in Z, evaluate the aggregation function $g^{te}(x^i|\boldsymbol{\lambda}^j, z^*)$, and store it in $\Delta(i, j)$.
b) Sort $\Delta(:, j)$ in an ascending order and keep the rank values in $R(:, j)$.
Step 2.2:

a) For each $i = 1, \ldots, M$, sort $R(i, :)$ in an ascending order.
b) Sort Z according to the lexicographical order of the rows in R.

Step 3 Association:

For each solution x in the sorted solution set Z, associate it with the subproblem whose weight vector has the minimum angle with the objective vector of x, based on (2).

Step 4 Selection:
For each $k = 1, \ldots, M$, do:

a) Set $A = \emptyset$;
b) For each subproblem, if the k-th solution associated with it is exist, then add it to A.
c) If $|P| + |A| \leq N$, then set $P = P \bigcup A$; else, the $N - |P|$ solutions in A are selected and added to P based on the rankings in Z, then break.

Step 5 Termination: Record the indexes of the subproblems in I, which are associated with solutions in P. Then return P and I.

3.3 Diversity Based Selection

In order to keep a good population diversity, the solutions will be selected according the subproblems. In **Step 4**, the best associated solution of each subproblem is selected and added to P, if they are exist. Then, the second best associated solution is selected, and so on. When $|P| + |A|$ is greater than N, the $N - |P|$

solutions in A are selected and added to P, depending on their performance. Here, the selection is based on their rankings in Z, in which solutions has been sorted according to their ensemble convergence performance in **Step 2**.

3.4 Termination

Algorithm 2 is terminated when the size of P reaches N. In **Step 5**, record the indexes of the subproblems in I, which are associated with solutions in P. Then return it and P as outputs.

3.5 Computational Cost of the SBS

In Algorithm 1, the calculation of matrix Δ (Step 2.1a) costs $O(mMN)$ computations, where m is the number of objectives. $O(NMlogM)$ comparisons are used to sort Δ (Step 2.1b). The sorting of matrix R and population Z needs $O(MNlogN)$ and $O(NMlogM)$ comparisons (Step 2.2), respectively. And association (Step 3) requires $O(mMN)$ computations. N solutions are selected from the population Z, so the complexity of selection (Step 4) is $O(N)$. In summary, the computational cost of the SBS is $O(NMlogM)$.

4 Algorithm

In this section, we present the whole algorithm, multiobjective evolutionary algorithm based on decomposition with diversity-based-sorting (MOEA/D-SBS). The pseudo-code of MOEA/D-SBS is demonstrated in Algorithm 2.

At each generation, MOEA/D-SBS maintains:

- a set of N subproblems, $S = \{s^1, \ldots, s^N\}$;
- a population of N solutions, $P = \{x^1, \ldots, x^N\}$;
- objective function values, $FV^1, \ldots, FV^N$, where FV^i is the F-value of x^i;

Let $\lambda^1, \ldots, \lambda^N$ be a set of even spread weight vectors [15] and z^* be the reference point. The MOP of (1) can be decomposed into N scalar optimization subproblems by using Tchebycheff approach. The k-th subproblem is:

$$\begin{aligned} &\text{minimize} \;\; g(x|\lambda^k, z^*) = \max_{1 \le i \le m} \{|f_i(x) - z_i^*|/\lambda_i^k\} \\ &\text{subject to} \;\; x \in \Omega. \end{aligned} \tag{3}$$

In initialization, for each $k = 1, \ldots, N$, let $B(k)$ be the set containing the indices of the T closest weight vectors to λ^k in terms of the Euclidean distance. If $i \in B(k)$, i-th subproblem is called a neighbor of k-th subproblem. I is a set of subproblem indices that each solution x^i belongs to. It is initialized by calling SBS function in Algorithm 1.

New solutions are generated in **Step 2**. In MOEA/D, each new solution is generated according to each subproblem, while in MOEA/D-SBS is according

Algorithm 2. MOEA/D-SBS

Input:

1. MOP(1);
2. a stopping criterion;
3. N: the number of subproblems;
4. $\boldsymbol{\lambda}^1, \ldots, \boldsymbol{\lambda}^N$: a set of N weight vectors;
5. T: the size of the neighborhood for each subproblem.

Output: population P
Step 1 Initialization:

a) Compute the Euclidean distances between any two weight vectors and obtain T closest weight vectors to each weight vector. For each $i = 1, \ldots, N$, set $B(i) = \{i_1, \ldots, i_T\}$ where $\lambda^{i_1}, \ldots, \lambda^{i_T}$ are the T closest weight vectors to λ^i.
b) Generate an initial population $P = \{x^1, \ldots, x^N\}$ randomly.
c) Initialize the ideal objective vector z^* by setting $z_i^* = \min\{f_i(x^1), \ldots, f_i(x^N)\}, i = 1, \ldots, m$.
d) Get the set P and I: $[P, I] = \text{SBS}(P, z^*, N)$.

Step 2 New Solution Generation:
For each $i = 1, \ldots, |P|$, do:

a) Selection of the Mating Solutions:
   1) Track the index k of the subproblem that x^i is associated with : $k = I(i)$.
   2) If $rand(0, 1) < \delta$, then set D is the set of solutions that are associated with the subproblem in $B(k)$, else, set $D = P$.

b) Reproduction: Set $r_1 = i$ and randomly select two indices r_2 and r_3 from D, and then generate a solution $\overline{y}$ from x^{r_1}, x^{r_2} and x^{r_3} by DE, and then perform a mutation operator on $\overline{y}$ with probability p_m to produce a new solution y^i.
c) Evaluation y^i **:** $FV^i = F(y^i)$.
d) Update of z^* **:** For each $j = 1, \ldots, m$, if $z_j^* > f_j(y^i)$, then set $z_j^* = f_j(y^i)$.

Step 3 Sorting-based-selection: $[P, I] = \text{SBS}(P \bigcup Y, z^*, N)$
Step 4 Stopping Criteria: If stopping criteria is satisfied, then stop and output P. Otherwise, go to **Step 2**.

to each individual in population. So the number of times that a subproblem is selected to generate offspring is equal to the number of solutions associated with it. In Step 2a, for each individual in population P, the subproblem k is tracked according to the set of subproblem indices I, then all individuals which are associated with the neighborhood of subproblem k in population P are selected as the possible mating range D. In Step 2b, two parent solutions are selected from the D, and then the differential evolution (DE) operator [13] and polynomial mutation [5] are applied to three parent solutions, x^{r_1}, x^{r_2} and x^{r_3}, to generate an offspring y^i.

In DE operator, each element $\overline{y}_j$ in $\overline{y} = (\overline{y}_1, \ldots, \overline{y}_n)^T$ is generated as follows:

$$\overline{y}_j = \begin{cases} x_j^{r_1} + F \times (x_j^{r_2} - x_j^{r_3}), & \text{if } rand \leq CR \text{ or } j = j_{rand} \\ x_j^{r_1}, & \text{otherwise} \end{cases} \tag{4}$$

where $j = 1, \ldots, n$, $rand \in [0, 1]$, $j_{rand} \in [1, n]$ is a random integer, and CR and F are two control parameters.

The mutation operator generates $y = (y_1, \ldots, y_n)^T$ from $\overline{y}$ in the following way:

$$y_j = \begin{cases} \overline{y}_j + \sigma_j \times (b_j - a_j), & \text{with probability } p_m \\ \overline{y}_j, & \text{with probability } 1 - p_m \end{cases} \tag{5}$$

with

$$\sigma_j = \begin{cases} (2 \times rand)^{\frac{1}{\eta+1}} - 1, & \text{if } rand < 0.5 \\ 1 - (2 - 2 \times rand)^{\frac{1}{\eta+1}}, & \text{otherwise} \end{cases} \tag{6}$$

where $rand$ is a uniformly random number from $[0, 1]$; the distribution index η and the mutation rate p_m are two control parameters; and a_j and b_j are the lower and upper bounds of the j-th decision variable, respectively.

The procedure (Step 2a–c) will be repeated N times, so an population $Y = \{y^1, \ldots, y^N\}$ will be got. In **Step 3**, the SBS, detailed in Sect. 4, is adopted to select offsprings from the merged population.

5 Experimental Studies

5.1 Experimental Setting

The UF test suite, which contains ten unconstrained MOP test instances (UF1 to UF10) from the CEC2009 MOEA competition [16], is considered in our experimental studies. For all UF test functions, the number of decision variables is set to 30.

All the algorithms were implemented in Matlab. The parameters of NSGA-II, MSOPS-II, MOEA/D-DE and MOEA/D-STM were set according to the corresponding references [6,7,9,10]. The parameter settings of our proposed MOEA/D-SBS are as follows:

1) Control parameters in DE and polynomial mutation: $CR = 1.0$ and $F = 0.5$ in DE operator; $\eta = 20$ and $p_m = 1/n$ in the polynomial mutation operator.
2) Probability used to select in the neighborhood: $\delta = 0.9$.
3) Population size: $N = 300$ for bi-objective test instances, 595 for the three-objective ones.
4) Neighborhood size: $T = 10$ for bi-objective test instances except for UF3, for which T is set to 20, $T = 595$ for three-objective ones.
5) Number of runs and stopping condition: Each algorithm is run 30 times independently on each test instance. The algorithm stops after 300,000 function evaluations.

It is worth noting that the population size and the number of function evaluations are set same for all compared algorithms.

In our experimental studies, we employ two widely used performance indicators inverted generational distance metric (IGD) [3] and hypervolume metric (I_H) [19]. Both of them can simultaneously measure the convergence and diversity of obtained solutions.

Table 1. Mean and standard deviation values of IGD, obtained by MOEA/D-SBS, MOEA/D-DE, MSOPS-II and NSGA-II on UF instances.

Instance	IGD			
	MOEA/D-SBS	MOEA/D-DE	MSOPS-II	NSGA-II
UF1	**0.0017** (5.67E-05)	0.0024 (4.94E-04)†	0.0743 (5.44E-03)†	0.0839 (1.17E-02)†
UF2	**0.0057** (1.69E-03)	0.0112 (3.21E-03)†	0.0572 (2.10E-02)†	0.0327 (2.32E-03)†
UF3	**0.0037** (1.72E-03)	0.0254 (2.12E-02)†	0.3141 (1.75E-02)†	0.0703 (1.14E-02)†
UF4	**0.0560** (2.67E-03)	0.0677 (2.80E-03)†	0.0567 (4.09E-03)	0.0761 (1.35E-02)†
UF5	**0.2469** (2.46E-02)	0.2901 (4.56E-02)†	0.3437 (9.80E-02)†	0.6793 (9.88E-02)†
UF6	**0.0927** (4.21E-02)	0.1868 (1.34E-01)†	0.2985 (2.29E-01)†	0.3217 (7.60E-02)†
UF7	**0.0023** (2.64E-04)	0.0041 (9.31E-04)†	0.0418 (7.21E-03)†	0.3504 (8.65E-03)†
UF8	**0.0522** (7.66E-03)	0.0658 (7.89E-03)†	0.1916 (4.92E-03)†	0.2693 (5.59E-02)†
UF9	**0.0278** (2.98E-03)	0.0720 (3.96E-02)†	0.2540 (2.29E-02)†	0.2054 (6.92E-02)†
UF10	1.3755 (2.51E-01)	0.4846 (5.52E-02)‡	**0.2147** (4.98E-02)‡	0.6429 (8.84E-02)‡

Wilcoxon's rank sum test at a 0.05 significance level is performed between MOEA/D-SBS and each of the other competing algorithms.

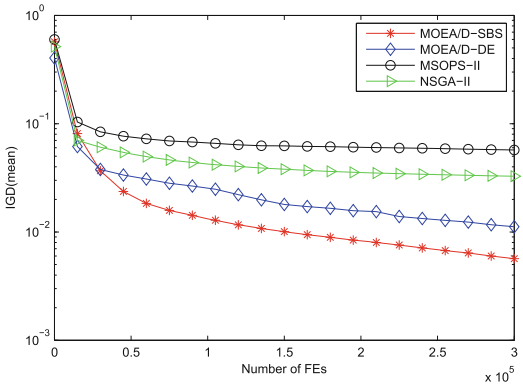
† and ‡ denotes that the performance of the corresponding algorithm is significantly worse than or better than that of MOEA/D-SBS, respectively. The best mean is highlighted in boldface

Table 2. Mean and standard deviation values of I_H, obtained by MOEA/D-SBS, MOEA/D-DE, MSOPS-II and NSGA-II on UF instances.

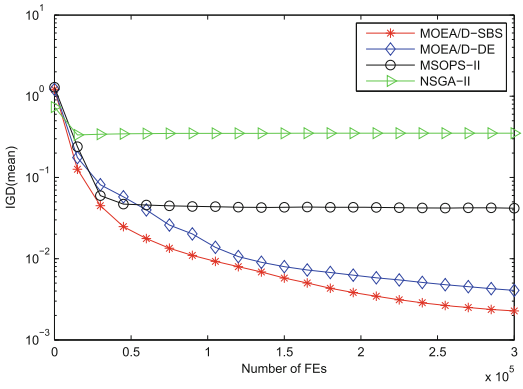
Instance	I_H			
	MOEA/D-SBS	MOEA/D-DE	MSOPS-II	NSGA-II
UF1	**3.6582** (0.0020)	3.6504 (0.0076)†	3.4418 (0.0918)†	3.3859 (0.0175)†
UF2	**3.6422** (0.0142)	3.6073 (0.0399)†	3.4618 (0.0884)†	3.6090 (0.0076)†
UF3	**3.6598** (0.0027)	3.5493 (0.1092)†	2.5711 (0.0457)†	3.5191 (0.0402)†
UF4	3.1672 (0.0117)	3.1352 (0.0165)†	**3.1876** (0.0064)‡	3.0787 (0.1345)
UF5	**2.8680** (0.0865)	2.5807 (0.1622)†	2.5875 (0.3309)†	1.5901 (0.2338)†
UF6	**3.1558** (0.0932)	2.9058 (0.2428)†	2.6652 (0.4505)†	2.6228 (0.1497)†
UF7	**3.4885** (0.0062)	3.4796 (0.0080)†	3.4029 (0.0627)†	2.5452 (0.0091)†
UF8	**7.3028** (0.0267)	6.9542 (0.2821)†	6.4215 (0.0098)†	6.6184 (0.3498)†
UF9	**7.6812** (0.0419)	6.4168 (0.7147)†	5.7602 (0.2120)†	7.0840 (0.2283)†
UF10	9.615 (4.7472)	16.224 (1.2106)‡	**24.269** (1.0457)‡	14.770 (1.4454)‡

Wilcoxon's rank sum test at a 0.05 significance level is performed between MOEA/D-SBS and each of the other competing algorithms.

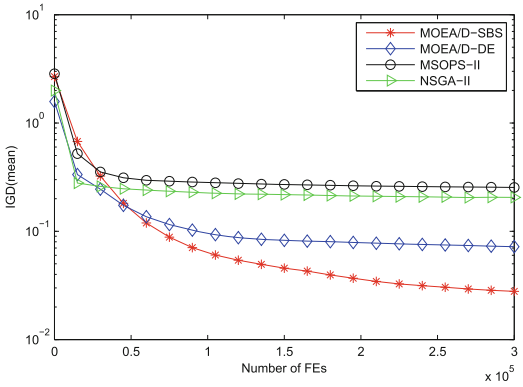
† and ‡ denotes that the performance of the corresponding algorithm is significantly worse than or better than that of MOEA/D-SBS, respectively. The best mean is highlighted in boldface



(a) UF2



(b) UF7



(c) UF9

Fig. 1. Convergence graphs in terms of *IGD* (mean) obtained by MOEA/D-DIS, MOEA/D-DE, MSOPS-II and NSGA-II on three UF instances.

5.2 Comparisons with Classical MOEAs

In this section, we compare MOEA/D-SBS with three classical MOEAs, NSGA-II, MSOPS-II and MOEA/D-DE. Each algorithm is run 30 times independently on each test instance. The comparison results of MOEA/D-SBS with the other three MOEAs in terms of IGD metric values are presented in Table 1. From the table, we can find that MOEA/D-SBS has obtained the best mean metric values on all the test problems except UF10. No algorithm can approximate UF10' PF very well, as it has many local PFs. MSOPS-II has best performance among all the compared algorithms on the UF test problem. In Table 1, all comparisons are statistically significant except the comparison of between MOEA/D-SBS with MSOPS-II on UF4. From Table 2, we can find that the comparison results in terms of I_H metric are similar to the ones in terms of IGD.

The evolution of the average IGD values versus the number of function evaluations in the four algorithms on UF test problems are plotted in Fig. 1. It can be observed from these figures that MOEA/D-SBS performs best on both the convergence speed and the quality of the final solution sets.

5.3 Comparison with MOEA/D-STM

In this section, MOEA/D-STM which has a good performance on UF test problems in [10], will be compared with the proposed MOEA/D-SBS. Table 3 presents their comparison results in terms of IGD metric. From the results, we can find that the performance of MOEA/D-SBS is significantly better than that

Table 3. Mean and standard deviation values of IGD, obtained by MOEA/D-SBS and MOEA/D-STM on UF instances.

Instance	IGD		
	MOEA/D-SBS	MOEA/D-STM	p-value
UF1	**0.001711** (5.67E-05)	0.001980 (6.32E-05)	3.69E-11
UF2	**0.005661** (1.69E-03)	0.007074 (1.84E-03)	1.17E-03
UF3	0.003658 (1.72E-03)	0.003721 (2.48E-03)	0.4918
UF4	0.055980 (2.67E-03)	0.056139 (3.75E-03)	0.9705
UF5	0.246915 (2.46E-02)	0.252158 (2.45E-02)	0.1907
UF6	0.092737 (4.21E-02)	0.082018 (3.75E-02)	0.0824
UF7	**0.002276** (2.64E-04)	0.002658 (6.35E-04)	5.61E-05
UF8	**0.052198** (7.66E-03)	0.067490 (1.26E-02)	3.32E-06
UF9	0.027832 (2.98E-03)	0.030200 (2.07E-02)	0.1087
UF10	1.375528 (2.51E-01)	**1.238261** (2.23E-01)	1.63E-02

Wilcoxon's rank sum test at a 0.05 significance level is performed between MOEA/D-SBS and MOEA/D-STM. Boldface denotes that the performance of the corresponding algorithm is significantly better than that of the other

of MOEA/D-STM on UF1, UF2, UF7 and UF8, but it is significantly worse on UF10, on the other test problems it is not statistically significant. So, we can claim that the proposed algorithm MOEA/D-SBS is competitive with MOEA/D-STM.

6 Conclusion

This paper proposed a sorting-based-selection (SBS) to select offspring population for MOEA/D. The proposed algorithm, MOEA/D-SBS, was tested on the 10 unconstrained problems for CEC09 algorithm competition. The comparison results with 4 multiobjective evolutionary algorithms (NSGA-II, MSOPS-II, MOEA/D-DE and MOEA/D-STM) show that MOEA/D-SBS outperforms other compared algorithms.

Further work includes investigation of the proposed SBS integrated to other framework of multiobjective evolutionary algorithms. We also intend to extend MOEA/D-SBS to tackle many optimization problems.

Acknowledgments. This work was supported in part by the National Natural Science Foundation of China (NSFC) under grant 61300159, by the Natural Science Foundation of Jiangsu Province under grant BK20130808, by the Research Fund for the Doctoral Program of Higher Education of China under grant 20123218120041.

References

1. Auger, A., Bader, J., Brockhoff, D., Zitzler, E.: Hypervolume-based multiobjective optimization: theoretical foundations and practical implications. Theor. Comput. **425**(2), 75–103 (2011)
2. Bader, J., Zitzler, E.: Hype: an algorithm for fast hypervolume-based many-objective optimization. Evol. Comput. **19**(1), 45–76 (2011). http://dx.doi.org/10.1162/EVCO_a_00009
3. Coello, C.A.C., Cortés, N.C.: Solving multiobjective optimization problems using an artificial immune system. Genet. Program. Evolvable Mach. **6**(2), 163–190 (2005)
4. Deb, K.: Multi-Objective Optimization using Evolutionary Algorithms. Wiley, Chichester (2001)
5. Deb, K., Goyal, M.: A combined genetic adaptive search (geneas) for engineering design. Comput. Sci. Inf. **26**, 30–45 (1996)
6. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002)
7. Hughes, E.: MSOPS-II: a general-purpose many-objective optimiser. In: IEEE Congress on Evolutionary Computation, CEC 2007, pp. 3944–3951, September 2007
8. Hughes, E.J.: Multiple single objective pareto sampling. In: Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003), vol. 4, pp. 2678–2684. IEEE Press, Canberra, December 2003
9. Li, H., Zhang, Q.: Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II. IEEE Trans. Evol. Comput. **13**(2), 284–302 (2009)

10. Li, K., Zhang, Q., Kwong, S., Li, M., Wang, R.: Stable matching based selection in evolutionary multiobjective optimization. IEEE Trans. Evol. Comput. **18**(6), 909–923 (2014)
11. Miettinen, K.: Nonlinear Multiobjective Optimization. Kluwer Academic Publishers, Boston (1999)
12. Phan, D.H., Suzuki, J.: R2-IBEA: R2 indicator based evolutionary algorithm for multiobjective optimization. In: 2013 IEEE Congress on Evolutionary Computation (CEC), pp. 1836–1845 (2013)
13. Price, K., Storn, R.M., Lampinen, J.A.: Differential Evolution: A Practical Approach to Global Optimization. Natural Computing Series. Springer, Heidelberg (2005)
14. Schaffer, J.D., Grefenstette, J.J.: Multiobjective learning via genetic algorithms. In: Proceedings of the 9th International Joint Conference on Artificial Intelligence (IJCAI-85), pp. 593–595. AAAI, Los Angeles (1985)
15. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**(6), 712–731 (2007)
16. Zhang, Q., Zhou, A., Zhao, S., Suganthan, P.N., Liu, W., Tiwari, S.: Multiobjective optimization test instances for the CEC 2009 special session and competition. University of Essex, Colchester, UK and Nanyang Technological University, Singapore, Special Session on Performance Assessment of Multi-objective Optimization Algorithms, Technical report (2008)
17. Zitzler, E., Künzli, S.: Indicator-based selection in multiobjective search. In: Yao, X., Burke, E.K., Lozano, J.A., Smith, J., Merelo-Guervós, J.J., Bullinaria, J.A., Rowe, J.E., Tiňo, P., Kabán, A., Schwefel, H.-P. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 832–842. Springer, Heidelberg (2004)
18. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: improving the strength pareto evolutionary algorithm. In: Giannakoglou, K., Tsahalis, D., Periaux, J., Papailou, P., Fogarty, T. (eds.) EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, pp. 95–100, Athens, Greece (2002)
19. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans. Evol. Comput. **3**(4), 257–271 (1999)