Differential Grouping with Spectral Clustering for Large Scale Global Optimization

Lin Li[1], Wei Fang[1,*], Quan Wang[2], Jun Sun[1]
[1] *Jiangsu Provincial Engineering Laboratory of Pattern Recognition and Computational Intelligence*
Department of Computer Science and Technology
Jiangnan University
Wuxi, China
fangwei@jiangnan.edu.cn

[2] *Wuxi SensingNet Industrialization Research Institute*
Wuxi, China
seepwq@163.com

***Abstract*—Cooperative co-evolution (CC) is an effective strategy for large scale global optimization (LSGO) problems, which divides the problem into several smaller sub-problems based on the idea of divide-and-conquer. The grouping of decision variables has an important impact on the optimization results. Differential grouping (DG) and its improved versions detect the interaction relationship between variables according to the differential values, achieving competitive grouping results on the CEC'2010 benchmark functions which have uniform subcomponent sizes. However, these algorithms still have difficulty to obtain good decomposition for the more complex functions, especially for overlapping functions. Due to some degree of overlap between subcomponents, there will be no unique optimal decomposition of the decision variables. We draw lessons from the idea of clustering that objects in the same cluster are similar to each other and objects in different clusters are quite different. In this paper, we propose a differential grouping with spectral clustering (DGSC) algorithm. All the variables are treated as edge-connected points in space to construct an undirected weighted graph. The design structure matrix derived from the differential values is used as the similarity matrix of spectral clustering such that the weight of edges is expressed by the interaction relationship between these variables. This graph is divided into several subcomponents by clustering to decompose decision variables, making the degree of interaction between subcomponents as weak as possible and within the subcomponent as strong as possible. The experimental results show that DGSC has promising performance on the latest LSGO benchmark functions.**

Keywords—large scale global optimization, spectral clustering, differential grouping, cooperative co-evolution

I. INTRODUCTION

In recent years, numerous large scale global optimization (LSGO) problems arise in the field of science or engineering, usually with thousands or even more decision variables [1,2]. There are four main challenges for solving LSGO problems [3]. The first challenge is as the dimension of variables increases, the search space becomes much more complex and the number of local optimal solutions grows exponentially. The second challenge is that the properties of the search space may change along with the scale of decision variables increases. Thirdly, the evaluation of large scale problems is often costly. Another challenge is the interaction between variables contributes to the difficulty of large scale optimization. In order to overcome these challenges, a great deal of approaches for solving LSGO problems have arisen.

Cooperative co-evolution (CC) [4] is an effective method for solving LSGO problems. The CC framework is based on the divide-and-conquer strategy [5], which decomposes the problem to several sub-problems, and applies a heuristic algorithm to solve each sub-problem. Finally, the solutions of these sub-problems are combined to obtain the solution of the original problem. Since the CC framework can embed multiple heuristic algorithms [6,7] and has good robustness, it is very suitable for solving LSGO problems. The grouping of variables has a significant impact on the final optimization result [8]. A better optimization result can be obtained by placing interactive variables in the same group and independent variables into different groups.

The grouping strategies can divide into three categories, including fixed grouping, random grouping, and grouping based on learning mechanisms. Fixed grouping means that the variable composition and size of each group are fixed, usually set manually, and remain unchanged throughout the optimization process [9]. Since the fixed grouping does not consider the interaction relationship between variables, it is far from the ideal grouping. On this basis, random grouping is proposed with the aim of placing interactive decision variables in the same group as much as possible. Yang et al. developed a multilevel cooperative coevolution method [10], which assigns a set of group sizes and use a self-adapted mechanism to select a decomposer according to its historical performance at the start of each cycle. Li et al. designed a PSO algorithm based on cooperative coevolution framework (CCPSO2) [11], which randomly distributes decision variables into each group to improve the probability of placing interactive variables in the same group. With the increase of the number of decision variables, random grouping has been difficult to solve these

978-1-7281-2153-6/19/$31.00 ©2019 IEEE

problems [12]. The grouping strategy based on learning mechanism comes into being, with the interaction relationship between variables obtained to realize the automatic grouping of variables. Chen et al. raised a two-stage optimization method (CCVIL) [13], which takes part of the fitness evaluations for grouping, and then combined with JADE for optimization according to the grouping result. Omidvar et al. proposed a differential grouping (DG) algorithm [14] that achieves higher grouping accuracy than CCVIL. Mei et al. developed a global differential grouping (GDG) algorithm [15], which obtains the threshold based on the fitness values of the randomly sampled K points. Ling et al. designed a graph-based differential grouping (gDG) algorithm, which applies graph theory to construct a modeling for the decomposition problem [16]. Omidvar et al. raised an improved version of DG, namely DG2 [8], markedly improves the efficiency and grouping accuracy. Sun et al. proposed a recursive differential grouping (RDG) [17] and an adaptive threshold parameter estimation with recursive differential grouping (RDG2) [18], achieving lower time complexity and higher decomposition accuracy. The existing grouping algorithms based on learning mechanisms have obtained good grouping results on the CEC'2010 benchmark functions with uniform subcomponent sizes [19]. However, the CEC'2013 benchmark functions introduce features of imbalance in the contribution of subcomponents, nonuniform subcomponent sizes, conforming and conflicting overlapping functions, and new transformations to the base functions [20]. These new features largely increase the difficulty of large scale problems, resulting in the CEC'2013 benchmark functions not being properly solved.

Among these new features mentioned above, overlapping contributes great to the difficulty of grouping. Due to some degree of overlap between subcomponents, there will be no unique optimal decomposition of the decision variables. In order to solve this problem, we learn from the idea of clustering that objects in the same cluster are very similar while objects in different clusters are quite different. A differential grouping with spectral clustering (DGSC) algorithm is proposed in this paper. Spectral clustering is an optimal graph partition problem based on spectral graph theory [21]. It treats all the data as edge-connected points in space to construct an undirected weighted graph and the weight of the edge is the similarity value between two points. Using certain graph cut method, the graph is divided into several connected subcomponents. In this paper, we regard the decision variables as points in space and take the interaction relationship between these variables as the weight of the edge. DGSC combines differential grouping with spectral clustering by using the design structure matrix derived from the differential values as the similarity matrix of spectral clustering to output the grouping result of decision variables. Then SaNSDE [22] is used as the subcomponent optimizer in the CC framework for optimization. The resultant algorithm, called DGSC-DECC, has achieved promising results on the CEC'2013 benchmark functions.

The rest of this paper is organized as follows: Section II introduces the theory and related algorithms of differential grouping. Section III proposes a differential grouping with spectral clustering algorithm. Section IV contains the experimental design and experimental results. Finally, Section V concludes the paper.

II. Background and Related Work

A. Definitions

The mathematical formulation of LSGO problems is stated as follows:

$$\min_{\mathbf{x}\in R^n} f(\mathbf{x}) \tag{1}$$

where $f: R^n \to R$ is a real value objective function, $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ is a *n*-dimensional decision vector, *n* is the dimension of variables.

Definition 1 ([3])**.** A function $f(\mathbf{x})$ is partially separable with *m* independent subcomponents iff:

$$\arg\min_{\mathbf{x}} f(\mathbf{x}) = \left(\arg\min_{\mathbf{x}_1} f(\mathbf{x}_1,\ldots),\ldots,\arg\min_{\mathbf{x}_m} f(\ldots,\mathbf{x}_m)\right) \tag{2}$$

where $\mathbf{x} = (x_1, x_2, \cdots, x_n)$ is a decision vector of *n* dimensions, $\mathbf{x}_1,\ldots,\mathbf{x}_m$ are disjoint sub-vectors of $\mathbf{x}$, and $2 \le m \le n$. If $m = n$, function $f(\mathbf{x})$ is fully separable.

Definition 2 ([3])**.** A function $f(\mathbf{x})$ is fully non-separable if all decision variables interact with each other.

Definition 3 ([3])**.** A function $f(\mathbf{x})$ is partially additively separable iff:

$$f(\mathbf{x}) = \sum_{i=1}^{m} f_i(\mathbf{x}_i) \tag{3}$$

where *m* is the number of independent subcomponents, and $f_i(\cdot)$ is corresponding sub-function. The definitions of $\mathbf{x}$ and $\mathbf{x}_i$ are the same as those in definition 1.

B. Differential Grouping

DG is an automatic decomposition strategy for LSGO problems proposed by Omidvar et al [14]. The following theorem is the core idea of DG to detect the interaction between decision variables.

Theorem 1 ([3])**.** Given an additively separable function $f(\mathbf{x})$, $\forall a, b_1 \ne b_2, \delta \in \mathbb{R}, \delta \ne 0$, variables x_i and x_j interact if the following condition holds:

$$\Delta_{\delta,x_i}[f](\mathbf{x})|_{x_i=a,x_j=b_1} \ne \Delta_{\delta,x_i}[f](\mathbf{x})|_{x_i=a,x_j=b_2} \tag{4}$$

where

$$\Delta_{\delta,x_i}[f](\mathbf{x}) = F(\ldots,x_i+\delta,\ldots) - F(\ldots,x_i,\ldots) \tag{5}$$

refers to the forward difference of f with respect to variable x_i with interval δ.

Theorem 1 states a definition of direct interaction, denoted as $x_i \leftrightarrow x_j$. In addition, there is a case of indirect interaction [23]. That is, when x_i and x_j do not satisfy the condition of direct interaction and $\exists$ a set of decision variables

$\{x_{k1},\ldots,x_{kt}\} \subset \mathbf{x}$, such that $x_i \leftrightarrow x_{k1} \leftrightarrow \cdots \leftrightarrow x_{kt} \leftrightarrow x_j$. Given a function $f(\mathbf{x}) = x_1x_2 + x_2x_3 + x_4^2$, $\mathbf{x} \in [-1,1]^4$, where x_1 and x_2 are direct interacted, x_1 and x_3 are indirect interacted, x_1 and x_4 are independent. Fig.1 shows a schematic diagram of the relationship between these variables.

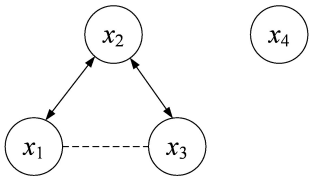

Fig. 1. Diagram of interaction relationship between variables.

DG detects the interaction between variables according to the principle of differential values, and can obtain an approximate optimal decomposition without the prior knowledge. However, the algorithm has three major drawbacks [15]. Firstly, some interaction may be missed due to premature deletion of variables. Second, the threshold ε is artificially selected and have an apparent impact on grouping accuracy. Thirdly, all separable variables are placed in a group, lacking efficient decomposition for fully separable and approximate fully separable functions. In order to solve the above problems, two improved algorithms GDG and DG2 have been proposed in [15] and [8].

C. Drawbacks in GDG and DG2

GDG has improved the grouping accuracy based on DG by introducing the design structure matrix and the global threshold technique. The global threshold takes values according to the minimum value of function values of the randomly generated K points, and considers computational errors to solve the sensitivity issue. However, the use of global threshold to detect interaction is not suitable for unbalanced functions with imbalance in the contribution of each subcomponent. DG2 proposes a faster and more accurate differential grouping algorithm, which reuses existing sample points to reduce computational cost and automatically calculates thresholds without artificial settings. Its main drawback is the lack of efficient decomposition for fully separable functions and overlapping functions, since all variables are placed in the same group.

Despite GDG and DG2 have achieved good grouping results for the CEC'2010 benchmark functions, they still have difficulty to get good grouping results for the CEC'2013 benchmark functions, especially for the overlapping functions. The subcomponents of overlapping functions have some degree of overlap with its neighboring subcomponents. According to the degree of overlap, three overlapping functions included in the CEC'2013 benchmark functions can be divided into two types. The first type of overlapping functions is f_{12}, in which each dimension of variable only directly interact to its previous and next variables, approximately close to the fully separable function. The second type of overlapping functions, including f_{13} and f_{14}, can be divided into 20 subcomponents with 5 dimensions of shared variables between any two adjacent subcomponents. The shared decision variables between two subcomponents in f_{13} have the same optimum value with respect to both sub-functions while that in f_{14} have a different optimum value. The existing grouping algorithms cannot find an effective decomposition of overlapping functions, usually put all variables in a single group to optimize, resulting in poor optimization values. For the overlapping functions, we do not consider whether the grouping is completely correct, but based on the interaction relationship between variables, using the idea of spectral clustering to group. This issue is discussed further in Section III.

III. Differential Grouping with Spectral Clustering

A. Spectral Clustering

Spectral clustering is a widely used clustering algorithm that evolved from algebraic graph theory [24]. It is simple to implement and performs well on high-dimensional data [25]. According to different criterion functions and spectral mapping methods, there are many implementations of spectral clustering. The specific spectral clustering method used in this paper is as follows.

Spectral clustering treats all the data as edge-connected points in space to construct an undirected weighted graph with the weight of the edge being the similarity value between two points. For a graph G, it can be described by a set of points V and a set of edges E, denoted as G (V, E), where $V = (v_1, v_2, ..., v_n)$ is all points in the data set. Any two points in V can be connected with or without edges.

Step 1. Obtain the similarity matrix W, in which w_{ij} is the weight between point v_i and point v_j, and $w_{ij} = w_{ji}$. If v_i and v_j are connected by a edge, then $w_{ij} > 0$. If not, $w_{ij} = 0$.

Step 2. Calculate the degree matrix D. For any point in the graph, its degree is defined as the sum of the weights of all the edges connected to it, that is

$$d_i = \sum_{j=1}^{n} w_{ij} . \tag{6}$$

Step 3. Compute unnormalized Laplacian matrix L.

$$L = D - W . \tag{7}$$

Step 4. Calculate the eigenvectors of L corresponding to the k smallest eigenvalues and construct eigenvector space.

Step 5. Use the kmeans algorithm to cluster eigenvectors in the eigenvector space, dividing the graph into several subgraphs. The sum of the edge weights between different subgraphs is as small as possible, and that inside the subgraphs is as large as possible, so as to achieve the purpose of clustering.

B. Differential Grouping with Spectral Clustering

Due to some degree of overlap between the subcomponents, there will be no unique optimal grouping of decision variables for the overlapping functions. We draw lessons from the idea of clustering that objects in the same cluster are very similar and objects in different clusters are quite different, combining

differential grouping with spectral clustering to solve this problem. In this section, DGSC is proposed for LSGO problems, with the purpose of achieving the highest degree of interaction within the subcomponent and the lowest degree of interaction among the subcomponents.

DGSC regards the decision variables as points in space, and take the interaction relationship between these variable as the weight of the edge, where the value is 1 when directly interacted, otherwise is 0. It combines differential grouping with spectral clustering by using the design structure matrix (DSM) $\mathbf{\Theta}$ derived from the differential values as the similarity matrix of spectral clustering. First of all, the matrix $\mathbf{\Theta}$ should be calculated. Among the existing algorithms, DG2 has the highest accuracy of matrix $\mathbf{\Theta}$, so we use this method to calculate it. Secondly, the degree matrix D is obtained as the sum of the weights of all the edges connected to it. After that, an unnormalized Laplacian matrix L is computed from the difference of D and W. Then the eigenvectors of L corresponding to the k smallest eigenvalues and construct eigenvector space are obtained. Finally, using the kmeans algorithm to cluster eigenvectors in the eigenvector space. Assuming an objective function has n decision variables, if we want to cluster its decision variables into k groups, the procedure of DGSC can be seen in Algorithm 1.

Algorithm 1: $(\mathbf{x}_1,\ldots,\mathbf{x}_k) = \mathrm{DGSC}(\mathbf{\Theta},k)$

$\mathbf{W} = \mathbf{\Theta}$;
$\mathbf{D} = \mathbf{0}_{n\times n}$;
for $i = 1 \to n$ **do**
    $\mathbf{D}(i,i) = \sum_{j=1}^{n} w_{ij}$;
end
$\mathbf{L} = \mathbf{D} - \mathbf{W}$;
$(\mathbf{Q},\mathbf{V}) = \mathrm{eigs}(\mathbf{L},k,'SA')$;
$\mathbf{C} = \mathrm{kmeans}(\mathbf{Q},k)$;
$groups = \mathrm{cell}(1,k)$;
for $i = 1 \to k$
    $groups\{i\} = \mathrm{find}(\mathbf{C} == i)$;
end

An example of overlapping function is given below. Given an object function $f(\mathbf{x}) = x_1x_2x_3x_4 + x_4x_5x_6$ with unique lower bound of -1 and upper bound of 1 for all the decision variables. The design structure matrix $\mathbf{\Theta}$ is shown in (8) and the graph corresponding to the adjacency matrix (8) is shown in Fig.2. This function has two overlapping subcomponents (x_1,x_2,x_3,x_4) and (x_4,x_5,x_6) with a shared decision variable x_4, where the grouping of x_4 is critical. According to DGSC, variables are grouped into (x_1,x_2,x_3,x_4) and (x_5,x_6), achieving the highest degree of interaction within the subcomponent and the lowest degree of interaction between the subcomponents.

$$\mathbf{\Theta} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}. \tag{8}$$

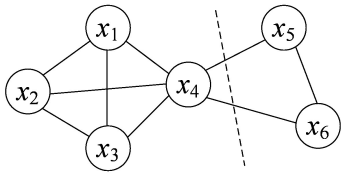

Fig. 2. The graph corresponding to the adjacency matrix (8).

A resultant algorithm named DGSC-DECC is presented in this section, which includes the three parts shown in Algorithm 2. Firstly, a design structure matrix representing the interaction relationship between decision variables is derived through differential values. Secondly, using the idea of spectral clustering, the decision variables are grouped according to the design structure matrix. Lastly, SaNSDE is used as the subcomponent optimizer under the CC framework to obtain the final optimization result.

Algorithm 2: $(\mathbf{x}^*, f^*) = \mathrm{DGSC\text{-}DECC}(f,n,\overline{\mathbf{x}},\underline{\mathbf{x}})$

// Calculating the matrix $\mathbf{\Theta}$
$(\mathbf{\Theta}) = \mathrm{DSM}(f,n,\overline{\mathbf{x}},\underline{\mathbf{x}})$;
// Grouping by spectral clustering
$(\mathbf{x}_1,\ldots,\mathbf{x}_k) = \mathrm{DGSC}(\mathbf{\Theta},k)$;
// Optimizing each subcomponent
$(\mathbf{x}^*, f^*) = \mathrm{DECC}(k,\mathbf{x}_1,\ldots,\mathbf{x}_k)$

IV. Experimental Results and Analysis

This section presents the effectiveness of DGSC in solving LSGO problems. The optimization results are compared with that of several state-of-the-art decomposition algorithms, namely DG, GDG and DG2. For the sake of fairness, all of these algorithms use SaNSDE as the subcomponent optimizer in the CC framework during the optimization stage.

A. Experimental Design

In order to verify the performance of DGSC algorithm, it is evaluated on the CEC'2013 benchmark functions, which includes 15 1000-dimensional benchmark functions that can be divided into the following four categories:

1) Fully separable functions (f_1-f_3);
2) Partially separable functions
   · Functions with a separable subcomponent (f_4-f_7);
   · Functions with no separable subcomponent (f_8-f_{11});

3) Overlapping functions (f_{12}-f_{14});

4) Nonseparable functions (f_{15}).

SaNSDE is used as the subcomponent optimizer in this paper, which is a variant of differential evolution (DE) [26]. The population size is set to 50 as suggested in [22] and the maximum number of fitness evaluations is 3×10^6 as suggested in [20]. All experimental results for each algorithm are based on 25 independent runs. The DGSC algorithm sets k = 5, 10, and 20 respectively, where k represents the number of subcomponents obtained by the spectral clustering algorithm.

B. Performance Comparison on DGSC with Various k Values

Table I shows the experimental results of DGSC-DECC on the CEC'2013 benchmark functions when clustering into 5 groups, 10 groups and 20 groups respectively. The median, mean and standard deviation of the 25 runs are recorded to express the optimization performance. The entries show in bold are significantly better using a two-sided Wilcoxon rank-sum test with α = 0.05. It's clear that when k = 5 the optimization results are significantly better on 10 out of 15 functions, whereas with k = 20 this number is reduced to 4 and with k = 10 this number is only 1. It should be noted that for the fully separable functions f_1-f_3 and the approximate fully separable function f_{12}, the optimization result is best when k = 20 and that is the worst when k = 5. This indicates that for this type of functions, increasing the number of groups to a certain extent is beneficial to the optimization result. For the other 11 functions, 10 functions perform best at k = 5. When k = 10, the optimization results are usually neither good nor bad.

TABLE I. EXPERIMENTAL RESULTS OF DGSC-DECC ON THE CEC'2013 BENCHMARK FUNCTIONS USING 25 INDEPENDENT RUNS. THE HIGHLIGHTED ENTRIES ARE SIGNIFICANTLY BETTER (WILCOXON TEST, α =0.05).

Functions		k=5	k=10	k=20
f_1	Median	5.15e-05	1.81e-11	**7.17e-13**
	Mean	2.60e-04	2.96e-11	1.55e-12
	Std	5.66e-04	3.70e-11	2.48e-12
f_2	Median	7.04e+02	1.13e+02	**4.72e+01**
	Mean	7.15e+02	1.08e+02	4.82e+01
	Std	1.61e+02	1.73e+01	1.08e+01
f_3	Median	2.07e+01	2.07e+01	**2.05e+01**
	Mean	2.07e+01	2.07e+01	2.05e+01
	Std	1.88e-02	1.57e-02	1.06e-02
f_4	Median	**3.81e+08**	4.84e+08	2.10e+09
	Mean	3.77e+08	5.78e+08	2.43e+09
	Std	9.61e+07	2.78e+08	9.30e+08
f_5	Median	**3.32e+06**	4.35e+06	6.23e+06
	Mean	3.27e+06	4.11e+06	6.33e+06
	Std	6.57e+05	8.95e+05	8.75e+05
f_6	Median	**1.06e+06**	1.06e+06	1.06e+06
	Mean	1.06e+06	1.06e+06	1.06e+06
	Std	1.27e+03	1.80e+03	1.03e+03
f_7	Median	**2.75e+05**	3.64e+06	1.77e+07
	Mean	4.83e+05	4.28e+06	2.07e+07
	Std	5.89e+05	2.38e+06	1.28e+07
f_8	Median	**1.50e+13**	4.36e+13	1.30e+14
	Mean	1.85e+13	4.49e+13	1.32e+14
	Std	1.20e+13	2.20e+13	5.46e+13
f_9	Median	**1.95e+08**	2.91e+08	3.43e+08
	Mean	1.79e+08	2.71e+08	3.36e+08
	Std	5.31e+07	5.73e+07	8.23e+07
f_{10}	Median	**9.39e+07**	9.42e+07	9.43e+07
	Mean	9.38e+07	9.42e+07	9.44e+07
	Std	3.32e+05	2.37e+05	3.45e+05
f_{11}	Median	3.23e+08	**7.85e+07**	2.75e+08
	Mean	6.92e+09	4.57e+09	1.41e+09
	Std	2.08e+10	1.46e+10	5.22e+09
f_{12}	Median	2.96e+03	2.29e+03	**1.76e+03**
	Mean	2.93e+03	2.30e+03	1.79e+03
	Std	2.17e+02	1.30e+02	1.26e+02
f_{13}	Median	**6.25e+08**	7.53e+08	3.08e+09
	Mean	8.63e+08	1.05e+09	3.50e+09
	Std	1.25e+09	1.17e+09	1.33e+09
f_{14}	Median	**1.14e+08**	3.40e+09	6.18e+10
	Mean	1.32e+08	8.61e+09	6.73e+10
	Std	8.12e+07	1.20e+10	5.69e+10
f_{15}	Median	**2.66e+07**	3.52e+07	2.58e+08
	Mean	2.67e+07	3.49e+07	2.78e+08
	Std	4.29e+06	7.84e+06	6.81e+07
No. Best		10	1	4

In Table II, the number of the best performance, the moderate performance and the worst performance of the three grouping methods on the four categories of functions are counted respectively. The statistical results show that when k = 5, it performs well on partially separable functions, overlapping functions and non-separable functions. Although the performance on the separable functions is not as good as that of k = 10 and k = 20, the optimization results have been greatly improved compared with putting all variables in one group. The optimization results are moderate at k = 10, usually neither best nor worst. When k = 20, it performs best on the separable functions, while it performs poorly on other functions. From the overall perspective, dividing into five groups is more suitable for DGSC algorithm.

TABLE II. THE NUMBER OF THE BEST, THE MODERATE AND THE WORST PERFORMERS ON THE FOUR CATEGORIES OF FUNCTIONS RESPECTIVELY WHEN k = 5, k = 10, k =20.

Categories	k = 5 b/m/w	k = 10 b/m/w	k = 20 b/m/w
C1: f_1-f_3	0/0/3	0/3/0	3/0/0
C2: f_4-f_{11}	7/0/1	1/7/0	0/1/7
C3: f_{12}-f_{14}	2/0/1	0/3/0	1/0/2
C4: f_{15}	1/0/0	0/1/0	0/0/1
Total	10/0/5	1/14/0	4/1/10

C. Performance Comparison on Grouping Methods

Table III presents the experimental results of DGSC-DECC on the CEC'2013 benchmark functions and compares it against several similar algorithms, including DG, GDG and DG2. The median, mean and standard deviation of 25 runs are recorded to express the optimization performance. The entries show in bold are significantly better using a two-sided Wilcoxon rank-sum test with α = 0.05. In Table IV, the number of the best performance of DG, GDG, DG2 and DGSC on the four categories of functions are counted respectively, where 'b' means the number of best and 't' means the number of total.

As can be seen from Table III and Table IV, the DGSC algorithm achieves significant better optimization results on 10 out of 15 benchmark functions, while that number of DG,

GDG, and DG2 is 2, 3 and 4 respectively. For functions f_1- f_3, all the variables are fully separable. Since the variables are divided into several subgroups, DGSC has obviously better optimization results on f_1 and f_2. The optimization results of these algorithms on f_3 are not much different because Ackley function is not additively separable. For the partially separable functions f_4-f_{11}, DGSC has significantly better optimization results on 5 out of 8 functions, which is also superior to other algorithms. DGSC has achieved the best optimization results on the overlapping functions f_{12}-f_{14}, which effectively solves the problem of grouping and optimization for overlapping functions to some extent. For the nonseparable function f_{15}, the optimization result of DGSC is not good. This is because any two variables of the function are directly interacted and are not suitable for grouping. Therefore, DG obtains the best optimization result on this function, which puts all variables in one group and has the most number of fitness evaluations remained at optimization stage.

TABLE III. PERFORMANCE COMPARISON OF DGSC AND OTHER GROUPING TECHNIQUES ON THE CEC'2013 BENCHMARK FUNCTIONS USING 25 INDEPENDENT RUNS. THE HIGHLIGHTED ENTRIES ARE SIGNIFICANTLY BETTER (WILCOXON TEST, α = 0.05).

Functions		DG	GDG	DG2	DGSC
f_1	Median	1.95e+06	4.08e+05	5.52e+05	**4.27e-05**
	Mean	6.28e+06	6.84e+05	1.72e+06	2.60e-04
	Std	1.28e+07	8.25e+05	3.61e+06	5.66e-04
f_2	Median	1.46e+04	1.49e+04	1.39e+04	**7.04e+02**
	Mean	1.45e+04	1.45e+04	1.40e+04	7.15e+02
	Std	1.54e+03	1.52e+03	1.26e+03	1.61e+02
f_3	Median	2.07e+01	**2.06e+01**	**2.06e+01**	2.07e+01
	Mean	2.07e+01	2.06e+01	2.06e+01	2.07e+01
	Std	8.53e-03	7.92e-03	7.00e-03	1.88e-02
f_4	Median	1.03e+11	4.26e+08	4.38e+08	**3.81e+08**
	Mean	1.19e+11	4.88e+08	5.13e+08	3.77e+08
	Std	7.37e+10	2.08e+08	2.34e+08	9.61e+07
f_5	Median	**2.61e+06**	**2.76e+06**	**2.44e+06**	3.32e+06
	Mean	2.69e+06	2.55e+06	2.47e+06	3.27e+06
	Std	6.78e+05	6.37e+05	4.79e+05	6.57e+05
f_6	Median	1.06e+06	1.06e+06	1.06e+06	**1.06e+06**
	Mean	1.06e+06	1.06e+06	1.06e+06	1.06e+06
	Std	9.88e+02	1.53e+03	1.37e+03	1.27e+03
f_7	Median	1.46e+08	**1.06e+05**	1.23e+06	2.75e+05
	Mean	1.52e+08	3.25e+05	1.60e+06	4.83e+05
	Std	5.17e+07	1.10e+06	1.01e+06	5.89e+05
f_8	Median	2.10e+15	4.07e+13	8.06e+13	**1.50e+13**
	Mean	2.29e+15	4.69e+13	9.68e+13	1.85e+13
	Std	1.31e+15	2.81e+13	5.16e+13	1.20e+13
f_9	Median	3.08e+08	3.54e+08	3.52e+08	**1.95e+08**
	Mean	2.96e+08	3.49e+08	3.24e+08	1.79e+08
	Std	6.00e+07	6.97e+07	9.13e+07	5.31e+07
f_{10}	Median	9.43e+07	9.43e+07	9.43e+07	**9.39e+07**
	Mean	9.43e+07	9.43e+07	9.42e+07	9.38e+07
	Std	2.87e+05	3.56e+05	3.31e+05	3.32e+05
f_{11}	Median	5.73e+09	5.21e+08	**2.45e+08**	3.23e+08
	Mean	1.81e+10	6.74e+08	2.16e+10	6.92e+09
	Std	2.31e+10	7.73e+08	4.61e+10	2.08e+10
f_{12}	Median	1.47e+11	4.00e+07	1.97e+07	**2.96e+03**
	Mean	1.46e+11	1.72e+08	6.10e+07	2.93e+03
	Std	1.60e+10	2.74e+08	6.83e+07	2.17e+02
f_{13}	Median	6.65e+09	7.71e+08	**6.65e+08**	**6.25e+08**
	Mean	6.76e+09	7.89e+08	7.84e+08	8.63e+08
	Std	1.47e+09	2.74e+08	4.47e+08	1.25e+09
f_{14}	Median	9.32e+09	1.17e+09	1.16e+09	**1.14e+08**
	Mean	1.37e+10	1.61e+09	1.82e+09	1.32e+08
	Std	1.28e+10	1.39e+09	1.70e+09	8.12e+07
f_{15}	Median	**4.93e+06**	5.79e+06	5.85e+06	2.66e+07
	Mean	5.16e+06	6.49e+06	7.05e+06	2.67e+07
	Std	1.58e+06	2.11e+06	3.29e+06	4.29e+06
No. Best		2	3	4	10

TABLE IV. THE NUMBER OF THE BEST PERFORMERS ON THE FOUR CATEGORIES OF FUNCTIONS RESPECTIVELY WITH DG, GDG, DG2 AND DGSC.

Categories	DG b/t	GDG b/t	DG2 b/t	DGSC b/t
C1: f_1-f_3	0/3	1/3	1/3	2/3
C2: f_4-f_{11}	1/8	2/8	2/8	5/8
C3: f_{12}-f_{14}	0/3	0/3	1/3	3/3
C4: f_{15}	1/1	0/1	0/1	0/1
Total	2/15	3/15	4/15	10/15

D. Performance Comparison on Convergence of Overlapping Functions

Fig. 3-5 show the convergence curves of DG, GDG, DG2 and DGSC on the overlapping functions. Each point on the curve is calculated by taking the average of 25 independent runs. It can be seen that DGSC converges quickly on the overlapping functions and has a small convergence value. Overall, the optimization results of DGSC are significantly better than other grouping algorithms on the CEC'2013 benchmark functions.

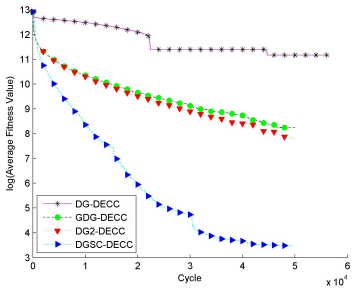

Fig. 3. Convergence curves of various algorithms on f_{12}.

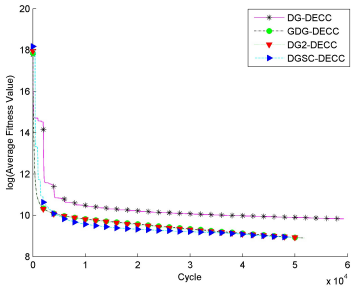

Fig. 4. Convergence curves of various algorithms on f_{13}.

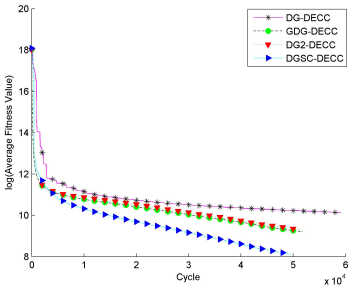

Fig. 5. Convergence curves of various algorithms on f_{14}.

V. Conclusion

In this paper, a differential grouping with spectral clustering algorithm for LSGO problems is proposed. Due to the introduction of features such as nonuniform, imbalance and overlapping, the existing grouping algorithms have difficulty to obtain good grouping results for the CEC'2013 benchmark functions. For the first time, DGSC combines differential grouping with spectral clustering to solve the LSGO problems and achieves encouraging results. DGSC derives the design structure matrix representing the interaction relationship between variables based on the differential values, and then uses this matrix as the input of spectral clustering to output the grouping result of decision variables. Finally, SaNSDE is used as the subcomponent optimizer under the CC framework. The resultant algorithm, called DGSC-DECC, has been proved to have promising results on the CEC'2013 benchmark functions, especially for separable functions and overlapping functions. Since spectral clustering does not need to consider whether the grouping is completely correct but is grouped according to the interaction relationship between variables, the problem that overlapping functions cannot find the optimal decomposition is solved. The disadvantage of this algorithm is that the number of groups needs to be artificially set. In the future, this value will be adaptively calculated to reduce the parameter.

Acknowledgment

This work was supported in part by the National Key R&D Program of China under Grant 2017YFC1601800, in part by the National Natural Science foundation of China, under Grant 61673194, and Grant 61672263, in part by the Key Research and Development Program of Jiangsu Province, China, under Grant BE2017630, in part by "Blue Project" in Jiangsu Universities, in part by the Postdoctoral Science Foundation of China under Grant 2014M560390, and in part by the Six Talent Peaks Project of Jiangsu Province under Grant DZXX-025.

References

[1] X. M. Hu, F. L. He, W. N. Chen, and J. Zhang, "Cooperation coevolution with fast interdependency identification for large scale optimization," Information Sciences, vol. 381, pp. 142–160, March 2017.

[2] H. Ge, S. Liang, G. Tan, C. Zheng, and C. L. P. Chen, "Cooperative hierarchical PSO with two stage variable interaction reconstruction for large scale optimization," IEEE Transactions on Cybernetics, vol. 47, pp. 2809–2823, September 2017.

[3] M. N. Omidvar, X. Li, and K. Tang, "Designing benchmark problems for large-scale continuous optimization," Information Sciences, vol. 316, pp. 419–436, September 2015.

[4] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in Proc. International Conference on Parallel Problem Solving from Nature, vol. 2, 1994, pp. 249–257.

[5] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," Information Sciences, vol. 178, pp. 2985–2999, August 2008.

[6] Y. Guo, J. Cheng, S. Luo, D. Gong, and Y. Xue, "Robust dynamic multi-objective vehicle routing optimization method," IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 15, pp. 1891–1903, November/December 2018.

[7] Y. Guo, J. Ji, J. Ji, D. Gong, J. Cheng, and X. Shen, "Firework-based software project scheduling method considering the learning and forgetting effect," Soft Computing, pp. 1–16, April 2018.

[8] M. N. Omidvar, M. Yang, Y. Mei, X. Li, and X. Yao, "DG2: A faster and more accurate differential grouping for large-scale black-box optimization," IEEE Transactions on Evolutionary Computation, vol. 21, pp. 929–942, April 2017.

[9] F. van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," IEEE Transactions on Evolutionary Computation, vol. 8, no. 3, pp. 225–239, July 2004.

[10] Z. Yang, K. Tang, and X. Yao, "Multilevel cooperative coevolution for large scale optimization," in Proc. IEEE Congress on Evolutionary Computation, pp. 1663–1670, June 2008.

[11] X. Li and X. Yao, "Cooperatively coevolving particle swarms for large scale optimization," IEEE Transactions on Evolutionary Computation, vol. 16, no. 2, pp. 210–224, April 2012.

[12] M. N. Omidvar, X. Li, Z. Yang, and X. Yao, "Cooperative co-evolution for large scale optimization through more frequent random grouping," in Proc. IEEE Congress on Evolutionary Computation, pp. 1754–1761, July 2010.

[13] W. Chen, T. Weise, Z. Yang, and K. Tang, "Large-scale global optimization using cooperative coevolution with variable interaction learning," in Proc. International Conference on Parallel Problem Solving from Nature, vol. 6239, pp. 300–309, Springer, 2011.

[14] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization," IEEE Transactions on Evolutionary Computation, vol. 18, pp. 378–392, January 2013.

[15] Y. Mei, M. N. Omidvar, X. Li, and X. Yao, "Competitive divide-and-conquer algorithm for unconstrained large scale black-box optimization," ACM Transaction on Mathematical Software, vol. 42, no. 2, p. 13, February 2015.

[16] Y. Ling, H. Li, and B. Cao, "Cooperative co-evolution with graph-based differential grouping for large scale global optimization," In 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), pp. 95–102, August 2016.

[17] Y. Sun, M. Kirley, and S. K. Halgamuge, "A recursive decomposition method for large scale continuous optimization," IEEE Transactions on Evolutionary Computation, vol. 22, no. 5, pp. 647–661, October 2018.

[18] Y. Sun, M. N. Omidvar, M. Kirley, and X. Li, "Adaptive threshold parameter estimation with recursive differential grouping for problem decomposition," In Proceedings of the Genetic and Evolutionary Computation Conference, pp. 889–896, July 2018.

[19] 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 and Applications Laboratory, USTC, China, Tech. Rep., January 2010.

[20] X. Li, K. Tang, M. N. Omidvar, Z. Yang, and K. Qin, "Benchmark functions for the CEC'2013 special session and competition on large-scale global optimization," RMIT University, Melbourne, Australia, Tech. Rep., December 2013.

[21] C. Chen, M. K. Ng, and S. Zhang, "Block spectral clustering methods for multiple graphs," Numerical Linear Algebra with Applications, vol. 24, November 2016.

[22] Z. Yang, K. Tang, and X. Yao, "Self-adaptive differential evolution with neighborhood search," in Proc. IEEE Congress on Evolutionary Computation , pp. 1110–1116, June 2008.

[23] Y. Sun, M. Kirley, and S. K. Halgamuge, "Extended differential grouping for large scale global optimization with direct and indirect variable interactions," in Proc. Conference on Genetic and Evolutionary Computation, pp. 313–320, July 2015.

[24] R. Shang, Z. Zhu, L. Jiao, W. Wang, and S. Yang, "Global discriminative-based nonnegative spectral clustering," Pattern Recognition, vol. 55, pp. 172–182, July 2016.

[25] Q. Li, Y. Ren, L. Li, and W. Liu, "Fuzzy based affinity learning for spectral clustering," Pattern Recognition, vol. 60, pp. 531–542, December 2016.

[26] R. Storn and K. Price, "Differential evolution: A simple and efficient heuristic for global optimization over continuous spaces," J. Global Optimization, vol. 11, no. 4, pp. 341–359, December 1997.