A Multiobjective Evolutionary Algorithm Based on Decomposition and Preselection

Jinyuan Zhang(✉), Aimin Zhou, and Guixu Zhang

Shanghai Key Laboratory of Multidimensional Information Processing,
Department of Computer Science and Technology, East China Normal University,
Shanghai 200235, China
jyzhang@ecnu.cn, {amzhou,gxzhang}@cs.ecnu.edu.cn

Abstract. The preselection aims to choose promising offspring solutions from a candidate set in evolutionary algorithms. Usually the preselection process is based on the real or estimated objective values, which might be expensive. It is arguable that the preselection is doing classification in nature, which requires to know a solution is good or not instead of knowing how good it is. In this paper we apply a *classification based preselection (CPS)* to a *multiobjective evolutionary algorithm based on decomposition (MOEA/D)*. In each generation, a set of candidate solutions are generated for each subproblem and only a good one is chosen as the offspring by the CPS. The modified MOEA/D, denoted as MOEA/D-CPS, is applied to a set of test instances, and the experimental results suggest that the CPS can successfully improve the performance of MOEA/D.

Keywords: Preselection · Classification · MOEA/D

1 Introduction

In scientific and engineering areas, many problems can be modeled as *multiobjective optimization problems (MOP)*. In this paper, we consider the following continuous MOP:

$$\begin{array}{l} \min F(x) = (f_1(x), \cdots, f_m(x))^T \\ \text{s.t } x \in \Pi_{i=1}^{n}[a_i, b_i] \end{array} \tag{1}$$

where $x = (x_1, \cdots, x_n)^T \in R^n$ is a decision variable vector, $\Pi_{i=1}^{n}[a_i, b_i] \subset R^n$ defines the feasible region of the search space, $f_i : R^n \rightarrow R, i = 1, \cdots, m$, is a continuous mapping, and $F(x)$ is an objective vector.

Since the objectives of (1) usually conflict with each other, there does not exit a solution that can minimize all the objects at the same time. Therefore, the tradeoff solutions, called *Pareto optimal solutions* [1], between the objectives are of interests. The set of all the Pareto optimal solutions is called the *Pareto set (PS)* in the decision space and the *Pareto front (PF)* in the objective space. Since evolutionary algorithms are able to approximate the PS (PF) in a single run, they have become the major method to deal with MOPs. A variety of *multiobjective evolutionary algorithms (MOEAs)* [2] have been proposed in last decades. Most of these algorithms can be classified into three categories.

1. Domination based MOEAs: In these algorithms, the selection operators are based on the Pareto domination relationship [3–5].
2. Indicator based MOEAs: These algorithms use the performance metrics as the objective to optimize and thus do selection [6–8].
3. Decomposition based MOEAs: This kind of algorithms decompose an MOP into a set of scalar-objective subproblems (SOPs), and solve them simultaneously [9,10].

The *multiobjective evolutionary algorithm based on decomposition (MOEA/D)* is one of the most popular decomposition based MOEAs. The basic idea of MOEA/D is to decompose the MOP into a set of scalar-objective subproblems. The neighboring subproblems collaborate with each other to generate new offspring solutions, and a new one will update not only the solution of the corresponding subproblem but also that of the neighboring subproblems. By this way, all the subproblems are tackled simultaneously, and the final solutions of the subproblems form an approximation to the PS (PF) of the original MOP. In MOEA/D and its variants, either the crossover and/or mutation operators [10,11] or probabilistic model based reproduction operators [12–14] are used to generate new offspring solutions.

This paper focuses on the preselection in MOEA/D, which has not been studied to the best of our knowledge. Following our previous work in preselection [15,16], a *classification based preselection (CPS)* is applied to MOEA/D. In each generation, a classification model is built according to a set of recorded training data set with either positive or negative labels. After that, a set of candidate offspring solutions are generated, the classification model is applied to label the candidate offspring solutions, and only those with positive labels are kept as the offspring solutions. In the approach, the nondominated sorting scheme [4] is used to maintain the training data set.

The rest of the paper is organized as follows. Section 2 presents the MOEA/D framework with CPS. The algorithm implementation details, such as classifier training, and offspring reproduction are introduced as well. The proposed approach is systematically studied in Sect. 3. Finally Sect. 4 concludes this paper with some future work remarks.

2 MOEA/D with Classification Based Preselection

This section introduces the proposed modified MOEA/D with the CPS. For simplicity, we denote this approach as MOEA/D-CPS.

2.1 Basic Idea

There is no reason that only one offspring solution is generated for each subproblem in MOEA/D. As shown in Fig. 1, there is a high probability to generate a good solution if more offspring solutions are generated. However, it may need more function evaluations as well. To deal with this problem, we can use the preselection to filter bad ones and keep promising ones.

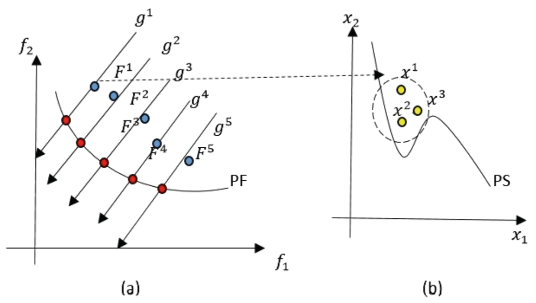


Fig. 1. An illustration of the advantage to generate more offspring solutions for each subproblem in MOEA/D

The purpose of preselection is to select some promising offspring solutions from a set of candidate offspring solutions. Usually, the preselection is based on the estimated objective values through surrogate models or meta models. However, the surrogate model building itself is an expensive procedure. Actually, the preselection only needs to know which solution is good and which one is bad instead of knowing how good or how bad it is. Thus the preselection can be naturally regarded as a classification process, i.e., to classify the candidate offspring solutions into two sets: the selected good ones and the unselected bad ones. Basically, a classification model needs to find a boundary between two classes of points while a surrogate model needs to find the function values of each point in the space. Therefore, the classification model building will be generally cheaper than the surrogate model building.

The key point here is to build a classifier to differentiate the candidate solutions. Fortunately, there are a variety of classification models in the community of statistical and machine learning that we can use directly [17–19]. Furthermore in the running process of MOEA/D, we can record some good and bad solutions found so far and use them as the classification model training samples.

2.2 Classifier Training

The classification model training problem can be formulated as follows. Let $\{< x, label >\}$ be a set of training data, where x is a feature vector of a data point, $label \in L$ is the label of the data point and L is a set of labels. The relationship between a feature vector and the corresponding label can be denoted as $label = Class(x)$. The target of a classifier training is to find a relationship $label = Cl\hat{a}ss(x)$ that can approximate the real relationship $label = Class(x)$ at most based on the training data set.

In our case, a solution x can be directly regarded as a feature vector, and $L = \{-1, +1\}$ in which $+1$ is the label of good solutions and -1 denotes bad solutions. Following our previous work [15,16], this paper will use *k-nearest neighbor (KNN)* [20] as the classification algorithm.

$$label = KNN(x) = sign\left(\sum_{y \in N(x)} label_y\right)$$

where $sign(x)$ returns $+1$ if $x > 0$ and -1 if $x < 0$, $N(x)$ denotes the K nearest neighbors of x from the training set, and K is an odd number.

To implement CPS, we introduce two external populations P_+ and P_-: P_+ contains some 'good' solutions found so far with label $+1$, and P_- contains some 'bad' solutions with label -1. In order to obtain P_+ and P_-, this paper uses the nondominated sorting scheme introduced in [4]. Let $q = NDS(p, n)$ denote this scheme, which selects the best n solutions from p and stores the selected ones in q. The details of this procedure are referred to [4]. Let P be the initial population and N be the size of P, P_+ and P_- are initialized as

$$P_+ = NDS\left(P, \left\lfloor \frac{N}{2} \right\rfloor\right)$$

and

$$P_- = P \backslash P_+.$$

Let Q be the set of newly generated solutions in each generation, Q_+ and Q_- contain the nondominated and dominated solutions in Q respectively. P_+ and P_- are updated as

$$P_+ = NDS\left(P_+ \cup Q_+, \left\lfloor \frac{N}{2} \right\rfloor\right)$$

and

$$P_- = NDS\left(P_- \cup Q_-, \left\lfloor \frac{N}{2} \right\rfloor\right).$$

2.3 Offspring Reproduction

Different reproduction operators can be used in MOEA/D. In this paper, we choose some operators based on *differential evolution (DE)* [11] and the polynomial mutation [10] to generate a set of candidate offspring solutions.

Multiple Operator Search. In [11], we proposed to use multiple DE operators to generate offspring solutions following the idea in [21]. Let x^i be the solution with the ith subproblem, $r1, r2, r3, r4, r5$ be five randomly selected neighboring subproblems, and F be randomly selected from $\{0.5, 0.7\}$. The three operators are defined as follows.

$$\begin{aligned}
\hat{y}^1 &= x^i + F(x^{r1} - x^{r2}).\\
\hat{y}^2 &= x^i + rand()(x^{r1} - x^{r2}) + F(x^{r3} - x^{r4}).\\
\hat{y}^3 &= x^{r1} + rand()(x^{r2} - x^{r3}) + F(x^{r4} - x^{r5}).
\end{aligned} \tag{2}$$

In [11], the best one according to real objective value is chosen as the offspring solution, while in our approach the best one according to the classifier is chosen.

Polynomial Mutation. It should be noted that after the above operation, each candidate solution $\hat{y}$ is repaired in (3) and then mutated by the polynomial mutation in (4) before the function evaluation or classification.

$$\bar{y}_j = \begin{cases} a_j + 0.5(x_j^i - a_j) & \text{if } \hat{y}_j < a_j \\ b_j - 0.5(b_j - x_j^i) & \text{if } \hat{y}_j > b_j \\ \hat{y}_j & \text{otherwise} \end{cases} \tag{3}$$

where $j = 1, \cdots, n$.

$$y_j = \begin{cases} \bar{y}_j + \sigma_j \times (b_i - a_i) & \text{if } rand() < p_m \\ \bar{y}_j & \text{otherwise.} \end{cases} \tag{4}$$

with

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

where $j = 1, \cdots, n$, $rand$ is a uniform random number in [0,1]. η and p_m are two control parameters. a_j and b_j are the lower and upper bound of the ith decision variable.

2.4 MOEA/D-CPS Framework

An MOEA/D decomposes an MOP into a set of scalar-objective subproblems and solves them simultaneously. The optimal solution of each subproblem will hopefully be a Pareto optimal solution of the original MOP, and a set of well selected subproblems may produce a good approximation to the PS (PF). Therefore, a key issue with MOEA/D is the subproblem definition. In this paper, we use the Tchebycheff technique as follows.

$$\min g(x|\lambda, z^*) = \max_{1 \le j \le m} \lambda_j |f_j(x) - z_j^*| \tag{5}$$

where $\lambda = (\lambda_1, \cdots, \lambda_m)^T$ is a weight vector with the subproblem,

$$z^* = (z_1^*, \cdots, z_m^*)^T$$

is a reference point, i.e., z_j^* is the minimal value of f_j in the search space. For simplicity, we use $g^i(x)$ to denote the above subproblem with the ith subproblem. In most cases, two subproblems with close weight vectors will have similar optimal solutions. Based on the distances between the weight vectors, MOEA/D defines the neighborhood of a subproblem where the subproblems with the nearest weight vectors. In MOEA/D, the offspring reproduction and solution selection are based on the concept of neighborhood.

Algorithm 1. Main Framework of MOEA/D-CPS

```
1  Initialize a set of subproblems $(x^i, F^i, B^i, g^i)$, $i = 1, \cdots, N$, initialize the
   reference point $z^*$ as $z_j^* = \min_{i=1,\cdots,N} f_j(x^i)$, $j = 1, \cdots, m$;
2  Set $P_+ = NDS\left(\{x^1, \cdots, x^N\}, \lfloor \frac{N}{2} \rfloor\right)$, and $P_- = \{x^1, \cdots, x^N\} \backslash P_+$;
3  while not terminate do
4      Set $Q = \emptyset$;
5      Train a classifier $label = \hat{Class}(x)$ with data set $P_+ \cup P_-$;
6      foreach $i \in \{1, \cdots, N\}$ do
7          Set the mating pool as
               $\pi = \begin{cases} B^i & \text{if } rand() < p_n \\ \{1, \cdots, N\} & \text{otherwise} \end{cases}$
8          Generate $M$ trial solutions $Y = \{y^1, \cdots, y^M\}$ by the mating pool $\pi$;
9          Set $V = \{y \in Y | \hat{Class}(y) = 1\}$, and reset $V = Y$ if $V = \emptyset$;
10         Randomly choose $y \in V$ as the offspring solution and evaluate it;
11         Update the reference point
               $z_j^* = \begin{cases} f_j(y) & \text{if } f_j(y) < z_j^* \\ z_j^* & \text{otherwise} \end{cases}$
12         Set counter $c = 0$;
13         foreach $j \in \pi$ do
14             if $g^j(y) < g^j(x^j)$ and $c < C$ then
15                 Replace $x^j$ by $y$;
16                 Set $c = c + 1$;
17             end
18         end
19         Set $Q = Q \cup \{y\}$;
20     end
21     Set $Q_+$ and $Q_-$be the nondominated and dominated solutions in $Q$
       respectively;
22     Update $P_+ = NDS\left(P_+ \cup Q_+, \lfloor \frac{N}{2} \rfloor\right)$;
23     Update $P_- = NDS\left(P_- \cup Q_-, \lfloor \frac{N}{2} \rfloor\right)$;
24 end
25 return the population P.
```

In MOEA/D, the ith $(i = 1, \cdots, N)$ subproblem will maintain following information:

- its weight vector λ^i and its objective function g^i,
- its current solution x^i and the objective vector of x^i, i.e. $F^i = F(x^i)$, and
- the index set of its neighboring subproblems, B^i.

The main framework of MOEA/D-CPS is shown in Algorithm 1. It is basically a general MOEA/D framework and only several steps with the classification model building and using are added. We would like to make the following comments.

- Some notations are as follows. N is the number of subproblems and it is also the population size. $K = |B^i|$ is the neighborhood size. p_n denotes the probability to generate trial solutions using the neighboring solutions. M is the number of generated candidate offspring solutions for each subproblem. C is the maximal number of old solutions that can be replaced by a new one. $rand()$ generates a random real number in $[0, 1]$.
- As in the original MOEA/D [10], the population is initialized in *Line 1*. The mating pool is set in *Line 7* and the candidate offspring solutions are generated in *Line 8* by the reproduction operators introduced in Sect. 2.3. The reference point is updated in *Line 11*. The population is updated by the offspring solutions in *Lines 12–18*.
- In the CPS, two external populations are maintained in *Line 2* and *Lines 21–23*. The two external populations are then used as a data set to train a classifier in *Line 5*, which is defined in Sect. 2.2. A promising offspring is chosen out according to the classifier in *Line 9*.

3 Experimental Study

3.1 Experimental Settings

In this section we apply MOEA/D-CPS to 9 test instances named as LZ1-LZ9, which are introduced in [10]. MOEA/D-MO [11], a modified MOEA/D that uses the reproduction operators introduced in Sect. 2.3 to generate new trial solutions, is chosen for the comparison study. For simplicity, we use MO to denote MOEA/D-MO that uses the multiple operator search strategy, and use CPS-MO to denote MOEA/D-CPS with the classification based search strategy.

The parameter settings are as follows.

- The number of decision variables are $n = 30$ for all the test instances.
- The algorithms are executed 50 times independently on each instance and stop after $150,000$ for bi-objective and $297,500$ for tri-objective problems respectively.
- The population size is $N = 300$ and $N = 595$ for bi-objective and tri-objective problems respectively, and the neighborhood size is $K = 15$ and $K = 30$ for bi-objective and tri-objective problems respectively.
- The other parameters are $T = 20$, $p_n = 0.9$, $C = 2$, $\eta = 20$ and $p_m = \frac{1}{n}$.
- In CPS, the number of nearest points used in KNN is $K = 3$.

All the algorithms are implemented in Matlab and executed in the same computer.

3.2 Performance Metrics

In this paper we use *Inverted Generational Distance (IGD)* metric [22] to assess the performance of the algorithms in our experimental.

Let P^* be a set of well-distributed Pareto optimal points from the PF, and P be the set of nondominated solutions found. The IGD metric is defined as follows.

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

where $d(x, P)$ is the minimum Euclidean distance between x and any point in P, and $|P^*|$ is the cardinality of P^*. The IGD value denotes some kind of distance from the given reference set P^* to the obtained set P. In our experiments, $10,000$ evenly distributed points in PF are generated as the P^*.

3.3 Experimental Results

Statistical Results. Table 1 shows the mean, std, min, max IGD value of CPS-MO and MO over 50 runs. In order to get statistically conclusions, the Wilcoxon's rank sum test at a 5 % significance level is employed to compare the IGD values obtained by different algorithms. In the table, $\sim$, $+$, and $-$ denote that the results obtained by the CPS version are similar to, better than, or worse than that obtained by the original version.

Table 1 clearly shows that with the Wilcoxon rank test on the obtained values, the CPS-MO beats the MO on 4 test instances: LZ2-LZ4, and LZ9. And on other 5 instance the two algorithms get similar results. According to the mean values, CPS-MO performs better than MO on LZ2-LZ9 and gets the same result on LZ1.

Runtime Performance. Figure 2 plots the run time performance in terms of the IGD values obtained by the two variants on the 9 test instances.

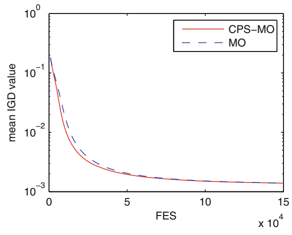
Figure 2 shows the run time performance of CPS-MO and MO. It is clear that the convergence speeds of the two algorithms are similar on LZ1, LZ3, LZ5, LZ6 while CPS-MO beats MO on LZ2, LZ4, LZ7-LZ9.

Visual Results. Figures 3 and 4 plot the final PF and PS approximations obtained by CPS-MO and MO over 50 runs respectively.

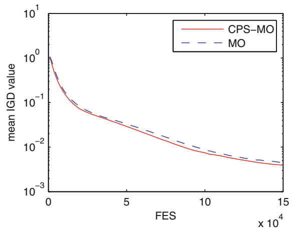
It is clear that the two algorithms perform similar. However there are also some differences. For example on LZ7, the convergence of the population obtained by CPS-MO is better than MO. Also the similar results can be found on LZ6 and LZ8.

Table 1. The statistical results obtained by CPS-MO and MO on LZ1-LZ9.

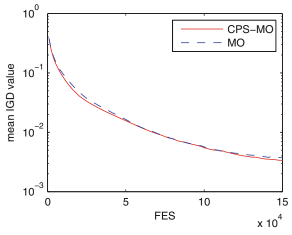
Instance	CPS-MO				MO			
	mean	std.	min	max	mean	std.	min	max
LZ1	**1.39e-03**(∼)	1.29e-05	1.36e-03	1.42e-03	**1.39e-03**	1.50e-05	1.36e-03	1.43e-03
LZ2	**3.92e-03**(+)	5.42e-04	3.14e-03	5.39e-03	4.51e-03	9.88e-04	3.25e-03	8.50e-03
LZ3	**3.39e-03**(+)	6.58e-04	2.67e-03	5.67e-03	3.71e-03	9.47e-04	2.76e-03	6.68e-03
LZ4	**4.03e-03**(+)	1.12e-03	2.82e-03	8.31e-03	4.87e-03	1.25e-03	3.12e-03	8.81e-03
LZ5	**7.66e-03**(∼)	1.44e-03	5.93e-03	1.49e-02	7.70e-03	9.80e-04	6.38e-03	1.01e-02
LZ6	**4.19e-02**(∼)	7.44e-03	3.02e-02	5.59e-02	4.22e-02	7.37e-03	2.91e-02	5.92e-02
LZ7	**1.20e-01**(∼)	1.02e-01	4.13e-03	3.66e-01	1.24e-01	1.07e-01	5.21e-03	3.40e-01
LZ8	**1.37e-02**(∼)	1.11e-02	2.66e-03	5.26e-02	1.48e-02	1.16e-02	3.08e-03	6.51e-02
LZ9	**4.99e-03**(+)	1.14e-03	3.19e-03	7.58e-03	5.96e-03	1.39e-03	3.84e-03	9.44e-03



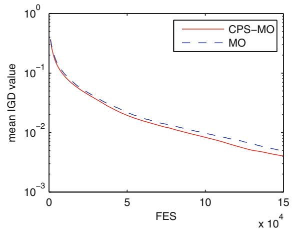
(a) LZ1



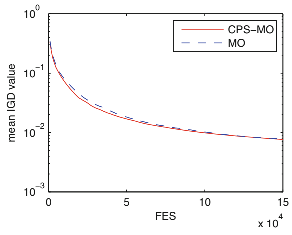
(b) LZ2



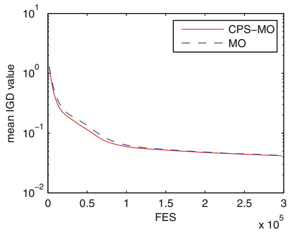
(c) LZ3



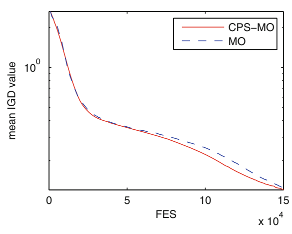
(d) LZ4



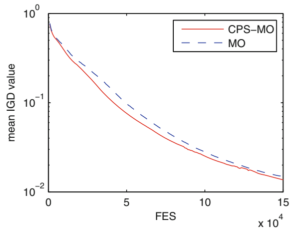
(e) LZ5



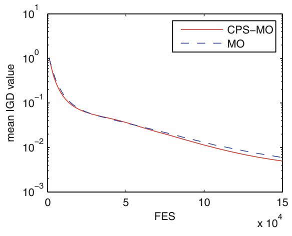
(f) LZ6



(g) LZ7



(h) LZ8



(i) LZ9

Fig. 2. The mean IGD values versus FES obtained by CPS-MO and MO over 50 runs.

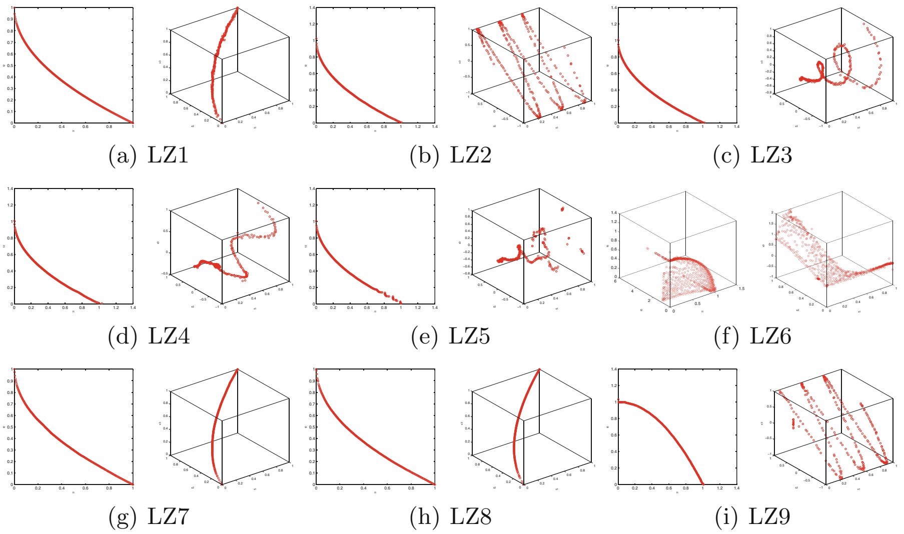

(a) LZ1 (b) LZ2 (c) LZ3
(d) LZ4 (e) LZ5 (f) LZ6
(g) LZ7 (h) LZ8 (i) LZ9

Fig. 3. The final PF and PS approximations obtained by CPS-MO over 50 runs.

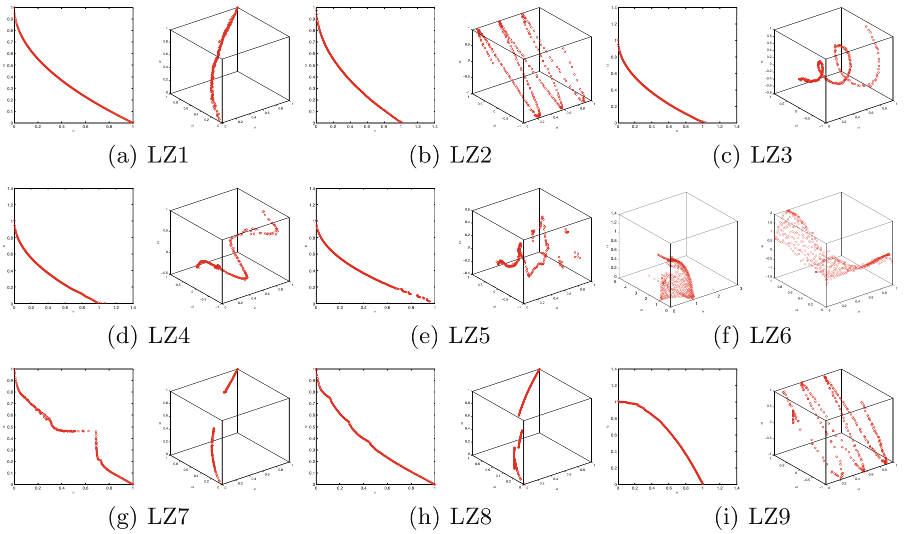

(a) LZ1 (b) LZ2 (c) LZ3
(d) LZ4 (e) LZ5 (f) LZ6
(g) LZ7 (h) LZ8 (i) LZ9

Fig. 4. The final PF and PS approximations obtained by MO over 50 runs.

4 Conclusion

This paper proposed to use a *classification based preselection (CPS)* to improve the performance of the *multiobjective evolutionary algorithm based on decomposition (MOEA/D)*. This algorithm, named as MOEA/D-CPS, utilizes the *non-domination sorting (NDS)* operator to choose some solutions to form a training set, and then builds a classifier based on the training set. In each generation, a set of candidate offspring solutions are generated for each subproblem and only a promising one according to the classifier is chosen as the offspring solution.

MOEA/D-CPS is applied to a test suite, and the experimental results suggest that the CPS can successfully improve the performance of an MOEA/D variant.

The work reported in this paper is preliminary and there are still some work that could be done in the future. For example, we could (a) improve the efficiency of the CPS, (b) try other classification models, and (c) apply this approach to other MOEAs.

Acknowledgment. This work is supported by China National Instrumentation Program under Grant No.2012YQ180132, the National Natural Science Foundation of China under Grant No.61273313, and the Science and Technology Commission of Shanghai Municipality under Grant No.14DZ2260800.

References

1. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. Wiley, Chichester (2001)
2. Zhou, A., Qu, B.Y., Li, H., Zhao, S.Z., Suganthanb, P.N., Zhang, Q.: Multiobjective evolutionary algorithms: a survey of the state of the art. Swarm Evol. Comput. **1**, 32–49 (2011)
3. Knowles, J., Corne, D.: Approximating the nondominated front using the pareto archived evolution strategy. Evol. Comput. **8**, 149–172 (2000)
4. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**, 182–197 (2002)
5. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: improving the strength pareto evolutionary algorithm. In: Evolutionary Methods for Design Optimisation and Control, pp. 95–100 (2001)
6. Basseur, M., Zitzler, E.: Handling uncertainty in indicator-based multiobjective optimization. Int. J. Comput. Intell. Res. **2**, 255–272 (2006)
7. Zitzler, E., Künzli, S.: Indicator-based selection in multiobjective search. In: Yao, X., et al. (eds.) PPSN 2004. LNCS, vol. 3242, pp. 832–842. Springer, Heidelberg (2004)
8. Bader, J., Zitzler, E.: HypE: an algorithm for fast hypervolume-based manyobjective optimization. Technical report TIK 286, Computer Engineering and Networks Laboratory, ETH Zurich 19, pp. 45–76 (2010)
9. Zhang, Q., Li, H.: MOEA/D: a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. **11**, 712–731 (2007)
10. Li, H., Zhang, Q.: Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. IEEE Trans. Evol. Comput. **13**, 284–302 (2009)
11. Li, Y., Zhou, A., Zhang, G.: An MOEA/D with multiple differential evolution mutation operators. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp. 397–404 (2014)
12. Zhang, Q., Zhou, A., Jin, Y.: RM-MEDA: a regularity model-based multiobjective estimation of distribution algorithm. IEEE Trans. Evol. Comput. **12**, 41–63 (2008)
13. Zhou, A., Zhang, Q., Zhang, G.: A multiobjective evolutionary algorithm based on decomposition and probability model. In: 2012 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8 (2012)

14. Zhou, A., Zhang, Y., Zhang, G., Gong, W.: On neighborhood exploration and subproblem exploitation in decomposition based multiobjective evolutionary algorithms. In: 2015 IEEE Congress on Evolutionary Computation (CEC) (2015)
15. Zhang, J., Zhou, A., Zhang, G.: A classification based preselection for evolutionary algorithms. Swarm and Evolutionary Computation (2015, under review)
16. Zhang, J., Zhou, A., Zhang, G.: A classification and pareto domination based multiobjective evolutionary algorithm. In: 2015 IEEE Congress on Evolutionary Computation (CEC) (2015)
17. Bishop, C.: Pattern Recognition and Machine Learning. Springer, New York (2006)
18. Weiss, S.M., Kapouleas, I.: An empirical comparison of pattern recognition, neural nets, and machine learning classification methods. In: Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, pp. 781–787 (1989)
19. Michie, D., Spiegelhalter, D.J., Taylor, C.: Machine Learning, Neural and Statistical Classification. Ellis Horwood, Upper Saddle River (1994)
20. Coomans, D., Massart, D.: Alternative k-nearest neighbour rules in supervised pattern recognition: Part 1. k-Nearest neighbour classification by using alternative voting rules. Anal. Chim. Acta **136**, 15–27 (1982)
21. Wang, Y., Cai, Z., Qingfu, Z.: Differential evolution with composite trial vector generation strategies and control parameters. IEEE Trans. Evol. Comput. **15**, 55–66 (2011)
22. Zhou, A., Zhang, Q., Jin, Y.: Approximating the set of Pareto-optimal solutions in both the decision and objective spaces by an estimation of distribution algorithm. IEEE Trans. Evol. Comput. **13**, 1167–1189 (2009)