Multiple Parents Guided Differential Evolution for Large Scale Optimization

Qiang Yang, *Student Member, IEEE*, Han-Yu Xie, Wei-Neng Chen, *Member, IEEE*
and Jun Zhang, *Senior Member, IEEE*

***Abstract*—Large scale optimization has become an important and challenging area in evolutionary computation. To solve this kind of problems efficiently, this paper proposes a multiple parents guided differential evolution (MPGDE) algorithm. Instead of using only one parent to guide each individual in traditional DE variants, multiple top ranked parents are utilized to direct each individual to search the space. Since the failed parents or trial vectors may also contain useful information, we maintain an archive to preserve these failed individuals and utilize a niching method to update the archive during evolution. Combining the above together, we put forward a new mutation strategy for DE. Cooperated with existing self-adaptive strategies for parameters in DE, MPGDE can afford a good balance between exploration and exploitation, so that promising performance can be obtained. Extensive experiments are conducted on 20 CEC'2010 large scale benchmark functions with 1000 dimensions to verify the efficacy and effectiveness of the developed MPGDE in comparison with several state-of-the-art algorithms dealing with large scale problems.**

I. Introduction

Differential evolution (DE) has received plenty of attention since it was proposed by Storn and Price [2]. In recent years, numerous remarkable new DE variants [1],[3-17] have been developed and many researchers have applied DE to solve real-world problems [18],[19].

Although most existing DE variants have witnessed great success on the tested problems [1],[3-10], they are specially designed for low dimensional problems (smaller than 500 dimensions). When it comes to high dimensional problems (generally larger than 500 dimensions), most existing DE algorithms would drastically lose their efficiency and effectiveness [20]. This is mainly caused by the exponentially enlarged search space when the dimensionality increases. In addition, when the dimensionality becomes larger and larger, the number of local optima generally becomes larger and larger as well. Both situations greatly enhance the difficulty in locating the global optima for DE [17] as well as other traditional evolutionary algorithms (EAs) [21-27].

To deal with high dimensional (or large scale) problems efficiently, at present, a frequently used technique is cooperative coevolution (CC) [28]. CC adopts the so-called *divide-and-conquer* strategy to decompose problems into several low dimensional sub-problems and then evolves each sub-problem separately. Since Potter and Jong [28] proposed this promising framework, many variants have been developed by proposing different decomposition strategies [20],[29-33]. Since CC optimizes each sub-problem individually, one key step for CC to achieve satisfactory performance is the decomposition strategy, for which the ideal method should put interdependent variables into the same group and meanwhile place independent variables into different groups. So far, the most satisfactory grouping methods are differential grouping (DG) [33] and its extended version (XDG) [34].

Even though these CC variants are promising for large scale optimization, they encounter two limitations: 1) Generally, it would cost a lot of function evaluations to obtain the accurate grouping, such as in the worst cases, both DG and XDG need $O(D^2)$ (D is the dimension size) function evaluations. This huge consumption greatly reduces the number of function evaluations used for evolution, leading to unsatisfactory performance, especially faced with multimodal problems. 2) Evolving each sub-group individually usually needs a large number of function evaluations, especially when the number of groups is large. These two limitations restrict the wide application of CC in dealing with practical large scale problems.

To conquer the above limitations, from the perspective of traditional DE algorithms, we propose a multiple parents guided DE algorithm (MPGDE) to deal with large scale optimization, which evolves all dimensions simultaneously. Specifically, in MPGDE, a new mutation strategy is developed, where each individual is guided by multiple top ranked parents. To preserve high diversity, the failed parents and trial vectors are reserved in an archive, which is updated during the evolution process using a niching strategy. For further enhancement of diversity, the evolution path is also added to direct the evolving of individuals. Through these techniques, the exploration and exploitation of the proposed algorithm are both promoted, so that the wide search space of large scale problems may be fully explored and exploited.

To verify the efficiency and effectiveness of the developed MPGDE, extensive experiments are conducted on 20 CEC'2010 benchmark problems with 1000 dimensions in comparison with several state-of-the-art EAs dealing with large scale optimization.

The rest of this paper is organized as follows. In Section II, several related DE variants and some existing EAs coping with large scale optimization are reviewed. Section III elaborates the proposed MPGDE in details, following which

Qiang Yang, Wei-Neng Chen, and Jun Zhang are with School of Computer Science and Engineering, South China University of Technology, Guangzhou, China, 510006. Corresponding authors: Wei-Neng Chen (email: cwnraul634@aliyun.com) and Jun Zhang (email: junzhang@ieee.org)

Qiang Yang and Han-Yu Xie are also students with School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China, 510006.

This work was supported in part by the NSFC projects Nos. 61379061, 61332002, 6141101191, in part by Natural Science Foundation of Guangdong for Distinguished Young Scholars No. 2015A030306024, in part by the "Guangdong Special Support Program" No. 2014TQ01X550, and in part by the Guangzhou Pearl River New Star of Science and Technology No. 151700098.

978-1-5090-0623-6/16/$31.00 ©2016 IEEE

is the experimental studies in Section IV to verify the promising performance of MPGDE in dealing with large scale optimization. Finally, Section V concludes this paper.

II. Related Work

Without loss of generality, in this paper, we consider minimization problems, which can be formulated as follows:

$$\min f(\boldsymbol{x}) \quad \boldsymbol{x} \in \boldsymbol{R}^D \tag{1}$$

where $f(\boldsymbol{x})$ is the function to be minimized and $\boldsymbol{x}$ is the variable vector containing D dimensions. For large scale problems, generally, D is larger than 500. In this paper, the function value is considered as the fitness value of one individual.

A. Differential Evolution

Generally speaking, a classical DE algorithm contains three operators: 1) mutation operator, 2) crossover operator, and 3) selection operator.

1) Mutation Operator

During each generation, for each individual, this operator creates a mutation vector $\boldsymbol{v}$ based on the current population. So far, the most frequently used mutation strategies in the literature include the following five ones:

DE/rand/1:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{r_1,G} + F \cdot (\boldsymbol{x}_{r_2,G} - \boldsymbol{x}_{r_3,G}) \tag{2}$$

DE/best/1:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{\text{best},G} + F \cdot (\boldsymbol{x}_{r_2,G} - \boldsymbol{x}_{r_3,G}) \tag{3}$$

DE/current-to-best/1:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{i,G} + F \cdot (\boldsymbol{x}_{\text{best},G} - \boldsymbol{x}_{i,G}) + F \cdot (\boldsymbol{x}_{r_1,G} - \boldsymbol{x}_{r_2,G}) \tag{4}$$

DE/rand-to-best/1:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{r_1,G} + F \cdot (\boldsymbol{x}_{\text{best},G} - \boldsymbol{x}_{r_1,G}) + F \cdot (\boldsymbol{x}_{r_2,G} - \boldsymbol{x}_{r_3,G}) \tag{5}$$

DE/current-to-pbest/1:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{i,G} + F \cdot (\boldsymbol{x}^{p}_{best,G} - \boldsymbol{x}_{i,G}) + F \cdot (\boldsymbol{x}_{r_1,G} - \boldsymbol{x}_{r_2,G}) \tag{6}$$

where $\boldsymbol{v}_{i,G}$ is the mutation vector of the ith individual $\boldsymbol{x}_{i,G}$ at generation G; r_1, r_2 and r_3 are selected randomly within [1, NP] and they are mutually different from each other and also different from the index of the current individual i; $\boldsymbol{x}_{\text{best},G}$ is the global best individual found so far; $\boldsymbol{x}^{p}_{best,G}$ is a parent randomly selected from the top p best parents in the population and F is the mutation factor, which controls the step that the target vector moves.

2) Crossover Operator

After mutation, it comes to the crossover operator, which would build a trial vector or an offspring for each individual. In the literature, there are two crossover strategies: 1) the binomial crossover operation and 2) the exponential crossover operation. Compared with the latter, the former one is more frequently used, which can be formulated as:

$$u_{j,i,G} = \begin{cases} v_{j,i,G}, & \text{if } rand(0,1) \le CR \text{ or } j = j_{rand} \\ x_{j,i,G}, & \text{otherwise} \end{cases} \tag{7}$$

where $\boldsymbol{u}_{i,G}$ is the trial vector of the ith individual $\boldsymbol{x}_{i,G}$ at generation G; $\boldsymbol{v}_{i,G}$ is the mutation vector generated in the last operator; j_{rand} is an integer randomly generated from [1, D], which is used to ensure that at least one dimension of the trial vector $\boldsymbol{u}_{i,G}$ is different from $\boldsymbol{x}_{i,G}$; $rand(0,1)$ is a real number generator, which can uniformly generate a real number within [0,1]; and CR is the crossover rate, which controls the number of dimensions that the trial vector $\boldsymbol{u}_{i,G}$ will inherit from the mutation vector $\boldsymbol{v}_{i,G}$.

3) Selection Operator

Following crossover is the selection operator. For the ith individual $\boldsymbol{x}_{i,G}$, after its offspring $\boldsymbol{u}_{i,G}$ is generated, they compete with each other, and only the better one can be kept to enter the next generation. For a minimization problem, this process can be formulated as:

$$\boldsymbol{x}_{i,G+1} = \begin{cases} \boldsymbol{u}_{i,G}, & \text{if } f(\boldsymbol{u}_{i,G}) < f(\boldsymbol{x}_{i,G}) \\ \boldsymbol{x}_{i,G}, & \text{otherwise} \end{cases} \tag{8}$$

For DE algorithms, the above processes are iteratively executed, until the termination criterion is met.

In general, the main difference among different DE variants [6-9],[35],[36-38] lies in the mutation operator. Except for the contribution in developing new mutation operators, many researchers also made great contribution to adaptively or self-adaptively adjusting the parameters (F and CR) in DE, such as jDE [39], JADE [1], SaNSDE [40] and IDE [9].

However, most existing DE variants are specially designed for low-dimensional problems (less than 500 dimensions). When it comes to large scale problems (more than 500 dimensions), they would drastically lose their effectiveness owing to the exponentially increased search space.

B. DE Variants for Large Scale Optimization

To deal with large scale optimization efficiently, a frequently used technique is cooperative coevolution (CC) [28], which adopts *divide-and-conquer* to decompose problems into smaller sub-problems and then evolves each sub-problem separately. So far, most existing CC algorithms [20],[30-33] cooperate with DE to cope with large scale optimization. Instead of developing DE, these CC algorithms mainly put forward various grouping strategies, which are the key step for CC to achieve satisfactory performance.

At first, random grouping [20],[30] was proposed, which randomly divides the whole dimensions into several groups with a fixed group size. During each generation, this grouping is executed to create new groups, which can increase the probability of putting two interdependent variables into the same group. To further improve the performance of random grouping, Yang *et al*. [30] further let the group size dynamically chosen from a group size pool, leading to multi-level CC (MLCC). Utilizing the historical information, MLCC could make a proper choice about the group size during evolution. Subsequently, based on the historical information, delta grouping [32] was developed, which divides the dimensions according to the accumulated difference of each dimension between two populations in the two consecutive generations. These mentioned grouping strategies can be considered as dynamic grouping strategies, because they are all executed during each generation, and thus the components of each group are dynamic.

To obtain the accurate grouping of dimensions, variable dependency should be taken into consideration. Therefore, Li *et al*. [33] proposed differential grouping (DG) utilizing

partial difference to detect variable dependency. Different from the dynamic grouping strategies mentioned above, DG is executed before evolution and once the dimension groups are obtained, they keep unchanged during the later optimization stage. Thus, actually, DG considers the variable dependency as global information. Since DG can only identify variables that interact directly, an extended version (XDG) [34] was developed by Sun *et al*, which can detect indirect variable dependency.

Even through the above two grouping strategies can decompose dimensions into groups as accurately as possible, they usually cost a large number of function evaluations to detect variable dependency. In the worst case, they need $O(D^2)$ function evaluations, which greatly reduces the number of function evaluations used for evolution. In addition, when it arrives at problems with dynamic variable dependency, such as piecewise functions, they would lose their feasibility.

In this paper, from the perspective of developing DE algorithms, we propose a multiple parents guided DE (MPGDE) algorithm to cope with large scale optimization, which will be detailed in the following section.

III. Multiple Parents Guided Differential Evolution

Observing Eqs. (4), (5) and (6), we can see that each individual is guided by only one best parent (either the global best parent or one of the top p best parents). Then, this guidance is disturbed by a difference vector created by two parents randomly selected from the current population. Such guidance may have limitations in the movement towards the global area, because generally one best individual only contains a part of useful information [41]. This situation is much more obvious in high dimensional problems.

Thus, to enhance the movement of each individual towards the global area, we propose a multiple parents guided DE (MPGDE) algorithm to deal with large scale optimization. The main idea of this algorithm is to use multiple best parents to guide individuals to move to promising areas.

Further, during evolution, the failed parents and trial vectors may be also beneficial for finding better searching directions for individuals, the former of which has been verified to be useful for evolution in [1]. Thus, in MPGDE, for the consideration of accelerating the convergence, we maintain an archive with the same size as the population to store the failed parents and trial vectors. During evolution, the archive is updated using a method like niching, which will be detailed in the following.

Generally, fast convergence usually is accompanied with stagnation and falling into local optima. To avoid this, in general, high diversity is required for the population. To this end, in MPGDE, we introduce the evolution path in the mutation operator.

Together, the mutation strategy in MPGDE can be formulated as follows:

$$v^j_{i,G} = x^j_{i,G} + F_{i_1}(x^j_{tbest_{r_1},G} - x^j_{i,G}) + F_{i_2}(x^j_{r_2,G} - x^j_{r_3,G}) + F(EP^j_G - EP^j_{G-1}) \quad (9)$$

where $\boldsymbol{x}_{tbest_{r_1},G}$ is an individual randomly selected from the top M ranked individuals among the population $\boldsymbol{P}$ and the archive $\boldsymbol{A}$, namely, r_1 is within[1, M]. It should be noted that for each dimension, r_1 is randomly generated, indicating that r_1 may be different for different dimensions. $\boldsymbol{x}_{r_2,G}$, and $\boldsymbol{x}_{r_3,G}$ are two random individuals selected from the population $\boldsymbol{P}$ and the archive $\boldsymbol{A}$**,** namely r_2 and r_3 are within [1,2**NP*] (when either is larger than *NP*, it means that this individual comes from the archive $\boldsymbol{A}$.). Different from r_1, these two random individuals are only selected once for each individual, meaning that for all dimensions of one individual, these two random individuals are kept unchanged. $\boldsymbol{EP}_G$ and $\boldsymbol{EP}_{G\text{-}1}$ are the centers of the top M ranked individuals in two consecutive generations G (the current generation) and G-1(the previous generation) respectively, the difference of which denotes the evolution path between two consecutive generations. Finally, F_{i_1}, F_{i_2} and F are three scale parameters controlling the influence of each part respectively and together, they control the step that each individual moves in each generation.

This mutation strategy (Eq. (9)) is called "*DE/current-to-Mtbest/1*" in this paper. The detailed elaboration of each part in this mutation strategy is presented as follows:

1) The first difference, $F_{i1}(x^j_{tbest_{r1},G} - x^j_{i,G})$, guides each dimension of the current individual to one of the top ranked M individuals. Since each of these M best individuals may only contain a part of useful information, we use multiple best parents to guide each individual, so that fast convergence can be achieved. Thus, in this part, r_1 is randomly selected within [1, M] for each dimension. In addition, different from Eq. (6), where the top p best individuals are only from the population, in MPGDE, the M top ranked individuals are from the combination of the population $\boldsymbol{P}$ and the archive $\boldsymbol{A}$. This is mainly because the failed parents and trial vectors at each generation may also contain potentially useful directions for individuals to move. Together, this part mainly reflects the idea of MPGDE.
2) The second difference, $F_{i2}(x^j_{r_2,G} - x^j_{r_3,G})$, is a random difference vector that disturbs the mutation of each individual. Instead of randomly selecting these two individuals only from the population, in MPGDE, these two are randomly generated from the combination of the population and the archive. This is mainly for the promotion of diversity. In addition, it should be noted that r_2 and r_3 are only generated once for each individual, which is very different from r_1. This is because the first difference vector is mainly for the guidance, while the second one is for disturbance.
3) The third difference, $F(EP^j_G - EP^j_{G\text{-}1})$, denotes the evolution path, which is the difference of centers of the top ranked M individuals in two consecutive generations G (the current generation) and G-1(the previous generation). The center ($\boldsymbol{EP}_G$) of the M individuals is the mean position of these M individuals, which can be calculated as follows:

$$EP^j_G = \frac{1}{M}\sum_{t=1}^{M} x^j_{tbest_t} \quad (10)$$

This difference vector is to utilize the useful information embedded in the evolution path, which can not only enhance the diversity of the population to some extent, but also lead the individuals to the promising areas with potentially useful directions, which may contribute to fast convergence.

As for the parameters, F_{i_1}, F_{i_2}, F and CR in MPGDE, we consider utilizing self-adaptive strategies to adjust these parameters since DE usually is very sensitive to these parameters. In the literature, many works have been done to self-adaptively adjust F and CR, such as jDE [39], JADE [1], SaNSDE [40] and IDE [9]. In this paper, we adopt the self-adaptive strategy in JADE [1] to adjust F_{i_1}, F_{i_2} and CR while set F as a fixed value (in this paper, it is set to be 1.0). That is, F_{i_1} and F_{i_2} are generated with Cauchy distribution for each individual, while CR is generated with Gaussian distribution for each individual. During the generation, those CR, F_{i_1} and F_{i_2} that can make the generated trial vectors successfully survive are kept and then are used to update the mean of the Gaussian and Cauchy distribution.

As for the archive $\boldsymbol{A}$, at the initial stage, it is empty. During the generation, the failed parents and trial vectors are gradually added into the archive. When the archive is full, namely the size of the archive is equal to the population size, we use a method like niching to update the archive. That is, when a new failed parent or trial vector arrives, we randomly select K individuals from the full archive and then compare the failed parent or trial vector with the worst one among the selected K individuals. If the failed parent or trial vector is better, it will replace the worst one; otherwise, it will be discarded. Through this, the diversity of the archive can be preserved, so that the diversity of the population can be preserved as well.

Overall, the complete framework of the proposed MPGDE is presented in **Algorithm 1**. In this algorithm, Lines 1 to 5 display the initialization stage, Lines 10 to 15 show the mutation operation, Line 16 is the crossover operation, Line 17 is the selection process and Lines 19 to 34 present the update of the archive. Comprehensively, we can see that MPGDE is simple to implement.

Algorithm 1: The pseudo code of MPGDE

Input: Population size NP, Maximum number of function evaluations Max_Fes, the initial uF_1, uF_2 and uCR (uF_1=0.7, uF_2=0.7 and uCR=0.5);

1: Fes = 0 and set the archive $\boldsymbol{A}$ to be an empty set;
2: Initialize the population randomly;
3: Evaluate individuals in the population $\boldsymbol{P}$;
4: $Fes = Fes + NP$;
5: Initialize $\boldsymbol{EP}_0$ to a zero vector;
6: Sort individuals in the population and record the indexes of the top best M individuals;
7: **While** $Fes < Max\_Fes$
8: Compute the mean position $\boldsymbol{EP}_G$ of these M individuals according to Eq. (10);
9: **For** $i = 1:NP$
//mutation operation
10: Generate F_{i_1}, F_{i_2} and CR_i for this individual according to the method in JADE [1] utilizing uF_1, uF_2 and uCR;
11: Randomly select two individuals (r_2 and r_3) from $\boldsymbol{P} \cup \boldsymbol{A}$;
12: **For** $j = 1:D$
13: Randomly select an individual r_1 from those M individuals;
14: Compute the jth dimension of mutation vector $\boldsymbol{v}_i$ according to Eq. (9);
15: **End For**
//crossover operator
16: Generate the trial vector $\boldsymbol{u}_i$ according to Eq. (7);
//selection operator
17: Evaluate the trial vector $\boldsymbol{u}_i$ and compare it with its parent $\boldsymbol{x}_i$ according to Eq. (8);
18: $Fes = Fes + 1$;
//update archive
19: **If** $\boldsymbol{u}_i$ survivals
20: **If** $|\boldsymbol{A}| < NP$
21: Put the failed parent $\boldsymbol{x}_i$ into the archive $\boldsymbol{A}$;
22: **Else**
23: Randomly select K indexes of individuals from the archive;
24: Compare the failed parent $\boldsymbol{x}_i$ with the worst one among the selected K individuals, and replace it if $\boldsymbol{x}_i$ is better;
25: **End If**
26: Record the successful F_{i_1}, F_{i_2} and CR_i;
27: **Else**
28: **If** $|\boldsymbol{A}| < NP$
29: Put the failed trial vector $\boldsymbol{u}_i$ into the archive $\boldsymbol{A}$;
30: **Else**
31: Randomly select K indexes of individuals from the archive;
32: Compare the failed trial vector $\boldsymbol{u}_i$ with the worst one among the selected K individuals, and replace it if $\boldsymbol{u}_i$ is better;
33: **End If**
34: **End If**
35: **End For**
36: Update uF_1, uF_2 and uCR according to the method in JADE [1];
37: Sort individuals in the population and the archive together and record the indexes of the top best M individuals;
38: Obtain the global best fitness and the global best individual;
39: **End While**
Output: The global best fitness and the global best individual.

IV. Experimental Studies

A. Experimental Setup

1) Benchmark Functions

To verify the performance of the proposed MPGDE, we conduct experiments on 20 CEC'2010 benchmark functions with 1000 dimensions [42]. For the detailed description of these benchmark functions, readers can be referred to [42].

TABLE I
PARAMETER SETTINGS FOR MPGDE

NP	M	K	uF_1	uF_2	uCR	F
200	20	50	0.7	0.7	0.5	1.0

2) Compared Algorithms

To comprehensively demonstrate the performance of MPGDE, we compare it with several state-of-the-art evolutionary algorithms that focus on dealing with large scale optimization. The compared algorithms are:

(1). CSO [21] is a PSO variant dealing with large scale optimization. It introduces a new competitive learning strategy for PSO and shows its promising performance in handling large scale optimization.
(2). SL_PSO [22] is another PSO variant that embeds a social learning strategy into PSO.
(3). DMS_PSO [43] is a multi-swam PSO variant, where multiple swarms are dynamically formed during each

generation.

(4). CCPSO2 [44] is a PSO variant that takes advantage of CC to deal with large scale optimization. In this algorithm, the grouping method is random grouping and the group size is randomly selected from a pool.

(5). DECC-DG [33] is a DE variant that makes use of CC to handle large scale optimization. In this method, the nearly accurate grouping, differential grouping, for fixed interdependent variables is utilized.

(6). JADE [1] is a self-adaptive DE, which maintains an

TABLE II

COMPARISON RESULTS AMONG DIFFERENT ALGORITHMS ON CEC'2010 FUNCTIONS WITH 1000 DIMENSIONS.

F	Quality	MPGDE	CSO	SL_PSO	DMS_PSO	CCPSO2	DECC-DG	JADE
F_1	Mean	1.74E-19	4.75E-12	7.73E-18	7.57E+05	1.88E+00	7.80E+03	2.98E+05
	Std	1.08E-19	7.90E-13	9.01E-19	2.20E+06	2.26E+00	2.66E+04	5.42E+05
	t-test	-	**3.29E+01**$^+$	**4.50E+01**$^+$	1.89E+00$^=$	**4.54E+00**$^+$	1.61E+00$^=$	**3.01E+00**$^+$
F_2	Mean	2.51E+03	7.48E+03	1.93E+03	7.38E+03	5.06E+00	4.43E+03	4.74E+03
	Std	9.98E+01	2.63E+02	8.32E+01	3.19E+02	1.10E+00	1.98E+02	3.37E+02
	t-test	-	**9.67E+01**$^+$	-2.43E+01$^-$	**7.97E+01**$^+$	-1.37E+02$^-$	**4.75E+01**$^+$	**3.48E+01**$^+$
F_3	Mean	9.28E+00	2.57E-09	1.84E+00	1.97E+01	5.61E-03	1.67E+01	7.02E+00
	Std	4.58E-01	1.85E-10	2.67E-01	9.64E-02	1.73E-03	3.10E-01	3.93E-01
	t-test	-	-1.11E+02$^-$	-7.68E+01$^-$	**1.22E+02**$^+$	-1.11E+02$^-$	**7.39E+01**$^+$	-2.05E+01$^-$
F_4	Mean	2.45E+11	6.87E+11	2.82E+11	3.69E+12	2.14E+12	4.95E+12	3.34E+13
	Std	5.15E+10	1.79E+11	8.88E+10	2.59E+12	1.54E+12	1.33E+12	4.39E+12
	t-test	-	**1.30E+01**$^+$	1.98E+00$^=$	**7.30E+00**$^+$	**6.71E+00**$^+$	**1.93E+01**$^+$	**4.14E+01**$^+$
F_5	Mean	2.46E+07	2.46E+06	3.04E+07	3.51E+08	4.56E+08	1.49E+08	1.50E+08
	Std	3.70E+06	1.35E+06	8.57E+06	8.13E+07	1.20E+08	2.02E+07	2.62E+07
	t-test	-	-3.08E+01$^-$	**3.38E+00**$^+$	**2.20E+01**$^+$	**1.97E+01**$^+$	**3.32E+01**$^+$	**2.60E+01**$^+$
F_6	Mean	1.11E+01	8.16E-07	1.95E+01	5.65E+06	1.79E+07	1.63E+01	2.29E+05
	Std	7.83E-01	2.60E-08	4.20E+00	5.64E+06	4.50E+06	3.82E-01	4.71E+05
	t-test	-	-7.74E+01$^-$	**1.08E+01**$^+$	**5.49E+00**$^+$	**2.18E+01**$^+$	**3.30E+01**$^+$	**2.66E+00**$^+$
F_7	Mean	8.80E-04	2.13E+04	6.48E+04	4.36E+06	2.48E+08	1.14E+04	1.29E+05
	Std	1.17E-03	4.60E+03	3.88E+04	2.11E+07	3.97E+08	8.52E+03	5.50E+04
	t-test	-	**2.53E+01**$^+$	**9.15E+00**$^+$	1.13E+00$^=$	**3.43E+00**$^+$	**7.34E+00**$^+$	**1.28E+01**$^+$
F_8	Mean	5.39E+02	3.86E+07	7.57E+06	1.20E+07	1.55E+07	2.30E+07	1.99E+06
	Std	1.33E+03	8.33E+04	2.48E+06	2.14E+07	1.85E+07	2.16E+07	2.85E+06
	t-test	-	**2.54E+03**$^+$	**1.67E+01**$^+$	**3.08E+00**$^+$	**4.58E+00**$^+$	**5.84E+00**$^+$	**3.82E+00**$^+$
F_9	Mean	2.44E+07	6.68E+07	3.34E+07	1.25E+08	1.00E+08	5.90E+07	4.43E+07
	Std	1.82E+06	4.47E+06	3.67E+06	1.21E+07	3.93E+07	9.97E+06	1.04E+07
	t-test	-	**4.81E+01**$^+$	**1.20E+01**$^+$	**4.49E+01**$^+$	**1.06E+01**$^+$	**1.87E+01**$^+$	**1.04E+01**$^+$
F_{10}	Mean	2.53E+03	9.58E+03	2.79E+03	8.36E+03	5.11E+03	4.55E+03	6.73E+03
	Std	7.96E+01	6.68E+01	1.30E+03	8.89E+02	8.80E+02	1.10E+02	2.41E+02
	t-test	-	**3.71E+02**$^+$	1.11E+00$^=$	**3.58E+01**$^+$	**1.60E+01**$^+$	**8.14E+01**$^+$	**9.04E+01**$^+$
F_{11}	Mean	6.12E+01	3.98E-08	2.42E+01	2.02E+02	1.98E+02	1.04E+01	1.96E+02
	Std	4.69E+00	3.25E-09	3.10E+00	1.38E+01	3.94E-01	8.19E-01	4.10E+01
	t-test	-	-7.15E+01$^-$	-3.60E+01$^-$	**5.27E+01**$^+$	**1.60E+02**$^+$	-5.84E+01$^-$	**1.79E+01**$^+$
F_{12}	Mean	4.02E+03	4.37E+05	1.54E+04	2.72E+04	4.09E+04	2.56E+03	4.09E+04
	Std	6.12E+02	6.61E+04	7.19E+03	5.66E+03	1.27E+04	5.31E+02	2.02E+04
	t-test	-	**3.59E+01**$^+$	**8.61E+00**$^+$	**2.23E+01**$^+$	**1.59E+01**$^+$	-9.90E+00$^-$	**1.00E+01**$^+$
F_{13}	Mean	7.46E+02	5.53E+02	9.79E+02	2.03E+03	1.32E+03	5.64E+03	7.72E+04
	Std	1.57E+02	1.78E+02	3.91E+02	6.02E+02	1.81E+02	3.02E+03	3.92E+04
	t-test	-	-4.46E+00$^-$	**3.02E+00**$^+$	**1.13E+01**$^+$	**1.31E+01**$^+$	**8.87E+00**$^+$	**1.07E+01**$^+$
F_{14}	Mean	6.54E+07	2.46E+08	8.55E+07	3.92E+08	2.58E+08	3.40E+08	1.19E+08
	Std	4.33E+06	1.31E+07	7.68E+06	2.83E+07	1.34E+08	2.23E+07	1.07E+07
	t-test	-	**7.15E+01**$^+$	**1.25E+01**$^+$	**6.24E+01**$^+$	**7.88E+00**$^+$	**6.63E+01**$^+$	**2.57E+01**$^+$
F_{15}	Mean	2.54E+03	1.01E+04	1.12E+04	9.35E+03	1.05E+04	5.86E+03	6.79E+03
	Std	1.12E+02	5.84E+01	9.77E+01	1.23E+03	1.36E+03	9.20E+01	1.90E+02
	t-test	-	**3.27E+02**$^+$	**3.20E+02**$^+$	**3.01E+01**$^+$	**3.17E+01**$^+$	**1.25E+02**$^+$	**1.05E+02**$^+$
F_{16}	Mean	2.00E+02	5.68E-08	2.36E+01	3.89E+02	3.97E+02	7.57E-13	3.40E+02
	Std	7.61E+00	6.32E-09	1.13E+01	3.93E+00	6.18E-01	6.67E-14	4.22E+01
	t-test	-	-1.44E+02$^-$	-7.09E+01$^-$	**1.21E+02**$^+$	**1.41E+02**$^+$	-1.44E+02$^-$	**1.78E+01**$^+$
F_{17}	Mean	4.66E+04	2.21E+06	8.74E+04	7.92E+04	1.32E+05	4.05E+04	1.77E+05
	Std	5.87E+03	2.10E+05	1.40E+04	1.19E+04	5.37E+04	2.05E+03	4.09E+04
	t-test	-	**5.62E+01**$^+$	**1.47E+01**$^+$	**1.35E+01**$^+$	**8.63E+00**$^+$	-5.38E+00$^-$	**1.73E+01**$^+$
F_{18}	Mean	1.75E+03	1.64E+03	2.92E+03	4.37E+07	2.91E+03	1.46E+10	6.43E+06
	Std	1.83E+02	8.27E+02	8.21E+02	8.13E+07	3.43E+02	2.82E+09	2.44E+07
	t-test	-	-6.92E-01$^-$	**7.64E+00**$^+$	**2.94E+00**$^+$	**1.64E+01**$^+$	**2.84E+01**$^+$	1.44E+00$^=$
F_{19}	Mean	1.33E+06	9.86E+06	5.22E+06	2.66E+06	1.53E+06	1.74E+06	9.09E+05
	Std	1.20E+05	5.13E+05	9.33E+05	4.93E+05	8.83E+04	9.89E+04	8.57E+04
	t-test	-	**8.87E+01**$^+$	**2.27E+01**$^+$	**1.43E+01**$^+$	**7.13E+00**$^+$	**1.42E+01**$^+$	-1.57E+01$^-$
F_{20}	Mean	1.87E+03	1.07E+03	1.73E+03	3.06E+07	2.15E+03	6.28E+10	1.35E+06
	Std	1.39E+02	1.72E+02	1.54E+02	6.44E+07	2.08E+02	8.48E+09	4.74E+06
	t-test	-	-1.99E+01$^-$	-3.70E+00$^-$	**2.60E+00**$^+$	**6.01E+00**$^+$	**4.06E+01**$^+$	1.56E+00$^=$
w/l/t		-	12/8/0	13/5/2	18/0/2	18/2/0	15/4/1	16/2/2

archive to keep the failed parents, but does not include the failed trial vectors.

3) Parameter Settings

In terms of the maximum number of function evaluations, algorithms, the main parameters are set as recommended in their corresponding papers.

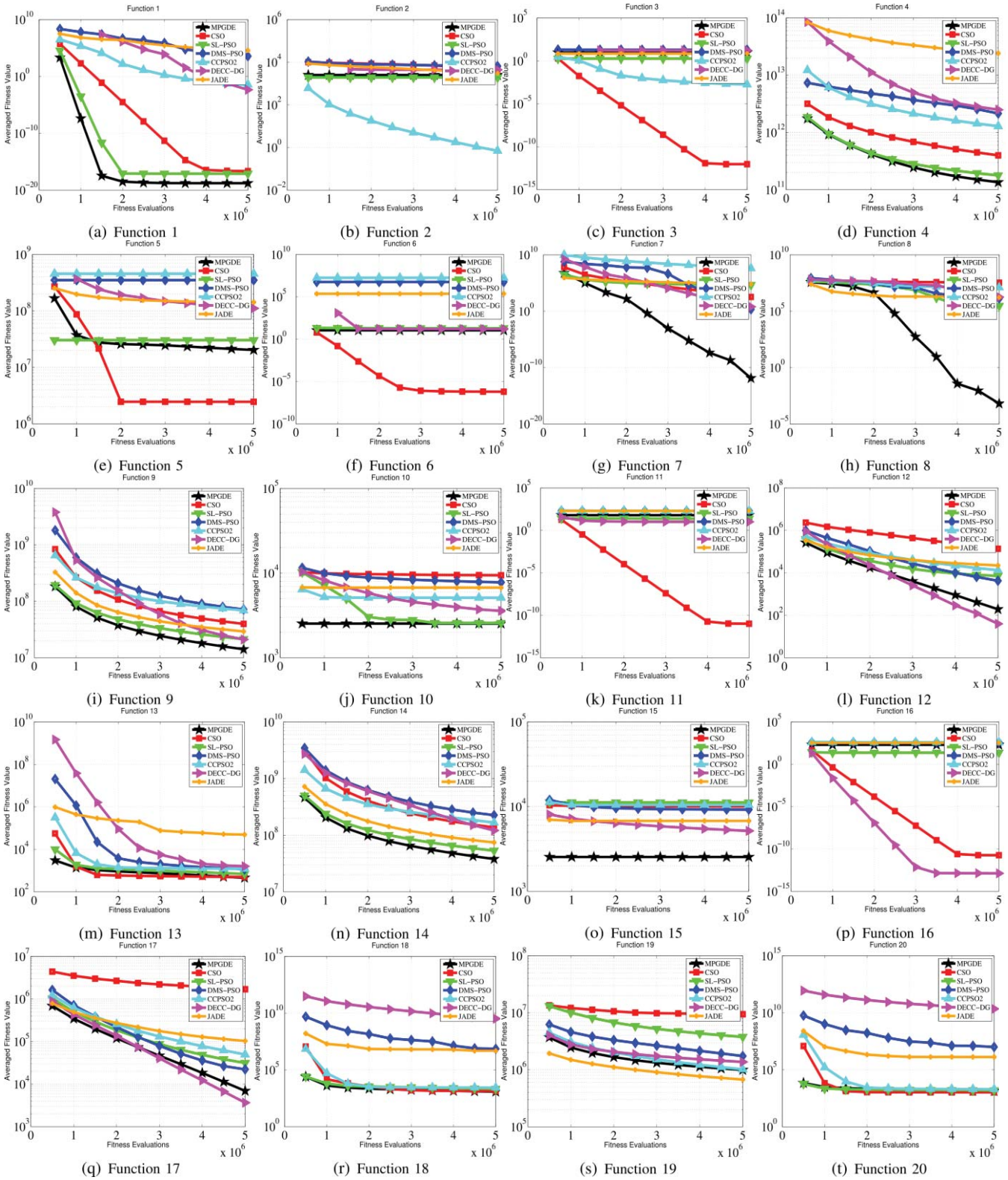

Fig. 1 Comparison results among different algorithms with respect to convergence behaviors on 20 1000-*D* CEC'2010 functions (better viewed in color).

As for the parameter settings, the main parameter settings of MPGDE are presented in Table I. For the compared without otherwise stated, it is set as 3×10^6, at which the comparison results are reported in Table II.

Additionally, it is worth mentioning that all experiments are conducted 30 runs for statistics. During the comparison between two algorithms, the two-tailed t-test at significance α=0.05 is conducted, at which the critical t-test value for 30 samples is 2.042. Based on the t-test results, in Table II, the best results are highlighted in bold and the symbols "+", "−" and "=" above *t* values indicate that MPGDE is significantly better than, worse than and equivalent to the compared methods respectively. Specially, the last row in Table II, termed as "*w/l/t*" counts the number of functions where MPGDE performs better than, worse than or similarly to the compared algorithms.

4) Experimental Results

Table II presents the detailed comparison results with respect to the mean value (Mean) and standard deviation (Std) among function values in 30 runs when the maximum function evaluations are exhausted. From this table, obviously, we can see that the proposed MPGDE is significantly better than the compared algorithms on most of the 20 benchmark functions. Specifically, compared with CSO and SL_PSO, MPGDE is better on 12 and 13 functions, respectively. In comparison with DMS_PSO and CCPSO2, MPGDE is much superior to both algorithms on 18 functions. During the competition with DECC-DG and JADE, MPGDE wins the competition on 15 and 16 functions respectively.

In summary, compared with PSO variants (CSO, SL_PSO, DMS_PSO and CCPSO2), MPGDE shows its promising superiority to PSO algorithms, demonstrating the potential of DE in handling large scale optimization. In comparison with DECC-DG, MPGDE shows that the without CC, evolving the dimensions simultaneously is also very promising or even better in coping with large scale optimization. The comparison results between MPGDE and JADE potentially verify the efficiency and effectiveness of the proposed mutation strategy named "*DE/current-to-Mtbest/1*".

Additionally, to comprehensively demonstrate the promising of MPGDE, we conduct experiments on CEC'2010 functions to make comparisons with respect to convergence behaviors of different algorithms. Fig. 1 shows the comparison results with the maximum number of fitness evaluations varying from 5×10^5 to 5×10^6.

From this figure, we can see that MPGDE is superior to all the compared algorithms on 6 functions (F_1, F_7~F_9, F_{14} and F_{15}). More specifically, MPGDE dominates CSO, CCPSO2 and DECC-DG on 12, 13 and 14 functions respectively, performs similarly to these three algorithms on 3, 5 and 5 functions respectively, and failed in the completion with the three methods only on 5, 2 and 1 functions respectively. Compared with DMS_PSO and JADE, MPGDE wins the competition on 17 and 16 functions. Specially, MPGDE is only dominated by JADE on 1 function and there is no failure in competition with DMS_PSO. In comparison with SL_PSO, MPGDE is better on 9 functions and performs similarly to SL_PSO on 9 functions as well.

As a whole, we can conclude that with respect to the convergence behavior, MPGDE can dominate the compared algorithms on most of the 20 functions.

Overall, the above verified superiority of the proposed MPGDE mainly benefits from three aspects. 1) The proposed multiple parents guided mutation strategy can afford good directions for individuals to move, which may contribute to fast convergence. Generally, one best individual only contains a part of useful information. This situation is more obvious for large scale problems. Using multiple top ranked parents to guide the search can potentially afford good combination of useful information in different parents and thus provide promising direction for individuals. 2) Maintaining an archive to reserve the failed parents and trial vectors can also potentially preserve the useful information that embeds in the failed individuals. In addition, all the individuals in the archive occupy chances to participate in the mutation of individuals. Thus, the diversity of the population can be potentially enhanced to a great extent. 3) Evolution path is introduced in the mutation of individuals. Generally, the evolution path contains the track that the population evolves and moves towards promising areas. Thus, the useful information embedded in the evolution path should be made full use of, which may contribute to fast convergence. Together, the above three techniques can potentially afford a good balance between exploration and exploitation, resulting in the promising performance of MPGDE.

V. CONCLUSION

This paper has proposed a multiple parents guided DE algorithm (MPGDE) to deal with large scale optimization. Since the failed parents and trial vectors may contain useful information, an archive is maintained to reserve these failed individuals and is updated with a method like niching to preserve diversity. Different from previous DE variants where only one parent is utilized to guide the search of each individual, in MPGDE, multiple top ranked parents among the population and the archive are made use of to direct each individual to move towards promising areas. Such guidance takes advantage of the fact that each top ranked individual may carry only a part of useful evolutionary information. Through the co-guidance of multiple parents, individuals may move faster to the promising area. In addition, to preserve high diversity during the evolution, the individuals in both the population and the archive participate in the mutation. Together, a new mutation strategy named "*DE/current-to-Mtbest/1*" is developed for MPGDE.

Experiments conducted on 20 CEC'2010 benchmark functions have verified the efficacy and effectiveness of the proposed MPGDE in comparison with the state-of-the-art evolutionary algorithms that concentrate on large scale optimization.

In the future, deep insight into this mutation strategy will be investigated and further improvement of MPGDE will be researched, including that more extensive experiments will be conducted to test the feasibility of this algorithm.

REFERENCE

[1] J. Zhang, and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," *IEEE Trans. Evol. Comput.,* vol. 13, no. 5, pp. 945-958, 2009.

[2] R. Storn, and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.,* vol. 11, no. 4, pp. 341-359, 1997.
[3] A. K. Qin, and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Proc. IEEE Congr. Evol. Comput.*, pp. 1785-1791, 2005.
[4] U. K. Chakraborty, *Advances in differential evolution*: Springer, 2008.
[5] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Trans. Evol. Comput.,* vol. 13, no. 3, pp. 526-553, 2009.
[6] S. Das, and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.,* vol. 15, no. 1, pp. 4-31, 2011.
[7] W.-J. Yu, M. Shen, W.-N. Chen, Z.-H. Zhan, Y.-J. Gong, Y. Lin, O. Liu, and J. Zhang, "Differential Evolution With Two-Level Parameter Adaptation," *IEEE Trans. Cybern.,* vol. 44, no. 7 pp. 1080 - 1099 2013.
[8] Y.-L. Li, Z.-H. Zhan, Y.-J. Gong, W.-N. Chen, J. Zhang, and Y. Li, "Differential Evolution with an Evolution Path: A DEEP Evolutionary Algorithm," *IEEE Trans. Cybern.,* vol. PP, no. 99 2014.
[9] L. Tang, Y. Dong, and J. Liu, "Differential evolution with an individual-dependent mechanism," *IEEE Trans. Evol. Comput.,* vol. 19, no. 4, pp. 560-574, 2015.
[10] Q. Fan, and X. Yan, "Self-Adaptive Differential Evolution Algorithm With Zoning Evolution of Control Parameters and Adaptive Mutation Strategies," *IEEE Trans. Cybern.,* vol. 46, no. 1, pp. 219-232, 2016.
[11] M. Weber, F. Neri, and V. Tirronen, "Shuffle or update parallel differential evolution for large-scale optimization," *Soft Computing,* vol. 15, no. 11, pp. 2089-2107, 2011.
[12] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "An improved self-adaptive differential evolution algorithm for optimization problems," *IEEE Trans. Ind. Informat.,* vol. 9, no. 1, pp. 89-99, 2013.
[13] M. Z. Ali, N. H. Awad, and P. N. Suganthan, "Multi-population differential evolution with balanced ensemble of mutation strategies for large-scale global optimization," *Applied Soft Computing,* vol. 33, pp. 304-327, 2015.
[14] N. Chen, W. N. Chen, Y. J. Gong, Z. H. Zhan, J. Zhang, Y. Li, and Y. S. Tan, "An Evolutionary Algorithm with Double-Level Archives for Multiobjective Optimization," *IEEE Trans. Cybern.,* vol. 45, no. 9, pp. 1851-1863, 2015.
[15] S. Das, A. Ghosh, and S. S. Mullick, "A Switched Parameter Differential Evolution for Large Scale Global Optimization–Simpler May Be Better," *Mendel 2015: Recent Advances in Soft Computing*, pp. 103-125, 2015.
[16] S.-M. Guo, C.-C. Yang, P.-H. Hsu, and J. S.-H. Tsai, "Improving Differential Evolution With a Successful-Parent-Selecting Framework," *IEEE Trans. Evol. Comput.,* vol. 19, no. 5, pp. 717-730, 2015.
[17] S. Das, S. S. Mullick, and P. Suganthan, "Recent advances in differential evolution–An updated survey," *Swarm and Evolutionary Computation,* vol. 27, pp. 1-30, 2016.
[18] T. Lixin, Z. Yue, and L. Jiyin, "An Improved Differential Evolution Algorithm for Practical Dynamic Scheduling in Steelmaking-Continuous Casting Production," *IEEE Trans. Evol. Comput.,* vol. 18, no. 2, pp. 209-225, 2014.
[19] V. Santucci, M. Baioletti, and A. Milani, "Algebraic Differential Evolution Algorithm for the Permutation Flowshop Scheduling Problem with Total Flowtime Criterion," *IEEE Trans. Evol. Comput.,* vol. PP, no. 99, pp. 1-1, 2015.
[20] Z. Yang, K. Tang, and X. Yao, "Differential evolution for high-dimensional function optimization," in *Proc. IEEE Congr. Evol. Comput.*, pp. 3523-3530, 2007.
[21] R. Cheng, and Y. Jin, "A Competitive Swarm Optimizer for Large Scale Optimization," *IEEE Trans. Cybern.,* vol. 45, no. 2, pp. 191-204, 2015.
[22] R. Cheng, and Y. Jin, "A social learning particle swarm optimization algorithm for scalable optimization," *Inf. Sci.,* vol. 291, pp. 43-60, 2015.
[23] W.-N. Chen, J. Zhang, H. S. Chung, W.-L. Zhong, W.-G. Wu, and Y.-H. Shi, "A novel set-based particle swarm optimization method for discrete optimization problems," *IEEE Trans. Evol. Comput.,* vol. 14, no. 2, pp. 278-300, 2010.
[24] W.-N. Chen, J. Zhang, Y. Lin, N. Chen, Z.-H. Zhan, H. S.-H. Chung, Y. Li, and Y.-h. Shi, "Particle Swarm Optimization With an Aging Leader and Challengers," *IEEE Trans. Evol. Comput.,* vol. 17, no. 2, pp. 241-258, 2013.
[25] Y.-J. Gong, J.-J. Li, Y. Zhou, Y. Li, H. S.-H. Chung, Y.-H. Shi, and J. Zhang, "Genetic Learning Particle Swarm Optimization," *IEEE Trans. Cybern.*, in press, 2015.
[26] X.-Y. Zhang, Y.-j. Gong, Z. Zhan, W.-N. Chen, Y. Li, and J. ZHANG, "Kuhn-Munkres Parallel Genetic Algorithm for the Set Cover Problem and Its Application to Large-Scale Wireless Sensor Networks," *IEEE Trans. Evol. Comput.*, in press, 2015.
[27] Q. Yang, W.-N. Chen, Y. Li, C. P. Chen, X.-M. Xu, and J. Zhang, "Multimodal Estimation of Distribution Algorithms," *IEEE Trans. Cybern.*, in press, 2016.
[28] M. A. Potter, and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving from Nature—PPSN III*, pp. 249-257, 1994 of Conference.
[29] Y.-j. Shi, H.-f. Teng, and Z.-q. Li, *Cooperative co-evolutionary differential evolution for function optimization*: Springer, 2005.
[30] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Inf. Sci.,* vol. 178, no. 15, pp. 2985-2999, 2008.
[31] W. Chen, T. Weise, Z. Yang, and K. Tang, "Large-scale global optimization using cooperative coevolution with variable interaction learning," in *Parallel Problem Solving from Nature, PPSN XI*, pp. 300-309, 2010 of Conference.
[32] M. N. Omidvar, X. Li, and X. Yao, "Cooperative co-evolution with delta grouping for large scale non-separable function optimization," in *Proc. IEEE Congr. Evol. Comput.*, pp. 1-8, 2010.
[33] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization," *IEEE Trans. Evol. Comput.,* vol. 18, no. 3, pp. 378 - 393, 2014.
[34] Y. Sun, M. Kirley, and S. K. Halgamuge, "Extended Differential Grouping for Large Scale Global Optimization with Direct and Indirect Variable Interactions," in *Proc. Conf. Genetic Evol. Comput.*, pp. 313-320, 2015.
[35] N. M. Hamza, D. L. Essam, and R. A. Sarker, "Constraint Consensus Mutation based Differential Evolution for Constrained Optimization," *IEEE Trans. Evol. Comput.,* vol. PP, no. 99, pp. 1-1, 2015.
[36] N. Hamza, D. Essam, and R. Sarker, "Constraint Consensus Mutation based Differential Evolution for Constrained Optimization," *IEEE Trans. Evol. Comput.,* vol. PP, no. 99, pp. 1-1, 2015.
[37] S. M. Guo, and C. C. Yang, "Enhancing Differential Evolution Utilizing Eigenvector-Based Crossover Operator," *IEEE Trans. Evol. Comput.,* vol. 19, no. 1, pp. 31-49, 2015.
[38] S. Biswas, S. Kundu, and S. Das, "Inducing Niching Behavior in Differential Evolution Through Local Information Sharing," *IEEE Trans. Evol. Comput.,* vol. 19, no. 2, pp. 246-263, 2015.
[39] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Trans. Evol. Comput.,* vol. 10, no. 6, pp. 646-657, 2006.
[40] Z. Yang, K. Tang, and X. Yao, "Self-adaptive differential evolution with neighborhood search," in *Proc. IEEE Congr. Evol. Comput.*, pp. 1110-1116, 2008.
[41] Z.-H. Zhan, J. Zhang, Y. Li, and Y.-H. Shi, "Orthogonal learning particle swarm optimization," *IEEE Trans. Evol. Comput.,* vol. 15, no. 6, pp. 832-847, 2011.
[42] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC 2010 special session and competition on large-scale global optimization," *Technical Report, Nature Inspired Computation and Applications Laboratory, USTC, China*, 2010.
[43] J. Liang, and P. N. Suganthan, "Dynamic multi-swarm particle swarm optimizer," in *Proc. IEEE SIS*, pp. 124-129, 2005.
[44] X. Li, and X. Yao, "Cooperatively coevolving particle swarms for large scale optimization," *IEEE Trans. Evol. Comput.,* vol. 16, no. 2, pp. 210-224, 2012.