2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

A Clustering Based Multiobjective Evolutionary Algorithm

Hu Zhang and Shenmin Song
Center for Control Theory and Guidance Technology, Harbin Institute of Technology, Harbin, China
jxzhanghu@126.com, songshenmin@hit.edu.cn

Aimin Zhou
Department of Computer Science and Technology, East China Normal University, Shanghai, China
amzhou@cs.ecnu.edu.cn

Xiao-Zhi Gao
Department of Electrical Engineering and Automation, Aalto University School of Electrical Engineering, Aalto, Finland
xiao-zhi.gao@aalto.fi

***Abstract*—In this paper, we propose a clustering based multiobjective evolutionary algorithm (CLUMOEA) to deal with the multiobjective optimization problems with irregular Pareto front shapes. CLUMOEA uses a k-means clustering method to discover the population structure by partitioning the solutions into several clusters, and it only allows the solutions in the same cluster to do the reproduction. To reduce the computational cost and balance the exploration and exploitation, the clustering process and evolutionary process are integrated together and they are performed simultaneously. In addition to the clustering, CLUMOEA also uses a distance tournament selection to choose the more similar mating solutions to accelerate the convergence. Besides, a cosine nondominated selection method considering the location and distance information of the solutions are further presented to construct the final population with good diversity. The experimental results show that, compared with some state-of-the-art algorithms, CLUMOEA has significant advantages on dealing with the given test problems with irregular Pareto front shapes.**

I. Introduction

In real world, many optimization problems have several objectives, and these objectives are usually conflicting. This kind of optimization problems are called multiobjective optimization problems (MOPs). Different from the scalar-objective optimization problems, it is impossible to find a solution to optimize all the objectives at the same time for a MOP. Instead, a set of tradeoff solutions (Pareto optimal solutions) among the different objectives, named Pareto set (PS) in the solution space and Pareto front (PF) in the objective space, are of interests [1], [2]. Obviously, for an MOP, there exists plenty of Pareto optimal solutions. However, in the solving process, we do not hope to obtain all of them, but limited number of representative ones, which converge to the PF as close as possible and distribute along the PF as uniforming as possible.

Evolutionary algorithms (EAs) play an important role in dealing with MOPs, and they have the potential to get multiple nondominated solutions to approximate the PFs in a single run. For this reason, multiobjective optimization evolutionary algorithms (MOEAs) have attracted more and more attention in both scientific research and engineering applications [3], [4], [5], [6], [8], [9]. In the evolutionary multiobjective optimization community, there are three kinds of well-known algorithms, i.e., indicator based MOEAs [7], dominance based MOEAs [5] and decomposition based MOEAs (MOEAs/D) [8], [9]. Usually, if the objective spaces of MOPs are given, and the PF shapes are close to a quarter of circle or spherical surface, the MOEA/D has significant advantages to deal with them. However, for the problems, whose objective spaces are unknown and the PF shapes are irregular, MOEA/D may not work well. This is due to the fact that, MOEA/D uses a set of uniformly distributed weight vectors to guide the solutions to approach to the PF. The location that each vector point projects to the PF is the solution's final convergence location guided by that vector. Obviously, when the PF shapes are similar to a quarter of circle or spherical surface, the weight vector points will have uniformly distributed projection locations, which means that the final solutions will have a uniform distribution on the PF. On the other hand, if the objective space of a MOP is unknown or its PF shape is irregular, it is hard to set the uniform weight vectors, and the projection points are uneven, which will lead the solutions to converge to undesired regions. To deal with this drawback, a dynamic weight design method based on the projection of the current nondominated solutions and equidistant interpolation was proposed in [15]. This strategy, however, may lead to additional computational cost.

According to the above analysis, we can see that, the weight vector is a key point for an MOEA/D, and it is highly related to the properties of the problems to tackle. The dominance based MOEAs do not need weight vectors to guide the convergence, and compared with MOEA/D, they might be more suitable to tackle the MOPs with unknown objective space or irregular PF shapes in a sense. Therefore, in this paper, we focus on dominance and propose a clustering based multiobjective optimization evolutionary algorithm (CLUMOEA), to deal with this kind of MOPs.

CLUMOEA is an approach of combining a machine learning (ML) method and an EA. Its main characteristic is that, an individual in the EA is regarded as a training example; the population forms a training data set, and the ML is used to extract population information and to guide the EA search. This combination is promising to improve the performance of the EA in a sense. In our previous work [11], we have applied a k-means clustering method [12] to EA and proposed a clustering based mating selection (CMS). The CMS was adopted in two scalar-objective EAs and it speeded up the convergence and improved the quality of final solutions dramatically. Different from the previous work, in this paper, our attention is paid on the the combination of a clustering method and an MOEA.

In CLUMOEA, a clustering based distance tournament mating selection (CDTMS) operator is designed. Similar to the

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

CMS, CDTMS uses a k-means clustering method to partition the population into several classes in each generation, and only the solutions in the same class are allowed to mate with each other. In the early stages, the classes might not be accurate, and this is suitable for exploration; while in the latter stages, the classes will be expected to be accurate, and this is suitable for exploitation. Different from CMS, CDTMS develops a distance tournament selection to select mating parents in the same class. The difference between the distance tournament selection and the well-known binary tournament selection [10] is that, when two individuals are competing to become a parent, the one with a smaller Euclidean distance to the current individual will become the winner. In addition to CDTMS, for CLUMOEA, a cosine crowding measurement based nondominated solution selection is proposed to improve its performance. Seven MOPs with irregular PS shapes are chosen to construct a test suite; and the representatives of dominance based MOEAs and decomposition based MOEAs, NSGA-II [5] and MOEA/D-DE [9], are utilized as the comparison algorithms to show the advantages of CLUMOEA.

The rest of the paper is organized as follows. Section II presents the proposed CDTMS including the k-means clustering method and CDTMS procedures. Section III introduces the cosine crowding measurement based selection for nondominated solutions. Section IV depicts the CLUMOEA procedures. Section V compares CLUMOEA with MOEA/D-DE and NSGA-II based on the constructed test suite, and analyzes the performance. Finally, the paper is concluded in Section VI.

II. Clustering Based Distance Tournament Mating Selection

This section introduces the CDTMS in details including the k-means clustering method which is used in this paper and the concrete procedures of CDTMS.

A. K-means Clustering Method

As stated before, CDTMS depends on the k-means clustering method to locate the population individuals into several classes. Suppose there are N data points $x_1, x_2, \ldots, x_N$ that need to be partitioned into K clusters. The k-means algorithm works as follows.

Step 1 **(Initialization)** Initialize the cluster centers $m_1, m_2, \ldots, m_K$, and set K empty clusters $C_1, C_2, \ldots, C_K$.

Step 2 **(Assignment)** For each data point $x_i, (i = 1, 2, \ldots, N)$, assign it to the k-th cluster C_k, which satisfies

$$k = \arg \min_{j=1,2,\ldots,K} dis(x_i, m_j),$$

where $dis(a, b)$ measures the distance between two points a and b.

Step 3 **(Update)** For each cluster $C_k, (k = 1, 2, \ldots, K)$, update its center as

$$m^k = \left| \frac{1}{C_k} \right| \sum_{x \in C_k} x.$$

Step 4 If the stop condition is not satisfied, go to **Step 2**; otherwise, terminate and return the partition $C_1, C_2, \ldots, C_K$.

It should be noted that (a) there are many ways to initialize the cluster centers in **Step 1**, and in this paper, we randomly choose K training points as the cluster centers; (b) we use the Euclidean distance in **Step 2** to measure the similarity among points; (c) the k-means iteration stops, once the partition does not change in two consecutive iterations.

B. Clustering Based Distance Tournament Mating Selection

From above description, we can see that the k-means method has heavy computational cost on the distance calculations and similarity comparisons. Especially, the k-means technique also contains an iteration process as an EA does, if it is called in each generation, which makes the computational cost even severe. In order to apply the k-means method with a low cost, the idea adopted here is to combine the clustering iteration process and the MOEA evolutionary process together. In each generation of CLUMOEA, the k-means method does the clustering operation on the population for only one iteration, then the population individuals are set into several clusters according to the operation results. Based on the population partition, only the solutions in the same cluster are selected to generate new trial solutions.

Mate selection always works with the mating mechanism. In CLUMOEA, differential evolution (DE) [13] is chosen as the reproduction method, which requires three mating solutions to produce a new solution. In our algorithm, the DE operations are performed for each solution, which means that, each individual will alternatively become the current individual to take part in the DE, and the other two individuals mating with current one are selected from the whole population. Therefore, the task of CDTMS is to choose proper mating solutions for each solution and assist their DE crossover .

In one generation, the CDTMS procedures are as follows:

Step 1 Perform a clustering operation with one iteration on population $P = (x_1, x_2, \ldots, x_N)$, and update the clustering model.

Step 2 Based on the current clustering model, partition the population P into K clusters $C_1, C_2, \ldots, C_K$.

Step 3 For each solution $x \in C_k (k = 1, 2, \ldots, K)$,

- **Step 3.1** Select two different mating solutions u_1, u_2 from C_k by using a distance tournament selection (DTS) method.
- **Step 3.2** x, u_1, u_2 make up a mating group (MG).
- **Step 3.3** Perform the DE operation on the MG, and generate a new trial solution.

We have several comments on above CDTMS procedures.

1) In **Step 2**, if there is an empty cluster, the cluster will be discarded.

2) In **Step 3.1**, if $x \in C_k$, and the size of C_k is less than 3, for x, the other two mating solutions are selected from the group composed of n closest solutions to it, $n = 20$ in this paper.
3) In **Step 3.1**, the method of using the DTS to choose a mating solution u_1 for solution x, where x belongs to cluster C_k, obeys following steps.

First	Randomly choose two solutions y_1, y_2 from C_k.
Second	Determine the distances of y_1 and y_2 to x, d_1 and d_2, which have been calculated in clustering process.
Last	If $d_1 < d_2$, y_1 is the winner, and $u_1 \leftarrow y_1$; or y_1 wins, and $u_1 \leftarrow y_2$.

With respect to CDTMS, the following analysis is made.

1) In CDTMS, in each generation of an EA, the k-means clustering process is not iteratively implemented, and instead it iterates only once. The iterations of the clustering and the evolutions of the EA are performed simultaneously. In this way, the computational cost of the clustering calculations is reduced greatly.
2) Usually, for an EA, in its early stages, we hope that the mating solutions are very dissimilar so that it can generate the solutions with big differences and maintain the population diversity well; but in the later convergence stages, we also expect that the mating solutions have a certain similarity. This is because that mating among the similar solutions is helpful to produce high quality solutions and to accelerate the convergence. In CDTMS, since it only does a rough clustering operation, in the early stages of the EA, the solutions assigned into the same cluster are quite different; and after converging to the later stages, the solutions in the same cluster are also similar. Thus, in a sense, the CDTMS selects the solutions in the same cluster to mate, and it can promote the EA to meet our requirements and to balance the exploration and exploitation.
3) A distance tournament selection method is designed in CDTMS, which aims to guide similar solutions to mate. In the early stages, since the differences among the solutions in the same cluster are very large, and the DTS does not have a significant effect; however, in the later stages, the candidate mating solutions in the same cluster have a high similarity, and the DTS guides the similar solutions to mate further, which is quite effective to improve the solution quality. Therefore DTS can accelerate the convergence speed greatly.

III. Cosine Crowding Measurement based Nondominated Solution Selection

It is well known that the selection for nondominated solutions is very important for a dominance based MOEA; it directly determines the convergence and distribution of the final nondominated solutions. To our best knowledge, the popular nondominated solution selection method is the fast nondominated sorting approach proposed by Deb in his NSGA-II algorithm [5]. In NSGA-II, each solution is first assigned a rank according to its dominance relationship with the others; next for each rank, it calculates the crowding distances of the solutions belonging to it; finally, the solutions with lower ranks and larger crowding distances are selected into the next generations. As a matter of fact, the NSGA-II method has its drawbacks. In the crowding distance measurement, it just considers the distances among the adjacent solutions, and the location information of each solution is not considered, which is not beneficial to obtain uniformly distributed solutions. Besides, the crowding assessments are limited within each rank, and the distance value of each solution represents its crowding degree with the solutions in the same rank, which can not reflect its actual crowding status with the solutions in the whole population. Thus, the selected solutions with large crowding distances might be not desired. In order to improve the performance of nondominated solution selection, in CLUMOEA, we propose a cosine crowding measurement based nondominated solution selection (CNDS) method. The basic idea of CNDS is that a cosine crowding measurement involving the location information is appied to calculate the distances among the solutions, and the crowding assessments are carried out from the perspective of the whole population so that the final nondominated solutions we obtain are as diverse and uniform as possible.

A. Cosine Crowding Measurement

The cosine crowding measurement (cCM) is a new calculation method for the crowding distances of the solutions. It is designed in our previous work [14]. To calculate the crowding degree of a solution, the basic idea of cCM is utilizing a cosine function to combine its location and distance information to the adjacent solutions and getting a comprehensive crowding degree. For a MOP $F = (f_1, \ldots, f_t, \ldots, f_T)$, there are N solutions; FV is the set of their objective values. The pseudo-code of using the cCM to calculate the crowding distances of the solutions are shown in Algorithm 1.

Algorithm 1 Pseudo-code of a cosine crowding measurement

```
1: proc CosineCrowdingMeasurement(FV)
2: CD = zeros(N,1);
3: for t = 1 : T do
4:    cd ← zeros(N,1);
5:    [s, p] ← sort(FV_[t]);
6:    for i = 1 : N do
7:       if i == 1 or i == N then
8:          cd(1) ← T;
9:          cd(N) ← T;
10:      else
11:         m ← (s(i + 1) + s(i − 1))/2;
12:         d ← s(i + 1) − s(i − 1);
13:         r ← (s(i) − m)/(d/2);
14:         cd(i) ← cos(r × pi/2) × d/(s(N) − s(1));
15:      end if
16:   end for
17:   CD ← CD + cd(p, 1);
18: end for
19: end proc CosineCrowdingMeasurement
```

When assessing the crowding degree of a solution, cCM

calculates its crowding distance along each objective, and sums them to get the total result. In Algorithm 1, CD is the set of the total crowding distance values of all the solutions. T is the MOP objective number. $FV_{[t]}$ denotes the set of the objective value of each solution along the tth objective. Along the tth objective, for the ith solution, Line 11 determines the central location of the interval constructed by two adjacent solutions; Line 12 measures the interval length; Line 13 gives the degree that the ith solution drifts from the central location, and Line 14 calculates the crowding distance. From the calculation, we can see that cCM sufficiently involves the distance and location information to assess the crowding degree. It is very helpful to obtain the solutions with good uniformity and diversity. Algorithm 1 only gives the main procedures of cCM, and the other details can be found in Ref. [14].

B. CNDS Procedures

In the nondominated solution selection method in NSGA-II, the crowding distance of a solution only reflects its crowding status with the solutions in the same rank, which does not consider the crowding degree from a perspective of the whole population. It is not beneficial to obtain the final uniformly distributed nondominated solutions. To overcome this shortcoming, we propose an improved mechanism in CNDS. The procedures of selecting N solutions from P, a population with size $2N$, are as follows:

Step 1 Assign a rank for each solution in population P according to the rank assignment method in NSGA-II, and the nondominated solutions are set in the first rank.

Step 2 Construct an auxiliary empty population P'; add the solutions in rank 1 into P', and measure the size of P' as N'. If $N' \geq N$, terminate the addition and output P'; otherwise, continue to add the solutions of rank 2 into P' and start another size measurement. Perform such additions repeatedly until $N' \geq N$.

Step 3 Judge whether $N' > N$, if yes, repeat the following deletions until $N' == N$; otherwise, stop the operation, and obtain the final population.

- regard the solutions in population P' as a whole to calculate their crowding distances using the above cosine crowding measurement method;
- delete the solution with the highest rank and the minimum crowding degree, and obtain a new P'.

From the procedures, we can see that CNDS can preserve the solutions in low ranks; and it maintains the diversity of the selected solutions from the perspective of the whole population by deleting the worse solutions one by one; besides, it can obtain the uniformly distributed solutions via cosine crowding measurement. According to our analysis, CNDS is well capable of selecting desired solutions.

IV. Algorithm Framework of CLUMOEA

Integrating the CDTMS and CNDS mechanisms, we give the procedures of CLUMOEA in this section as follows.

Step 1 Randomly initialize a population $P = (x_1, x_2, \ldots, x_N)$, and set the generation counter $g = 0$.

Step 2 If $mod(g, G) = 0$, initialize the cluster centers $m_1, m_2, \ldots, m_K$.

Step 3 Based on the cluster centers defined, apply CDTMS and DE operations to each solution of population P, generate a new population P_o, and update the cluster centers $m_1, m_2, \ldots, m_K$.

Step 4 Perform a polynomial mutation [19] on P_o.

Step 5 Combine P and P_o to get a union population P_u.

Step 6 Select N solutions from P_u by CNDS to form the population of next generation P.

Step 7 If g is less than $maxGens$, set $g = g + 1$, and go to **Step 2**; otherwise, terminate and output the final population P as the approximation of PF.

We provide more details of simulations as follows.

- In **Step 1**, for a MOP with two and three objectives, the population size N is defined as 100 and 300, respectively.
- In **Step 2**, the cluster centers are re-initialized every G generations, and $G = 20$. It is aimed at prevent the clustering process trapped in local optima. The cluster number used in CLUMOEA is $K = 10$.
- In **Step 3**, two control parameters of DE are $F = 0.5$ and $CR = 1$, respectively.
- In **Step 4**, the mutation probability is $P_m = 1/d$, and d is the variable dimension of the MOPs to be solved.
- In **Step 7**, $maxGens$ denotes the maximum evolutionary generations, and $maxGens = 1000$.

V. Comparison with Other MOEAs

This section compares CLUMOEA with other MOEAs to assess its performance. The comparison algorithms chosen are MOEA/D-DE [9] and NSGA-II [5]. All the parameters of these two algorithms used here are the same as the original literatures except for the population sizes and maximum evolutionary generations. In order to have a fair comparison, the population sizes of these three MOEAs are all 100 for the MOPs with two objectives and 300 for the three objective MOPs. 1000 is set as the maximum evolutionary generations.

A. Test Suite

To show the advantages of CLUMOEA, we select seven test MOPs with irregular PS shapes from [15][16][17] to construct a test suite in Table II. All the problems are minimization problems.

B. Performance Metric

The performance of MOEAs for a problem is often evaluated from two aspects, convergence and diversity. we use the inverted generational distance (IGD) metric [18] to assess them. Let P^* be a set of uniformly distributed Pareto optimal

TABLE I. TEST MOPS WITH IRREGULAR PS SHAPES

MOPs	n	Objective functions	Variable bounds
MOP1	10	$\begin{cases} f_1(x) = (1+g)x_1 \\ f_2(x) = (1+g)(2 - x_1 - sign(\cos(2\pi x_1))) \end{cases}$ where $g = \sum_{i=2}^{n}(x_i - \cos(2\pi x_1 + \frac{i\pi}{10}))^2$.	$[0,1] \times [-1,1]^{n-1}$
MOP2	10	$\begin{cases} f_1(x) = (1+g)(1 - \cos(\frac{\pi x_1}{2})) \\ f_2(x) = (1+g)(10 - 10\sin(\frac{\pi x_1}{2})) \end{cases}$ it has the same g expression with MOP1	$[0,1] \times [-1,1]^{n-1}$
MOP3	10	$\begin{cases} f_1(x) = (1+g)x_1 \\ f_2(x) = (1+g)\begin{cases} 1 - 19x_1 & \text{if } x_1 \leq 0.05 \\ \frac{1}{19} - \frac{x_1}{19} & \text{else} \end{cases} \end{cases}$ it has the same g expression with MOP1	$[0,1] \times [-1,1]^{n-1}$
MOP4	10	$\begin{cases} f_1(x) = (1+g)x_1 \\ f_2(x) = (1+g)(2 - 2x_1^{0.5}\cos(3\pi x_1^2)^2) \end{cases}$ it has the same g expression with MOP1	$[0,1] \times [-1,1]^{n-1}$
MOP5	30	$\begin{cases} f_1(x) = x_1 \\ f_2(x) = 1 - \sqrt{\frac{f_1}{g}} - \frac{f_1}{g}\sin(10\pi f_1) \end{cases}$ where $g = 1 + 9\sum_{i=2}^{n} x_i$	$[0,1]^n$
MOP6	12	$\begin{cases} f_1(x) = \cos(\frac{\pi x_1}{2})\cos(\frac{\pi(1+2gx_2)}{4+4g})(1+g) \\ f_2(x) = \cos(\frac{\pi x_1}{2})\sin(\frac{\pi(1+2gx_2)}{4+4g})(1+g) \\ f_3(x) = \sin(\frac{\pi x_1}{2})(1+g) \end{cases}$ where $g = \sum_{i=3}^{n}(x_i - 0.5)^2$	$[0,1]^n$
MOP7	12	$\begin{cases} f_1(x) = \cos(\frac{\pi x_1}{2})\cos(\frac{\pi(1+2gx_2)}{4+4g})(1+g) \\ f_2(x) = \cos(\frac{\pi x_1}{2})\sin(\frac{\pi(1+2gx_2)}{4+4g})(1+g) \\ f_3(x) = \sin(\frac{\pi x_1}{2})(1+g) \end{cases}$ where $g = \sum_{i=3}^{n} x_i^{0.1}$	$[0,1]^n$

TABLE II. THE MEAN AND MINIMUM VALUES OF CLUMOEA, MOEA/D-DE AND NSGA-II IN 20 INDEPENDENT RUNS FOR EACH TEST PROBLEM

IGD-values	mean			minimum		
Problems	CLUMOEA	MOEA/D-DE	NSGA-II	CLUMOEA	MOEA/D-DE	NSGA-II
MOP1	0.002214	0.003713	0.702780	0.00216153	0.00371350	0.70278020
MOP2	0.033620	0.436478	5.001584	0.03334574	0.43647814	5.00158370
MOP3	0.005100	0.018697	0.239214	0.00509396	0.01869650	0.23921392
MOP4	0.005753	0.014467	0.925396	0.00572749	0.01446658	0.92539580
MOP5	0.005035	0.010665	0.005370	0.00473140	0.01066528	0.00536952
MOP6	0.001772	0.007349	0.001829	0.00158892	0.00734901	0.00182897
MOP7	0.001403	0.007191	0.142951	0.00140189	0.00719133	0.14295100

points in the true PF. Let P be an nondominated front of the problems. The IGD metric is defined as

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

where $d(v, P)$ is a minimum distance between v and any point in P, and $|P^*|$ is the cardinality of P^*. A lower IGD value is desirable. To get a low value, P must be close to the true PF, and cannot miss any part of the whole PF.

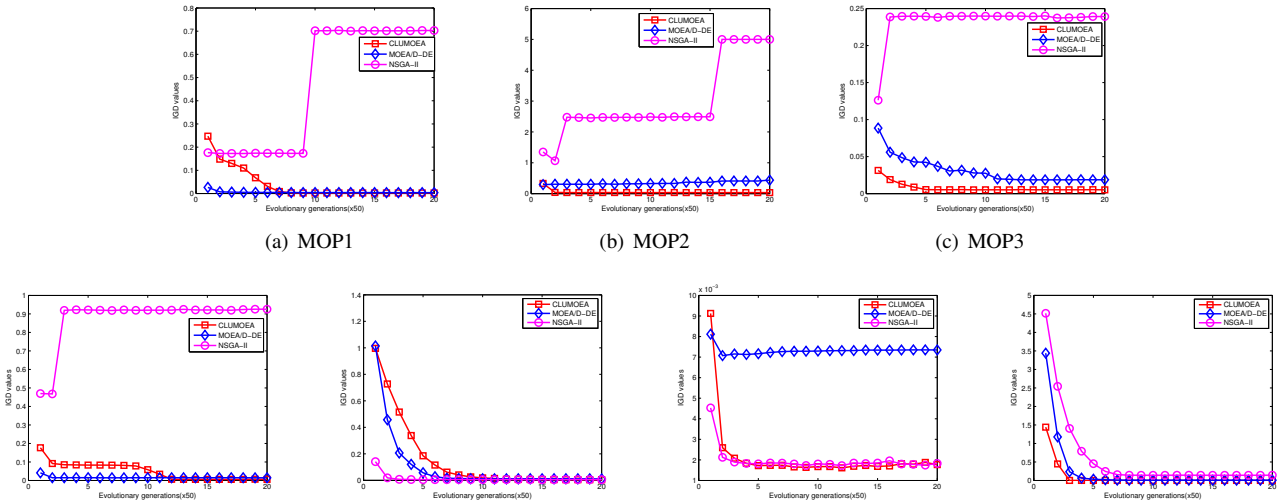
C. Results

To learn about the statistical performance of CLUMOEA, we run CLUMOEA, MOEA/D-DE, and NSGA-II to solve each test problem independently for 20 times and get 20 final IGD values. The mean and minimum values are shown in Table II. The values with dark gray and light gray backgrounds are the best and second best ones, respectively.

In order to observe the IGD variation along with the evolutions, we plot the variation curves in Fig.1. In the evolutionary processes, the IGD values are calculated every 50 generations, and since each algorithm has 20 independent runs for each test problem, in the curves, each point is the mean of these 20 metric values.

Among the 20 runs while the algorithms deal with each MOP, the final PF approximations with the lowest IGD value are also plotted in Fig. 2 and Fig. 3.

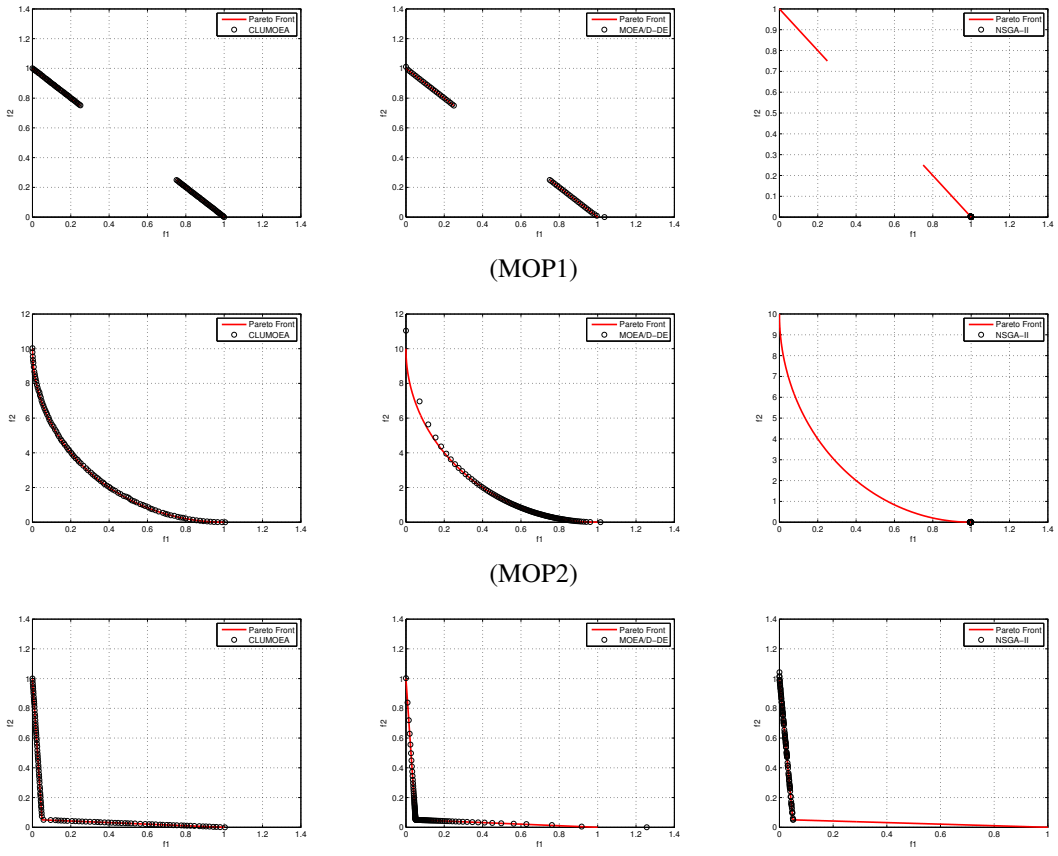
It is clear from Table II that, for every test problem, among



(a) MOP1 (b) MOP2 (c) MOP3

(d) MOP4 (e) MOP5 (f) MOP6 (g) MOP7

Fig. 1. The variation curves of IGD values in the processes that CLUMOEA, MOEA/D-DE and NSGA-II solve seven MOPs in 20 times.



(MOP1)

(MOP2)

(MOP3)

Fig. 2. The PF approximations with the lowest IGD values found by CLUMOEA, MOEA/D-DE and NSGA-II in 20 runs in the search space on MOP1-MOP3

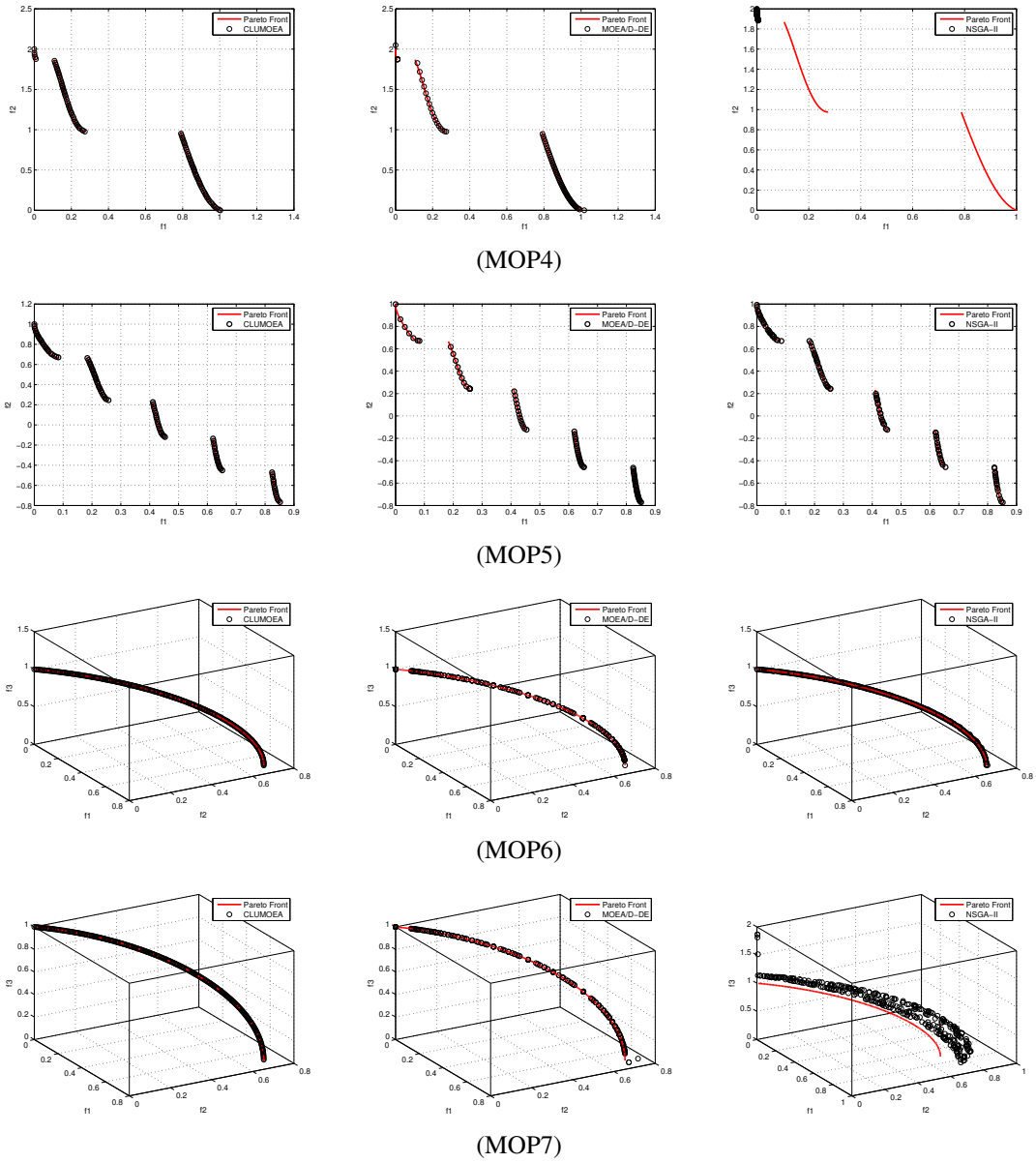


Fig. 3. The PF approximations with the lowest IGD values found by CLUMOEA, MOEA/D-DE and NSGA-II in 20 runs in the search space on MOP4-MOP7

the 20 IGD values, CLUMOEA always has the best mean and minimum values. Especially for problems MOP2, MOP3, MOP4, MOP7, the advantages of CLUMOEA are obvious. These results confirm that CLUMOEA is effective for the MOPs with irregular PS shapes than MOEA/D-DE and NSGA-II.

In Fig. 1, we can see that the IGD values of CLUMOEA decrease rapidly to reach the locations lower than those of MOEA/D-DE and NSGA-II; it indicates that, in the evolutionary processes, CLUMOEA does not fall into the local optimal points, it promotes the populations to approximate the PFs and to get the final populations with good convergence and diversity efficiently. The above phenomenon reveals that CLUMOEA has a rapid convergence and is helpful to balance the exploration and exploitation. Regarding the IGD curves of NSGA-II, for MOP1, MOP2, MOP3, MOP4, NSGA-II seems to get stuck in the premature convergence as shown in Fig. 2.

Figure. 2 and 3 show that, compared with MOEA/D-DE and NSGA-II, along with the PFs of the test problems, the final populations obtained by CLUMOEA spread more widely and uniformly.

VI. Conclusions

This paper has proposed a clustering based multiobjective optimization evolutionary algorithm, called CLUMOEA, to deal with the multiobjective optimization problems with irregular Pareto front shapes. ClUMOEA is a dominance based multiobjective evolutionary algorithm. The idea is using the machine learning techniques to guide its search. In CLUMOEA, a k-means cluster method is applied to partition the population of the MOEA into several clusters, and the solutions in the same cluster are allowed to perform the reproduction; in

the evolutionary process, CLUMOEA evolves one time and the k-means algorithm also performs only an iteration to reduce the computational cost and maintain the balance between exploration and exploitation. In addition to the clustering method, CLUMOEA also uses a distance tournament selection method to choose the more similar mating solutions to accelerate convergence, and a cosine nondominated selection method including the location and distance information of the solutions is further presented to select the final solutions with good diversity. The experimental results suggested that, compared with other algorithms, CLUMOEA has a faster convergence speed, and the obtained final solutions distribute more widely and uniformly.

The work aims to provide a new approach of utilizing machine learning to guide the search in MOEAs. In the future work, the k-means cluster method can be replaced by other machine learning algorithms to discover the structures of the populations of MOEAs, and the proposed algorithm needs to be applied into the test problems with more complicated PS and PF shapes to verify its performance. Besides, other machine learning strategies of improving the algorithm performance are also worth studying.

ACKNOWLEDGMENT

This work is supported by the National Basic Research Program of China (Grant No. 2012CB821205), the Foundation for Creative Research Groups of the National Natural Science Foundation of China (Grant No. 61021002), the National Natural Science Foundation of China (Grant No. 61174037, 51275274 and 61273313), and the Innovation Funds of China Academy of Space Technology (Grant No. CAST20120602).

REFERENCES

[1] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms.* Wiley, Chichester, UK, 2001.

[2] K.M. Miettinen, *Nonlinear Multiobjective Optimization.* Kluwer Academic Publishers, AA Dordrecht, Netherlands, 1999.

[3] M. Kim, T. Hiroyasu, M. Miki, S. Watanabe, SPEA2+: "Improving the Performance of the Strength Pareto Evolutionary Algorithm 2," *Parallel problem solving from nature-PPSN VIII*, pp. 742-751, Springer, Germany, 2004.

[4] J. Knowles, D. Corne, "The pareto archived evolution strategy: A new baseline algorithm for pareto multiobjective optimisation," *Proceedings of the 1999 Congress on Evolutionary Computation (CEC-1999)*, vol. 1, IEEE Computer Society, USA, 1999.

[5] K. Deb, A. Pratap, S. Agarwal, et al., "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182-197, Apr. 2002.

[6] H. Zhang, S.M. Song, A.M. Zhou, "MCGA: A Multiobjective Cellular Genetic Algorithm based on a 3D Grid," *Proceedings of the 14th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL'2013)*, pp. 453-462, Springer, Heidelberg, Germany, 2013.

[7] M. Basseur, E. Zitzler, "Handling uncertainty in indicator-based multiobjective optimization," *International Journal of Computational Intelligence Research*, vol. 2, no. 3, pp. 255-272, Apr. 2006.

[8] Q.F. Zhang, H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712-731, Dec. 2007.

[9] H. Li, Q.F. Zhang, "Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284-302, Apr. 2009.

[10] D.E. Goldberg, *Genetic algorithms in search, optimization, and machine learning.* Addison-Wesley, Massachusetts, USA, 2001.

[11] A.M. Zhou, Q.X. Liao, G.X. Zhang "A clustering based mate selection for evolutionary algorithms," *Neurocomputing*, under review, Oct. 2013.

[12] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, vol. 1, pp. 281-291, University of California Press, USA 1967.

[13] A.K. Qin, V.L. Huang, and P.N. Suganthan, "Differential evolution algorithm with atrategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284-302, Apr. 2009.

[14] H. Zhang, S.M. Song, and A.M. Zhou, "A multiobjective cellular genetic algorithm based on 3D structure and cosine crowding measurement," *International Journal of Machine Learning and Cybernetics*, under review.

[15] F.Q. Gu, H.L. Liu, K.C. Tan, "A multiobjective evolutionary algorithm using dynamic weight design method," *International Journal of Innovative Computing, Information and Control*, vol. 8, no. 5(B), pp. 3677-3688, May 2012.

[16] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, "Scalable multi-objective optimization test problems," *Proceedings of the Congress on Evolutionary Computation (CEC-2002)*, pp. 825-830, IEEE Computer Society, USA, 2002.

[17] E. Zitzler, K. Deb, L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173-195, 2000.

[18] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, V.G. Da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117-132, Apr. 2003.

[19] K. Deb and R.B. Agrawal, "Simulated binary crossover for continuous search space," *Complex System*, vol. 9, no. 2, pp. 115-148, Apr. 1995.