A Multi-optimizer Cooperative Coevolution Method for Large Scale Optimization

[1]Wen Shi, [1*]Wei-Neng Chen, [1,2]Qiang Yang and [1]Jun Zhang
[1]School of Computer Science and Engineering, South China University of Technology, Guangzhou, China
[2]School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China

Abstract—**Cooperative coevolution framework is an effective strategy to deal with large scale optimization problems. However, most studies on cooperative coevolution framework utilize the same optimizer for all subcomponents, which may not be effective enough. In this paper, we propose a novel multi-optimizer cooperative coevolution method for large scale optimization problems which randomly chooses an optimization strategy for each subcomponent independently. Four extensively used differential evolution algorithms are utilized as candidate optimizers. Two of them have good exploration properties while the other two have good exploitation properties. Experimental results utilizing differential grouping algorithm as decomposition strategy of the cooperative coevolution framework show that this multi-optimizer CC method performs better on most of the CEC'2010 large-scale global optimization (LSGO) benchmark functions than each single-optimizer CC framework where all of the subcomponents use the same optimizer. What is more, experimental results also show that this multi-optimizer CC method is suitable for not only fixed decomposition strategy (DG, XDG, and GDG) but also dynamic decomposition strategy (Delta Grouping).**

Keywords—multi-optimizer; cooperative coevolution; large scale optimization; differential evolution

I. INTRODUCTION

Large scale optimization problem has become increasingly popular in the recent years for a large amount of real world problems can be abstracted as large scale optimization problem. However, solving large scale optimization problem is a challenging task due to the deterioration of the performances of evolutionary algorithms (EAs) as the dimensions increase.

To settle this problem, two kinds of solutions have been proposed. One is to decompose the large scale problem into several smaller ones and the other is to apply hybridization method.

Cooperative coevolution framework [1], which utilizes a divide-and-conquer technique is the most important approach to solve large scale optimization problems. It divides a high dimensional problem into several lower dimensional sub-components and then evolves each subcomponent separately and finally implements coevolution to gain interdependencies between different subcomponents. This CC framework has been shown to be attractive for solving large scale complex optimization problems. Since there are plenty of candidate EAs for the optimization process and the coadaptation process is embedded in fitness evaluation operations, the decomposition process becomes a vital process. Thus, dozens of papers have studied the grouping methods of CC framework.

The original papers only applied simple decomposition mechanisms such as one-dimensional based strategy [2], splitting-in-half strategy [3], and random grouping strategy [4], as time goes by, more and more complicated strategies have been proposed, such as multilevel cooperative coevolution (MLCC) [5], delta grouping [6], differential grouping (DG) [7] and so on.

Although a lot of researches have been made on the decomposition strategies of CC framework, most of them utilize the same algorithm to optimize all of the subcomponents so far. It may not be effective enough to use the same optimizer to evolve subcomponents in different stages, for some of them may in the early stage of optimization and need to use strategies with good exploitation properties to speed up the pace of convergence while the others may be in the later stage of optimization that have already converged and need to use strategies with good exploration properties to increase the diversity of the population and search more spaces to avoid stagnation in the local optima.

This paper pays attention to the difference of the subcomponents and proposes a new multi-optimizer CC method which randomly chooses an optimization strategy from the candidate optimizers for each subcomponent. Four widely used differential evolution algorithms [8] are utilized as the candidate optimizers in the proposed multi-optimizer CC method, two of them can speed up the pace of convergence ("DE/current-to-best/1/bin" and "DE/current-to-best/2/bin") and the other two can increase the diversity of the population ("DE/rand/1/bin" and "DE/rand/2/bin"). The parameters in the DEs are set based on individual-dependent parameter (IDP) setting strategy which is proposed in [9].

The rest of the paper is organized as follows. In Section II, a review of the differential evolution, the cooperative coevolution framework and some of the decomposition strategies are given. Section III elaborates the proposed multi-optimizer CC method utilizing different strategies in different subcomponents. Section IV presents the experiment results to prove the effectiveness of the proposed multi-optimizer CC

* **Corresponding Author** (Email: cwnraul634@aliyun.com)
This work was supported under the National Natural Science Foundation of China (NSFC) Grant Nos. 61622206, 61379061, 61332002, the Natural Science Foundation of Guangdong No. 2015A030306024, the Pearl River New Star of Science and Technology No. 201506010002 and the Guangdong Special Support Program No. 2014TQ01X550.

978-1-5090-4240-1/16/$31.00 ©2016 IEEE

method and shows that this method is suitable for differential grouping (DG), extended differential grouping (XDG), global differential grouping (GDG) and delta grouping. Finally, the conclusions and summaries of the whole paper are provided in Section V.

II. Related Work

A. Cooperative Coevolution Framework

The cooperative coevolution (CC) framework [1] was first proposed to improve the performance of genetic algorithms (GAs) [10] in the optimization of the high-dimensional problems and has received great success. In recent years, efforts have been made to apply CC framework to DE algorithm [11]. The main idea of CC framework is to divide a complicated problem into several smaller ones to make it easier to deal with and then conquer them separately. The detailed information is concluded as follows.

1) Decomposition: Suppose the dimension of the given problem is n, which is an extremely huge number. The decomposition process is executed to divide it into m smaller groups. Initial works on cooperative coevolution framework utilized simple strategies in this step, such as one-dimensional based strategy [2], where $m = n$, splitting-in-half strategy [3], where $m = 2$ and random grouping strategy [4], where m is a factor of n. One-dimensional based strategy, dividing the n-dimensional vector into n independent variables, is effective when dealing with separable problems, but meets with reverse when dealing with nonseparable problems. Splitting-in-half strategy decompose the n-dimensional vector into two $n/2$-dimensional ones, which are still complicated when n is large. Random grouping strategy divides the n-dimensional vector into m smaller groups of the same size randomly, where a large amount of work is needed to find the suitable value of m since it varies from problem to problem. Nowadays, a huge number of improved decomposition strategies have been proposed [5-7]. They can be divided into two categories: the ones which decompose the problem into fixed subcomponents during the whole iterations and the ones whose subcomponents keep changing during the iterations, the detailed information of these strategies will be discussed in the following subsection.

2) Optimization: After decomposition, each sub-component evolves separately. EAs [12, 13], which have a strong global search capability, can effectively optimize each sub-component. EAs such as particle swarm optimization (PSO) [14-16], genetic algorithm (GA) [10], ant colony optimization (ACO) [17], the estimation of distribution algorithms (EDA) [18] and differential evolution algorithm (DE) [8] have been widely used in this step. In the multi-optimizer CC method proposed by this paper, DE algorithms are utilized as the optimization strategies due to its effectiveness in global optimization. The particular information about DE algorithm will be elaborated later.

3) Coadaptation: Since the vectors have been divided, the fitness value cannot be calculated by using each single subcomponent exclusively in the optimization process. Therefore, the individual with the best fitness value of each subcomponent is recorded and utilized to calculate the fitness value together with each subcomponent. That is, noted the individual with the global fitness value as $X_{best} = (X_1^{best}, X_2^{best}, \ldots X_m^{best})$, where X_a^{best}, $a = 1,2,\ldots m$ is the individual with the best fitness value up to the current generation in subcomponent a. When optimizing the subcomponent in the gth generation, $X = (X_1^{best}, \ldots, X_{a-1}^{best}, X_{a,g}, X_{a+1}^{best}, \ldots, X_m^{best})$ is used to compute the fitness value.

B. Decomposition Strategies

1) Fixed Decomposition Strategies: Differential grouping algorithm (DG) [7] is the most vital fixed decomposition algorithm, it is an algorithm extensively applied as the decomposition strategy of CC framework due to its effectiveness for both separable and nonseparable problems and its ability to get the number of subgroups automatically.

Although DG algorithm has been proven to be effective, there are also many drawbacks, therefore, studies have been made to further improve its performance:

Extended Differential Grouping (XDG) [19]: addresses the limitation of DG that can only identify the direct interaction between variables.

Global Differential Grouping (GDG) [20]: increases the accuracy of DG by maintaining global information.

2) Dynamic Decomposition Strategies: Multilevel Cooperative Coevolution (MLCC) [5], which can choose the proper size of the subcomponent self-adaptively and Delta Grouping [6], which uses delta value (the amount of changes in each of the decision variables in every iterations) to identify interacting variables and divides subcomponents according to it, are two important dynamic decomposition strategies.

Apart from these two algorithms, there are also other dynamic decomposition algorithms such as EACC-G [21], which applies an adaptive weighting mechanism in the coadaptation steps; CCEA-AVP [22], which partitions the variables based on the correlation coefficient; DECC-ML [23], which improves the performance of MLCC by grouping more frequently; DECC-CIG [24], which proposes a new method named symmetrical uncertainty to identify the interaction between variables.

C. Differential Evolution

DE algorithm was proposed by Storn and Price in 1997 [8], like other evolutionary algorithms, it randomly generates the initial population $X_{i,0} = (x_{i,1,0}, x_{i,2,0}, \ldots x_{i,n,0})$, $x_{i,j,0} \in [x_j^l, x_j^u]$, $i = 1,2,\ldots NP$ with uniform distribution, where NP is the population size, n is the dimension of the given problem, x_j^l and x_j^u are the lower bounds and upper bounds of the jth dimension. And then get into the iterations of evolution process which consists of mutation, crossover and selection.

1) Mutation: At each generation g, for each individual $X_{i,g}$, the mutant vector is denoted as $V_{i,g}$ and the learning strategies of DE are usually denoted by DE/x/y/z, where x indicates the vector to be mutated while y indicates the number of difference vectors used and z stands for the crossover mechanism. The mutation strategies with the largest utilization ratio are the following three:

DE/rand/1: $V_{i,g} = X_{r_1,g} + F_{i,g} \cdot (X_{r_2,g} - X_{r_3,g})$

DE/current-to-best/1:

$V_{i,g} = X_{i,g} + F_{i,g} \cdot (X_{best,g} - X_{i,g}) + F_{i,g} \cdot (X_{r_1,g} - X_{r_2,g})$

DE/best/1: $V_{i,g} = X_{best,g} + F_{i,g} \cdot (X_{r_1,g} - X_{r_2,g})$

where $F_{i,g}$ is the mutation vector of the ith individual in generation g which lies in [0,2]; r_1, r_2, r_3 are three random and mutually exclusive integers uniformly chosen from the range $[1,NP]\backslash\{i\}$; $X_{best,g}$ is the individual with the best fitness value in generation g.

2) Crossover: After mutation, crossover operation is applied. In this process, the binomial crossover operation is executed to each pair of mutant vector $V_{i,g}$ and $X_{i,g}$ to get the trail vector $U_{i,g}$, where

$$u_{i,j,g} = \begin{cases} v_{i,j,g}, & \text{if } U(0,1) \le CR_{i,g} \text{ or } j = j_{rand} \\ x_{i,j,g}, & \text{otherwise} \end{cases}, j = 1,2,...n$$

$CR_{i,g}$ is the crossover rate of the ith individual in generation g which lies in [0,1]; $U(0,1)$ is an uniform random number on the interval [0,1]; j_{rand} is an integer randomly chosen from $[1,n]$.

3) Selection: Finally, the selection operation is utilized to compare the fitness value of the trail vector $U_{i,g}$ with the parent vector $X_{i,g}$ and choose the better one as the parent vector $X_{i,g+1}$ of the next generation.

$$X_{i,g+1} = \begin{cases} U_{i,g}, & \text{if } f(U_{i,g}) < f(X_{i,g}) \\ X_{i,g}, & \text{otherwise} \end{cases}$$

$f(U_{i,g})$ and $f(X_{i,g})$ are the fitness value of $U_{i,g}$ and $X_{i,g}$.

III. MULTI-OPTIMIZER CC METHOD

A. Experiment Analysis of Each Subgroup

It has been acknowledged that apply different strategies in different stages of optimization process can enhance the performance of the algorithm and there have already been researches based on this idea [9]. But the discrepancy of each group is always ignored and almost all of the proposed algorithms utilize the same strategy to optimize all of the subgroups although many researches have been made on CC framework and plenty of decomposition strategies have been proposed to divide the vector into several groups.

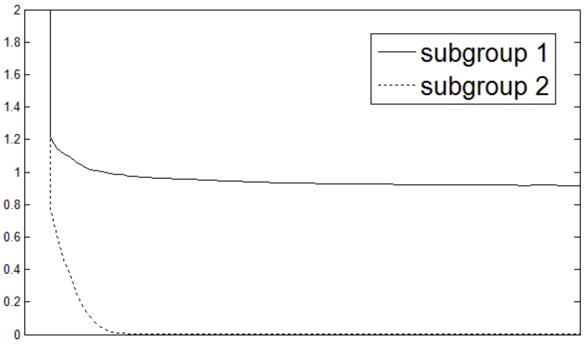

Fig. 1. The diversity of each subgroup of Single-group Shifted and 50-rotated Rastrigin's Function

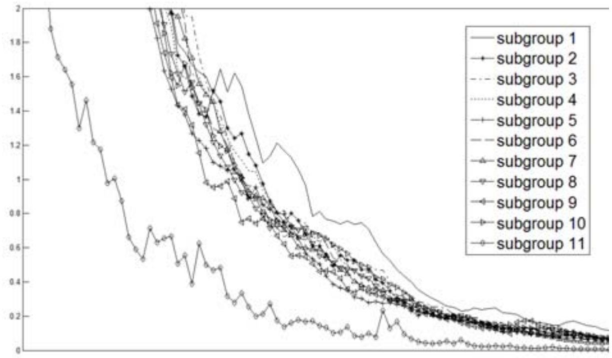

Fig. 2. The diversity of each subgroup of 10-group Shifted 50-dimensional Schwefel's Problem 1.2

Figs. 1 and 2 shows the diversity of each subgroup of Single-group Shifted and 50-rotated Rastrigin's Function and 10-group Shifted 50-dimensional Schwefel's Problem 1.2. It can easily discover from Fig. 1-2 that the characteristic of each group are totally different for these two problems. Some of the groups are in the early stage of the optimization and need to utilize strategies with a good convergence property to speed up the convergence rate while the others are in the later stage of the optimization and need to utilize strategies with good diversity to search more spaces to avoid stopping in the local optima.

B. Parameter Settings in DEs

Although a large amount of self-adaptive parameter setting strategies of DE algorithm have been proposed and proven to be effective, most of them need to use the historical information of the former iterations [9, 25-30], which is not suitable for our new CC method. For most of the decomposition strategies we utilized, the iteration time of each subcomponent is set to be 1 so that there is no history information that can be gained. A subcomponent may use different optimization strategies in different iterations and it is unreasonable to apply the information of an optimizer to the others. Therefore, we use a parameter setting strategy that merely needs to use the information of the current population, which is proposed in [9].

The value of $F_{i,g}$ and $CR_{i,g}$ are set based on the rank of $X_{i,g}$: reindex all individuals in the current population in ascending order of their fitness values, i.e., individual $X_{i,g}$ is the ith superior one. So that $F_{i,g}$ and $CR_{i,g}$ can be set as

$$F_{i,g} = CR_{i,g} = \frac{i}{NP}$$

However, superior individuals are not always close to the global best individual because many local minima spread all over the searching space and confuse the search. Based on this observation, we randomize the parameters using a normal distribution with the mean specified to the original value and the standard deviation specified to 0.1. Then, the parameter settings can be modified as

$$F'_{i,g} = N(F_{i,g}, 0.1)$$
$$CR'_{i,g} = N(CR_{i,g}, 0.1)$$

C. Multi-Optimizer CC Method

Base on the discovery that the subgroups in CC framework have different characteristic, this paper proposes a multi-optimizer CC method randomly choose an optimization strategy for each group.

We select four learning strategies as candidates: "DE/rand/1/bin", "DE/rand/2/bin", "DE/current-to-best/1/bin", "DE/current-to-best/2/bin", which have been extensively used in global optimization, the probability of choosing each of them to the *a*th subgroup is set to be $p_{a,1} = p_{a,2} = p_{a,3} = p_{a,4} = 0.5$, $a = 1,2,\ldots m$.

Among the four candidate strategies, "DE/rand/1/bin" and "DE/rand/2/bin" shows good exploration properties while DE/current-to-best/1/bin" and "DE/ current-to-best/2/bin" shows good exploitation properties.

The flow chart of the new proposed multi-optimizer CC method is given in Fig. 3.

Firstly, the initial population is randomly generated with the given bound and the variables are decomposed using a certain decomposition strategy. Then computes the fitness value of each individual to get the global best one and begins to cycle until the halting criteria are satisfied. For each group, generate a uniform random number on interval [0,1]. If it is in the range [0,0.25), DE/current to best/1 is used to optimize it; if it is in the range [0.25,0.5), DE/current to best/2 is used to optimize it; if it is in the range [0.5,0.75), DE/rand/1 is used to optimize it; if it is in the range [0.75,1], DE/rand/2/ is used to optimize it.

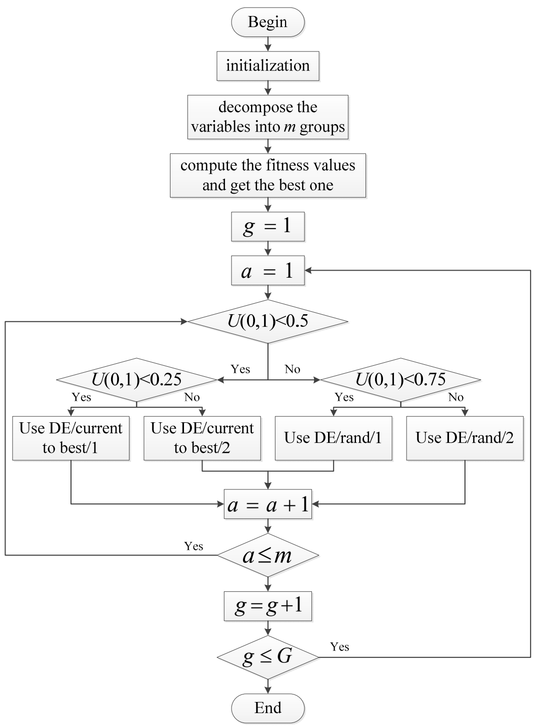

Fig. 3. The flow chart of multi-optimizer CC method

IV. Experiment Results

A. Benchmark Function

To evaluate the performance of the new proposed algorithm, benchmark functions for the CEC'2010 special session and competition on large-scale global optimization [31] is utilized. It consists of 20 benchmark functions with dimension $n = 1000$, the number of variables in each group $l = 50$ and it includes three separable functions ($f_1 - f_3$), five single-group l-non-separable functions ($f_4 - f_8$), five $n/2l$-group l-nonseparable functions ($f_9 - f_{13}$), five n/l-group l-nonseparable functions ($f_{14} - f_{18}$) and two nonseparable functions (f_{19}, f_{20}).

B. Parameter Settings

For each function, each algorithm is run 30 runs and FES=3e6. The population size *NP* is set to be 50 and the iteration time of each subcomponent is set to be 1 as suggested in [6, 7, 19, 20]. $\varepsilon = 10^{-3}$ in DG as suggested in [7]; $\varepsilon = 10^{-1}$ in XDG as suggested in [19] and $\varepsilon = 10^{-10}$ in GDG as suggest in [20].

C. Comparison with DECC-DGs

The Comparison of the multi-optimizer algorithm with single-optimizer algorithms using each candidate DE individually and SaNSDE, which is the optimization strategy the original work of DECC-DG uses, is given and all of them use DG algorithm as the decomposition strategy. Table I shows the mean value, the standard deviation of the multi-optimizer CC and each single-optimizer CC, and the p-value of Wilcoxon rank sum test of each single optimizer CC with multi-optimizer CC.

Table I shows that the new multi-optimizer CC method outperforms the single-optimizer CC methods that use optimizer SaNSDE, DE/current-to-best/1/bin, DE/current-to-best/2/bin, DE/rand/1/bin, DE/rand/2/bin individually in 8, 12, 16, 11, 14 out of 20 functions. Therefore, we can come to a conclusion that the multi-optimizer CC method cooperates well with DG.

D. Comparison with Other DECCs

It has been shown that this multi-optimizer CC method suits DG algorithm well, this subsection is to discuss whether it is suitable for XDG, GDG and Delta Grouping as well. We still compare the multi-optimizer DECCs with CC-SaNSDEs. The mean value, the standard deviation and the p-value of Wilcoxon rank sum test of the new multi-optimizer DECCs and the original DECCs with SaNSDE are shown in Table II.

Table II shows that the multi-optimizer DECC_XDG outperforms the original DECC_XDG with SaNSDE in 8 out of 20 functions, the multi-optimizer DECC_GDG outperforms the DECC_GDG with SaNSDE in 9 out of 20 functions and the multi-optimizer DECC_D outperforms the DECC_D with

SaNSDE in 9 out of 20 functions. Consequently, this multi-optimizer CC method is suitable for XDG, GDG and Delta Grouping as well. The results seem not good enough for the multi-optimizer DECCs outperform the original DECCs with SaNSDE in less than half of the benchmark functions for each decomposition strategy. However, we can discover from Table II that only 3 out of 20 functions underperform the original DECCs with SaNSDE for each decomposition strategy. The rest of them behave almost the same for the multi-optimizer CC method uses a random strategy for choosing optimizer, which makes the results have a great randomness. Since the multi-optimizer CC method underperform the single-optimizer CC framework in just few of the benchmark functions, we can draw a conclusion that this multi-optimizer CC method is suitable for XDG, GDG and Delta Grouping as well.

V. CONCLUSION

In this paper, we propose a new multi-optimizer CC method which randomly choose an optimization algorithm for each subcomponent. Experimental results show that this multi-optimizer CC method outperforms the single-optimizer CC methods in most of the CEC'2010 large-scale global optimization benchmark functions and this multi-optimizer CC method is suitable for fixed decomposition strategies such as DG, XDG and GDG as well as dynamic decomposition strategy such as Delta Grouping. Moreover, other optimization strategies can be also applied as the candidate optimizers in this algorithm and the number of candidates can increase or decrease as required. However, it seems that this method meets with the failure when cooperate with some of the dynamic decomposition strategies such as MLCC, which remains a problem to be solved.

REFERENCES:

[1] M. A. Potter and K. A. D. Jong, "A cooperative coevolutionary approach to function optimization," *Lecture Notes in Computer Science,* vol. 866, pp. 249--257, 1994.

[2] Y. Liu, X. Yao, Q. Zhao, and T. Higuchi, "Scaling up fast evolutionary programming with cooperative coevolution," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, 2001, pp. 1101-1108 vol. 2.

[3] Y. J. Shi, H. F. Teng and Z. Q. Li, *Cooperative Co-evolutionary Differential Evolution for Function Optimization*: Springer Berlin Heidelberg, 2005.

[4] V. D. B. Frans and A. P. Engelbrecht, "A Cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation,* vol. 8, pp. 225-239, 2004.

[5] Z. Yang, K. Tang and X. Yao, "Multilevel cooperative coevolution for large scale optimization," in *Evolutionary Computation*, 2008, pp. 1663-1670.

[6] M. N. Omidvar, X. Li and X. Yao, "Cooperative Co-evolution with delta grouping for large scale non-separable function optimization," in *Evolutionary Computation*, 2010, pp. 1 - 8.

[7] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative Co-evolution with Differential Grouping for Large Scale Optimization," *Evolutionary Computation IEEE Transactions on,* vol. 18, pp. 378-393, 2014.

[8] R. Storn and K. Price, "Price, K.: Differential Evolution - A Simple and Efficient Heuristic for Global Optimization Over Continuous Spaces. Journal of Global Optimization 11(4), 341-359," *Journal of Global Optimization,* vol. 11, pp. 341-359, 1997.

[9] L. Tang, Y. Dong and J. Liu, "Differential Evolution With an Individual-Dependent Mechanism," *IEEE Transactions on Evolutionary Computation,* vol. 19, pp. 560-574, 2015.

[10] J. Holland, "Genetic algorithms," *Scientific American,* vol. 267, pp. 66-72, 1992.

[11] Z. Yang, K. Tang and X. Yao, "Differential evolution for high-dimensional function optimization," *Evolutionary Computation .cec .ieee Congress on,* pp. 3523-3530, 2007.

[12] 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 Transactions on Cybernetics,* vol. 45, pp. 1851-1863, 2014.

[13] X. Wen, W. Chen, Y. Lin, T. Gu, H. Zhang, Y. Li, Y. Yin, and J. Zhang, "A Maximal Clique Based Multiobjective Evolutionary Algorithm for Overlapping Community Detection," *IEEE Transactions on Evolutionary Computation,* p. 1-1.

[14] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *IEEE International Conference on Neural Networks, 1995. Proceedings*, 1995, pp. 1942-1948 vol.4.

[15] W. N. Chen, J. Zhang, Y. Lin, and N. Chen, "Particle Swarm Optimization With an Aging Leader and Challengers," *IEEE Transactions on Evolutionary Computation,* vol. 17, pp. 241-258, 2013.

[16] W. N. Chen, J. Zhang, H. S. H. Chung, and W. L. Zhong, "A Novel Set-Based Particle Swarm Optimization Method for Discrete Optimization Problems," *IEEE Transactions on Evolutionary Computation,* vol. 14, pp. 278-300, 2010.

[17] Q. Yang, W. N. Chen, Z. Yu, and T. Gu, "Adaptive Multimodal Continuous Ant Colony Optimization," *IEEE Transactions on Evolutionary Computation,* p. 1-1, 2016.

[18] Q. Yang, W. N. Chen, Y. Li, C. L. Chen, X. M. Xu, and J. Zhang, "Multimodal Estimation of Distribution Algorithms.," *Cybernetics IEEE Transactions on,* pp. 1-15, 2016.

[19] Y. Sun, M. Kirley and S. K. Halgamuge, "Extended Differential Grouping for Large Scale Global Optimization with Direct and Indirect Variable Interactions," in *The Genetic and Evolutionary Computation Conference*, 2015, pp. 313-320.

[20] Y. Mei, X. Yao, X. Li, and M. N. Omidvar, "A Competitive Divide-and-Conquer Algorithm for Unconstrained Large Scale Black-Box Optimization," *Acm Transactions on Mathematical Software,* 2015.

[21] Z. Yang, T. Ke and Y. Xin, "Large scale evolutionary optimization using cooperative coevolution ☆," *Information Sciences,* vol. 178, pp. 2985-2999, 2008.

[22] T. Ray and X. Yao, "A cooperative coevolutionary algorithm with Correlation based Adaptive Variable Partitioning," in *Evolutionary Computation, 2009. CEC '09. IEEE Congress on*, 2009, pp. 983-989.

[23] M. N. Omidvar, X. Li, Z. Yang, and X. Yao, "Cooperative Co-evolution for large scale optimization through more frequent random grouping," in *Evolutionary Computation*, 2010, pp. 1 - 8.

[24] M. N. Omidvar, X. Li, Z. Yang, and X. Yao, "Cooperative Co-evolution for large scale optimization through more frequent random grouping," in *Evolutionary Computation*, 2010, pp. 1 - 8.

[25] A. K. Qin and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, 2005, pp. 1785-1791.

[26] J. Brest, S. Greiner, B. Boskovic, and M. Mernik, "Self-Adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems," *Evolutionary Computation IEEE Transactions on,* vol. 10, pp. 646-657, 2006.

[27] Z. Yang, K. Tang and X. Yao, "Self-adaptive differential evolution with neighborhood search," in *Evolutionary Computation*, 2008, pp. 1110-1116.

[28] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Transactions on Evolutionary Computation,* vol. 13, pp. 526-553, 2010.

[29] J. Zhang and A. C. Sanderson, "JADE: Adaptive Differential Evolution With Optional External Archive," *Evolutionary Computation IEEE Transactions on,* vol. 13, pp. 945-958, 2009.

[30] M. Leon and N. Xiong, "Greedy adaptation of control parameters in differential evolution for global optimization problems," in *IEEE Conference on Evolutionary Computation*, 2015, pp. 169-172.

[31] 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," *Nature Inspired Computation & Applications Laboratory,* pp. 1-23, 2010.

TABLE I. COMPARISON OF THE MULTI-OPTIMIZER DECC-DG WITH DECC-DG WITH SANSDE AND SINGLE-OPTIMIZER DECC-DGS

DG							
		Multi optimizer	***SaNSDE***	***current to best/1***	***current to best/2***	***rand/1***	***rand/2***
Function 1	Mean	6.5493E+03	2.392E+03	1.5820E+07	7.8314E+04	7.2968E-01	4.3901E+02
	Std	1.6065E+04	4.058E+03	9.0601E+06	4.1554E+05	1.2929E+00	1.1841E+03
	p-value	-	6.952E-01	**3.0199E-11^{+}**	**2.2658E-03^{+}**	3.0199E-11^{-}	1.4298E-05^{-}
Function 2	Mean	4.1170E+03	4.429E+03	7.4058E+03	7.6327E+03	1.5795E+03	1.6425E+03
	Std	3.2375E+02	1.121E+02	2.6166E+02	8.6720E+02	1.3249E+02	9.7337E+01
	p-value	-	**2.388E-04^{+}**	**3.0199E-11^{+}**	**3.0199E-11^{+}**	3.0199E-11^{-}	3.0199E-11^{-}
Function 3	Mean	1.6391E+01	1.670E+01	1.9458E+01	1.6091E+01	5.8154E+00	4.7031E+00
	Std	3.1249E-01	4.042E-01	1.3834E-01	1.8521E+00	4.1293E-01	4.6399E-01
	p-value	-	**9.031E-04^{+}**	**3.0199E-11^{+}**	3.1466E-02^{-}	3.0199E-11^{-}	3.0199E-11^{-}
Function 4	Mean	8.9690E+12	5.240E+12	1.9026E+12	1.8484E+13	1.1554E+14	1.2816E+14
	Std	3.7613E+12	1.457E+12	7.3794E+11	4.9761E+12	2.3207E+13	3.1410E+13
	p-value	-	1.019E-05^{-}	3.0199E-11^{-}	**6.5183E-09^{+}**	**3.0199E-11^{+}**	**3.0199E-11^{+}**
Function 5	Mean	9.6216E+07	1.484E+08	1.4432E+08	1.2977E+08	1.1477E+08	1.2517E+08
	Std	1.8169E+07	2.210E+07	2.8289E+07	1.7031E+07	1.8249E+07	1.7588E+07
	p-value	-	**1.612E-10^{+}**	**7.1152E-09^{+}**	**6.0079E-08^{+}**	**4.7129E-04^{+}**	**9.5299E-07^{+}**
Function 6	Mean	1.6047E+01	1.630E+01	1.7405E+06	1.5553E+01	5.4412E+00	4.5045E+00
	Std	4.7979E-01	3.369E-01	7.0714E+05	1.8267E+00	4.8181E-01	3.7994E-01
	p-value	-	5.188E-02	**2.9860E-11^{+}**	1.6798E-03^{-}	3.0199E-11^{-}	3.0199E-11^{-}
Function 7	Mean	1.5841E+04	1.027E+04	2.9636E+03	8.9516E+05	6.2188E+08	3.8474E+09
	Std	1.8286E+04	8.511E+03	7.6470E+03	1.0923E+05	1.8962E+08	9.8731E+08
	p-value	-	7.618E-01	2.7829E-07^{-}	**3.0199E-11^{+}**	**3.0199E-11^{+}**	**3.0199E-11^{+}**
Function 8	Mean	5.5508E+06	3.331E+07	7.9735E+05	6.4698E+06	5.7077E+07	5.1104E+07
	Std	4.3243E+06	2.719E+07	1.6219E+06	4.7645E+06	2.7132E+07	2.9687E+07
	p-value	-	**4.200E-10^{+}**	3.3228E-08^{-}	3.1830E-01	**3.0199E-11^{+}**	**3.0199E-11^{+}**
Function 9	Mean	9.2259E+07	5.586E+07	3.5385E+07	1.7424E+08	1.0030E+09	1.0898E+09
	Std	7.9389E+06	7.019E+06	1.1201E+07	1.4562E+07	5.0675E+07	8.9721E+07
	p-value	-	3.020E-11^{-}	3.3384E-11^{-}	**3.0199E-11^{+}**	**3.0199E-11^{+}**	**3.0199E-11^{+}**
Function 10	Mean	3.3639E+03	4.550E+03	4.7292E+03	5.8988E+03	3.1899E+03	4.4511E+03
	Std	2.2012E+02	1.249E+02	1.6241E+02	3.1330E+02	1.9647E+02	2.2203E+02
	p-value	-	**3.020E-11^{+}**	**3.0199E-11^{+}**	**3.0199E-11^{+}**	5.8282E-03^{-}	**3.0199E-11^{+}**
Function 11	Mean	1.0692E+01	1.020E+01	3.7484E+01	8.2833E+00	2.2941E+00	3.6372E+00
	Std	8.9005E-01	8.778E-01	2.0647E+00	1.4151E+00	4.5082E-01	3.8277E-01
	p-value	-	4.515E-02^{-}	**3.0199E-11^{+}**	5.5329E-08^{-}	3.0199E-11^{-}	3.0199E-11^{-}
Function 12	Mean	2.3389E+03	2.803E+03	1.1492E+04	8.5738E+03	2.8752E+05	3.8822E+05
	Std	4.8567E+02	9.205E+02	6.1795E+03	8.3287E+02	1.9599E+04	2.0796E+04
	p-value	-	**2.151E-02^{+}**	**3.6897E-11^{+}**	**3.0199E-11^{+}**	**3.0199E-11^{+}**	**3.0199E-11^{+}**
Function 13	Mean	5.1251E+03	5.081E+03	8.5640E+03	1.0935E+04	4.0351E+03	1.0408E+05
	Std	3.2128E+03	3.459E+03	4.0711E+03	4.6067E+03	2.6216E+03	3.3407E+04
	p-value	-	8.766E-01	**9.5207E-04^{+}**	**1.3853E-06^{+}**	1.8090E-01	**3.0199E-11^{+}**
Function 14	Mean	3.7814E+08	3.413E+08	7.9230E+07	5.6933E+08	2.8601E+09	2.9648E+09
	Std	2.4956E+07	2.522E+07	7.4440E+06	2.7128E+07	1.5463E+08	1.6052E+08
	p-value	-	4.744E-06^{-}	3.0199E-11^{-}	**3.0199E-11^{+}**	**3.0199E-11^{+}**	**3.0199E-11^{+}**
Function 15	Mean	3.1117E+03	5.858E+03	2.8469E+03	3.4036E+03	5.3612E+03	5.7949E+03
	Std	1.2830E+02	7.669E+01	1.4284E+02	1.7158E+02	1.5953E+02	2.0166E+02
	p-value	-	**3.020E-11^{+}**	3.6459E-08^{-}	**1.8500E-08^{+}**	**3.0199E-11^{+}**	**3.0199E-11^{+}**
Function 16	Mean	2.9302E-02	7.854E-13	3.5895E+01	2.0477E-09	1.4899E-09	5.2340E-04
	Std	1.6049E-01	7.854E-14	4.0588E+00	2.2811E-10	1.4296E-10	6.4441E-05
	p-value	-	5.449E-10^{-}	**2.9506E-11^{+}**	5.4608E-10^{-}	5.4608E-10^{-}	5.4608E-10^{-}
Function 17	Mean	3.7225E+04	4.036E+04	1.2189E+02	8.2180E+04	8.0258E+05	1.0102E+06
	Std	3.5008E+03	2.627E+03	2.4827E+01	4.7476E+03	3.5993E+04	2.8669E+04
	p-value	-	**4.218E-04^{+}**	3.0199E-11^{-}	**3.0199E-11^{+}**	**3.0199E-11^{+}**	**3.0199E-11^{+}**
Function 18	Mean	1.3019E+10	1.422E+10	3.1141E+10	2.2909E+10	3.1642E+09	1.7228E+10
	Std	2.0613E+09	2.147E+09	4.9082E+09	3.4440E+09	7.8626E+08	2.0347E+09
	p-value	-	5.012E-02	**3.0199E-11^{+}**	**1.4643E-10^{+}**	3.0199E-11^{-}	**1.1023E-08^{+}**
Function 19	Mean	2.0981E+06	1.744E+06	1.5661E+06	2.6572E+06	1.3669E+07	1.6446E+07
	Std	1.6202E+05	7.731E+04	1.2314E+05	1.6195E+05	3.2640E+06	1.5925E+06
	p-value	-	3.020E-11^{-}	3.0199E-11^{-}	**1.6132E-10^{+}**	**3.0199E-11^{+}**	**3.0199E-11^{+}**
Function 20	Mean	4.9609E+10	4.593E+10	6.6937E+10	8.4712E+10	5.9734E+10	1.1979E+11
	Std	8.8147E+09	6.369E+09	1.1512E+10	1.1222E+10	6.7087E+09	1.3007E+10
	p-value	-	6.787E-02	**2.3768E-07^{+}**	**3.6897E-11^{+}**	**1.9963E-05^{+}**	**3.0199E-11^{+}**
b/e/w			**8/6/6**	**12/0/8**	**16/1/3**	**11/1/8**	**14/0/6**

TABLE II. COMPARISON OF THE MULTI-OPTIMIZER DECC WITH CC-SANSDE USING XDG, GDG, DELTA GROUPING AS DECOMPOSITION STRATEGY

		XDG		GDG		Delta Grouping	
		Multi optimizer	***SaNSDE***	***Multi optimizer***	***SaNSDE***	***Multi optimizer***	***SaNSDE***
Function 1	Mean	3.672E+04	3.427E+03	3.813E+01	4.192E+03	0.000E+00	1.549E-24
	Std	1.866E+05	8.006E+03	6.936E+01	2.267E+04	0.000E+00	8.484E-24
	p-value	-	1.715E-01	-	8.236E-02	-	8.152E-02
Function 2	Mean	4.246E+03	4.469E+03	4.158E+03	4.402E+03	2.598E+02	2.939E+02
	Std	3.773E+02	1.234E+02	2.646E+02	1.504E+02	5.787E+01	2.507E+01
	p-value	-	**7.617E-03**$^{+}$	-	**2.531E-04**$^{+}$	-	1.055E-01
Function 3	Mean	1.639E+01	1.666E+01	1.628E+01	1.663E+01	1.235E-13	1.260E-13
	Std	3.353E-01	3.640E-01	3.281E-01	3.697E-01	5.877E-15	4.545E-15
	p-value	-	**3.501E-03**$^{+}$	-	**2.006E-04**$^{+}$	-	7.033E-02
Function 4	Mean	2.848E+11	7.213E+11	8.942E+11	8.576E+11	2.610E+12	3.641E+12
	Std	1.252E+11	1.794E+11	3.601E+11	3.250E+11	8.785E+11	1.415E+12
	p-value	-	**1.613E-10**$^{+}$	-	7.958E-01	-	**3.183E-03**$^{+}$
Function 5	Mean	1.004E+08	1.550E+08	9.580E+07	1.390E+08	2.407E+08	2.465E+08
	Std	1.674E+07	2.125E+07	2.114E+07	2.040E+07	5.708E+07	5.846E+07
	p-value	-	**8.153E-11**$^{+}$	-	**9.260E-09**$^{+}$	-	6.952E-01
Function 6	Mean	1.614E+01	1.629E+01	1.607E+01	1.626E+01	3.908E-09	5.329E-09
	Std	3.697E-01	3.273E-01	4.144E-01	3.590E-01	1.084E-09	1.807E-09
	p-value	-	1.055E-01	-	**4.515E-02**$^{+}$	-	**1.030E-03**$^{+}$
Function 7	Mean	1.197E+03	9.826E+02	8.060E+01	2.103E+02	4.770E+08	4.332E+08
	Std	1.619E+03	1.284E+03	1.099E+02	7.250E+02	2.327E+08	2.483E+08
	p-value	-	8.073E-01	-	3.042E-01	-	3.183E-01
Function 8	Mean	3.987E+05	3.987E+05	5.321E+05	6.646E+05	4.483E+07	1.093E+08
	Std	1.216E+06	1.216E+06	1.378E+06	1.511E+06	5.089E+07	8.779E+07
	p-value	-	2.921E-02^{-}	-	5.395E-01	-	**4.353E-05**$^{+}$
Function 9	Mean	1.352E+08	1.116E+08	1.039E+08	6.949E+07	5.562E+07	5.936E+07
	Std	1.625E+07	1.117E+07	1.266E+07	7.284E+06	6.102E+06	6.269E+06
	p-value	-	4.444E-07^{-}	-	3.690E-11^{-}	-	**2.151E-02**$^{+}$
Function 10	Mean	3.954E+03	5.256E+03	3.455E+03	4.682E+03	1.238E+04	1.307E+04
	Std	2.147E+02	1.480E+02	2.096E+02	1.124E+02	3.137E+02	2.221E+02
	p-value	-	**3.020E-11**$^{+}$	-	**3.020E-11**$^{+}$	-	**5.072E-10**$^{+}$
Function 11	Mean	1.082E+01	1.067E+01	1.033E+01	1.062E+01	5.860E-02	1.508E-13
	Std	7.858E-01	7.932E-01	1.061E+00	7.279E-01	2.230E-01	9.362E-15
	p-value	-	6.204E-01	-	1.297E-01	-	2.587E-03^{-}
Function 12	Mean	1.212E+04	1.276E+04	4.333E+03	4.252E+03	3.967E+06	4.281E+06
	Std	2.775E+03	1.771E+03	1.398E+03	9.452E+02	2.325E+05	2.337E+05
	p-value	-	8.771E-02	-	6.520E-01	-	**3.157E-05**$^{+}$
Function 13	Mean	3.489E+03	3.297E+03	1.201E+03	1.366E+03	1.444E+03	1.230E+03
	Std	1.594E+03	1.254E+03	7.338E+02	7.496E+02	7.436E+02	4.906E+02
	p-value	-	8.534E-01	-	2.009E-01	-	2.707E-01
Function 14	Mean	5.751E+08	5.870E+08	4.566E+08	4.552E+08	1.879E+08	2.012E+08
	Std	3.900E+07	3.626E+07	2.927E+07	2.500E+07	1.096E+07	1.553E+07
	p-value	-	2.519E-01	-	9.470E-01	-	**8.120E-04**$^{+}$
Function 15	Mean	3.639E+03	6.351E+03	3.327E+03	6.077E+03	1.482E+04	1.595E+04
	Std	1.532E+02	9.595E+01	1.150E+02	8.087E+01	3.026E+02	3.073E+02
	p-value	-	**3.020E-11**$^{+}$	-	**3.020E-11**$^{+}$	-	**3.020E-11**$^{+}$
Function 16	Mean	3.724E-09	1.778E-08	2.930E-02	5.379E-11	7.656E-02	2.217E-13
	Std	4.265E-10	1.540E-09	1.605E-01	5.383E-12	2.931E-01	1.521E-14
	p-value	-	**3.020E-11**$^{+}$	-	5.573E-10^{-}	-	6.230E-08^{-}
Function 17	Mean	1.130E+05	1.277E+05	6.551E+04	7.363E+04	6.383E+06	7.450E+06
	Std	6.260E+03	5.085E+03	4.441E+03	4.102E+03	3.512E+05	3.258E+05
	p-value	-	**6.121E-10**$^{+}$	-	**3.081E-08**$^{+}$	-	**8.993E-11**$^{+}$
Function 18	Mean	1.299E+03	1.359E+03	1.085E+03	1.250E+03	2.161E+03	1.990E+03
	Std	1.627E+02	1.371E+02	1.577E+02	1.220E+02	5.567E+02	6.487E+02
	p-value	-	1.624E-01	-	**4.943E-05**$^{+}$	-	1.154E-01
Function 19	Mean	2.116E+06	1.731E+06	2.293E+06	1.880E+06	1.854E+07	1.884E+07
	Std	1.388E+05	1.048E+05	1.909E+05	1.084E+05	1.558E+06	1.197E+06
	p-value	-	7.389E-11^{-}	-	8.993E-11^{-}	-	5.201E-01
Function 20	Mean	1.441E+05	1.051E+07	8.229E+03	1.413E+04	1.243E+03	1.176E+03
	Std	3.779E+05	5.590E+07	1.041E+04	3.038E+04	1.165E+02	8.566E+01
	p-value	-	2.399E-01	-	1.809E-01	-	1.273E-02-
b/e/w		**8/9/3**		**8/9/3**		**9/8/3**	