JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 33, 37-50 (2017)
DOI: 10.6688/JISE.2017.33.1.3

An Enhanced Multi-Objective Group Search Optimizer Based on Multi-producer and Crossover Operator

XIANG-WEI ZHENG[1,2], XIAO-MEI YU[1,2], YAN LI[1,2] AND HONG LIU[1,2]
[1]*School of Information Science and Engineering*
Shandong Normal University
Ji'nan, 250014 P.R. China
[2]*Shandong Provincial Key Laboratory for Distributed Computer Software Novel Technology*
Ji'nan, 250014 P.R. China
E-mail: xwzhengcn@163.com

In order to enhance the convergence ability of multi-objective group search optimizer and improve solution distribution of non-dominated Pareto set, we put forward a novel multi-objective group search optimizer based on multiple producers and crossover operator of genetic algorithm (MCGSO) in this paper. The producer of MCGSO is extended from one to multiple ones, which explores more solutions and improves solution distribution of non-dominated Pareto set. For the purpose of preventing a local optimal solution, the metropolis rule of simulation annealing algorithm is introduced into the search pattern of producers. Rangers' search strategies and crossover operator are combined to enhance algorithm's ability to find new solutions and expand the range of non-dominated optimal set. Experimental results on DZTL serial benchmark functions demonstrate that MCGSO can effectively and efficiently solve multi-objective optimization problems compared with other similar multi-objective evolutionary algorithms.

Keywords: multi-objective optimization, group search optimizer, multi-producer, crossover operator, Pareto set

1. INTRODUCTION

Many metaheuristics have been applied to solve Multi-objective Optimization Problems (MOPs) in engineering fields and scientific researches recently [1]. With the development of evolutionary algorithms, Schaffer creatively applied genetic algorithm to MOPs and proposed the Vector Evaluated Genetic Algorithm (VEGA) [2] in 1985. Due to numerous advantages of evolutionary algorithms such as non-demand on derivation and other auxiliary knowledge, generating multiple solutions at a runtime, easiness to be implemented and so on, evolutionary algorithms become effective candidate methods for MOPs [3]. A series of multi-objective algorithm such as Multi-objective Genetic Algorithms (MOGA) [4], Strength Pareto Evolutionary Algorithm 2 (SPEA2) [5], Non-dominated Sorting Genetic Algorithm II (NSGA-II) [6], Multi-objective Particle Swarm Optimizer (MOPSO) [7] have been proposed since then.

Recently, He, Wu and Saunders proposed a swarm intelligence optimization algorithm named Group Search Optimizer (GSO), which simulates animal's foraging behavior and is formulated based on producer-scrounger model [8]. This algorithm has demonstrated good performance in function optimization for high-dimensional optimization problems and possesses obvious superiority to solve complex engineering problems.

Received November 27, 2015; revised February 2, 2016; accepted March 11, 2016.
Communicated by Xiaohong Jiang.

Several multi-objective GSOs have been proposed in last few years and have obtained certain progress, but the disadvantages such as low convergence precision and a small distribution range of problem solutions still exist.

To overcome the above disadvantages, we put forward an enhanced multi-objective group search optimizer based on multiple producers and crossover operator of genetic algorithm (MCGSO). The improvements include the following. Firstly, we extend the number of producers from one to multiple and the metropolis rule is introduced into the search pattern of producers. Secondly, the crossover operator of genetic algorithm is combined with the rangers' search strategies. At last, multi-objective GSO with single producer (MGSO) [8] and NSGA-II [6] are compared with the proposed MCGSO. The experimental results show the effectiveness of the proposed MCGSO on DZTL benchmark functions.

The remainder of this paper is organized as follows. Section 2 introduces GSO and reviews related work of multi-objective GSO. Section 3 describes the proposed MCGSO in detail, including improved strategy of multi-producer, combination of crossover operator with ranger's search pattern and archive of Pareto optimal set. Section 4 presents experimental results and discussion of MCGSO on DZTL benchmark functions. Section 5 concludes the paper and points out future work.

2. GSO AND RELATED WORK

2.1 GSO

GSO is a novel intelligent optimization algorithm inspired from animal's foraging behavior and is formulated based on producer-scrounger model. Members in GSO are divided into three categories: producer, scroungers and rangers. Among them, the producer is responsible for finding resources in each iteration, while the scroungers move toward the producer to share its information. In order to maintain the diversity of group and avoid being entrapped in local optimum, GSO adopts a wandering strategy, in which rangers perform random walks to explore the whole search space.

In the n-dimensional search space, the position of i^{th} member at k^{th} iteration is expressed as $X_i^k \in R^n$, while the member has a head angle $\varphi_i^k=(\varphi_{i1}^k, \ldots, \varphi_{i(n-1)}^k) \in R^{n-1}$ and a head direction $D_i^k(\varphi_{i1}^k)=(d_{i1}^k, \ldots, d_{in}^k) \in R^n$ that can be calculated from the head angle via a polar to Cartesian coordinate transformation as defined in Eq. (1).

$$\begin{cases} d_{i1}^k = \prod_{p=1}^{n-1} \cos(\varphi_{ip}^k) \\ d_{ij}^k = \sin(\varphi_{i(j-1)}^k) \cdot \prod_{p=i}^{n-1} \cos(\varphi_{ip}^k) \\ d_{in}^k = \sin(\varphi_{i(n-1)}^k) \end{cases} \tag{1}$$

For limited space, member behaviors are briefly reviewed in the following and detailed introduction to GSO can be found in [8].

(A) Producer's behavior

The producer's behavior can be summarized as follows.

First, the producer will begin to scan from 0°angle and randomly sample three points in the scanning field: one point at 0°, one point in the right side of the hypercube and the other one in the left side of the hypercube, which can be described in Eq. (2).

$$\begin{cases} X_z = X_p^k + r_1 l_{\max} D_p^k(\varphi^k) \\ X_r = X_p^k + r_1 l_{\max} D_p^k(\varphi^k + r_2 \theta_{\max/2}) \\ X_l = X_p^k + r_1 l_{\max} D_p^k(\varphi^k - r_2 \theta_{\max/2}) \end{cases} \tag{2}$$

Where $r_1 \in R^1$ is a normally distributed random value with mean 0 and standard deviation 1, and $r_2 \in R^{n-1}$ is a uniformly distributed random value in the range (0, 1).

Second, if the producer finds a better position among the three points, it will move to that location; otherwise, it will return to the previous position and turn its head to a new angle as defined in Eq. (3).

$$\varphi^{k+1} = \varphi^k + r_2 \alpha_{\max} \tag{3}$$

Where $\alpha_{\max}$ is the maximum turning angle.

At last, if the producer can't find a better position after α iterations, it will turn back to original angle as defined in Eq. (4).

$$\varphi^{k+\alpha} = \varphi^k \tag{4}$$

Where α is a user-defined constant.

(B) Scrounger's behavior

In each iteration, the scroungers follow the producer in a random step according to Eq. (5).

$$X_i^{k+1} = X_i^k + r_3(X_p^k - X_i^k) \tag{5}$$

Where $r_3 \in R^n$ is a uniformly distributed random value in the range (0,1).

(C) Ranger's behavior

In each iteration, rangers move to a new point based on a random head angle and a random distance as defined in Eq. (6).

$$\begin{cases} \varphi^{k+1} = \varphi^k + r_2 \alpha_{\max} \\ l_i = a \cdot r_1 l_{\max} \\ X_i^{k+1} = X_i^k + l_i D_i^k(\varphi^{k+1}) \end{cases} \tag{6}$$

2.2 Related Work

The original GSO has been studied and improved with different strategies, including GSO member update and parameter optimization [9-11], co-evolutionary GSO [15, 16], hybrid GSO algorithms [17, 18], applications to complex practical problems [19, 20] and so on.

Furthermore, GSO is applied to solve multi-objective optimization. Wang, Zhong *et al*. proposed a novel multi-objective group search optimizer named NMGSO [12] for solving multi-objective optimization problems. The scanning strategy of the original GSO is replaced by the limited pattern search procedure but a special mutation with a controlling probability is designed to balance the exploration and exploitation at different searching stages. Wu, Lu *et al*. formulated reactive power dispatch incorporating flexible AC transmission system (FACTS) devices as a nonlinear constrained multi-objective optimization problem [13]. A group search optimizer with multiple producers (GSOMP) was designed and applied to minimize the real power loss and improve voltage profile. Guo, Zhan *et al*. proposed a group search optimizer with multiple producers (GSOMP) to deal with the multi-objective dynamic economic emission dispatch (DEED) of power systems [14]. A technique for order preference similar to an ideal solution (TOPSIS) is developed to determine the final solution by considering a decision maker's preference.

In summary, several conclusions can be drawn as follows:

(1) GSO demonstrates its advantages in high-dimensional function optimization. However, GSO has its disadvantages such as slow convergence and entrapment in local optima. Therefore, GSO is still worthy of being improved with new strategies.
(2) GSO simulates swarm behavior and is evolved with predator-scrounger model. The group members are classified as producer, scroungers and rangers, which is similar to multi-population structure. However, GSO can't efficiently explore problem space in certain cases for the fact that all scroungers follow the single producer.
(3) Related work on GSO mostly deals with single objective optimization problems. However, only a few works touch on multi-objective optimization problems and generally solves two-objective MOPs.

Therefore, we are inspired to develop a multi-objective group search optimizer based on multi-producer and crossover operator to improve its performance and solve multi-objective optimization problems.

3. A MULTI-OBJECTIVE GROUP SEARCH OPTIMIZER BASED ON MULTI-PRODUCER AND CROSSOVER OPERATOR（MCGSO）

In this section, some key components of MCGSO are presented firstly and then algorithmic procedure is described in detail.

3.1 Multi-Producer and Metropolis Rule

In the existing multi-objective group search optimizer based on single producer, the Pareto solution with the largest crowding distance is generally selected as the producer and all the scroungers move to the only producer. Therefore, the single producer is bound to affect the convergence speed and the diversity of solution sets. However, for MOPs, the Pareto solutions constitute a non-domination set and are not a single solution. The producer can be extended to more than one. Search strategy with multiple producers enables the algorithm to simultaneously search in multiple directions, and achieves the best solution in less iterations. This can also avoid excessive concentration of solutions

in certain extent. In order to guarantee the convergence precision and the diversity of solution, the non-dominated set will be sorted according to their crowding distance. The former members of *top*% will be chosen as producers of the current iteration, and the scroungers will randomly choose a producer to follow according to Eq. (5).

In fact, once the producer moves to a wrong position, the algorithm will be easily trapped into a local optimum. In our study, the metropolis rule of simulation annealing (SA) is introduced. This method can lead the algorithm to effectively jump out of local optimal [9]. In the k^{th} iteration of multi-objective group search algorithm with metropolis rule, each producer will generate three new solutions x_{i1}^k, x_{i2}^k, x_{i3}^k after three times of search in different directions according to Eq. (2). If a solution x_{im}^k, $m \in (1,2,3)$ can dominate the current solution, the producer will move to x_{im}^k; otherwise, after calculating the optimal increment $\Delta t = f(x_{im}^k) - f(x_i^k)$, the producer will accept x_{im}^k as a new solution according to a probability of $\exp(-\Delta t / T)$, which is defined in Eq. (7).

$$X_i^{k+1} = \begin{cases} X_{im}^k, if\ (\exp(\dfrac{f_j(X_i^k) - f_j(X_{im}^k)}{T}) > random) \\ \qquad\qquad .or.(f_j(X_i^k) > f_j(X_{im}^k)) \\ X_i^k, otherwise \end{cases} \tag{7}$$

Where, T is the total number of iterations, and j is the number of objective functions, j=1, 2, 3, ⋯ .

In summary, the update strategy of multi-producer can be described as follows:

For (each producer x_i^k in k^{th} iteration) {
  //i=1, …, N and N is group size, k is current generation
  (1) The producer moves according to Eq. (2) and three new solutions x_{i1}^k, x_{i2}^k, x_{i3}^k are generated;
  (2) **If** (x_{im}^k (m=1, 2, 3) dominates x_i^k) {
    If there is only one x_{im}^k dominating x_i^k {
      x_i^k is replaced with to x_{im}^k;
    Else
      Randomly selects a new solution x_{im}^k to replace x_i^k;
    }// end If
  ElseIf (all the objective functions of x_{im}^k are inferior to x_i^k and $\exp(-\Delta t / T) >$ *random* is satisfied)
    Randomly select a new solution x_{im}^k to replace x_i^k;
  }//end If
  If a better solution can't be found after α iterations, the producer moves back to the original position, and turn its head to a new angle according to Eq. (3);
}//end For

3.2 Ranger Search Combining with Crossover Operator

In GSO, the fact that most members (scroungers) move to the minority (only one producer) leads GSO to concentrate only on part of solutions. For some complex prob-

lems, it is difficult to find ideal optimal solutions. Therefore, crossover operator of genetic algorithm is introduced to multi-objective group search optimizer, which ensures the diversity of non-dominated set in solution space.

For each ranger, crossover operator is executed with the probability R_{ran} while the original updating strategy is executed with probability $(1-R_{ran})$. The update strategy of rangers is described as follows:

For (each ranger x_i^k in k^{th} iteration) {
(1)Generate a random value r in (0,1);
(2)**If** ($r > R_{ran}$) {
Randomly select two members represented as p_{i1}^k, p_{i2}^k from producers which act as the parent individuals of the crossover operator;
The parent individuals produce offspring according to Eq. (8), and two children individuals are generated and represented as c_{i1}^k, c_{i2}^k;

$$\begin{cases} c_{i1}^k = \frac{1}{2}[(1-\beta)\mathrm{p}_{i1}^k + (1+\beta)\mathrm{p}_{i2}^k] \\ c_{i2}^k = \frac{1}{2}[(1+\beta)\mathrm{p}_{i1}^k + (1-\beta)\mathrm{p}_{i2}^k] \end{cases} \tag{8}$$

Here, β is calculated according to Eq. (9).

$$\beta(\mathrm{u}) = \begin{cases} (2\,\mathrm{u})^{\frac{1}{\eta_c+1}}, if\ (\mathrm{u} \le 0.5) \\ (2(1-\mathrm{u}))^{\frac{1}{\eta_c+1}}, otherwise \end{cases} \tag{9}$$

Where u is a random value in (0, 1), and η_c is the distribution index for crossover operator;
Calculate fitness value of c_{i1}^k, c_{i2}^k on each objective function;
Select a better one as a new solution, otherwise randomly select one if they are non-dominant to each other;
Else
Update x_i^k according to Eq. (6);
}//end If
}//end For

3.3 Archive Strategy

By referencing the elitist strategy of NSGA-II, update procedure of MCGSO external archive is formulated as follows:

(1) Combine new population Q_k of k^{th} generation with its parent population P_k to R^k, and the size of R^k is $2N$.
(2) Sort R^k based on non-domination, generate non-dominated sets F_1, F_2, …, F_j, … and then calculate the crowding distance of each member. Sort each set F_j by their crowding distance.

(3) Put F_1 into the new parent population P_{k+1}.
(4) If the size of P_{k+1} is less than N, put next level of non-dominated set F_2, F_3, …F_j into P_{k+1} consistently until the size of P_{k+1} exceed N. Execute crowding-comparison operator for F_j, and put the first $\{|F_j|-(|P_{k+1}|-N)\}$ individuals into P_{k+1} until the size of P_{k+1} is equal to N. The new parent population P_{k+1} is eventually formed.

3.4 Description of MCGSO

The algorithmic procedure of MCGSO is described as follows.

(1) Population initialization. Initialize position x_i^1 and angle φ_i^1 randomly for each member, and the population is initialized as R^1. Calculate all the objective function values for all the members. k=1.
(2) j=1, P_k=∅. Sort R^k based on non-domination, generate non-dominated sets F_1, F_2, …, F_j, … and then calculate crowding distance of each member. Sort each set F_j by their crowding distance.
(3) Calculate $|P^k|+|F_j|$. If $|P^k|+|F_j|$ is greater than N, go to 5).
(4) $P^k=P^k \cup F_j$, $j=j+1$, go to 3).
(5) Put the former $\{|F_j|-(|P_k|-N)\}$ individuals of F_j into P_k.
(6) Sort P_k according to their crowding distance and choose the former members of *top*% as producers of this iteration.
(7) For each member x_i^k in P_k (i=1, …, N)
   i. If it's a producer, it will be updated according to producer update strategy described in Section 3.1;
   ii. If it's a scrounger, it will choose a producer at first, and then be updated according to Eq. (5);
   iii. If it's a ranger, it will be updated according to the steps described in Section 3.2.
   iv. After all the members are completely updated, new population Q_k is constructed.
(8) $k=k+1$. If k is greater than maximum number of iterations, output the result.
(9) Combine P_{k-1} and Q_{k-1} to construct R^k whose size is $2N$. Go to 2).

4. SIMULATION EXPERIMENTS AND DISCUSSION

4.1 Evaluation Metrics

In order to compare the performance of MCGSO with selected comparative algorithms, three frequently referenced metrics are adopted in this paper.

(A) Spacing (SP)

SP is used to measure the distribution of Pareto solution set [21]. It is defined in Eq. (10).

$$\begin{cases} SP = \sqrt{\dfrac{1}{n-1}\sum_{i=1}^{n}\left(\bar{d}-d_i\right)^2} \\ d_i = min_{j=1,2,\cdots,n}\left(\sum_{k=1}^{M}\left|f_k^i - f_k^j\right|\right) \\ \bar{d} = \dfrac{1}{n}\sum_{i=1}^{n} d_i \end{cases} \tag{10}$$

Where M is the number of objective functions and n is the number of final set of Pareto optimal solutions.

(B) Generational Distance (GD)

GD measures the distance between true Pareto front and approximate Pareto front obtained by optimization algorithm, which can evaluate convergence ability of an algorithm [21]. It is mathematically defined in Eq. (11).

$$GD = (\sqrt{\sum_{i=1}^{n} d_i^2}) / n \tag{11}$$

Where n is the number of final set of Pareto optimal solutions, and d_i is the Euclidean phenotypic distance between each member i of achieved Pareto front and its closest neighbor member in true Pareto front.

(C) Maximum Spread (MS)

MS can embody the coverage of Pareto front considering the extreme solution for the situation [21]. Larger value of MS shows that the Pareto front achieved by the algorithm is widely distributed. It is mathematically defined in Eq. (12).

$$\sqrt{\frac{1}{M}\sum_{i=1}^{M}\left(\frac{\min(f_i^{\max}, f_{itrue}^{\max}) - \max(f_i^{\min}, f_{itrue}^{\min})}{f_{itrue}^{\max} - f_{itrue}^{\min}}\right)} \tag{12}$$

Where $f_i^{\max}$ and $f_i^{\min}$ are the maximum and minimum values of i^{th} objective function for appropriate Pareto set and $f_{itrue}^{\max}$, $f_{itrue}^{\min}$ are those for the true Pareto set.

4.2 Experimental Result and Discussion

In this paper, three-objective DZTL are selected as benchmark functions for comparative algorithms and the details of DZTL can be found in [22]. To validate the effectiveness of multi-objective group search algorithm in solving MOPs, we compared the MCGSO with MGSO [8] and NSGA-II [6] on benchmark functions.

Main experimental parameters are listed in Table 1 and are not especially tuned for each benchmark functions. All the benchmark functions employed identical ones. Therefore, some experimental results are not prominent due to the selected parameters. Intensive tests on DZTL benchmark functions were conducted in Matlab R2010b setting. Each algorithm is independently run 30 times.

Table 2 summarizes experimental results of SP, GD and MS on DZTL1~ DZTL7 benchmark functions. The best numerical results are in bold. Sub-graph (a) of Figs. 1-7 show comparison of the Parent front achieved by MCGSO and true Pareto front. Sub-graph (b) of Figs. 1-7 illustrate changes of GD for MCGSO, MGSO and NSGA –II. Sub-graph (c) of Figs. 1-7 are boxplot of GD for three algorithms. In the boxplots, the bottom and top of the box are the first and third quartiles. The middle horizontal lines are the median of samples while the upper and lower dotted lines represent the maximum and minimum samples respectively [19].

For DZTL1, the Pareto front achieved by MCGSO is close to true Pareto front. The trend can be analyzed from GD curves in sub-graph (b) of Fig. 1. The GD of MCGSO is

similar to NSGA II but smaller than MCGSO. However, MCGSO is not steady as MGSO and NSGA II as shown in sub-graph (c) of Fig. 1.

For DZTL2, the Pareto front achieved by MCGSO is approximate to true Pareto front except three separated points. The GD curve of MCGSO in sub-graph (b) of Fig. 2 approaches to zero and is smaller than NSGA II. In addition, MCGSO is steady as NSGA II as shown in sub-graph (c) of Fig. 2.

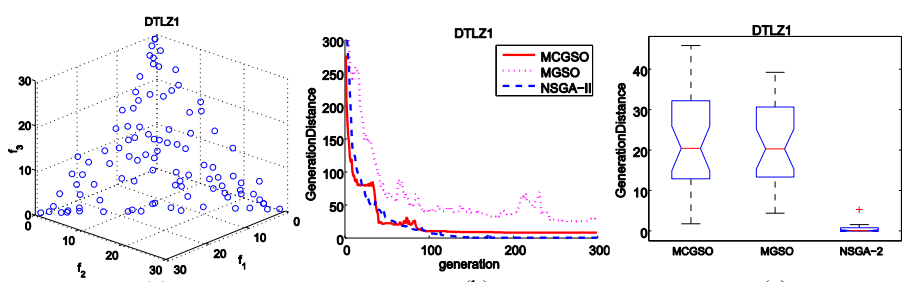
For DZTL3 and DZTL4, MCGSO doesn't show expected results and maybe it is not suitable to this kind of multi-objective problem. The Pareto front achieved by MCGSO is not identical to true Pareto front. The GD curves of MCGSO in sub-graph (b) of Figs. 3 and 4 are between MGSO and NSGA II.

Table 1. Main experimental parameters.

Algorithm	Parameter	Value
NSGA-II	crossover probability	0.9
	mutation probability	1/*n*
MGSO	Producer number	1
	Scrounger number	80%
MCGSO	Producer number	20%
	Scrounger number	60%
	Ranger crossover probability	0.5
All Algorithms	Population Size	100
	Generations	300

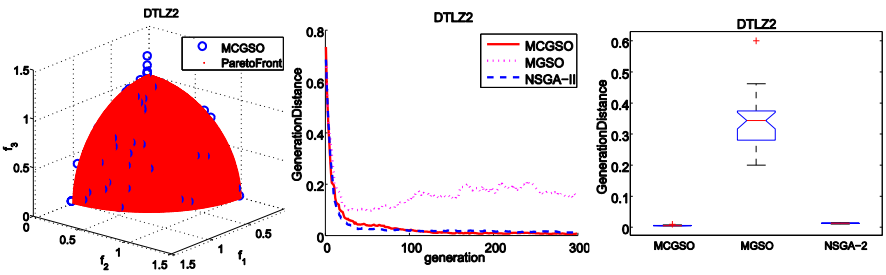
Table 2. Mean and variance of GD and SP.

Metrics	Algorithms	DTLZ1	DTLZ2	DTLZ3	DTLZ4	DTLZ5	DTLZ6	DTLZ7
GD	MCGSO	24.2124 (13.3229)	**0.0052 (0.0010)**	97.8983 (38.4990)	0.8268 (0.3716)	**0.0041 (0.0001)**	**0.0015 (0.0004)**	**0.0034 (0.0026)**
	MGSO	21.5240 (9.4403)	0.3385 (0.0785)	0.3434 **(0.1106)**	0.3343 (0.0795)	0.0931 (0.0478)	0.3725 (0.7040)	0.2998 (0.1037)
	NSGA-II	**0.4957 (1.0092)**	0.0126 (0.0011)	**0.0213** (6.1842)	**0.0060 (0.0011)**	0.0048 (0.0003)	0.0195 (0.0004)	(0.0034)
SP	MCGSO	**0.5888 (0.3101)**	**0.0431 (0.0020)**	3.9981 (1.8223)	0.0738 (0.0206)	**0.0070 (0.0003)**	**0.0070 (0.0002)**	**0.0211 (0.0120)**
	MGSO	0.9959 (0.4769)	0.0920 (0.0097)	**0.0954 (0.0231)**	0.0909 (0.0109)	0.0212 (0.0070)	0.0497 (0.1089)	0.0676 (0.0218)
	NSGA-II	0.6210 (3.0295)	0.0585 (0.0049)	8.0416 (10.7732)	**0.0565 (0.0041)**	0.0094 (0.0010)	0.0126 (0.0007)	0.0732 (0.0123)



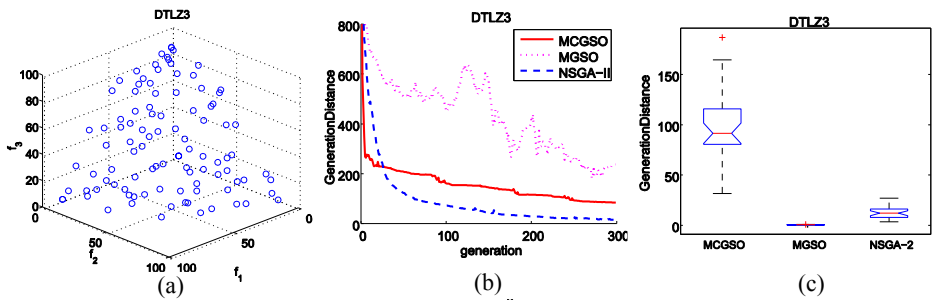
(a) (b) (c)

Fig. 1. Pareto fronts, GD curves and GD boxplots of DTLZ1.



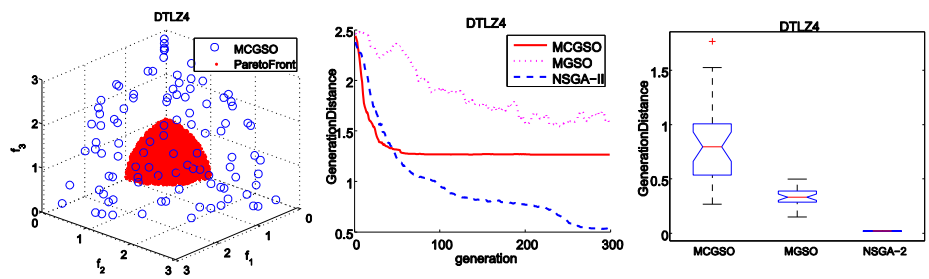
(a) (b) (c)

Fig. 2. Pareto fronts, GD curves and GD boxplots of DTLZ2.



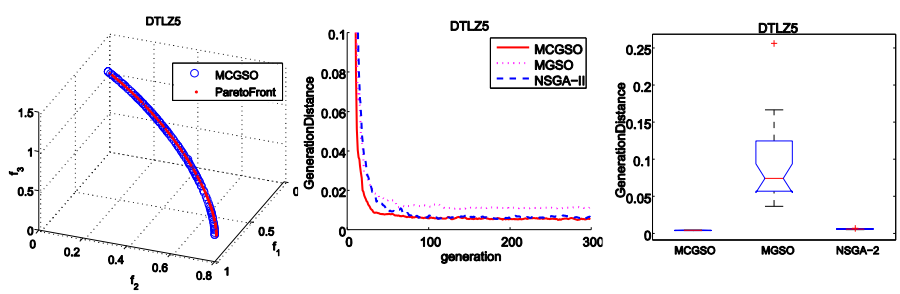
(a) (b) (c)

Fig. 3. Pareto fronts, GD curves and GD boxplots of DTLZ3.



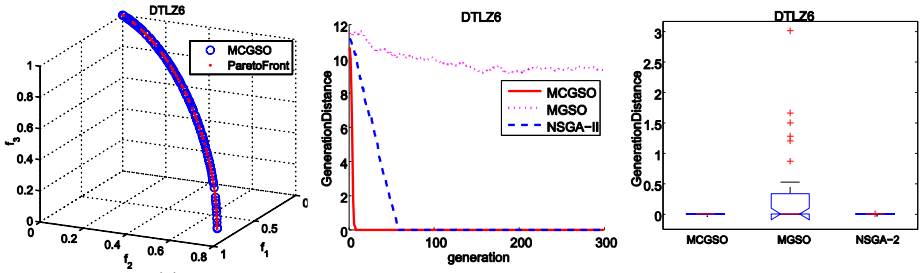
(a) (b) (c)

Fig. 4. Pareto fronts, GD curves and GD boxplots of DTLZ4.



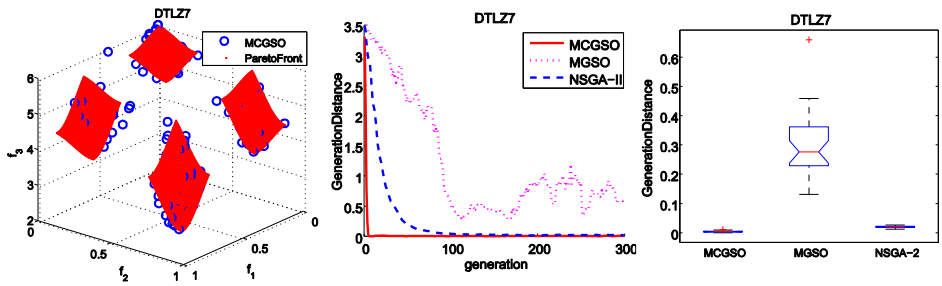
(a) (b) (c)

Fig. 5. Pareto fronts, GD curves and GD boxplots of DTLZ5.



(a) (b) (c)

Fig. 6. Pareto fronts, GD curves and GD boxplots of DTLZ6.



(a) (b) (c)

Fig. 7. Pareto fronts, GD curves and GD boxplots of DTLZ7.

For DZTL5, the Pareto front achieved by MCGSO is close to true Pareto front. GD curves in sub-graph (b) of Fig. 5 show that all algorithms have similar result. However, MGSO is not steady as MCGSO and NSGA II as shown in sub-graph (c) of Fig. 5.

For DZTL6 and DZTL7, MCGSO show similar performance. The Pareto front achieved by MCGSO is very close to true Pareto front. The GD curve of MCGSO in sub-graph (b) of Figs. 6 and 7 approaches to zero in fewer generations than NSGA II. And, MCGSO and NSGA II are steady as shown in sub-graph (c) of Figs. 6 and 7. MGSO shows worse results on GD and convergence.

In conclusion, as shown in Table 2 and Sub-graph (a) of Figs. 1-7, the Pareto front achieved by MCGSO is consistent with the true Pareto front. MCGSO outperforms MGSO and NSGA II on DTLZ2, DTLZ5~DTLZ7. In terms of the algorithmic convergence, MCGSO has the minimum value of GD than other algorithms. As shown in sub-graph (b) of Figs. 1-7, the convergence speed of MCGSO is significantly faster than other algorithms. However, the SP of MCGSO for DTLZ2 and DTLZ3 are worse than NSGA-II.

5. CONCLUSIONS AND FUTURE WORK

Considering the superiority of GSO in high dimensional function optimization, we studied how to apply GSO to solve MOPs in this paper. Therefore, we put forward an enhanced multi-objective group search optimizer based on multiple producers and crossover operator of genetic algorithm. The experiments on DTLZ benchmark functions demonstrate

the effectiveness of MCGSO on various multi-objective optimization problems.

However, MCGSO is still not utilized to solve practical applications. Future work will be continued. On the one hand, we attempt to formulate a multi-objective virtual network embedding model and plan to employ it as optimization algorithm. On the other hand, based on the tests on practical applications, MCGSO will be further improved to suit practical problems.

ACKNOWLEDGMENTS

We are grateful for the support of the National Natural Science Foundation of China (61373149, 61472232, 61402270) and Shandong Provincial Natural Science Foundation (ZR2014FQ009).

REFERENCES

1. G. R. Zavala, A. J. Nebro, and F. Luna, "A survey of multi-objective metaheuristics applied to structural optimization," *Structural and Multidisciplinary Optimization*, Vol. 49, 2014, pp. 537-558.
2. J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proceedings of the 1st International Conference on Genetic Algorithms*, 1985, pp. 93-100.
3. M. Reyes-Sierra and C. A. C. Coello, "Multi-objective particle swarm optimizers: A survey of the state-of-the-art," *International Journal of Computational Intelligence Research*, Vol. 2, 2006, pp. 287-308.
4. C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multiobjective optimization: formulation discussion and generalization," in *Proceedings of the 5th International Conference on Genetic Algorithms*, 1993, pp. 416-423.
5. E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, Vol. 3, 1999, pp. 257-271.
6. K. Deb, A. Pratap, and S. Agarwal, "A fast and elitist multi-objective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, Vol. 6, 2002, pp. 182-197.
7. S. Mohankrishna, D. Maheshwari, P. Satyanarayana, *et al*., "A comprehensive study of particle swarm based multi-objective optimization," in *Proceedings of International Conference on Information Systems Design and Intelligent Applications*, 2012, pp. 689-701.
8. S. He, Q. H. Wu, and J. R. Saunders, "Group search optimizer: an optimization algorithm inspired by animal searching behavior," *IEEE Transactions on Evolutionary Computation*, Vol. 13, 2009, pp. 973-990.
9. D. B. Chen, J. T. Wang, F. Zou, W. B. Hou, and C. X. Zhao, "An improved group search optimizer with operation of quantum-behaved swarm and its application," *Applied Soft Computing*, Vol. 12, 2012, pp. 712-725.
10. G. He, Z. Cui, and J. Zeng, "Group search optimizer with interactive dynamic neighborhood," *Artificial Intelligence and Computational Intelligence*, 2011, pp. 212-219.

11. X. Yan, W. Yang, and H. Shi, "A group search optimization based on improved small world and its application on neural network training in ammonia synthesis," *Neurocomputing*, Vol. 97, 2012, pp. 94-107.
12. L. Wang, X. Zhong, and M. Liu, "A novel group search optimizer for multi-objective optimization," *Expert Systems with Applications*, Vol. 39, 2012, pp. 2939-2946.
13. Q. H. Wu, Z. Lu, and M. S. Li, "Optimal placement of FACTS devices by a group search optimizer with multiple producer," *IEEE World Congress on Evolutionary Computation*, 2008, pp. 1033-1039.
14. C. X. Guo, J. P. Zhan, and Q. H. Wu, "Dynamic economic emission dispatch based on group search optimizer with multiple producers," *Electric Power Systems Research*, Vol. 86, 2012, pp. 8-16.
15. L. D. S. Pacifico and T. B. Ludermir, "Cooperative group search optimization," *IEEE Congress on Evolutionary Computation*, 2013, pp. 3299-3306.
16. H. Shen, Y. Zhu, and B. Niu, "An improved group search optimizer for mechanical design optimization problems," *Progress in Natural Science*, Vol. 19, 2009, pp. 91-97.
17. Q. Kang, T. Lan, and Y. Yan, "Group search optimizer based optimal location and capacity of distributed generations," *Neurocomputing*, Vol. 78, 2012, pp. 55-63.
18. L. Wang, X. Hu, and J. Ning, "A modified group search optimizer algorithm for high dimensional function optimization," *Information Computing and Applications*, 2012, pp. 219-226.
19. S. He, Q. H. Wu, and J. R.Saunders, "Breast cancer diagnosis using an artificial neural network trained by group search optimizer," *Transactions of the Institute of Measurement and Control*, Vol. 31, 2009, pp. 517-531.
20. M. M. Dalvand, B. M. Ivatloo, and A. Najafi, "Continuous quick group search optimizer for solving non-convex economic dispatch problems," *Electric Power Systems Research*, Vol. 93, 2012, pp. 93-105.
21. C. A. C. Coello, D. A. V. Veldhuizen, and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Springer, NY, 2007.
22. K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multiobjective optimization test problems," in *Proceedings of the Congress on Evolutionary Computation*, 2002, pp. 825-830.

Xiang-Wei Zheng (郑向伟) received the B.S. and M.S. degrees both in computer science from Shandong Normal University in 1995 and 1998 respectively. He received his Ph.D. in Management Science and Engineering from Shandong Normal University in 2008. He is currently a Professor in School of Information Science and Engineering, Shandong Normal University, China. His current research interests include computational intelligence, and computer networks. He has been involved in several national natural science foundation of China and is author of more than 30 national and international publications in conferences and journals.

Xiao-Mei Yu (于晓梅) received the B.S. and M.S. in Management Science and Engineering from Shandong Normal University in 1996 and 2004 respectively. Currently she is an Associate Professor in School of Information Science and Engineering in Shandong Normal University. Her research interests include data mining and big data.

Yan Li (李焱) received the B.S. and M.S. degrees both in Computer Science in Shandong Normal University. His research interests include computer animation, computational intelligence and data mining. Currently, he is a Lecturer in School of Information Science and Engineering, Shandong Normal University. He has been involved in several national natural science foundation of China and has published several papers in conferences and journals.

Hong Liu (刘弘) is currently the President of School of Information Science and Engineering, Shandong Normal University, Jinan, China. She is also the Director of Shandong provincial key laboratory for distributed computer software novel technology. She has contributed over 100 articles to professional journals. Her current research interests include distributed artificial intelligence, software engineering and computer aided design.