Large-scale global optimisation using cooperative co-evolution with self-adaptive differential grouping

Wei Fang*
Jiangsu Provincial Engineering Laboratory of Pattern Recognition and Computational Intelligence,
Department of Computer Science and Technology,
Jiangnan University,
Wuxi, China
Email: fangwei@jiangnan.edu.cn
*Corresponding author

Ruigao Min
Department of Computer Science and Technology,
Jiangnan University,
Wuxi, China
Email: 727891427@qq.com

Quan Wang
Wuxi SensingNet Industrialization Research Institute,
Wuxi, China
Email: seepwq@163.com

Abstract: Cooperative co-evolution (CC) provides the divide-and-conquer framework for solving large-scale global optimisation (LSGO) problems. Identification of variable interactions is the main challenge in CC. Differential grouping (DG) is a competitive approach to find the identification and Global DG (GDG) is its improvement by introducing the global information. In this paper, a self-adaptive DG (SDG) is proposed for further improving the grouping accuracy of GDG. The threshold for grouping in SDG can adjust adaptively along with the magnitude of different functions and is determined by only two points which is a randomly sampled point and its corresponding opposite point. A self-adaptive pyramid allocation (SPA) for computational resources is also studied. The proposed algorithm, with SDG, SPA, and the optimiser SaNSDE, is used to solve the CEC’2010 LSGO benchmark suite. Experimental results show that SDG achieved ideal decomposition for all the functions and the proposed algorithm obtained competitive optimisation performance.

Keywords: large-scale global optimisation; differential grouping; cooperative co-evolution; problem decomposition.

Biographical notes: Wei Fang received his PhD in Information Technology and Engineering of Light Industry from the Jiangnan University, Wuxi, China, in 2008. He is a Professor of Computer Science in the Jiangnan University. He was a Visiting Scholar in the CEACIA, University of Birmingham, Birmingham, UK, from 2013 to 2014. His current research interests involve the evolutionary computation, swarm intelligence, multiobjective optimisation, and large-scale global optimisation. He has published over 50 scientific papers in journals and international conferences. He serves as an editorial board member of the International Journal of Swarm Intelligence Research and the International Journal of Computing Science and Mathematics.

Ruigao Min received his BSc in Computer Science and Technology from the Jiangnan University, Wuxi, China, in 2016. His research interests include parallel realisation for swarm intelligence.

Quan Wang received his PhD from the Department of Electrical and Computer Engineering, University of Western Ontario, Canada, in 2015. He obtained his Master from the Wuhan University of Technology, Wuhan, China, in 2005. Before he began his PhD program, he worked as an Assistant Professor at the Chongqing University of Posts and Telecommunications for four years. From 2015 to 2016, he was a Post-doctoral Research Fellow in the Department of Electrical and Computer Engineering, at the University of Western Ontario, Canada. He is currently the Vice President at the Wuxi SensingNet Industrialization Research Institute, China. He is convenor of IEC/ISO JTC1 SC 41WG4, which is the international standardisation working group for IoT platform and interoperability. His main fields of research interests include industrial wireless sensor networks, and applications of internet of things.

1 Introduction

Swarm intelligence (SI) is the collective behavior of individuals that coordinate with decentralised control and self-organisation. The individuals in SI systems interact with each other and their environment that follow very simple rules. Ant colony optimisation (ACO) (Dorigo et al., 1996) and particle swarm optimisation (PSO) (Kennedy and Eberhart, 1995; Shi and Eberhart, 1998; Maiti et al., 2018; Lu and Tang, 2017) are two typical algorithms inspired from SI and they have shown excellent search abilities in many optimisation problems. Some other SI algorithms have also attracted a lot researches, such as bacterial foraging optimisation algorithm (Gorripotu et al., 2018; Sudha and Bai, 2017), grey wolf optimisation technique (Roy et al., 2017), and gravitational search algorithm (Kannan et al., 2018; Banerjee, 2018). However, many real-world optimisation problems face with a large number of variables, which is known as large-scale global optimisation (LSGO) problems. The performance of ACO, PSO and
other algorithms inspired by SI suffers from great deficiency as the dimensionality of the problem increases (Omidvar et al., 2015) and then these algorithms have difficulty to get good solutions when tackling with LSGO problems. There are four main factors leading to the performance deteriorates for solving LSGO problems (Mahdavi et al., 2015; Omidvar et al., 2015). Firstly, the search space of a problem exponentially increases as the number of the variables increases, which makes it impossible for the algorithm to search the entire search space sufficiently. Secondly, the complexity of problem increases and properties of problem may change as the problem dimension increases. The extensive computational burden for evaluating LSGO problems and the complex interaction among a large number of variables are the other two factors that contributes to the difficulty of LSGO problems.

To address the factors in LSGO problems, some approaches with SI algorithms have been studied by the researchers in recent years. Among these approaches, cooperative co-evolution (CC) with problem decomposition is a popular approach for LSGO problems. In CC approach, LSGO problems are decomposed into low dimensional subcomponents following the divide-and-conquer mechanism and then each of subcomponents is optimised by a separate optimisation algorithm. In order to achieve good performance based on CC approach for LSGO problems, two critical aspects are concerned. One is the variable grouping strategy for LSGO problems and another one is selection of the optimisation algorithms. van den Bergh et al. firstly proposed CC approach with PSO algorithms (CPSO) based on two static grouping strategy where the decision variables are assigned in fixed groups. Static grouping-based CC approach is effective on unimodal and low-dimensional problems but ineffective when dealing with non-separable and LSGO problems (Yang et al., 2008a). Different from the static grouping strategy in CPSO, Yang et al. (2008a) proposed a random grouping strategy by randomly allocating the decision variables into subcomponents for tackling non-separable LSGO problems. Although random grouping strategy in Yang et al. (2008a) increases the probability of assigning two interacting variables into the same group and shows good performance on scalable non-separable problems, the number of subcomponents is still fixed, each of which has the pre-determined size during the whole evolutionary process. When the number of interacting variable is more than two, the performance of random grouping becomes ineffectively (Omidvar et al., 2010b). In Omidvar et al. (2010b), designed a more frequent random grouping strategy for increasing the probability of grouping interacting variables in one subcomponent. Li and Yao (2009, 2012) introduced random grouping in CPSO with the adaptive weighting schemes (CCPSO) and they later proposed an improved random grouping strategy with dynamically changing group size based on CCPSO. Instead of using the predefined group size for the CC approach, Yang et al. (2008b) took group size as parameter and divided the decision variables into low dimensional with different group sizes based on random grouping strategy.

Different from allocating interacting variables in the same subcomponent with probability in random grouping strategy, some learning techniques were proposed to identify the interactive relation among variables. In Omidvar et al. (2010a), the delta method was proposed to capture interacting variables by measuring the average difference in the specified variables and it is more suitable for only one group of interacting variables. Chen and Tang (2013) studied the variable interaction learning (VIL) technique in CC approach for detecting interacting variables. Recently, Omidvar et al. (2014) proposed the differential grouping (DG) to uncover the underling interaction
structure within the decision variables. The DG can decompose interacting variables into the same subcomponent with greater accuracy for most of the benchmark functions than the above-mentioned grouping strategies. However, the grouping accuracy of DG is sensitive to the parameter for different kinds of functions. The time complexity for DG is high when tackling the problems with much more separable variables. To address the parameter sensitive problem and variables interaction missing problem in DG, Mei et al. (2016) used the graph structure to represent the interactions between variables and proposed the global differential grouping (GDG) decomposition method. Although GDG has successfully improved the grouping accuracy on the CEC’2010 large-scale benchmark test problems, it still cannot divide all the large-scale problems in CEC’2010 with 100% accuracy even by adjusting the parameters of GDG. As studied in Omidvar et al. (2016, 2011, 2017), contribution-based strategy that guides the computational burden to different subproblems is helpful for the optimisation results.

In this paper, an improved version of DG based on GDG is proposed in order to further improve the grouping accuracy of GDG on the CEC’2010 large-scale benchmark suite. The proposed grouping approach is termed as self-adaptive differential grouping (SDG) that can determine the threshold by two sampled points in the decision space. For the two sampled points, one point is selected randomly at first and then its opposite point is chosen in the decision space. By the selected two points, the proposed SDG can measure the magnitude of computational error adaptively. Furthermore, the proposed SDG addresses the relation missing problem among variables in DG. At the same time, we propose a self-adaptive pyramid allocation (SPA) strategy that can allocate different computational resource to subcomponents with the aim to improve the optimisation performance.

The rest of this paper is organised as follows. The definition of LSGO, ideas in CC, and the details of DG are introduced in Section 2. The proposed DG approach and SPA strategy for computational resource are given in Section 3. Experimental results are shown in Section 4. Finally, Section 5 concludes the paper.

2 Background

2.1 Large-scale global optimisation

For a single-objective unconstrained minimisation problem as the following,

$$
\min_{\vec{x} \in \mathbb{R}^D} f(\vec{x})
$$

(1)

where $f$ is the objective function mapping from $\mathbb{R}^D$ to $\mathbb{R}$, and $D$ is the number of decision variables or the dimension of the problem. As the LSGO problem is concerned, it is generally regarded that the number of variables is large from several hundreds to thousands or more (Mahdavi et al., 2015). For the decision variables in LSGO problem, two kinds of relationships, which are interaction and independence, can be defined based on the differential theory (Omidvar et al., 2014; Mei et al., 2016).

**Definition 1 (interaction):** For any two different decision variables $x_i$ and $x_j$ in a function $f_i(\vec{x})$, they are said to interact with each other, iff $\exists (a, b), \frac{\partial^2 f}{\partial x_i \partial x_j}|_{x_i = a, x_j = b} \neq 0$. 
Definition 2 (independence): For any two different decision variables $x_i$ and $x_j$ in a function $f(\vec{x})$, they are said to be independent with each other, iff $\forall (a, b)$, 
$$\frac{\partial^2 f}{\partial x_i \partial x_j} |_{x_i=a,x_j=b} = 0.$$ 

Regarding the interaction among the decision variables in a problem, three types of definitions for LSGO problems can be obtained, which are partially separable problem, fully non-separable problem, and partially additively separable problem (Li et al., 2013).

Definition 3 (Omidvar et al., 2014): A function $f(x)$ is partially separable if it can divide into several subcomponents as the following form
\begin{equation}
\arg\min_{\vec{x}} f(\vec{x}) = (\arg\min_{\vec{x}_1} f(\vec{x}_1, \cdots), \cdots, \arg\min_{\vec{x}_m} f(\cdots, \vec{x}_m)),
\end{equation}
where $\vec{x}_1, \cdots, \vec{x}_m$ are mutually exclusive sub-vectors of $\vec{x}$ and $2 \leq m \leq D$. A function is fully separable if $m = D$.

Definition 4 (Omidvar et al., 2014): A function $f(\vec{x})$ is fully non-separable if every two decision variables interact with each other.

Definition 5 (Omidvar et al., 2014): Partially additive separability is a special form of partially separability with the following formulation
\begin{equation}
f(\vec{x}) = \sum_{i=1}^{m} f_i(\vec{x}_i)
\end{equation}
where $\vec{x}_i$ and $\vec{x}$ share the same meaning as those in Definition 3, $f_i$ is a non-separable sub-function and $m$ denotes the number of non-separable sub-functions. When $m = D$, the fully separable problem is also called completely additively separable.

2.2 Cooperative co-evolution

CC (Potter and De Jong, 1994) is one of the prominent way to tackle LSGO problems, which follows the divide-and-conquer mechanism. In CC, the problem is divided into several subcomponents by decomposition strategies and then evolved individually by optimisation approaches. In general, there are three steps in the framework of CC, including problem decomposition, subcomponents optimisation (co-evolution), and subcomponents combination (cooperation).

Problem decomposition is the first step of CC and is also the crucial step (Potter and De Jong, 2000). The task of problem decomposition is to find the interaction among the decision variables and place the interacting decision variables in the same subcomponent. As introduced in Section 1, the decomposition strategies have been studied from static grouping, random grouping to learning-based grouping before or along with the evolutionary process (Mahdavi et al., 2015). Each subcomponent is evolved in subcomponent optimisation step. The fitness of each subcomponent for a solution is evaluated by integrating itself with the other subcomponents. The subcomponents combination step is therefore called by subcomponents optimisation step.
2.3 Differential grouping

2.3.1 Principles in DG

DG was proposed by Omidvar et al. (2014), etc. for problem decomposition and has shown superior performance on grouping accuracy. The core principle in DG is its interaction detection approach described in the following theorem (Omidvar et al., 2014):

Theorem 1: If $f(\vec{x})$ is a partially additively separable function and $\forall a, b_1 \neq b_2, \delta \in \mathbb{R}, \delta \neq 0$, and holds the following conditions

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

(4)

where

$$\Delta_{\delta,x_i}[f](x) = f(\cdots, x_i + \delta, \cdots) - f(\cdots, x_i, \cdots)$$

(5)

Indicates the difference of function value $f$ only on the decision variable $x_i$ with interval $\delta$, then two decision variables, $x_i$ and $x_j$, are regarded to interact with each other.

Following the conditions in equation (4), the decision variables of a problem can be grouped and then the problem is decomposed.

2.3.2 Drawbacks of DG

Let $\Delta_1$ and $\Delta_2$ denote the left hand side and right hand side of equation (4) respectively, $\epsilon$ be a small positive number. Then from Theorem 1, we can get

$$|\Delta_1 - \Delta_2| > \epsilon \Rightarrow \text{non-separability.}$$

(6)

According to the relationship between the proposition and contraposition ($p \Rightarrow q \equiv q \Rightarrow \neg p$), equation (6) can be inferred to

Independence (Separability) $\Rightarrow |\Delta_1 - \Delta_2| \leq \epsilon$.  

(7)

From equations (6) and (7), $|\Delta_1 - \Delta_2| > \epsilon$ is the sufficient condition that two variables are dependent rather than the necessary condition. However, $|\Delta_1 - \Delta_2| \leq \epsilon$ is the necessary condition that two variables are independent rather than the sufficient condition. That is to say, by the comparison between $|\Delta_1 - \Delta_2|$ and $\epsilon$, the former difference is larger than the later value means two variables are definitely dependent with each other; but if the later value is not less than the former difference, this cannot explain whether two variables are dependent or not. Here is an example to illustrate this point.

Given a problem with the formulation

$$f(x_1, x_2, x_3, x_4) = x_1^2 + x_2x_3^{-3} + x_3x_4 + x_4^2$$

(8)
It is clear that \( x_2, x_3, \) and \( x_4 \) are dependent with each other and fall in the same group, and \( x_1 \) forms a single group with itself. By using DG to detect the interaction between \( x_2 \) and \( x_4 \), the following two equations can be easily got,

\[
\Delta_1 = (x_1^2 + (x_2 + \delta)x_3^{-3} + x_3x_4 + x_4^2) \\
- (x_1^2 + x_2x_3^{-3} + x_3x_4 + x_4^2) = x_3^{-3}\delta,
\]

(9)

\[
\Delta_1 = (x_1^2 + (x_2 + \delta)x_3^{-3} + x_3(x_4 + \varphi) + (x_4 + \varphi)^2) \\
- (x_1^2 + x_2x_3^{-3} + x_3(x_4 + \varphi) + (x_4 + \varphi)^2) = x_3^{-3}\delta.
\]

(10)

We find that \( \Delta_1 = \Delta_2 \) and then \( |\Delta_1 - \Delta_2| \leq \epsilon \). However, \( x_2 \) and \( x_4 \) cannot be inferred to interact with each other from equation (7). Therefore, the first drawback of DG is the theoretic defect that leads to mistakes in variable grouping.

The second drawback of DG lies in its comparison process that may miss many interactions between decision variables. To address this issue, Mei et al. (2016) proposed the GDG method based on DG and introduced a matrix to maintain the relationship between every two decision variables. From Theorem 1 and equation (6), the grouping accuracy is sensitive to the threshold parameter \( \epsilon \), which comes the third drawback of DG. In Mei et al. (2016), a randomly sample approach was used to determine the threshold. Recently, the inventor of DG, Omidvar et al. (2017) proposed am improved DG (DG2) to address the third drawback. In Omidvar et al. (2017), DG2 was also designed to improve the grouping accuracy when the problem is with overlapping components. In this paper, the proposed grouping approach aims to address the three drawbacks in DG for CEC’2010 large-scale benchmark functions and does not intend to find interactions of the problem with overlapping components.

3 Cooperative co-coevolution with SDG and SPA for computational resource

3.1 Self-adaptive differential grouping

Both GDG and DG2 have proposed the strategies to address the sensitive problem for determining the threshold in DG. For GDG, the threshold is depended on each function by estimating the computational error along with the objective space, which is not fixed for all functions. By considering the magnitude correlation between computational error and objective function, a reliable threshold estimation approach is proposed in DG2. These two approaches show their superior group accuracy over DG. However, one can find that neither approach can get the 100% success rate of grouping on the CEC’2010 large-scale benchmark functions (Omidvar et al., 2017). In this paper, inspired by GDG and DG2, we introduce an SDG approach in order to further improve the grouping accuracy of GDG and DG2 on the CEC’2010 large-scale benchmark suite.

In SDG, the main idea is also to find the threshold \( \epsilon \) that can adjust adaptively along with the magnitude of different functions. Different from GDG that randomly sample several points in the decision space, only two points are needed in SDG, which are a randomly sampled point and its corresponding opposite point in the decision space. Let \( P = (x_1, \cdots, x_D), x_i \in [LB_i, UB_i], i \in [1, D] \) be a randomly sampled point in the
decision space. \( LB_i \) and \( UB_i \) are the lower bound and upper bound of the \( i \)th dimension. Then the opposite point \( P^* \) of \( P \) is \( P^* = (x_i^1, \cdots, x_i^D) \), where
\[
x_i^* = LB_i + UB_i - x_i.
\] (11)

The fitness values of these two points are denoted as \( \text{fitness}(P) \) and \( \text{fitness}(P^*) \) respectively and let \( \Delta_P = |\text{fitness}(P) - \text{fitness}(P^*)| \). The value of \( \Delta_P \) is used to determine the threshold \( \epsilon \). A small \( \epsilon \) is needed when \( \Delta_P \) is small. On the contrary, when \( \Delta_P \) is large, a large \( \epsilon \) is required. Furthermore, small \( \Delta_P \) may mean smaller value of \( |\Delta_1 - \Delta_2| \), which implies high sensitivity to the threshold \( \epsilon \). In this case, logarithmic values of \( \text{fitness}(P) \) and \( \text{fitness}(P^*) \) are used to calculate \( \Delta_P \).

After the threshold \( \epsilon \) is determined, the values of all \( |\Delta_1 - \Delta_2| \) are calculated and compared with \( \epsilon \) to form the entire matrix \( \Theta_{n \times n} \) as that in Mei et al. (2016). The entry \( \Theta_{ij} \) equals 1 if \( \Lambda_{ij} > \epsilon \), and 0 otherwise, where \( \Lambda_{ij} \) represents the \( |\Delta_1 - \Delta_2| \) value between the variables \( x_i \) and \( x_j \). Therefore, the matrix \( \Theta_{n \times n} \) can be used to find the interactions between any two variables. However, since the theoretic defect in DG as discussed in Subsection 2.3, decomposition results by DG may have mistakes. To address this issue, we propose a revision approach on the matrix \( \Theta_{n \times n} \). By checking each row of \( \Theta_{n \times n} \), the indexes \( i \) and \( j \) of the entry \( \Theta_{i \times j} \) that takes 1 are recorded in a 1 – \( D \) array. New index combinations can be generated by crossover operator on the elements in the array. And then the entry \( \Theta_{i \times j} \) with new index combinations takes 1 except those on the principal diagonal. By the revision approach, the missing interactions can be found. Given the problem as equation (8), the \( \Theta \) matrix by DG is shown in equation (12). Through scanning the matrix, the 1 – \( D \) array is \([3, 2, 4] \) and the new index combinations are \((2, 3), (2, 4), (3, 2), (3, 4), (4, 2), \) and \((4, 3) \). The revised matrix can be obtained easily as equation (13), where the correct entire matrix is found. The pseudocode of SDG is described in Algorithm 1. In Algorithm 1, the parameters \( \alpha \), \( \beta \), and \( \mu \) take the values of 50, \( 10^{-8} \), and \( 10^2 \) respectively, which are determined by several attempts.

\[
\Theta = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}
\] (12)

\[
\Theta' = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}
\] (13)

Algorithm 1

1: \( P = LB + \text{rand}() \ast (UB - LB), P^* = LB + UB - P \)
2: \( P_1 = LB, F_1 = f(P_1), FEs = 1 \)
3: \( \text{dims} = \{1, 2, 3, \cdots, \text{dim}\}, \text{seps} = \{\}, \text{nonseps} = \{\} \)
4: \( \Delta_P = |f(P) - f(P^*)| \)
5: \( FEs = FEs + 2 \)
6: if $\Delta_p < \mu$ then
7:  $\epsilon = |\log(f(P)) - \log(f(P^*))| \ast \alpha$
8: else
9:  $\epsilon = \Delta_p \ast \beta$
10: end if
11: for $i = 1 : \dim$ do
12:  $P_2 = P$, $P_2(i) = UB(i)$, $F_2(i) = f(P_2)$
13:  $P_3 = P$, $P_3(i) = 0$, $F_3(i) = f(P_3)$
14:  $FEs = EEs + 2$
15: end for
16: for $i = 1 : \dim - 1$ do
17:  for $j = i + 1 : \dim$ do
18:    $P_4 = P$, $P_4(i) = UB(i)$, $P_4(j) = 0$
19:    $F_4(i,j) = F_4(j,i) = f(P_4)$
20:    $FEs = EEs + 1$
21: end for
22: end for
23: for $i = 1 : \dim$ do
24:  $\Delta_1 = F_1 - F_2(i)$, $\Delta_2 = F_3(j) - F_4(i,j)$
25:  $\phi_{ij} = |\Delta_1 - \Delta_2|$
26: if $\phi > \epsilon$ then
27:  $\theta_{ij} = 1$
28: end if
29: end for
30: for $i = 1 : \dim$ do
31:  temp = \{i\}
32:  for $j = 1 : \dim$ do
33:    if $\theta == 1$ then
34:      temp = temp $\cup$ j
35:    end if
36: end for
37: $\theta_{temp,temp} = 1$
38: end for
39: $\theta_{diagonal} = 0$
40: for $i \in \text{dims}$ do
41:  group = \{i\}
42:  for $j \in \text{dims} \neq j$ do
43:    if $\theta_{ij} == 1$ then
44:      group = group $\cup$ j
45:    end if
46: end for
47: if $\text{length(group)} = 1$ then
48:  seps = seps $\cup$ group
49: else
50:  nonseps = nonseps $\cup$ \{group\}
51: end if
52: dims = dims $-$ group
53: end for
54: return (seps, nonseps, EEs)
We also find that the scale of a variable group after decomposition may be too large, which is still difficult to be optimised even though the group result is with high accuracy. To solve this problem, the maximum number of variables in a group of different types is setup during the decomposition process. For the separable groups, the scale of each group is set to 100, and 50 for nonseparable groups.

3.2 Self-adaptation pyramid allocation strategy for computational resource

In general, the number of maximum fitness evaluations is regarded as the termination condition for solving LSGO problems based on EAs. For CC algorithms, a considerable number of fitness evaluations is carried out for variable grouping and the remaining fitness evaluations are used for subsequent optimisation, which should be taken efficient utilisation for good performance. For the imbalance problems, the contribution to the global fitness of each subcomponent is likely different and the computational resource can be allocated to the subcomponents according to the contribution of subcomponents to the global objective (Omidvar et al., 2011, 2016). Compared to the round-robin way for all subcomponents, the contribution-based CC shows more confidential results. In this paper, we propose a SPA strategy that can allocate different computational resource to subcomponents and consider their efforts to the global fitness. Figure 1 illustrates the proposed SPA strategy. In the proposed computational resource allocation strategy, the number of fitness evaluations for each subcomponent is different as shown in Figure 1 like a ‘pyramid’. When the pre-defined maximum number of fitness evaluations is reached for a subcomponent, the allocation process then goes back to the top of the ‘pyramid’. The allocation process is a continuous circular loop and stops until the maximum number of fitness evaluations is reached. During the allocation process, the contribution to the global objective of each subcomponent is monitored and more fitness evaluations will spend on this subcomponent.

![Figure 1](image.png)

The pseudocode of cooperative co-coevolution with SDG and SPA for computational resource is shown in Algorithm 2. The subcomponent optimiser is SaNSDE (Yang et al., 2008c) which is a self-adaptive version of NSDE and has significant performance on the difficult non-linear continuous optimisation problems.
Algorithm 2

1: \( \text{maxFEs} = \text{masFEs} - \text{FEs}, \text{FEs} = 0 \)
2: \( \text{group_num} = \text{size}(\text{grouping}, 2) \)
3: \( \text{SaNSDE}\_\text{FEs} = 50, \text{cycle} = 1 \)
4: \( \text{while} \ \text{FEs} < \text{maxFEs} \ \text{do} \)
5: \( \text{for} \ i = 1 : \text{group_num} \ \text{do} \)
6: \( \text{onemaxFEs} = \text{SaNSDE}\_\text{FEs} \)
7: \( \text{if} \ \text{cycle} < 1,000 \ \text{then} \)
8: \( \text{SaNSDE}\_\text{FEs} = \text{SaNSDE}\_\text{FEs} + \text{addSize} \)
9: \( \text{else} \)
10: \( \text{SaNSDE}\_\text{FEs} = \text{SaNSDE}\_\text{FEs} - \text{minuSize} \)
11: \( \text{end if} \)
12: \( \text{if} \ \text{SaNSDE}\_\text{FEs} < 50 \ \text{then} \)
13: \( \text{cycle} = 1 \)
14: \( \text{end if} \)
15: \( \text{if} \ \text{FEs} + \text{SaNSDE}\_\text{FEs} > \text{maxFEs} \ \text{then} \)
16: \( \text{SaNSDE}\_\text{FEs} = \text{maxFEs} - \text{FEs} \)
17: \( \text{end if} \)
18: \( (\text{best}) = \text{SaNSDE}(\text{groupingi}, \text{SaNSDE}\_\text{FEs}, \text{best}) \)
19: \( \text{cycle} = \text{cycle} + 1 \)
20: \( \text{FEs} = \text{FEs} + \text{SaNSDE}\_\text{FEs} \)
21: \( \text{end for} \)
22: \( \text{end while} \)
23: \( \text{return} \ (\text{best}) \)

4 Experimental studies

The IEEE CEC’2010 LSGO test suite is used to evaluate the performance of the proposed CC-SPA-SDG. There are 20 1,000-dimensional benchmark functions \((f_1 \text{ to } f_{20})\) in the LSGO test suites with five types. The functions \(f_1 \text{ to } f_3\) and \(f_{19} \text{ to } f_{20}\) are fully separable and fully non-separable functions respectively. The other 15 benchmark functions are partially separable with the details in Table 1. It should be noted that the IEEE CEC’2013 LSGO test suite has been extended greatly based on the IEEE CEC’2010 LSGO test suite. The characteristics of imbalance, overlapping, and non-uniform have been employed in the IEEE CEC’2013 LSGO benchmark functions. In this paper, we have not test the proposed algorithm on the IEEE CEC’2013 LSGO test suite since the great difference between the test suites. The accuracy of SDG is firstly compared with DG and GDG and then the optimisation results of CC-SPA-SDG with \(\text{SaNSDE}\) is compared with state-of-the-art algorithms, including MLCC (Yang et al., 2008b), DECC-DML (Omidvar et al., 2010b), DECC-D (Omidvar et al., 2010b), DECC-DG (Omidvar et al., 2014), and DECC-I (Omidvar et al., 2014). MA-SW-chains (Molina et al., 2010), which is a memetic algorithm and has shown good performance on LSGO problems without problem decomposition, is also taken into comparison. The maximum number of fitness evaluations is \(3 \times 10^6\), which is commonly used as terminal condition. For each compared algorithm on each test function, 25 independent runs are performed. Parameter settings for DG, GDG, and SDG are shown in Table 2.
Large-scale global optimisation using cooperative co-evolution

### Table 1

The 15 partially separable functions in CEC’2010 LSGO test suites

<table>
<thead>
<tr>
<th>Problems</th>
<th>Separable groups</th>
<th>Non-separable groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_4$ to $f_8$</td>
<td>950 separable variables</td>
<td>1 50-dimensional nonseparable group</td>
</tr>
<tr>
<td>$f_9$ to $f_{13}$</td>
<td>500 separable variables</td>
<td>10 50-dimensional nonseparable groups</td>
</tr>
<tr>
<td>$f_{14}$ to $f_{18}$</td>
<td>No separable variables</td>
<td>20 50-dimensional nonseparable groups</td>
</tr>
</tbody>
</table>

### Table 2

The parameter settings of DG, GDG, and SDG

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Parameter settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>DG</td>
<td>$\epsilon = 10^{-3}$</td>
</tr>
<tr>
<td>GDG</td>
<td>$K = 10, \alpha = 10^{-10}$</td>
</tr>
<tr>
<td>SDG</td>
<td>$\alpha = 50, \beta = 10^{-8}, \mu = 10^{2}$</td>
</tr>
</tbody>
</table>

#### 4.1 Performance metrics

Three performance metrics are used to measure the decomposition accuracy on the relationship of interaction, independence, and both of interaction and independence (Mei et al., 2016). The formulations of these three performance metrics are as follows:

\[
\rho_1 = \frac{\sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} (\Theta \circ \Theta^*)_{i,j}}{\sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} (\Theta^*)_{i,j}} \times 100\% \quad (14)
\]

\[
\rho_2 = \frac{\sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} ((1_{n \times n} - \Theta) \circ (1_{n \times n} - \Theta^*))_{i,j}}{\sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} (1_{n \times n} - \Theta^*)_{i,j}} \times 100\% \quad (15)
\]

\[
\rho_3 = \frac{\sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} (1_{n \times n} - |\Theta - \Theta^*|)_{i,j}}{n(n-1)/2} \times 100\% \quad (16)
\]

where the $\Theta$ matrix stores the decomposition results by the designed algorithm, the $\Theta^*$ matrix records the ideal decomposition results, all the elements in the matrix $1_{n \times n}$ is 1, and the operator $\circ$ means the entrywise product of two matrices. In this paper, we propose an extend metrics to calculate the overall decomposition performance as the following:

\[
\rho = \frac{\sum_{i=1}^{n} \min(\rho_1, \rho_2, \rho_3)}{n} \times 100\% \quad (17)
\]
4.2 Sensitivity analysis on the parameters of SDG

In order to investigate the sensitivity of the parameters on the performance of SDG, we carried the following tests by running SDG with different combinations among the parameters. All the tests were used to get the variable grouping results on the IEEE CEC’2010 test suites. The values of performance metrics $\rho_1$, $\rho_2$, and $\rho_3$ were recorded and then the overall value of performance metrics $\rho$ was obtained and given in Table 3. From Table 3, it can be seen that the parameter configuration influences the variable grouping accuracy. The parameter combinations of $\{\alpha = 50, \beta = 10^{-8}, \mu = 10^0\}$, $\{\beta = 10^{-8}, \mu = 10^2, \alpha = 10^1\}$, $\{\alpha = 50, \mu = 10^2, \beta = 10^{-10}\}$, and $\{\alpha = 50, \mu = 10^2, \beta = 10^{-9}\}$ can get the ideal decomposition.

Table 3  Test results for SDG with different combinations of parameters

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>$\beta$</th>
<th>$\mu$</th>
<th>$\rho$</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>$10^{-8}$</td>
<td>$10^{-3}$</td>
<td>93.7%</td>
</tr>
<tr>
<td>50</td>
<td>$10^{-8}$</td>
<td>$10^{-2}$</td>
<td>96.5%</td>
</tr>
<tr>
<td>50</td>
<td>$10^{-8}$</td>
<td>$10^{-9}$</td>
<td>97.8%</td>
</tr>
<tr>
<td>$10^2$</td>
<td>100%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^2$</td>
<td>100%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$10^2$</td>
<td>$10^{-12}$</td>
<td>96.2%</td>
<td>99.0%</td>
</tr>
<tr>
<td>$10^2$</td>
<td>$10^{-10}$</td>
<td>100%</td>
<td>97.7%</td>
</tr>
<tr>
<td>$10^2$</td>
<td>$10^{-10}$</td>
<td>100%</td>
<td>97.7%</td>
</tr>
</tbody>
</table>

4.3 Comparison on the number of fitness evaluations for variable grouping

The number of fitness evaluations for variable grouping of DG is calculated as the following equations:

$$GF Es = 2 \left( S + \frac{n}{m} \right),$$

$$S = \frac{n}{2m} (n + m - 2),$$

where $n$ is the number of variables, $m$ is the number of variable groups. For the fully separable functions $f_1$ to $f_3$, there are 1000 groups and then the $GF Es$ is calculated as

$$GF Es = 2 \times \left( \frac{1,000}{2} \times (1,000 + 1 - 2) + \frac{1,000}{1} \right) = 1,001,000.$$

Following the similar calculation, for the fully non-separable functions $f_{19}$ to $f_{20}$, the number of fitness evaluations on grouping is 2,000. For $f_4$ to $f_{18}$, which are partially separable functions, the number of fitness evaluations for variable grouping is between 2,000 and 1,001,000 with regard to the individual functions. For GDG and SDG, the numbers of fitness evaluations on grouping are the same for all the functions and are calculated as equations (21) and (22) respectively,

$$GF Es = 10 + 1 + 1,000 \times 2 + \frac{1,000 \times 999}{2} = 501,511,$$

$$GF Es = 10 + 1 + 1,000 \times 2 + \frac{1,000 \times 999}{2} = 501,511.$$
Large-scale global optimisation using cooperative co-evolution

\[ GFEs = 2 + 1 + 1,000 \times 2 + \frac{1,000 \times 999}{2} = 501,503. \] (22)

4.4 Comparison on grouping accuracy

Table 4 lists the experimental results on the interaction, independence, and both of them of DG, GDG, and the proposed SDG on the CEC’2010 LSGO benchmark suite. In the table, the symbol '-' represents the result cannot be calculated since the specific structure of the benchmark function. For functions \( f_1 \), \( f_2 \), and \( f_3 \), the values of \( \rho_1 \) are not available as all the elements in the ideal decomposition matrices are zero. Functions \( f_{19} \) and \( f_{20} \) are two fully non-separable problems and therefore the summary of \( I_{nxn} - \Theta^* \) in \( \rho_2 \) is zero. For DG, it cannot obtain good decomposition accuracy on the functions of \( f_7 \), \( f_{13} \), \( f_{18} \), and \( f_{20} \) since the incomplete variable comparison during the grouping process. GDG addressed the problems in DG and obtained perfect decomposition results on all almost all the benchmark functions except on \( f_3 \) and \( f_{11} \). The reason for incorrect decomposition of \( f_3 \) and \( f_{11} \) can be found by analyse the formulation of them. Both of these two functions are formed by Ackley function which contains the exponential function. The exponential functions in \( f_3 \) and \( f_{11} \) will cause large computational errors and make it difficult to select a proper threshold parameter \( \epsilon \). By investigating the results of three independent performance metrics and the overall performance metric in Table 4, one can find that the proposed SDG got ideal decomposition results on all the 20 benchmark functions. The high accuracy by SDG attributes to the self-adaptive mechanism according to different functions and the revision approach on the interaction matrix as described in Subsection 3.1. In Sun et al. (2015), an extended DG (XDG) for LSGO problem was proposed and the grouping accuracy of XDG is higher than DG. With different parameter selection in XDG, the grouping accuracy changes on some benchmark functions. The proposed SDG in this paper shows similar performance on the variable decomposition with XDG.

<table>
<thead>
<tr>
<th>Function</th>
<th>DG</th>
<th>GD</th>
<th>SDG</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_1 )</td>
<td>- 100% 100%</td>
<td>- 100% 100%</td>
<td>- 100% 100%</td>
</tr>
<tr>
<td>( f_2 )</td>
<td>- 100% 100%</td>
<td>- 100% 100%</td>
<td>- 100% 100%</td>
</tr>
<tr>
<td>( f_3 )</td>
<td>100% 84.70% 84.70%</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
</tr>
<tr>
<td>( f_4 )</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
</tr>
<tr>
<td>( f_5 )</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
</tr>
<tr>
<td>( f_6 )</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
</tr>
<tr>
<td>( f_7 )</td>
<td>55.60% 81.70% 81.60%</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
</tr>
<tr>
<td>( f_8 )</td>
<td>81.30% 77.40% 77.40%</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
</tr>
<tr>
<td>( f_9 )</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
</tr>
<tr>
<td>( f_{10} )</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
</tr>
<tr>
<td>( f_{11} )</td>
<td>99.60% 100% 99.90%</td>
<td>100% 75.50% 76.10%</td>
<td>100% 100% 100%</td>
</tr>
<tr>
<td>( f_{12} )</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
</tr>
<tr>
<td>( f_{13} )</td>
<td>24.30% 93.60% 91.90%</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
</tr>
<tr>
<td>( f_{14} )</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
<td>100% 100% 100%</td>
</tr>
</tbody>
</table>
Table 4  Grouping accuracy of DG, GDG, and SDG on the CEC’2010 LSGO benchmark suite (continued)

<table>
<thead>
<tr>
<th></th>
<th>DG</th>
<th>GDG</th>
<th>SDG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\rho_1$</td>
<td>$\rho_2$</td>
<td>$\rho_3$</td>
</tr>
<tr>
<td>$f_{15}$</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>$f_{16}$</td>
<td>99.20%</td>
<td>100%</td>
<td>99.90%</td>
</tr>
<tr>
<td>$f_{17}$</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>$f_{18}$</td>
<td>38.30%</td>
<td>76.90%</td>
<td>75.00%</td>
</tr>
<tr>
<td>$f_{19}$</td>
<td>100%</td>
<td>-</td>
<td>100%</td>
</tr>
<tr>
<td>$f_{20}$</td>
<td>8.90%</td>
<td>-</td>
<td>8.90%</td>
</tr>
<tr>
<td>$\rho$</td>
<td>79.40%</td>
<td>93.90%</td>
<td>100%</td>
</tr>
</tbody>
</table>

4.5 Comparison on the optimisation results

The experimental results are shown in Table 5 with mean and standard deviation values obtained by the compared algorithms using 25 independent runs on the CEC’2010 LSGO benchmark functions. The best mean values are highlighted in ital. Table 6 shows the $t$-test results on every benchmark function of the unpaired two-tailed test with significance level of 0.05 between two algorithms. Regarding the term Algorithm 1 vs. Algorithm 2, the signatures ‘+’, ‘-’, and ‘=’ in Table 6 indicate that Algorithm 1 is significantly better than, worse than, and equal to Algorithm 2 respectively. The number of ‘+’, ‘-’, and ‘=’ with regard to every paired comparison are also listed in Table 6, which are denoted as ‘b/w/e’. In order to show the overall comparison results, the difference between the number of significantly better and significantly worse is also calculated which is denoted as ‘gm’ in Table 6. In Table 6, the results of the multiple comparison procedure analysis of variance (ANOVA) with 0.05 as the level of significance are also recorded. The resulting rank of each algorithm is shown on the top right of the mean values and the total rank of each algorithm is given in the bottom of the table.

Random grouping for variable decomposition is used in MLCC and the algorithm shows better performance on the fully separable functions, especially for $f_1$ and $f_2$ where best mean fitness values are obtained. For DECC-D and DECC-DML, they both carry out delta decomposition, which is a kind of variable decomposition strategy between random grouping and DG, and has best mean fitness values on $f_3$, $f_6$, $f_{11}$, and $f_{20}$. From Table 5, it can be seen that the proposed CC-SPA-SDG shows superior performance on the partially separable functions and wins 5 out of 15 functions which are $f_4$, $f_5$, $f_7$, $f_9$, and $f_{13}$. From the statistical results in Table 6, CCSPA-SDG outperforms DECC-DG, DECC-I, MLCC, DECC-DML, and DECC-D significantly and has similar performance with MA-SW-CHAINS which was the winner in 2010. From the values of total rank, MA-SW-chains ranks the best and the proposed CCSPA-SDG is the second best algorithm.
Table 5  Mean and standard deviation of fitness values and ranking results obtained by the compared algorithms using 25 independent runs on the CEC’2010 LSGO benchmark functions

<table>
<thead>
<tr>
<th>f</th>
<th>DECC-DG</th>
<th>DECC-I</th>
<th>DECC-DML</th>
<th>DECC-DG-CCSPA-SDG</th>
</tr>
</thead>
<tbody>
<tr>
<td>f1</td>
<td>(4.48E+03)</td>
<td>(9.26E+00)</td>
<td>(7.96E-27)</td>
<td>(1.34E+00)</td>
</tr>
<tr>
<td></td>
<td>(2.10E-14)</td>
<td>(1.35E-25)</td>
<td>(5.29E-09)</td>
<td></td>
</tr>
<tr>
<td>f2</td>
<td>(4.47E+03)</td>
<td>(7.66E+00)</td>
<td>(7.96E-27)</td>
<td>(1.34E+00)</td>
</tr>
<tr>
<td></td>
<td>(1.99E-14)</td>
<td>(1.63E-25)</td>
<td>(8.09E-09)</td>
<td></td>
</tr>
<tr>
<td>f3</td>
<td>(1.63E+04)</td>
<td>(2.55E+00)</td>
<td>(9.29E-24)</td>
<td>(7.66E-27)</td>
</tr>
<tr>
<td></td>
<td>(1.99E-14)</td>
<td>(1.63E-25)</td>
<td>(8.09E-09)</td>
<td></td>
</tr>
<tr>
<td>f4</td>
<td>(4.78E+12)</td>
<td>(6.13E+11)</td>
<td>(3.64E+12)</td>
<td>(9.61E+12)</td>
</tr>
<tr>
<td></td>
<td>(2.23E+12)</td>
<td>(2.67E+02)</td>
<td>(3.72E+01)</td>
<td>(1.35E+02)</td>
</tr>
<tr>
<td>f5</td>
<td>(1.57E+08)</td>
<td>(1.34E+08)</td>
<td>(2.56E+08)</td>
<td>(3.84E+08)</td>
</tr>
<tr>
<td></td>
<td>(2.04E+07)</td>
<td>(2.31E+00)</td>
<td>(1.81E-01)</td>
<td>(5.05E-15)</td>
</tr>
<tr>
<td>f6</td>
<td>(1.64E+01)</td>
<td>(1.64E+01)</td>
<td>(5.40E-09)</td>
<td>(1.62E+07)</td>
</tr>
<tr>
<td></td>
<td>(3.85E-01)</td>
<td>(2.66E-01)</td>
<td>(1.81E-09)</td>
<td>(4.97E+06)</td>
</tr>
<tr>
<td>f7</td>
<td>(1.02E+04)</td>
<td>(2.97E+01)</td>
<td>(6.58E+07)</td>
<td>(9.20E+07)</td>
</tr>
<tr>
<td></td>
<td>(2.19E+07)</td>
<td>(4.28E+02)</td>
<td>(8.70E+00)</td>
<td>(1.47E-07)</td>
</tr>
<tr>
<td>f8</td>
<td>(2.56E+07)</td>
<td>(3.19E+05)</td>
<td>(9.20E+07)</td>
<td>(6.11E+08)</td>
</tr>
<tr>
<td></td>
<td>(2.19E+07)</td>
<td>(4.28E+02)</td>
<td>(8.70E+00)</td>
<td>(1.47E-07)</td>
</tr>
<tr>
<td>f9</td>
<td>(5.59E+07)</td>
<td>(7.51E+02)</td>
<td>(3.19E+05)</td>
<td>(9.20E+07)</td>
</tr>
<tr>
<td></td>
<td>(2.19E+07)</td>
<td>(4.28E+02)</td>
<td>(8.70E+00)</td>
<td>(1.47E-07)</td>
</tr>
<tr>
<td>f10</td>
<td>(5.54E+03)</td>
<td>(5.34E+03)</td>
<td>(3.70E+08)</td>
<td>(7.51E+02)</td>
</tr>
<tr>
<td></td>
<td>(2.30E+07)</td>
<td>(1.09E+06)</td>
<td>(8.70E+00)</td>
<td>(1.47E-07)</td>
</tr>
<tr>
<td>f11</td>
<td>(5.54E+03)</td>
<td>(5.34E+03)</td>
<td>(3.70E+08)</td>
<td>(7.51E+02)</td>
</tr>
<tr>
<td></td>
<td>(2.30E+07)</td>
<td>(1.09E+06)</td>
<td>(8.70E+00)</td>
<td>(1.47E-07)</td>
</tr>
</tbody>
</table>

Notes: Best fitness values are highlighted in ital.
Table 5
Mean and standard deviation of fitness values and ranking results obtained by the compared algorithms using 25 independent runs on the CEC'2010 LSGO benchmark functions (continued)

<table>
<thead>
<tr>
<th></th>
<th>DECC-DG</th>
<th>DECC-I</th>
<th>DECC-D</th>
<th>MLCC</th>
<th>MA-SW-chains</th>
<th>DECC-DML</th>
<th>CCSPA-SDG</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_{15}$</td>
<td>$(5.86E+03)^i$</td>
<td>$(5.87E+03)^i$</td>
<td>$(1.59E+04)^i$</td>
<td>$(7.11E+03)^i$</td>
<td>$(2.72E+03)^i$</td>
<td>$(1.54E+04)^i$</td>
<td>$(4.59E+03)^i$</td>
</tr>
<tr>
<td></td>
<td>8.03E+01</td>
<td>9.89E+01</td>
<td>3.09E+02</td>
<td>1.34E+03</td>
<td>1.22E+02</td>
<td>5.17E+02</td>
<td>1.24E+02</td>
</tr>
<tr>
<td>$f_{16}$</td>
<td>$(7.53E-13)^i$</td>
<td>$(2.47E-13)^i$</td>
<td>$(5.88E-02)^i$</td>
<td>$(7.36E+02)^i$</td>
<td>$(1.01E+02)^i$</td>
<td>$(1.56E-01)^i$</td>
<td>$(1.56E+01)^i$</td>
</tr>
<tr>
<td></td>
<td>5.13E-14</td>
<td>9.17E-15</td>
<td>2.94E-02</td>
<td>4.71E+01</td>
<td>1.45E+01</td>
<td>4.33E+00</td>
<td>3.28E+00</td>
</tr>
<tr>
<td>$f_{17}$</td>
<td>$(4.06E+04)^i$</td>
<td>$(3.91E+04)^i$</td>
<td>$(7.32E+06)^i$</td>
<td>$(1.59E+05)^i$</td>
<td>$(1.24E+00)^i$</td>
<td>$(6.58E+06)^i$</td>
<td>$(1.60E+02)^i$</td>
</tr>
<tr>
<td></td>
<td>2.41E+03</td>
<td>2.75E+03</td>
<td>4.58E+05</td>
<td>1.43E+04</td>
<td>1.25E+01</td>
<td>3.43E+05</td>
<td>1.45E+02</td>
</tr>
<tr>
<td>$f_{18}$</td>
<td>$(1.44E+10)^i$</td>
<td>$(1.17E+03)^i$</td>
<td>$(1.87E+03)^i$</td>
<td>$(7.09E+03)^i$</td>
<td>$(1.30E+03)^i$</td>
<td>$(2.19E+03)^i$</td>
<td>$(1.71E+03)^i$</td>
</tr>
<tr>
<td></td>
<td>2.60E+09</td>
<td>9.65E+01</td>
<td>6.55E+02</td>
<td>4.77E+03</td>
<td>4.36E+02</td>
<td>4.73E+02</td>
<td>3.28E+02</td>
</tr>
<tr>
<td>$f_{19}$</td>
<td>$(1.73E+06)^i$</td>
<td>$(1.74E+06)^i$</td>
<td>$(1.90E+07)^i$</td>
<td>$(1.36E+06)^i$</td>
<td>$(2.85E+05)^i$</td>
<td>$(1.53E+07)^i$</td>
<td>$(2.73E+07)^i$</td>
</tr>
<tr>
<td></td>
<td>9.19E+04</td>
<td>9.54E+04</td>
<td>1.06E+06</td>
<td>7.33E+04</td>
<td>1.73E+04</td>
<td>1.09E+06</td>
<td>1.40E+07</td>
</tr>
<tr>
<td>$f_{20}$</td>
<td>$(6.64E+10)^i$</td>
<td>$(4.14E+03)^i$</td>
<td>$(1.15E+03)^i$</td>
<td>$(2.05E+03)^i$</td>
<td>$(1.07E+03)^i$</td>
<td>$(9.97E+02)^i$</td>
<td>$(9.89E+05)^i$</td>
</tr>
<tr>
<td></td>
<td>9.15E+09</td>
<td>8.14E+02</td>
<td>7.40E+01</td>
<td>1.80E+02</td>
<td>7.29E+01</td>
<td>3.42E+01</td>
<td>6.60E+05</td>
</tr>
</tbody>
</table>

Notes: Best fitness values are highlighted in ital.
Large-scale global optimisation using cooperative co-evolution

Table 6 The t-test results of comparison among CCSPA-SDG and the other five compared algorithms

<table>
<thead>
<tr>
<th>t-test</th>
<th>f1</th>
<th>f2</th>
<th>f3</th>
<th>f4</th>
<th>f5</th>
<th>f6</th>
<th>f7</th>
<th>f8</th>
<th>f9</th>
<th>f10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC-SPA-SDG vs. DECC-DG</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>=</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>CC-SPA-SDG vs. DECC-I</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>CC-SPA-SDG vs. MLCC</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>CC-SPA-SDG vs. DECC-DML</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>=</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>CC-SPA-SDG vs. MA-SW-CHAIN</td>
<td>-</td>
<td>+</td>
<td>=</td>
<td>=</td>
<td>=</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CC-SPA-SDG vs. DECC-D</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>t-test</th>
<th>f11</th>
<th>f12</th>
<th>f13</th>
<th>f14</th>
<th>f15</th>
<th>f16</th>
<th>f17</th>
<th>f18</th>
<th>f19</th>
<th>f20</th>
<th>h/w/e/gm</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC-SPA-SDG vs. DECC-DG</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>16/2/2/14</td>
</tr>
<tr>
<td>CC-SPA-SDG vs. DECC-I</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>14/5/1/9</td>
</tr>
<tr>
<td>CC-SPA-SDG vs. MLCC</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>15/5/0/10</td>
</tr>
<tr>
<td>CC-SPA-SDG vs. DECC-DML</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>12/7/1/5</td>
</tr>
<tr>
<td>CC-SPA-SDG vs. MA-SW-CHAIN</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>=</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>8/7/5/1</td>
</tr>
<tr>
<td>CC-SPA-SDG vs. DECC-D</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>13/7/0/6</td>
</tr>
</tbody>
</table>

5 Conclusions

In this paper, a SDG for variable decomposition with SPA for computational resource allocation is proposed for solving LSGO problems. SDG is an improved version of the GDG which is used to divide the problems into several subcomponents. SDG has obtained the 100% grouping accuracy on all the twenty LSGO test problems of CEC'2010 which is better than that of GDG. Based on the SPA strategy, different computational resources are allocated to different subcomponents by considering their contributes to the objective functions. The proposed SDG and SPA are working with the optimiser SaNSDE to solve the CEC'2010 LSGO test problems and shows superior performance than the state-of-the-art results. In the future, we will continue to study SDG in order to identify the ideal partitions on the more complex LSGO problem, e.g., CEC'2013 LSGO test suite.

Acknowledgements

This work was supported in part by the National Key R&D Program of China under Grant 2017YFC1601800 and 2017YFC1601000, 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.
References


Large-scale global optimisation using cooperative co-evolution


