This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Multimodal Optimization Enhanced Cooperative Coevolution for Large-Scale Optimization

Xingguang Peng, *Member, IEEE*, Yaochu Jin, *Fellow, IEEE*, and Handing Wang, *Member, IEEE*

***Abstract*—Cooperative coevolutionary (CC) algorithms decompose a problem into several subcomponents and optimize them separately. Such a divide-and-conquer strategy makes CC algorithms potentially well suited for large-scale optimization. However, decomposition may be inaccurate, resulting in a wrong division of the interacting decision variables into different subcomponents and thereby a loss of important information about the topology of the overall fitness landscape. In this paper, we suggest an idea that concurrently searches for multiple optima and uses them as informative representatives to be exchanged among subcomponents for compensation. To this end, we incorporate a multimodal optimization procedure into each subcomponent, which is adaptively triggered by the status of subcomponent optimizers. In addition, a nondominance-based selection scheme is proposed to adaptively select one complete solution for evaluation from the ones that are constructed by combining informative representatives from each subcomponent with a given solution. The performance of the proposed algorithm has been demonstrated by comparing five popular CC algorithms on a set of selected problems that are recognized to be hard for traditional CC algorithms. The superior performance of the proposed algorithm is further confirmed by a comprehensive study that compares 17 state-of-the-art CC algorithms and other metaheuristic algorithms on 20 1000-dimensional benchmark functions.**

***Index Terms*—Cooperative coevolutionary (CC) algorithm, information compensation, large-scale optimization (LSO), multimodal optimization (MMO).**

I. Introduction

As a very powerful optimization tool, evolutionary algorithms (EAs) have successfully been applied to a wide range of real-world optimization problems. Nevertheless, EAs have not yet convincingly used for solving large-scale optimization (LSO) problems, which are commonly seen in many science and engineering domains, such as engineering design [1], computational biology [2], operational research [3], just to name a few. Typically, the performance of EAs rapidly decreases as the number of decision variables increases because of the increase in the search space of an optimization problem. Moreover, the increase of the number of decision variables may even lead to a change in the property of an optimization problem [4].

Cooperative coevolutionary (CC) algorithms are a class of EAs that perform optimization in a *divide-and-conquer* manner by decomposing a problem into several relatively small subcomponents and concurrently optimizing them. To fully exploit the capability of CC algorithms for LSO, two important steps in CC algorithms, i.e., problem decomposition and optimization of the subcomponents, must be properly designed.

If interacting variables are divided into separate subcomponents, a great deal of fitness landscape information in a CC algorithm may be lost. Without the right landscape information, coevolutionary individuals will be incorrectly assessed and converge to a Nash equilibrium rather than an optimum [5]. Various problem decomposition methods have been reported in [6]–[10]. Very recently, differential grouping (DG) [9] and global DG (GDG) [10] have been proposed, which are demonstrated to exhibit a very high accuracy of decomposition on the CEC'2010 benchmark functions [11]. However, these techniques often require a large number of fitness evaluations, which makes them impractical for some real-world LSO problems where fitness evaluations are computationally expensive. In addition, it is hardly possible to exactly decompose complex LSO problems that have a nonstationary fitness landscape or constraints.

For CC algorithms, designing a powerful optimization method for each subcomponent is as important as problem decomposition. It has been theoretically proved that insufficient random collaborators (collaborative solutions from the counterpart subcomponents that are used to construct complete solutions for fitness evaluation) might lead to poor fitness estimates and could also result in convergence to suboptimal solutions [12]. In addition, exchanging more collaborators does not necessarily mean more information exchange. An extreme variant of the conventional CC is shown in [13], which exchanges all individuals between subcomponents, is still unable to find the global optimum without a sufficiently large population size. From the above findings, we can conclude

Manuscript received October 9, 2017; revised April 7, 2018; accepted June 5, 2018. This work was supported in part by the National Nature Science Foundation of China under Grant 61473233, in part by the Fundamental Research Funds for the Central Universities of China under Grant 3102016ZY007, and in part by the EPSRC under Grant EP/M017869/1. This paper was recommended by Associate Editor F. Herrera. *(Corresponding author: Yaochu Jin.)*

X. Peng is with the School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: xgpeng.nwpu@gmail.com).

Y. Jin is with the Department of Computer Science, University of Surrey, Guildford GU2 7XH, U.K., and also with the Department of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan 030024, China (e-mail: yaochu.jin@surrey.ac.uk).

H. Wang is with the Department of Computer Science, University of Surrey, Guildford GU2 7XH, U.K. (e-mail: wanghanding.patch@gmail.com).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2018.2846179

that it is essential to develop new CC algorithms that are able to increase the amount of representative information of the collaborators. To achieve this, the diversity or distribution of coevolutionary populations should be properly maintained to prevent from converging to a local optimum but also the informative collaborators should be carefully selected.

To address the above issues, a small number of ideas have been proposed to improve the conventional CC framework, including using random plus historical best individuals as the collaborators [12], or identifying informative collaborators [13]. In our previous work [14], a multipopulation scheme-based CCEA proposed to continuously find multiple local or global optima of subcomponents. These optima are exchanged between subcomponents as informative collaborators. Although multioptimum collaborators have shown to be effective in helping the CCEA achieve much better performance on low-dimensional problems, the following two main challenges remain to be addressed in the CCEA to handle LSO problems.

First, it is challenging to locate multiple optima on a dynamic high-dimensional landscape. To simultaneously locate or maintain multiple optima in a population, certain diversity metrics or neighborhood schemes based on the Euclidian distance are widely utilized. However, these metrics may become ineffective when the dimensionality becomes relatively high because its capability to characterize similarity in terms of Euclidian distance (L2-norm) seriously degrades in a high-dimensional space [15].

Second, constructing a reasonable number of collaborators to evaluate a given solution is not straightforward either. According to the interactive nature of CC algorithms, an individual has to be combined with collaborators provided by the other subcomponents to construct a number of complete solutions for fitness evaluation. An intuitive way to construct such solutions is to mix solutions of one subcomponent with all other collaborators. Unfortunately, the number of completely mixed solutions increases exponentially with the number of subcomponents, which is intractable for solving problems with many subcomponents. For example, if a problem is decomposed into two subcomponents and each subcomponent provides two collaborators, only two fitness evaluations are required to evaluate an individual of a subcomponent. However, this number will dramatically increase to $2^{(20-1)} = 524\,288$ if an LSO problem is decomposed into 20 subcomponents. Such a vast number of fitness evaluations will consume most of the computing resource for problem solving.

This paper aims to achieve efficient information compensation by adaptively identifying the most informative collaborators between subcomponents to avoid exhaustive combinations of an given individual with all collaborators provided by other subcomponents, thereby significantly reducing the number of fitness evaluations as the number of subcomponents increases. The main contributions of this paper are summarized as follows.

1) A multimodal optimization (MMO) procedure is incorporated into the CCEA for information compensation. The main idea is to simultaneously obtain multiple global or local optima of a subcomponent, which are considered to be the informative representatives and occasionally exchanged among subcomponents to achieve efficient information compensation.
2) A collaborator construction scheme is proposed to avoid combinatorial explosion when evaluating a given individual. Within such a scheme, a small number of most informative collaborators are selected according to their fitness values and the degree of diversity. Moreover, after a short phase in which the best collaborators are counted, only one collaborator is eventually used for fitness evaluations afterwards.
3) A modified covariance matrix adaptation evolution strategy (CMA-ES) [16] has been adopted as the subcomponent optimizer that is able to adaptively trigger the MMO procedure. The subcomponent optimizer and the MMO can work together to guide the optimization process.

To demonstrate the effectiveness and efficiency of the proposed algorithm, comprehensive empirical studies have been carried out. The proposed algorithm is first compared with five state-of-the-art CC algorithms on a set of relative-overgeneralization-featured test problems, which are considered to be hard for traditional CC algorithms. In addition, the proposed algorithm is compared with 11 popular CC algorithms and six widely used metaheuristic algorithms, including the winners of the CEC competitions from 2010 to 2015, on the CEC'2010 benchmark suite for large-scale global optimization. At last, the effect of varying the main parameters of the proposed algorithm is also examined and analyzed.

The remainder of this paper is organized as follows. In Section II, we review LSO in the field of evolutionary computation and introduce the basic framework of CC algorithm. In Section III, the details of the proposed algorithm are described. Experiments are conducted in Sections IV and V. The conclusions and future work are given in Section VI.

II. Background

A. Review of EAs for LSO

A number of pieces of research work have been reported on solving LSO in the literature, which can roughly be divided into metaheuristics and divide-and-conquer methods.

1) Metaheuristic Algorithms for LSO: Many metaheuristic optimization methods and their variants have directly been employed to solve high-dimensional optimization problems. The memetic algorithm using Solis Wets [17] based local search chains (termed MA-SW-Chains) [18], the winner of CEC'2010 competition on LSO, embeds adaptive local search processes in [19] to exploit the most promising areas represented in the EA population. A comprehensive comparison of MA-SW-Chains with several well-known local search optimizers, including CMA-ES, multiple trajectory search [20] and Simplex [21], was also made on LSO test functions [18]. In [22], both population-based and local search algorithms are integrated in a multiple offspring sampling (MOS) framework [23]. By dynamically combining several well-known and a newly proposed optimizer in a sequential manner according to some quality measure, the MOS-based hybrid algorithm can utilize positive properties of different optimizers to achieve

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

satisfactory search performance. This algorithm was the winner of competitions at both CEC'2013 and CEC'2015 and a wide range of comparisons on CEC'2013 benchmark suite can also be found in [4].

In addition to the algorithms mentioned above, efforts have been made to explore the way to combine multipopulation scheme with subregional harmony search [24] or to incorporate estimation of distribution algorithm (EDA) with local search techniques [25] and variable mesh optimization [26]. Recently, a new variant of particle swarm optimization that introduces pairwise competition between particles has shown to perform very well on LSO [27]. A comprehensive survey of metaheuristic algorithms for large-scale global continuous optimization can be found in [28].

2) Divide-and-Conquer-Based Methods for LSO: In contrast to metaheuristic algorithms that directly search the whole high-dimensional decision space, the divide-and-conquer approaches in evolutionary computation, mainly CC algorithms, decompose a given LSO problem into a number of small subcomponents that are separately optimized by different EAs. Many CC algorithms have been proposed that largely fall into two categories depending on when the problem decomposition is conducted.

a) Decomposition before coevolution: If a problem is decomposed before co-evolution, the problem solving process is divided into two sequential stages, namely decomposition and optimization. Chen *et al.* [8] proposed a CC algorithm using variable interaction learning and JADE [29] (an adaptive differential evolution with optional external archive). To decompose an LSO problem at a reasonable computation cost, a more efficient and effective decomposition method, named DG was proposed by Omidvar *et al.* [9]. Every two pairwise variables are checked via a differential technique and the decomposition is achieved by iteratively grouping interacting variables. In addition, two contribution-based CC, termed CBCC1 and CBCC2, respectively, were also proposed based on DG. These algorithms are reported to have significantly outperformed MLCC [30] [a multilevel CC (MLCC) algorithm] and MA-SW-Chains [31].

Although it has achieved satisfied decomposition accuracy, DG is ineffective in detecting variable interactions where a variable interacts with more than one other variable. To tackle this problem, very recently, Mei *et al.* proposed a GDG [10] method which incorporates the *design structure matrix* [32] to the DG to maintain the global information of interactions between variables. In addition, a group of separable variables are further divided into several smaller subsets whose sizes are set empirically to make the subcomponent optimizer (a modified CMA-ES) perform better. It is shown that the GDG can very effectively decompose almost all of functions in the CEC'2010 competition benchmark suite.

Despite their great success in problem decomposition, most divide-and-conquer based LSO algorithms mentioned above only exchange the best-so-far solutions between subcomponents. Such a conventional CC framework has been theoretically proved to be insufficient to compensate information in the presence of decomposition error [12]. Considering the challenges of precise decomposition of real-world LSO problems, more effort should be made to improve the conventional CC framework.

b) Decomposition during coevolution: Yang *et al.* [6] incorporate a *random grouping* strategy with differential evolutionary cooperative coevolution (DECC). In the resultant algorithm termed DECC-G, according to the *random grouping* strategy, a problem is dynamically decomposed into a fixed number of subcomponents by randomly allocating decision variables to the subcomponents in every generation. DECC-G have been extended to improve *random grouping*. For example, in DECC-ML [33] (DECC with a modified multilevel cooperative coevolution technique extended from [30]) more frequent random grouping was suggested. a CC particle swarm optimization [34] adopts an adaptive scheme to dynamically determine the subcomponent sizes for random grouping during a run. Very recently, Kabán *et al.* [35] introduced an ensemble of random projections to solve LSO within the framework of EDA with fixed subcomponent sizes.

Another competitive decomposition strategy is *delta grouping* [7]. It measures the amount of change (*delta* value) in each of the decision variables to identify the interacting variables. The variables with smaller *delta* values are grouped in one subcomponent. To avoid using fixed subcomponent sizes, an MLCC algorithm [30] was proposed. In MLCC a set of possible subcomponent sizes are provided and a proper size value is assigned to each subcomponent according to the performance of using these values during the coevolution. To scale up the CMA-ES to LSO problems, Liu and Tang [36] proposed a CC-based CMA-ES, named CC-CMA-ES. A decomposition strategies pool is constructed and used in an adaptive manner to decompose a given problem.

While most problem decomposition techniques examine the correlation relationship between variables, a few algorithms focus on the influence of different decision variables on the convergence and diversity in large-scale multiobjective optimization. García-Sánchez *et al.* [37] extended the distributed coevolutionary island-model [38] based on the parallel execution of the subpopulations, whose individuals explore different domains of the decision space. Ma *et al.* [39] proposed a decision variable analysis method based on dominance relationships to divide the decision variables into three groups, i.e., convergence-related variables, diversity-related variables, and variables related to both convergence and diversity. A more effective division method was suggested by Zhang *et al.* [40] using the k-means clustering method with respect to the angles between the solutions and the direction of convergence. As a result, all decision variables are labeled as either convergence-related or diversity-related.

B. Brief Introduction to CC Algorithms

CC algorithms perform optimization in a *divide-and-conquer* manner by decomposing a problem into several subcomponents, each of which can be concurrently optimized by a coevolutionary algorithm. The main difference between classical EAs and CC algorithms is that in CC algorithms an individual encodes only a segment (according to its subcomponent) of the solution. To evaluate such a solution segment,

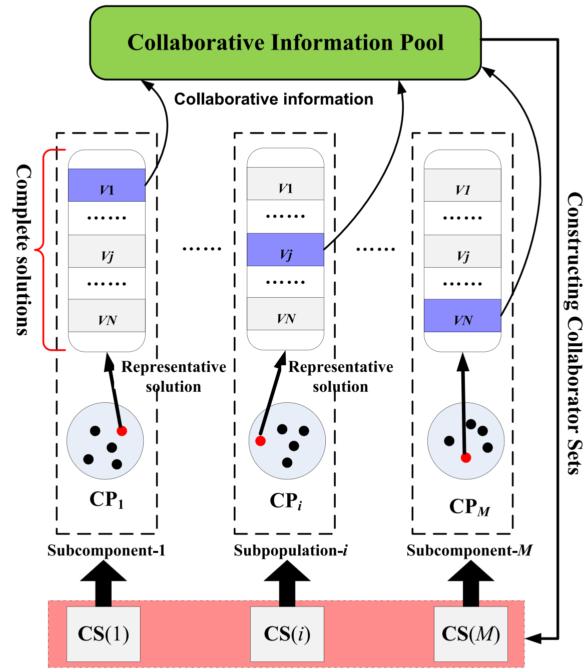

Fig. 1. Diagram of a generic CC algorithm.

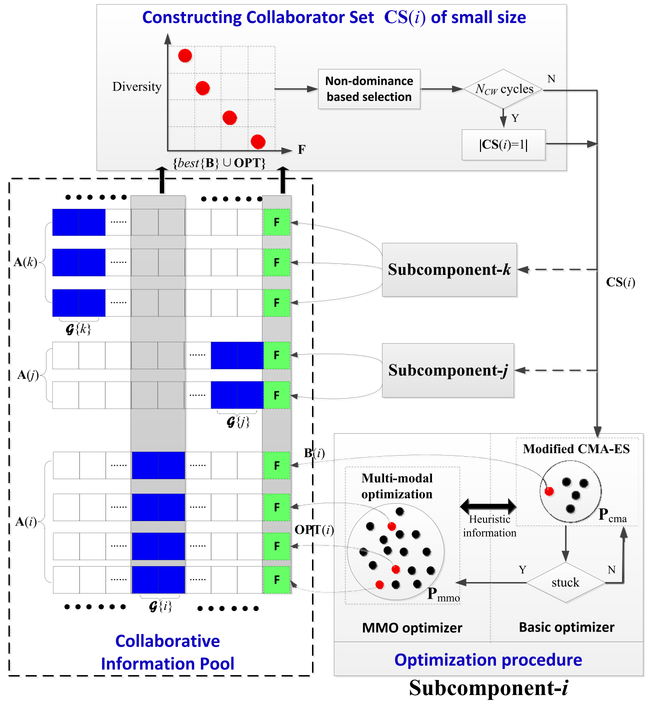

Fig. 2. Diagram of the MMO-CC framework.

one has to collect collaborators from the other subpopulations to construct complete solutions for fitness calculation.

A generic diagram of CC algorithms is shown in Fig. 1, where an N-dimensional problem is divided into M subcomponents $\mathcal{G}(i)$, $i = 1, \ldots, M$, each of which is evolved by a coevolutionary subpopulation, $\mathbf{CP}(i), i = 1, \ldots, M$ in a sequential or parallel manner. During the coevolution process, the coevolutionary subpopulations continuously exchange their representative collaborators, with which a virtual collaborative information pool can be built up. From the collaborative information pool one can construct the corresponding collaborator set $\mathbf{CS}(i)$ for $\mathbf{CP}(i)$. With such collaborators a given individual (a segment of problem solution) in $\mathbf{CP}(i)$ can be reconstructed and the corresponding complete solutions can be obtained for fitness evaluation. Generally speaking, more than one representative solution can be provided by the collaborative information pool, any solution that can effectively represent the status of the corresponding subcomponent can be treated as a representative collaborator.

III. Proposed CC Algorithm for Large-Scale Optimization

To tackle the above-mentioned challenges of extending CC to large-scale global optimization, we present an MMO enhanced CC framework in this section. According to the categories of CC in Section II-A, our CC framework belongs to the two-stage CC. We focus on the optimization stage with a given problem decomposition.

A. Framework of MMO Enhanced CC (MMO-CC)

The framework of the proposed MMO-CC is illustrated in Fig. 2. Each subcomponent (the ith component) undergoes two optimization procedures. In the first procedure, a modified CMA-ES (referring to Section III-C) continues performing optimization and updates the corresponding best individual $B(i)$ at each cycle. Second, an MMO procedures (referring to Section III-B) is occasionally triggered if CMA-ES gets stuck. After $\mathrm{NG}_{\mathrm{mmo}}$ generations, a set of current (global or local) optima $\mathbf{OPT}(i)$ can be obtained. Therefore, at each cycle, the ith subcomponent provides the collaborative information pool with a set of local representative $\mathcal{A}(i) = B(i) \cup \mathbf{OPT}(i)$. Note that, as shown in Fig. 2 the elements in $\mathcal{A}(i)$ are maintained in form of complete solutions together with their fitness values. In addition, these two optimizers mutually provide information when the optimization procedure is switching between them.

During the whole CC process, the ith subcomponent has to construct its collaborative set $\mathbf{CS}(i)$ so that the populations of MMO and CMA-ES optimizers can be evaluated. As shown in Fig. 2, a collaborator construction scheme (referring to Section III-D) is employed to keep $|\mathbf{CS}(i)|$ at a reasonable level. Within such scheme, two techniques are proposed to minimize the number of fitness evaluations. First, after the MMO procedure, a nondominance-based selection is used to construct a small number of collaborators from the pool according to the fitness values and diversity. In addition, after a short phase, only one collaborator is eventually fixed and used for fitness evaluations afterwards until another MMO is triggered.

B. Multimodal Optimization Procedure

A number of niching techniques have been developed to simultaneously locate multiple optima or preferred part of search space to prevent the population from converging to a single solution. Recently, a few bi-objective

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

methods [41]–[44] are proposed to solve MMO, thereby avoiding defining a threshold for the radius distance often used in many conventional niching techniques. The basic idea behind the bi-objective methods is to define a second objective in addition to the original objective.

In this paper, inspired by the ideas in [42] and [43], we define a second objective that maximizes the diversity of the individuals, mainly because that diversity is essential for the population to maintain multiple optima and is less problem-dependent. Considering a given subcomponent with population $\mathbf{P}_{\text{mmo}}$ conducting the MMO procedure, the ith individual X_i in $\mathbf{P}_{\text{mmo}}$ is evaluated using the following two objectives:

$$\begin{cases} f_1(i) = f(X_i) \\ f_2(i) = \frac{1}{|\mathbf{P}_{\text{mmo}}|} \sum_{j=1}^{|\mathbf{P}_{\text{mmo}}|} \left\| \overrightarrow{X_i} - \overrightarrow{X_j} \right\|^{\text{LP}} \end{cases} \tag{1}$$

where f is the original objective function calculated using the corresponding collaborator set. $\| * \|^{\text{LP}}$ means the Minkowski distance, where LP is set to $1/\min\{|\mathcal{G}(i)|, 50\}$ considering the relatively large high-dimensional search space of a subcomponent.

The elitist nondominated sorting genetic algorithm (NSGA-II [45]) is adopted to solve the above bi-objective optimization problem. Note that in every generation, the parent and offspring populations are combined into one single population, from which the parent population of the next generation will be selected based on nondominated sorting and the crowding distance. This means that the size of P_{mmo} will temporally change to $2|\mathbf{P}_{\text{mmo}}|$, which is used in (1). In this paper, the values of the two objectives of X_j are normalized as follows for better dominance comparisons:

$$\begin{cases} f_1(i)' = \frac{f_1(i)}{\sqrt[2]{\sum_{j=1}^{2|\mathbf{P}_{\text{mmo}}|} f_1(j)^2}} \\ f_2(i)' = 1 - \frac{f_2(i)}{\sqrt[2]{\sum_{j=1}^{2|\mathbf{P}_{\text{mmo}}|} f_2(j)^2}}. \end{cases} \tag{2}$$

Note that when the MMO procedure is triggered in the ith subcomponent (i.e., the CMA-ES optimizer gets stuck), the best-ever solution provided by the CMA-ES optimizer is used to replace a randomly selected dominated individual in $\mathbf{P}_{\text{mmo}}(i)$. Actually, in MMO it is not necessary to precisely locate the optima of each subcomponent, rather, rough information about the regions where optima are located is sufficient. With such information, the most potential optimum of a subcomponent will be further improved by the corresponding CMA-ES optimizer. This means that in the bi-objective optimization-based MMO, rough Pareto solutions of a subcomponent is sufficient. In addition, as for the multiobjective optimization, the Pareto front is usually dramatically improved and closed to the real one by a running multiobjective EA for a short period [46], [47]. Therefore, it is reasonable to use the MMO procedure to explore potential optima approximately with a small number of evolution generations (NG_{mmo}). With these optima (i.e., $\mathbf{OPT}(i)$), the other subcomponents could obtain an estimation about highly interested areas in ith subcomponent.

C. Modified CMA-ES Optimizer

Each subcomponent employs an independent CMA-ES optimizer whose status information $\mathcal{S}(i)$ is continuously maintained and updated in every coevolution cycle. $\mathcal{S}(i)$ is a structured data set containing the information of population $\mathbf{P}_{\text{cma}}(i)$, e.g., the mean and standard deviation, the covariance matrix, and the best so far solution *bestEver* since reinitialization.

In this paper, we adopt a variant of the CMA-ES by modifying the basic version (*purecmaes.m*)[1] provided by Hansen. The CMA-ES optimizers should be considered to be reinitialized after the MMO procedure. As for the ith subcomponent, the best solution in $\mathbf{OPT}(i)$ is selected according to the original objective function. Then this solution is used to reinitialize the mean value of initial CMA-ES population $\mathbf{P}_{\text{cma}}(i)$ if it is better than $\mathcal{S}(i).bestEver$. Meanwhile, the standard deviation of the initial CMA-ES population is set to 30% of the range of the decision variables as suggested in [10]. Thus, the contribution of the MMO procedure to the global optimum can also be utilized in the coevolution process.

As mentioned in Section III-A, the MMO procedure is triggered when the CMA-ES procedure gets stuck. Here, a stagnation is recognized if the CMA-ES procedure stagnates for *MaxStk* cycles which can be calculated as follows:

$$MaxStk = \min\left\{ \text{mean}\left(30 \times \left\lceil \frac{|\mathbf{P}_{\text{cma}}(i)|}{|\mathcal{G}(i)|} \right\rceil \right), 200 \right\} \tag{3}$$

where $i \in M$.

D. Constructing Collaborator Sets for Fitness Evaluation

Here we present a scheme to construct $\mathbf{CS}(i)$ of a reasonably small size. Two techniques are employed in turn to reduce the size of $\mathbf{CS}(i)$. First, after the MMO procedure, a nondominance-based selection is used to select a small number of collaborators from the collaborative information pool. Second, after a short phase, only one collaborator is eventually fixed and used for the afterward fitness evaluation until another MMO is triggered.

1) Nondominance-Based Collaborator Selection: First, we can obtain a relatively large collaborator set $\Omega_i = \{\{\mathbf{OPT}\} \cup \text{best}\{\mathbf{B}\}\}$. Then the unique elements are kept by comparing the values of the ith subcomponent's decision variables $\mathcal{G}(i)$, i.e., $\Omega(i) = \text{unique}(\Omega_i, \mathcal{G}(i))$. More specifically, if there are more than two elements whose values for the decision variables $\mathcal{G}(i)$ are the same, only the one with the higher fitness will be kept. To reduce the number of elements in Ω_i, it is intuitive to remove the ones with a relatively poor fitness and a low degree of diversity. Therefore, like the MMO procedure, we also the use fitness value and diversity as the two criteria to conduct the nondominance selection of Ω_i

$$\begin{cases} C_1(i,j) = F(\Omega_i(j)) \\ C_2(i,j) = \frac{1}{|\Omega_i|} \sum_{k=1}^{|\Omega_i|} \|\Omega_i(j, \mathcal{G}(i)) - \Omega_i(k, \mathcal{G}(i))\|^{\text{LP}} \end{cases} \tag{4}$$

where $i \in M$, $j = 1, \ldots, |\Omega_i|$. Note that here F does not need to be calculated, as it is already stored together with an

[1] https://www.lri.fr/~hansen/purecmaes.m

element. $\| * \|^{\mathrm{LP}}$ means the Minkowski distance, where *LP* is set to $1/\min\{|\mathcal{G}(i)|, 50\}$.

To better conduct nondominance comparison, the above criteria are normalized as follows:

$$\begin{cases} C_1(i,j)' = \frac{C_1(i,j)}{\sqrt[2]{\sum_{k=1}^{|\Omega_i|} C_1(i,k)^2}} \\ C_2(i,j)' = 1 - \frac{C_2(i,j)}{\sqrt[2]{\sum_{k=1}^{|\Omega_i|} C_2(i,k)^2}}. \end{cases} \tag{5}$$

After all elements in Ω_i have been compared using (5), nondominated elements can be determined. Considering that the dimensionality of a subcomponent could be still relatively high and the second criterion (mean diversity) may still result in many nondominated elements, we further select the nondominated elements using a grid-based method. Actually, the grid-based selection has been widely used in multiobjective optimization domain and also has been well studied both theoretically and experimentally [48]–[52]. In this paper, each normalized criterion is uniformly divided into N_{grid} parts each of which has a value range of $[(\max\{C_\lambda(j)'\} - \min\{C_\lambda(j)'\})/N_{\mathrm{grid}}]$, $\lambda = 1, 2, i \in M, j \in |\Omega_i|$. Thus, there are N_{grid}^2 grids in a 2-D space. In a certain grid, if there is only one nondominated element this element will be directly selected. If a grid is occupied by more than one nondominated element, only the one with the best $C_1{}'$ measurement is selected. By using the grid-based nondominance selection, only a small number (which can be controlled by N_{grid}) of the nondominated elements are selected to construct the collaborator set $\mathbf{CS}(i)$.

2) Minimize the Collaborator Size: For ith subcomponent, let *cIdx*(i) denote the index of the most frequently used element in $\mathbf{CS}(i)$, which is updated in each CMA-ES optimization cycle. More especially, when conducting *best-of-N* fitness evaluation for a given individual, the fitness is actually determined by the best collaborating element in $\mathbf{CS}(i)$. We record the index of the best collaborating element when evaluating the whole population $\mathbf{P}_{\mathrm{cma}}(i)$. Accordingly, the most frequent index *cIdx*(i) in the record refers to the element that is the most frequently used to determine the individuals' fitness.

Actually, it is not always necessary to use all elements in $\mathbf{CS}(i)$ to assemble solutions for fitness evaluations. For a given computational budget, a large number of collaborators is helpful only at the early stage of coevolution and this number should be dynamically adjusted along the coevolution process, as the empirical results in [53] indicated. Our pilot studies also show that when conducting the *best-of-N* fitness evaluations the index of the corresponding element tends to become gradually fixed as the coevolution proceeds. Based on these findings, we introduce a control parameter termed counting window N_{CW}. It is the number of cycles that are expected to be used to obtain a fixed collaborator index of a $\mathbf{CS}$. In other words, if the CMA-ES has evolved for N_{CW} cycles after the MMO procedure, only the *cIdx*(i)th element in $\mathbf{CS}(i)$ will be used for fitness evaluation afterwards. Therefore, in the rest cycles before another MMO procedure is retriggered, an individual is evaluated using only one objective evaluation, which can effectively save the computational resource.

E. Implementation of the MMO-CC

The pseudo-code of the MMO-CC is shown in Algorithm 1. In a particular cycle, the CMA-ES is considered to be getting stuck if it fails to improve its best-ever solution at a level more than 1%. If the number of cycles in which the CMA-ES has been successively got stuck is larger than *MaxStk*, a stagnation is detected [i.e., `detect_stagnation()`] and an MMO procedure will be triggered. Let *subSequence* denote the sequence in which the subcomponents are co-evolved. In this paper, the *subSequence* is not fixed. It is intuitive that the subcomponent with larger improvement may have the priority to be processed in the next cycle. Therefore, the *subSequence* is updated every cycle according to the deviation of $f(B(i))$ in every two consecutive cycles. N_{lc} denotes the number of cycles that a CMA-ES has evolved after an MMO procedure is performed.

In the initialization and decomposition stage, algorithmic parameters are initialized and the whole problem is decomposed into M subcomponents according to a certain grouping method (e.g., DG [9] or GDG [10]). The maximal number of fitness evaluations (*MaxFEs*) equals to total available fitness evaluations (*TotalFEs*) minus the consumed fitness evaluations for problem decomposition.

In the main loop, i.e., the optimization stage, collaborator set $\mathbf{CS}(i)$ is constructed at first for each subcomponent using `get_collab` procedure (see Section III-D). In a single coevolutionary cycle, each subcomponent is basically optimized by an independent `CMA-ES` optimizer in a round-robin fashion. If the coevolutionary process gets stuck, the `MMO` procedure is triggered to update $\mathbf{OPT}(i)$ in $\mathcal{A}(i)$. Note that $\mathcal{S}(i)$.*bestEver* of `CMA-ES` is used to guide `MMO` so that latest information about the global optimum obtained by `CMA-ES` could contribute to the `MMO` procedure. Meanwhile, a small part (controlled by parameter α) of $\mathbf{P}_{\mathrm{mmo}}(i)$ is randomly reinitialized to enhance diversity. On the other hand, if the best element (according to the underlying objective function) of $\mathbf{OPT}(i)$ obtained by the MMO procedure is better than $\mathcal{S}(i)$.*bestEver*, `CMA-ES` is reinitialized according to this element. Finally, if *nFEs* $>$ *MaxFEs*, the algorithm terminates and an optimization solution is outputted. In the proposed algorithm, fitness evaluations are conducted using *best-of-N* strategy. That is, an individual in $\mathbf{CP}(i)$ is combined with the collaborators in $\mathbf{CS}(i)$, the fitness values of the resulting complete solutions are then evaluated among which the best one is assigned to that individual.

As shown in [10], the complexity of CC algorithms comes from the optimizer for each subcomponent (denoted as s-dimensional subcomponent). In the proposed algorithm, CMA-ES is employed for search, whose complexity is $O(s^3)$. The basic CC framework for M subcomponents in the proposed algorithm has a complexity of $O(Ms^3)$, which equals to existing CC algorithms. The collaborator selection and MMO procedure which are novel in the proposed algorithm

Algorithm 1 Pseudo Code of the MMO-CC

INPUT: $\{\mathcal{G}, MaxFEs\}$

```
/* Initialization & decomposition stage*/
Decompose the problem into M subcomponents;
Calculate MaxStk according to Eq. 3;
bStk := false, nFEs =: 0;
subSequence := [1 : M], N_lc(i) := 0, i = 1, ..., M;
Randomly initiate P_mmo(i) and S(i) i = 1, ..., M;
Initialize each S(i), i = 1, ..., M;
B(i) := best{P_mmo(i) ∪ P_cma(i)};
/* Main loop: optimization stage */
while nFEs ≤ MaxFEs do
    for i = subSequence(j), j = 1, ..., M do
        CS(i) := get_collab (N_lc(i), A(i));
        if bStk = true then
            [OPT(i), ΔFEs] := MMO (S(i).bestEver, CS(i));
            if best(OPT(i)) is better than S(i).bestEver then
                Re-initialize CMA-ES using best(OPT(i));
            end if
            Randomly set the worst ⌈α * |P_mmo(i)|⌉ individuals in
            P_mmo(i);
            N_lc(i) := 0
        else
            [S(i), ΔFEs] := CMA-ES (S(i), CS(i));
            N_lc(i) := N_lc(i) + 1;
        end if
        B(i) := best{OPT(i) ⋃ S(i).bestEver};
        nFEs := nFEs + ΔFEs;
    end for
    Update subSequence according to the deviation of f(B(i)) in
    every two consecutive cycles;
    bStk := detect_stagnation();
end while
Output better{best(B), {B(i, G(i))}}, i ∈ M;
```

bring additional computational complexity. The collaborator selection is based on nondominated sorting for a bi-objective problem with M solutions, therefore, its complexity is $O(M)$ [54]. For each single MMO procedure carried out by NSGA-II, the complexity is $O(N^2)$ [45] for a population size N. In the worst case, the MMO procedure is triggered in the search of every subcomponent, leading to a complexity of $O(MN^2)$. In total, the complexity of the proposed algorithm becomes $\max\{O(Ms^3), O(M), O(MN^2)\}$ in the worst case, i.e., $\max\{O(Ms^3), O(MN^2)\}$. Generally, $N^2 \ll s^3$ holds for the proposed algorithm, thus, its complexity is estimated to be $O(Ms^3)$, which is at the same order of existing CC algorithms using CMA-ES as the optimizer.

IV. Effectiveness on Relative-Overgeneralization-Featured Problems

Relative overgeneralization (RO) was reported as a typical pathological behavior of CC algorithms [5] that may prevent conventional CC algorithms from converging to global optima. To validate the effectiveness of the proposed algorithm, we first test the MMO-CC on 2-D deceiving problems that are likely to result in RO to conventional CC algorithms. Note that the deceiving property is particularly for the CC approach, which might not be a problem for other EAs.

The following popular CC algorithms are used for comparison.

1) *Traditional CCEA (tCCEA) [55]:* tCCEA conducts the *best-of-N fitness* evaluation with K randomly chosen individuals.
2) *Biased CCEA (bCCEA) [56]:* The fitness of an individual is partly biased by a reward obtained when collaborating with the collaborators like tCCEA. The remaining part of the fitness is based on collaborating with the historical best collaborator.
3) *Complete CCEA (cCCEA):* cCCEA can be seen as an extreme case of the tCCEA. An individual has to access the whole population to conduct the *best-of-N* fitness evaluation.
4) *DECC:* DECC uses self-adaptive differential evolution with neighborhood search (SaNSDE) [57] as the subcomponent optimizer. It has been used as the optimizer of several successful CC algorithms, including DECC-G [6], DECC-DG [9] , DECC-ML [33] on LSO problems.
5) *Multipopulation Strategy Based CCEA (mCCEA) [14]:* mCCEA utilizes a dynamic multipopulation strategy in each co-evolutionary population to dynamically search multiple optima for information compensation.

A. RO-Featured Test Problems

Here we use a class of problems called the maximum of two quadratics (MTQ) that have been widely used to test the global optimization ability of CCEAs in [12], [13], and [56]. These problems include a global optimum and a local suboptimum, where the suboptimum covers a much wider range of the search space and is thus difficult for conventional CCEAs to escape from.

The joint reward function for the MTQ class is defined as follows [56]:

$$\mathrm{MTQ}(x, y) \leftarrow \max \begin{cases} H_1 * \left(1 - \frac{16*(x-X_1)^2}{S_1} - \frac{16*(y-Y_1)^2}{S_1}\right) \\ H_2 * \left(1 - \frac{16*(x-X_2)^2}{S_2} - \frac{16*(y-Y_2)^2}{S_2}\right) \end{cases} \quad (6)$$

where x and y may take values ranging between 0 and 1. H_1, H_2, X_1, Y_1, X_2, Y_2, S_1, and S_2 are parameters that affect the difficulty of the problem domain. More particularly, H_1 and H_2 affect the height of the two peaks; S_1 and S_2 affect the area that the two peaks cover: a higher value may result in a wider coverage of the specific peak, which makes the algorithm more likely to converge to this peak, even though it may be suboptimal; X_1, Y_1, X_2, and Y_2 characterize the location of the center of the two quadratics, which affects the relationship of the two peaks.

B. General Experimental Settings

Here we detail the general experimental settings used in the following experiments. tCCEA, bCCEA, cCCEA, and mCCEA are implemented using the GA toolbox[2] with the

[2]http://www-illigal.ge.uiuc.edu/~kumara

TABLE I
STATISTICAL RESULTS OBTAINED BY 50 INDEPENDENT RUNS ON 3 MAXIMIZATION MTQ FUNCTIONS WITH $H2 = 150$. p-VALUES INDICATE THE WILCOXON SIGNED RANK TEST BETWEEN MMO-CC AND THE OTHER ALGORITHMS

Algorithm	S2=1/32				S2=1/64				S2=1/128			
	Mean	Std	p-value	Conv. rate	Mean	Std	p-value	Conv. Rate	Mean	Std	p-value	Conv rate
tCCEA	60	30.3	9.96E-10	10%	60	30.3	1.63E-09	10%	52	14.14	1.09E-09	2%
cCCEA	89.76	75.14	8.53E-10	6%	58.87	82.86	7.56E-10	2%	56.85	76.17	7.56E-10	2%
bCCEA	39.92	69.23	7.56E-10	2%	34.4	59.3	7.56E-10	0%	35.48	46.96	7.56E-10	0%
DECC	126.27	38.89	7.55E-10	20%	114.31	39.38	9.62E-10	14%	98.08	44.78	5.65E-09	4%
mCCEA	121.73	40.66	1.50E-08	44%	100.48	44.94	3.78E-09	24%	83.9	43.35	2.23E-09	10%
MMO-CC	**149.97**	0.06	/	**100%**	**149.41**	1.63	/	**86%**	**147.86**	14.13	/	**96%**

default settings: the population size of each co-evolutionary population is set to 50, tournament selection size is 2, simulated-binary crossover and polynomial mutation rates are set to 0.9 and 0.1, respectively. For a fair comparison, parameter K in tCCEA and bCCEA is set to 3. The bias ratio δ of bCCEA is initialized to 1 and decreases linearly until it reaches to 0 at 75% of the total number of fitness evaluations. It then remains to be 0 until the end of the run.

We used the following parameter settings for the proposed MMO-CC: The population size of CMA-ES is $4 + \lfloor 3\log(|\mathcal{G}(i)|)\rfloor$ by default, $i \in M$. $|\mathbf{P}_{\text{mmo}}|$, NG_{mmo}, N_{CW}, N_{grid} and α are set to 50, 50, 5, 5, and 0.1, respectively.

We use the following parameter settings for MTQ problems in the first set of experiments: $H_1 = 50$, $X_1 = 0.75$, $Y_1 = 0.75$, $X_2 = 0.25$, $Y_2 = 0.25$, and $S_1 = 1.6$. In the second set of experiments, we set H_2 to 70, 150, 300 and set S_2 to 1/32, 1/64, 1/128, respectively, to examine the performance of MMO-CC on RO-featured problems with variant difficulty. All algorithms terminate when the number of fitness evaluations (FEs) reaches $TotalFEs = 1.5E4$. Most of the compared algorithms cannot conduct a sufficient search or adaptation to find the global optimum with such a limited number of FEs. For example, tCCEA and bCCEA can only evolve $1.5E4/(50 \times 2 \times 3) = 50$ generations. Therefore, to obtain satisfied performance, an algorithm is expected not only to conduct a global search but also to consume fewer FEs.

In order to investigate the dynamics of the proposed algorithm, population diversity of the main optimizer (i.e., the modified CMA-ES) is calculated and plotted based on $L1$ norm in the rest of this paper as follows [58]:

$$\text{diversity} = \frac{1}{n}\sum_{j=1}^{n}\left(\frac{1}{m}\sum_{i=1}^{m}\left|x_{ij} - \frac{1}{m}\sum_{k=1}^{m}x_{kj}\right|\right) \tag{7}$$

where m and n denote the population size and the number of decision variables, respectively. x_{ij} means the value of the jth decision variable of the ith individual.

C. Results and Analysis

The statistical results of 50 independent runs on 3 MTQ functions with $H2 = 150$ are given in Table I. More comparisons when $H2$ is set to 70, 150, 300, respectively, are listed in Table S-I in the supplementary material. It can be seen from these results that the proposed MMO-CC significantly outperforms all the compared algorithms according to the Wilcoxon signed rank test with a confidence level of 0.05. In addition, the MMO-CC can find the global optimum with a higher success rate. This indicates that the MMO procedure can provide CC algorithms with more informative collaborators to conduct information compensation. To better understand this, we further investigate the dynamics of the MMO-CC including the fitness, diversity and number of collaborators in each cycle. Fig. 1 presents the dynamics of a single run on an MTQ function ($H2 = 70$). As seen from the curve of fitness value, the MMO-CC converges to the suboptimum after a few cycles of coevolution. In this phase the two coevolutionary populations actually converge to the Nash equilibrium with the RO behavior. Benefiting from the collaborative information provided by the MMO procedure, the MMO-CC successfully breaks the Nash equilibrium (since the first MMO event) and converges to the global optimum. Note that the number of collaborators and diversity in each subcomponent temporarily increases after every MMO event takes place. This is beneficial to a CC algorithm to find the global optimum. In addition, the number of collaborators used in fitness evaluation can be adaptive adapted, which can not only provide sufficient collaborative information but also be helpful to reduce the number of FEs. Thus, the algorithm can evolve better without requiring additional computational resources.

Compared with DECC, the proposed MMO-CC substantially modifies the CC framework by exchanging one or more informative collaborators, resulting in a significant improvement of optimization performance. As for the mCCEA, it also focuses on modifying the canonical CC framework by dynamically discovering and maintaining multiple optima. Despite of this, it simply exchanges all the optima among subcomponents, which may lead to large consumption of computational resource. Comparing with the mCCEA, the proposed MMO-CC uses Pareto-dominance-based selection to reduce the number of collaborators while maintain sufficient information compensation. In addition, the subcomponent optimizer of MMO-CC is more efficient than that of mCCEA (simple GA). Both aspects help the MMO-CC achieve a better global optimization performance than that of the mCCEA.

V. PERFORMANCE ON LARGE-SCALE OPTIMIZATION PROBLEMS

In this section, we examine the scalability of the proposed MMO-CC. To this end, we conduct a comprehensive experimental study using the benchmark suite proposed in the competition on large-scale global optimization at CEC'2010, which contains 20 test functions that can be classified into four categories as shown in Table S-II in the supplementary materials. We first compare the MMO-CC with eleven state-of-the-art CC algorithms. Then six metaheuristic non-CC algorithms that have been shown to perform competitively on LSO problems are also compared using an online toolkit named MIDAS.[3]

Note that all experimental results in this section are obtained from 25 independent runs. In each run, an algorithm terminates when the number of FEs exceeds $TotalFEs = 3E6$. Note that in the following experiments the algorithmic parameters of the compared algorithms are the same as those suggested in the papers in which these algorithms were proposed. The settings of the MMO-CC are the same to that given in Section IV-B.

A. Comparison With the State-of-the-Art CC Algorithms

Here we conduct two sets of experiments according to different grouping techniques. First, we compare the MMO-CC with a CC algorithm with the DG technique. Then, we compare MMO-CC with CC algorithms which do not use DG. The compared CC algorithms are summarized in Table S-III in the supplementary material.

1) Comparison With CC Algorithms Using Differential Grouping: DG [9] has a rigorous theoretical background that guarantees the correctness of the detected interactions between decision variables. It has been empirically verified and represents the state-of-the-art decomposition method. Very recently, a new variant of DG, termed GDG [10] has been proposed. It has been empirically shown that GDG is able to detect the interactions between a variable and several independent variables, thereby further improving the decomposition accuracy.

In the following set of experiments, we compare MMO-CC with three DG-based CC algorithms, namely, DECC-DG [9], CBCC1-DG [59], and CBCC2-DG [59], and one GDG-based CC algorithm, i.e., CC-GDG-CMAES [10]. To further challenge the performance of the MMO-CC, ideal grouping (IG) is considered and the resultant DECC-IG [9] and CC-IG-CMAES [10] are also compared with MMO-CC.

Recall that CC algorithms using either DG of GDG adopt a two-stage problem solving procedure, in which a given problem is decomposed using DG or GDG before optimization. Then the grouping is fixed during the optimization. As mentioned in Section III-A, MMO-CC is also a two-stage CC algorithm. Unless otherwise stated, MMO-CC adopts DG to decompose a given problem in the following comparisons, and the control parameters of DG and GDG are set to $1E-6$ and $1E-10$, respectively.

Table II shows the statistical results obtained from 25 independent runs on CEC'2010 benchmark suite. As suggested in [60], the Wilcoxon rank test is used to statistically compare the performance between two algorithms. It can be seen that MMO-CC performs the best on 13 functions, followed by CC-IG-CMAES and CC-GDG-CMAES that perform the best on 10 and 9 functions, respectively. It is noteworthy that although the GDG method can obtain better decomposition accuracy, MMO-CC using the DG decomposition method can still significantly outperform CC-GDG-CMAES (10 versus 5), and CC-IG-CMAES (9 versus 5), where ideal decomposition is assumed.

With the same grouping method (i.e., DG), MMO-CC outperforms DECC-DG (16 versus 3), CBCC1-DG (16 versus 4) and CBCC2-DG (16 versus 3) at a higher level of significance. This can also be observed from the average fitness curves shown in Figs. S-2 and S-3 in the supplementary material, MMO-CC has better convergence performance than that of the other three algorithms. In addition, even with IG, DECC-IG is still outperformed by MMO-CC on most of the test functions (15 versus 5).

The above statistical comparisons confirm the effectiveness of MMO-based information compensation and Pareto-dominance/learning-based collaborator construction strategies. With effective information compensation, MMO-CC can evaluate the solutions of a subcomponent more accurately. On the other hand, the number of FEs for constructing collaborators is reduced by the Pareto-dominance/learning-based collaborator construction strategies and more computational resources could be used to search better solutions.

2) Comparison With CC Algorithms Using Nondifferential Grouping: Here we conduct another set of experiments where we compare MMO-CC with five CC algorithms, i.e., DECC-D [7], DECC-G [6], MLCC [30], CC-CMA-ES [36], and DECC-DML [7], that decompose a given problem with nondifferential method during coevolution.

The statistical results are shown in Table S-VII in the supplementary material. It can be seen that MMO-CC performs the best on most of the test functions (14 out of 20 functions). MMO-CC also shows clear superiority compared with each of the five CC algorithms, all of which adopt a powerful subcomponent optimizer such as SaNSDE or CMA-ES. MMO-CC significantly outperforms DECC-D (15 versus 3), DECC-G (15 versus 3), MLCC (17 versus 2), CC-CMA-ES (12 versus 2), and DECC-DML (15 versus 4) on the test functions.

Note, however, that MMO-CC is outperformed by CC-GDG-CMAES and CC-CMA-ES on 7th, 13th, and 18th functions. The main reason might be that DG performs fairly poorly (68%, 25.2%, and 17.3%, respectively) in decomposition on these test problems. This means that problem decomposition also plays a very important role in solving large scale problems using divide-and-conquer.

3) Performance Analysis: The overall performance comparison is given in Table S-VIII in the supplementary material in terms of the average Friedman ranking. The test functions are grouped into three categories: 1) fully separable ($F1-F3$); 2) partially separable ($F4-F18$); and 3) completely nonseparable ($F19-F20$) functions. Compared with other CC algorithms studied in this paper, MMO-CC obtains the best overall ranking result. Specifically, MMO-CC performs the best on fully separable functions and partially separable functions. This indicates that information compensation provided by the MMO procedure is beneficial for the problems with many nonseparable components. By contrast, completely nonseparable functions appear not to be able to benefit from the MMO procedure.

[3] http://vps128.cesvima.upm.es/

TABLE II
PERFORMANCE COMPARISONS AMONG CC ALGORITHMS USING DG OR GDG METHODS. STATISTICAL RESULTS ARE OBTAINED BY 25 INDEPENDENT RUNS ON CEC'2010 BENCHMARK SUITE ACCORDING TO WILCOXON SIGNED RANK TEST WITH A SIGNIFICANT LEVEL OF 0.05. THE BEST PERFORMING ALGORITHMS ARE MARKED IN BOLD. "NO. BEST" MEANS THE NUMBER OF FUNCTIONS ON WHICH MMO-CC PERFORMS THE BEST

Functions		DECC-IG	DECC-DG	CBCC-1	CBCC-2	CC-IG-CMAES	CC-GDG-CMAES	MMO-CC
F1	Mean	3.83E+05	2.08E+06	1.96E+06	6.38E+06	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std	6.51E+05	2.05E+06	1.99E+06	1.82E+07	0.00E+00	0.00E+00	0.00E+00
F2	Mean	4.39E+03	4.22E+03	4.33E+03	4.18E+03	1.60E+03	1.60E+03	**1.43E+03**
	Std	2.96E+02	3.80E+02	3.04E+02	5.38E+02	5.29E+01	5.29E+01	8.43E+01
F3	Mean	1.10E+01	1.09E+01	1.12E+01	1.10E+01	**0.00E+00**	2.17E+01	**0.00E+00**
	Std	6.23E-01	8.53E-01	8.96E-01	7.32E-01	0.00E+00	1.06E-02	0.00E+00
F4	Mean	2.71E+10	5.06E+11	1.81E+11	1.65E+10	1.60E+10	1.60E+10	**7.64E+06**
	Std	1.24E+10	1.96E+11	1.08E+11	3.62E+09	1.25E+10	1.25E+10	1.31E+06
F5	Mean	6.86E+07	7.36E+07	7.02E+07	**6.43E+07**	1.02E+08	1.02E+08	3.34E+08
	Std	1.24E+07	9.56E+06	1.05E+07	1.31E+07	1.32E+07	1.32E+07	1.54E+08
F6	Mean	1.63E+01	1.58E+01	8.14E+04	4.11E+04	6.03E+06	6.03E+06	**5.77E-01**
	Std	9.69E-01	7.30E-01	2.84E+05	2.05E+05	9.88E+06	9.88E+06	1.32E+00
F7	Mean	1.17E+04	2.79E+04	1.23E+05	1.26E+10	**0.00E+00**	**0.00E+00**	2.41E+10
	Std	3.96E+03	2.03E+04	1.09E+05	1.48E+10	0.00E+00	0.00E+00	6.26E+09
F8	Mean	8.06E+05	2.78E+07	**7.50E+06**	3.72E+07	2.90E+07	2.90E+07	2.63E+08
	Std	1.63E+06	3.19E+07	1.84E+07	3.47E+07	2.59E+07	2.59E+07	5.29E+08
F9	Mean	4.76E+07	3.65E+07	1.02E+07	3.40E+08	1.74E+03	1.74E+03	**8.99E+01**
	Std	5.30E+07	1.49E+07	3.84E+06	2.67E+08	6.95E+02	6.95E+02	4.64E+01
F10	Mean	3.13E+03	3.33E+03	2.59E+03	4.90E+03	1.81E+03	1.81E+03	**1.63E+03**
	Std	1.68E+02	1.92E+02	1.48E+02	6.37E+02	8.89E+01	8.89E+01	9.10E+01
F11	Mean	2.51E+01	2.64E+01	2.69E+01	2.75E+01	5.07E+01	6.53E+01	**2.99E+00**
	Std	2.72E+00	2.95E+00	2.64E+00	3.18E+00	2.04E+01	2.82E+01	3.97E+00
F12	Mean	2.44E+04	3.21E+04	3.53E+04	5.07E+04	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std	7.12E+03	1.06E+04	1.11E+04	1.10E+04	0.00E+00	0.00E+00	0.00E+00
F13	Mean	1.29E+04	2.89E+07	9.06E+04	1.29E+07	**2.72E+02**	**2.72E+02**	3.05E+04
	Std	4.34E+03	1.57E+07	6.11E+04	7.36E+06	1.77E+02	1.77E+02	9.43E+04
F14	Mean	2.14E+07	2.10E+07	2.24E+07	5.35E+09	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std	2.06E+06	2.25E+06	2.27E+06	6.00E+08	0.00E+00	0.00E+00	0.00E+00
F15	Mean	2.84E+03	2.88E+03	2.84E+03	3.22E+03	**2.00E+03**	**2.00E+03**	**2.05E+03**
	Std	1.86E+02	2.76E+02	2.65E+02	4.17E+02	6.74E+01	6.74E+01	9.39E+01
F16	Mean	1.93E+01	1.97E+01	1.87E+01	1.91E+01	9.67E+01	9.67E+01	**8.87E+00**
	Std	3.77E+00	3.61E+00	3.83E+00	2.76E+00	3.78E+01	3.78E+01	9.25E+00
F17	Mean	7.08E+00	7.76E+00	1.49E+01	1.24E+02	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std	1.76E+00	1.89E+00	7.01E+00	5.72E+01	0.00E+00	0.00E+00	0.00E+00
F18	Mean	1.15E+03	2.01E+10	4.10E+09	1.23E+11	**8.63E+01**	**8.63E+01**	3.37E+04
	Std	1.65E+02	4.82E+09	1.83E+09	1.45E+10	8.76E+01	8.76E+01	2.75E+04
F19	Mean	**8.95E+05**	**9.01E+05**	**9.12E+05**	**9.11E+05**	2.87E+06	2.87E+06	1.54E+07
	Std	6.24E+04	6.14E+04	7.11E+04	6.02E+04	6.61E+05	6.61E+05	1.59E+06
F20	Mean	1.67E+07	6.53E+08	1.41E+07	6.97E+09	**8.54E+02**	**8.54E+02**	1.10E+03
	Std	3.30E+07	6.71E+08	1.96E+07	1.12E+09	6.71E+01	6.71E+01	1.51E+02
No. Best		1	1	2	2	10	9	13
MMO-CC vs.		15 vs 5	16 vs 3	16 vs 4	16 vs 3	9 vs 5	10 vs 5	/

To better understand the behavior of the proposed MMO-CC, the dynamics of fitness, diversity and the number of collaborators are investigated on the selected functions from different categories of the test functions. Taking $F11$ for instance, the population diversity of the main optimizer (i.e., the CMA-ES), the best-so-far fitness and the total collaborator number of all subcomponents are given at each co-evolutionary cycle in Fig. 3. Note that, the population diversity of each subcomponent is added by $N_{\mathrm{ID}} - 1$ (N_{ID} denotes the component ID for clarity). As shown in Fig. 3, the solution's fitness has improved dramatically in the first one hundred iterations, after that, the optimization procedure gradually stagnates and an MMO procedure is triggered at about 380th cycle. Such dynamics can also be explained by the change of diversity, which in most cases descends to a near-zero level within the first one hundred iterations. After that, the co-evolutionary populations lose the exploration ability and the optimization process stagnates. When an MMO procedure is triggered, diversity of the populations is promoted again because some of the CMA-ES optimizers are heuristically reinitialized by their corresponding MMO procedures. Moreover, for a short period (N_{CW} iterations) after the MMO procedure, the number of collaborators of each subcomponent temporarily increases (see the bottom subfigure). The increase of the diversity and collaborator number is helpful to reactivate the optimization procedure. Similar behaviors occur several times, which constantly drives the optimization process toward the global optimum.

In addition to $F11$, functions $F3$, $F6$, $F16$, and $F19$ are also selected for investigation, and the results are shown in Fig. S-4 in the supplementary material. Similar behaviors can be observed on another two partially separable functions ($F6$ and $F16$). However, when solving nonseparable function $F19$, all decision variables are grouped into a single

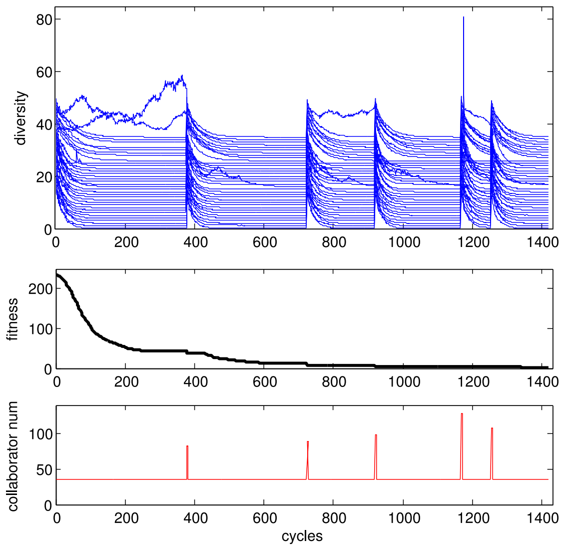

Fig. 3. Dynamics of the MMO-CC on $F11$.

component and a noncoevolutionary optimization is actually conducted. Therefore, no significant contribution of MMO is observed although it is constantly triggered. Besides, as for the fully separable function $F3$, the problem is decomposed into 50 subcomponents without information loss. Therefore, the conventional CC procedure with the CMA-ES optimizers is capable of obtaining a global optimum. The MMO procedures triggered in the later period are not helpful.

B. Comparison With State-of-the-Art Metaheuristic Non-CC Algorithms

Many non-CC metaheuristic algorithms are recognized as effective approaches for solving large scale optimization problems. Actually, as documented in the summary reports of the competitions on large-scale global optimization at CEC'2010, CEC'2012, CEC'2013, and CEC'2015, the winner has been a non-CC metaheuristic algorithm. For this reason, we also compare MMO-CC with six metaheuristic non-CC algorithms: 2S-Ensemble [25], GaDE [61], jDElscop [62], MA-SW-Chains [31], MOS-CEC2012 [63], and MOS-CEC2013 [22]. The descriptions of these compared algorithms are summarized in Table S-IV in the supplementary material. Note that DECC-G is used for comparison as a baseline algorithm.

We conduct the comparisons using an online toolkit MIDAS, which is repository of some state-of-the-art algorithms for large-scale global optimization [64]. Table III shows the overall comparison results according to the Friedman average ranking and Formula 1 ranking, respectively. It can be seen that MMO-CC performs the best, followed by MOS-CEC2013 and MOS-CEC2012, according to the Formula 1 ranking results. On the other hand, MOS-CEC2013 wins the first place, followed by MMO-CC and MA-SW-Chains according to Friedman ranking results. The significance of the performance differences of the compared algorithms is given in Table S-V (MMO-CC is the control algorithm) and Table S-VI (MOS-CEC2013 is the control algorithm) in the supplementary material in terms of Family-Wise-Error (FWER) at a

TABLE III
AVERAGE FRIEDMAN AND FORMULA 1 RANKINGS OF THE COMPARED ALGORITHMS

Algorithm Name	Friedman ranking	Formula 1 ranking
2S-Ensemble	4.25	249
DECC-G	6.65	138
GaDE	6.08	156
jDElscop	4.35	234
MA-SW-Chains	3.98	254
MMO-CC	3.65	**325**
MOS-CEC2012	4.13	281
MOS-CEC2013	**2.93**	323

An algorithm performs better when its Friedman ranking value is smaller and Formula 1 ranking value is larger. The best performing algorithm is marked in bold.

significant level of 0.05. From both tables, we can see that only GaDE and DECC-G perform significantly worse in the comparisons.

To further compare the performance on the fully separable, partially separable, and completely nonseparable functions, we investigate the average Friedman rankings (see Table S-IX in the supplementary material) and the proportion of the functions for which each algorithm obtains the best results in terms of the mean error (see Fig. S-5 in the supplementary material). Similar to the previous comparison with the other CC algorithms, MMO-CC performs poorly on nonseparable functions as well. It obtains the second worst average Friedman ranking and never performs the best on the nonseparable functions. On the partially separable functions, MMO-CC's average Friedman ranking is the second best, which is close to that of the MOS-CEC-2013.

Note that although MMO-CC obtains the best results on the largest proportion (45%, followed by MOS-CEC2012 and MOS-CEC2013 with a percentage of 20% and 15%, respectively, as shown in Fig. S-5 in the supplementary material) of the whole benchmark suite, it only achieves the second best average Friedman ranking. This may be attributed to the following reasons.

First, although MMO-CC obtains the best results on the largest proportion, it performs poorly on $F7$, $F8$, $F13$, $F18$, and $F19$. The ranking values of MMO-CC for those functions are 8, 7, 8, 8, and 8, respectively. This leads to substantial increase (deterioration) of the MMO-CCs rankings and therefore MMO-CC has only achieved the second best place according to the average Friedman ranking. Similarly, in the context of Formula 1 ranking, MMO-CC only obtains 1 or 2 points on those five functions while any best-performed algorithm will obtain 15 points. This may dramatically degrade the overall performance of MMO-CC and therefore MMO-CC has only 2 points superiority over the compared algorithms.

Second, CMA-ES is well-known for its invariance to transformations of the search space including rotation, reflection, and translation, to which the advantage of MMO-CC over the compared metaheuristic algorithms on the rotated functions could be attributed. However, the advantage of CMA-ES on nonrotated functions is not as significant as that on rotated functions, which may explain the relatively poor performance of MMO-CC on $F7$, $F8$, $F13$, $F18$, and $F19$.

In addition, the difficulty in solving nonrotated functions may further degrade the quality of the collaborators.

As mentioned in Section III, the quality of the solutions includes diversity and fitness. While diversity is maintained by the MMO procedure, the fittest collaborator mainly depends on the main optimizer. In particular, for $F7$, $F8$, $F13$, and $F18$ the property of the main optimizer weakens the ability of obtaining high quality fittest collaborator. Therefore the performance of the MMO-CC is poorly ranked when compared with non-CC metaheuristic algorithms. As for F19, the MMO procedure is not meaningful because there is only a single component and no cooperative coevolution takes place. It is rather difficult for CMA-ES to find the global optimum in a 1000-D nonrotated landscape.

Note that the above analysis does not mean that information compensation of MMO-CC fails to work on nonrotated functions. It is simply insufficient. For example, for F13 and F18, there are 10 and 20 50-D nonseparable subcomponents, respectively (see Table S-II in the supplementary material). The corresponding grouping accuracy for them is only 25.2% and 17.3%, respectively, as reported in [9]. The information compensation helps MMO-CC significantly outperform DECC-DG [9] and its variants CBCC-1 and CBCC-2 (see Table II) with the same grouping method, but it turns out to be insufficient when compared with the well-designed non-CC metaheuristic algorithms (decomposition is not needed), such as MOS-CEC2013 and MA-SW-Chains. As for $F7$ and $F8$, there is only one group of nonseparable variables (50-D). The potential benefit of information compensation is relatively limited in solving $F7$ and $F8$ compared with in solving $F13$ and $F18$. Thus, MMO-CC is outperformed by both DECC-DG and non-CC metaheuristic algorithms on these two functions. This also implies that the information compensation is more beneficial for the functions with many nonseparable (but not all) subcomponents which are more general in the real-world applications.

In summary, MMO-CC may perform poorly on the functions with both inaccurate grouping and nonrotated property. Although MMO-CC perform poorly on $F7$, $F8$, $F13$, $F18$, and $F19$, it outperforms all compared algorithms studied in this paper on most of the remaining functions, especially on $F1$, $F3$, $F4$, $F6$, $F9$, $F11$, $F12$, $F14$, $F16$, and $F17$.

C. Sensitivity Analysis of Parameter Settings

The algorithmic parameters of the MMO-CC can be classified into two categories. First, the parameters involved in the existing algorithms (i.e., CMA-ES and NSGA-II). Second, some new parameters introduced along with the NMO procedures and collaborator set construction. Since the main contribution of the proposed MMO-CC is to identify and utilize informative collaborators with a reasonable computational budget in the context of LSO, we focus on the parameters in the second category and carry out sensitivity analysis in the context of LSO.

There are three parameters involved in the MMO procedure, namely, $|P_{\text{mmo}}|$, NG_{mmo}, and α, and two parameters in collaborator set construction, i.e., N_{CW} and N_{grid}. To analyze the sensitivity, each parameter will be set to 2 more values in addition to the default parameter settings given in Section IV-B. Note that when examining the sensitivity of a certain parameter, the other four parameters are set to their corresponding default values. Considering the fact that the information compensation introduced by the MMO and some other related techniques is expected to be more beneficial on the problems containing several nonseparable subcomponents, we do the analysis on functions $F9 \sim F13$ (10 50-D-nonseparable groups and a 500-D-separable group) and $F10 \sim F18$ (20 50-D-nonseparable groups) in the CEC'2010 LSO benchmark suite.

The mean and standard deviation over 25 independent runs are plotted in Figs. S-6 and S-7 in the supplementary material. On a certain function, when analyzing the influence of assigning three typical values to a certain parameter, we can calculate the percentage enhancement from the smallest mean value to the largest one. Thus, only 7 out of 50 cases report a percentage increase larger than 100%. In 37 out of 50 cases, such percentage is smaller than 50%.

VI. Conclusion

The divide-and-conquer approach adopted by CC algorithms offers a promising means to solve LSO problems. To make full use of the strength of CC algorithms, two major issues, i.e., problem decomposition and CC optimization procedure, must be addressed. In contrast to problem decomposition, the study of CC optimization procedure in terms of LSO has not been well studied. However, we argue that development of new CC framework is of great importance since the canonical CC framework has been shown to be not able to guarantee global optimization. For a CC framework to be able to find a global optimum using the divide-and-conquer approach, it is essential to compensate the information lost in problem decomposition. Although some methods have been proposed for information compensation in the canonical CC framework, they are not easily scalable to high-dimensional problems.

In this paper, a multimodel optimization approach is proposed to achieve more effective information compensation in the CC framework. To this end, a bi-objective-based MMO is incorporated into the optimization procedure of subcomponents, where both the original objective function and the diversity are simultaneously optimized to locate the multiple optima in each subcomponent. The obtained local or global optima are then used as representatives for information exchange between subcomponents, thereby achieving more effective information compensation. In addition to the enhanced information compensation between the subcomponents, a modified CMA-ES has been incorporated as the subcomponent optimizer whose status is used to adaptively trigger the bi-objective-based MMO. To tackle the issue of exponential increase in the number of complete solutions due to combinatorial explosion, a nondominance-based selection scheme is introduced to reduce the number of complete solutions, which is further reduced to one. In this way, large amount of computational cost for evaluating the complete solutions can be saved.

To verify the effectiveness of the proposed algorithm, we have compared the proposed MMO-CC with five representative CC algorithms on a set of 2-D RO-featured problems. A significant improvement in optimization performance has been observed on all problems used in the study. Furthermore, a comprehensive comparison has been conducted on the benchmark suite of the CEC'2010 competition on large-scale global optimization. We have compared MMO-CC with eleven state-of-the-art CC algorithms, which decompose a problem either before or during the coevolution. Finally, six state-of-the-art metaheuristic algorithms that perform optimization without decomposition have also been compared with MMO-CC. Statistical results in all these comparisons show that MMO-CC outperforms the compared algorithms on a majority of the tested instances.

Much work remains to be investigated in the future. For example, it is of great importance to incorporate more effective decomposition methods into the MMO-CC framework to conduct cooperative coevolution on problems that are more difficult to be decomposed. Meanwhile, it is also of interest to develop self-adaptive parameter turning strategies. Finally, other population-based optimization algorithms will be investigated to develop more effective main optimizers for LSO.

REFERENCES

[1] B. Akay and D. Karaboga, "Artificial bee colony algorithm for large-scale problems and engineering design optimization," *J. Intell. Manuf.*, vol. 23, no. 4, pp. 1001–1014, 2012.

[2] J. Liu, Y. Chi, and C. Zhu, "A dynamic multiagent genetic algorithm for gene regulatory network reconstruction based on fuzzy cognitive maps," *IEEE Trans. Fuzzy Syst.*, vol. 24, no. 2, pp. 419–431, Apr. 2016.

[3] Y. Mei, X. Li, and X. Yao, "Cooperative coevolution with route distance grouping for large-scale capacitated arc routing problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 435–449, Jun. 2014.

[4] M. N. Omidvar, X. Li, and K. Tang, "Designing benchmark problems for large-scale continuous optimization," *Inf. Sci.*, vol. 316, pp. 419–436, Sep. 2015.

[5] R. P. Wiegand, "An analysis of cooperative coevolutionary algorithms," Ph.D. dissertation, Dept. Comput. Sci., George Mason Univ., Fairfax, VA, USA, 2003.

[6] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Inf. Sci.*, vol. 178, no. 15, pp. 2985–2999, 2008.

[7] M. N. Omidvar, X. Li, and X. Yao, "Cooperative co-evolution with delta grouping for large scale non-separable function optimization," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, 2010, pp. 1–8.

[8] W. Chen, T. Weise, Z. Yang, and K. Tang, "Large-scale global optimization using cooperative coevolution with variable interaction learning," in *Parallel Problem Solving From Nature PPSN XI*. Heidelberg, Germany: Springer, 2010, pp. 300–309.

[9] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 378–393, Jun. 2014.

[10] Y. Mei, M. N. Omidvar, X. Li, and X. Yao, "A competitive divide-and-conquer algorithm for unconstrained large-scale black-box optimization," *ACM Trans. Math. Softw.*, vol. 42, no. 2, pp. 1–24, 2016.

[11] 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," Nat. Inspired Comput. Appl. Lab., Hefei, China, Rep., 2009.

[12] L. Panait, "Theoretical convergence guarantees for cooperative coevolutionary algorithms," *Evol. Comput.*, vol. 18, no. 4, pp. 581–615, 2010.

[13] L. Panait, S. Luke, and J. F. Harrison, "Archive-based cooperative coevolutionary algorithms," in *Proc. Genet. Evol. Comput. Conf.*, Seattle, WA, USA, 2006, pp. 345–352.

[14] X. Peng, K. Liu, and Y. Jin, "A dynamic optimization approach to the design of cooperative co-evolutionary algorithms," *Knowl. Based Syst.*, vol. 109, no. 2, pp. 174–186, Oct. 2016.

[15] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, *On the Surprising Behavior of Distance Metrics in High Dimensional Space*. Heidelberg, Germany: Springer, 2001.

[16] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *Proc. IEEE Int. Conf. Evol. Comput.*, Nagoya, Japan, 1996, pp. 312–317.

[17] F. J. Solis and R. J.-B. Wets, "Minimization by random search techniques," *Math. Oper. Res.*, vol. 6, no. 1, pp. 19–30, 1981.

[18] D. Molina, M. Lozano, A. M. Sánchez, and F. Herrera, "Memetic algorithms based on local search chains for large scale continuous optimisation problems: MA-SSW-chains," *Soft Comput.*, vol. 15, no. 11, pp. 2201–2220, 2011.

[19] D. Molina, M. Lozano, C. García-Martínez, and F. Herrera, "Memetic algorithms for continuous optimisation based on local search chains," *Evol. Comput.*, vol. 18, no. 1, pp. 27–63, Mar. 2010.

[20] L.-Y. Tseng and C. Chen, "Multiple trajectory search for large scale global optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2008, pp. 3052–3059.

[21] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Comput. J.*, vol. 7, no. 4, pp. 308–313, 1965.

[22] A. LaTorre, S. Muelas, and J.-M. Peña, "Large scale global optimization: Experimental results with MOS-based hybrid algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Cancún, Mexico, 2013, pp. 2742–2749.

[23] A. LaTorre de la Fuente, "A framework for hybrid dynamic evolutionary algorithms: Multiple offspring sampling (MOS)," Ph.D. dissertation, Facultad de Informática, Universidad Politécnica de Madrid, Madrid, Spain, 2009.

[24] S.-Z. Zhao, P. N. Suganthan, and S. Das, "Dynamic multi-swarm particle swarm optimizer with sub-regional harmony search," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, 2010, pp. 1–8.

[25] Y. Wang and B. Li, "Two-stage based ensemble optimization for large-scale global optimization," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, 2010, pp. 1–8.

[26] A. Puris, R. Bello, D. Molina, and F. Herrera, "Variable mesh optimization for continuous optimization problems," *Soft Comput.*, vol. 16, no. 3, pp. 511–525, 2012.

[27] R. Cheng and Y. Jin, "A competitive swarm optimizer for large scale optimization," *IEEE Trans. Cybern.*, vol. 45, no. 2, pp. 191–204, Feb. 2015.

[28] S. Mahdavi, M. E. Shiri, and S. Rahnamayan, "Metaheuristics in large-scale global continues optimization: A survey," *Inf. Sci.*, vol. 295, pp. 407–428, Feb. 2015.

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

[30] Z. Yang, K. Tang, and X. Yao, "Multilevel cooperative coevolution for large scale optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2008, pp. 1663–1670.

[31] D. Molina, M. Lozano, and F. Herrera, "MA-SW-Chains: Memetic algorithm based on local search chains for large scale continuous global optimization," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, 2010, pp. 1–8.

[32] T. R. Browning, "Applying the design structure matrix to system decomposition and integration problems: A review and new directions," *IEEE Trans. Eng. Manag.*, vol. 48, no. 3, pp. 292–306, Aug. 2001.

[33] 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 Congr. Evol. Comput.*, Barcelona, Spain, 2010, pp. 1–8.

[34] X. Li and X. Yao, "Cooperatively coevolving particle swarms for large scale optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 2, pp. 210–224, Apr. 2012.

[35] A. Kabán, J. Bootkrajang, and R. J. Durrant, "Toward large-scale continuous EDA: A random matrix theory perspective," *Evol. Comput.*, vol. 24, no. 2, pp. 255–291, 2016.

[36] J. Liu and K. Tang, "Scaling up covariance matrix adaptation evolution strategy using cooperative coevolution," in *Intelligent Data Engineering and Automated Learning–IDEAL 2013*. Heidelberg, Germany: Springer, 2013, pp. 350–357.

[37] P. García-Sánchez, J. Ortega, J. González, P. A. Castillo, and J. J. Merelo, "Addressing high dimensional multi-objective optimization problems by coevolutionary islands with overlapping search spaces," in *Proc. Eur. Conf. Appl. Evol. Comput.*, Porto, Portugal, 2016, pp. 107–117.

[38] B. Dorronsoro, G. Danoy, A. J. Nebro, and P. Bouvry, "Achieving super-linear performance in parallel multi-objective evolutionary algorithms by means of cooperative coevolution," *Comput. Oper. Res.*, vol. 40, no. 6, pp. 1552–1563, 2013.

[39] X. Ma *et al.*, "A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 275–298, Apr. 2016.

[40] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 97–112, Feb. 2018.

[41] J. Yao, N. Kharma, and P. Grogono, "Bi-objective multipopulation genetic algorithm for multimodal function optimization," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 80–102, Feb. 2010.

[42] K. Deb and A. Saha, "Multimodal optimization using a bi-objective evolutionary algorithm," *Evol. Comput.*, vol. 20, no. 1, pp. 27–62, Mar. 2012.

[43] A. Basak, S. Das, and K. C. Tan, "Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 666–685, Oct. 2013.

[44] Y. Wang, H.-X. Li, G. G. Yen, and W. Song, "MOMMOP: Multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems," *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 830–843, Apr. 2015.

[45] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[46] O. Schütze, S. Mostaghim, M. Dellnitz, and J. Teich, "Covering Pareto sets by multilevel evolutionary subdivision techniques," in *Evolutionary Multi-Criterion Optimization*. Heidelberg, Germany: Springer, 2003, pp. 118–132.

[47] S. Mostaghim and J. Teich, "Covering Pareto-optimal fronts by subswarms in multi-objective particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, vol. 2. Portland, OR, USA, 2004, pp. 1404–1411.

[48] K. Deb, M. Mohan, and S. Mishra, "Evaluating the ϵ domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions," *Evol. Comput.*, vol. 13, no. 4, pp. 501–525, 2005.

[49] J. Knowles and D. Corne, "Properties of an adaptive archiving algorithm for storing nondominated vectors," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 100–116, Apr. 2003.

[50] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evol. Comput.*, vol. 10, no. 3, pp. 263–282, Sep. 2002.

[51] L. Rachmawati and D. Srinivasan, "Dynamic resizing for grid-based archiving in evolutionary multi objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, Singapore, 2007, pp. 3975–3982.

[52] S. Yang, M. Li, X. Liu, and J. Zheng, "A grid-based evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 721–736, Oct. 2013.

[53] L. Panait and S. Luke, "Time-dependent collaboration schemes for cooperative coevolutionary algorithms," in *Proc. AAAI Fall Symp. Coevolutionary Coadaptive Syst.*, 2005, pp. 1–8.

[54] H. Wang and X. Yao, "Corner sort for Pareto-based many-objective optimization," *IEEE Trans. Cybern.*, vol. 44, no. 1, pp. 92–102, Jan. 2014.

[55] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *Parallel Problem Solving From Nature PPSN III*. Heidelberg, Germany: Springer, 1994, pp. 249–257.

[56] L. Panait, S. Luke, and R. P. Wiegand, "Biasing coevolutionary search for optimal multiagent behaviors," *IEEE Trans. Evol. Comput.*, vol. 10, no. 6, pp. 629–645, Dec. 2006.

[57] Z. Yang, K. Tang, and X. Yao, "Self-adaptive differential evolution with neighborhood search," in *Proc. IEEE Congr. Evol. Comput.*, 2008, pp. 1110–1116.

[58] S. Cheng and Y. Shi, "Diversity control in particle swarm optimization," in *Proc. IEEE Symp. Swarm Intell. (SIS)*, Paris, France, 2011, pp. 1–9.

[59] M. N. Omidvar, X. Li, and X. Yao, "Smart use of computational resources based on contribution for cooperative co-evolutionary algorithms," in *Proc. 13th Annu. Conf. Genet. Evol. Comput.*, Dublin, Ireland, 2015, pp. 1115–1122.

[60] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, 2011.

[61] Z. Yang, K. Tang, and X. Yao, "Scalability of generalized adaptive differential evolution for large-scale continuous optimization," *Soft Comput.*, vol. 15, no. 11, pp. 2141–2155, 2011.

[62] J. Brest and M. S. Maučec, "Self-adaptive differential evolution algorithm using population size reduction and three strategies," *Soft Comput.*, vol. 15, no. 11, pp. 2157–2174, 2011.

[63] A. LaTorre, S. Muelas, and J.-M. Peña, "Multiple offspring sampling in large scale global optimization," in *Proc. IEEE Congr. Evol. Comput.*, Brisbane, QLD, Australia, 2012, pp. 1–8.

[64] A. LaTorre, S. Muelas, and J.-M. Peña, "A comprehensive comparison of large scale global optimizers," *Inf. Sci.*, vol. 316, pp. 517–549, Sep. 2015.

Xingguang Peng (M'11) received the B.Eng., M.Eng., and Ph.D. degrees from Northwestern Polytechnical University, Xi'an, China, in 2003, 2006, and 2009, respectively.

He is currently an Associate Professor with the School of Marine Science and Technology, Northwestern Polytechnical University. His current research interests include nature-inspired computation, large-scale optimization, dynamic optimization, swarm robotics, unmanned underwater vehicles, and real-world problems.

Dr. Peng is a member of the Task Force on Data-Driven Evolutionary Optimization of Expensive Problems Technical Committee of IEEE Computational Intelligence Society.

Yaochu Jin (M'98–SM'02–F'16) received the B.Sc., M.Sc., and Ph.D. degrees from Zhejiang University, Hangzhou, China, in 1988, 1991, and 1996, respectively, and the Dr.-Ing. degree from Ruhr University Bochum, Bochum, Germany, in 2001.

He is a Professor in computational intelligence with the Department of Computer Science, University of Surrey, Guildford, U.K., where he heads the Nature Inspired Computing and Engineering Group. He is also a Finland Distinguished Professor funded by the Finnish Agency for Innovation (Tekes) and a Changjiang Distinguished Visiting Professor appointed by the Ministry of Education, China. His current research is funded by EC FP7, U.K. EPSRC and industry. His current research interests include interdisciplinary areas that bridge the gap between computational intelligence, computational neuroscience, computational systems biology, nature-inspired, and real-world driven problem-solving. He has (co)-authored over 200 peer-reviewed journal and conference papers and been granted eight patents on evolutionary optimization.

Dr. Jin was a recipient of the 2014 and 2016 IEEE Computational Intelligence Magazine Outstanding Paper Award, the 2017 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award, and the Best Paper Award of the 2010 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology. He is the Editor-in-Chief of the IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS and *Complex and Intelligent Systems*. He is also an Associate Editor or Editorial Board Member of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON CYBERNETICS, the IEEE TRANSACTIONS ON NANOBIOSCIENCE, *Evolutionary Computation*, *BioSystems*, *Soft Computing*, and *Natural Computing*. He has delivered over 25 invited keynote speeches at international conferences. He has been an IEEE Distinguished Lecturer since 2017 and was the Vice President for Technical Activities of the IEEE Computational Intelligence Society from 2014 to 2015.

Handing Wang (S'10–M'16) received the B.Eng. and Ph.D. degrees from Xidian University, Xi'an, China, in 2010 and 2015, respectively.

She is currently a Research Follow with the Department of Computer Science, University of Surrey, Guildford, U.K. Her current research interests include nature-inspired computation, multiobjective optimization, multiple criteria decision making, surrogate-assisted evolutionary optimization, and real-world problems.

Dr. Wang is an Associate Editor of *Complex and Intelligent Systems*, the Chair of the Task Force on Intelligence Systems for Health within the Intelligent Systems Applications Technical Committee of IEEE Computational Intelligence Society.

Supplementary materials

Multi-Modal Optimization Enhanced Cooperative Coevolution for Large-Scale Optimization

Xingguang Peng, *Member, IEEE,* Yaochu Jin, *Fellow, IEEE,* Handing Wang, *Member, IEEE,*

TABLE S- I
STATISTICAL RESULTS OBTAINED BY 50 INDEPENDENT RUNS ON 9 MAXIMIZATION MTQ FUNCTIONS WITH VARIANT DIFFICULTY. EACH ALGORITHM TERMINATES WHEN FITNESS EVALUATION NUMBER IS LARGER THAN 1.5E4. p-VALUES INDICATE THE WILCOXON SIGNED RANK TEST BETWEEN MMO-CC AND THE OTHER ALGORITHMS. AN ALGORITHM GLOBALLY CONVERGES IF THE DEVIATION BETWEEN ITS OUTPUT SOLUTION'S VALUE AND THE GLOBAL OPTIMUM IS SMALLER THAN 1.

H2	S2	Statistics	tCCEA	cCCEA	bCCEA	CCDE	mCCEA	MMO-CC
70	1/32	Mean	51.6	35.66	32.2	60.51	63.65	69.98
		Std	5.48	54.85	47.38	8.91	8.01	0.1
		p-value	1.03E-09	7.56E-10	7.56E-10	7.47E-10	7.56E-10	/
		Conv.rate %	8.00	16.00	4.00	30.00	48.00	100.00
	1/64	Mean	51.2	40.89	38.57	57.56	59.26	69.99
		Std	4.8	44.15	38.25	8.49	8.49	0.03
		p-value	1.06E-09	7.56E-10	7.56E-10	7.44E-10	7.56E-10	/
		Conv.rate %	6.00	8.00	4.00	16.00	18.00	100.00
	1/128	Mean	50.8	21.87	37.07	54.93	56.25	69.99
		Std	3.96	54.8	38.11	7.41	8.3	0.07
		p-value	6.79E-10	7.56E-10	7.56E-10	7.34E-10	7.56E-10	/
		Conv.rate %	4.00	2.00	0.00	4.00	16.00	100.00
150	1/32	Mean	60	89.76	39.92	126.27	121.73	149.97
		Std	30.3	75.14	69.23	38.89	40.66	0.06
		p-value	9.96E-10	8.53E-10	7.56E-10	7.55E-10	1.50E-08	/
		Conv.rate %	10.00	6.00	2.00	20.00	44.00	100.00
	1/64	Mean	60	58.87	34.4	114.31	100.48	149.41
		Std	30.3	82.86	59.3	39.38	44.94	1.63
		p-value	1.63E-09	7.56E-10	7.56E-10	9.62E-10	3.78E-09	/
		Conv.rate %	10.00	2.00	0.00	14.00	24.00	86.00
	1/128	Mean	52	56.85	35.48	98.08	83.9	147.86
		Std	14.14	76.17	46.96	44.78	43.35	14.13
		p-value	1.09E-09	7.56E-10	7.56E-10	5.65E-09	2.23E-09	/
		Conv.rate %	2.00	2.00	0.00	4.00	10.00	96.00
300	1/32	Mean	100	187.99	81.84	242.15	232.94	292.9
		Std	101.01	124.76	107.54	94.24	101.49	35.43
		p-value	1.46E-08	4.78E-09	8.53E-10	5.49E-07	1.02E-04	/
		Conv.rate %	20.00	2.00	0.00	8.00	36.00	76.00
	1/64	Mean	75	172.12	52.81	221.26	185.18	298.66
		Std	75.75	128.92	90.84	103.52	110.87	2.87
		p-value	1.02E-09	7.56E-10	8.03E-10	2.10E-09	4.54E-08	/
		Conv.rate %	10.00	2.00	0.00	4.00	22.00	78.00
	1/128	Mean	55	98.15	38	148.02	139.51	298.9
		Std	35.34	132.09	85.73	114.98	113.38	2.68
		p-value	7.54E-10	7.56E-10	7.56E-10	8.53E-10	1.02E-09	/
		Conv.rate %	2.00	2.00	0.00	4.00	8.00	82.00

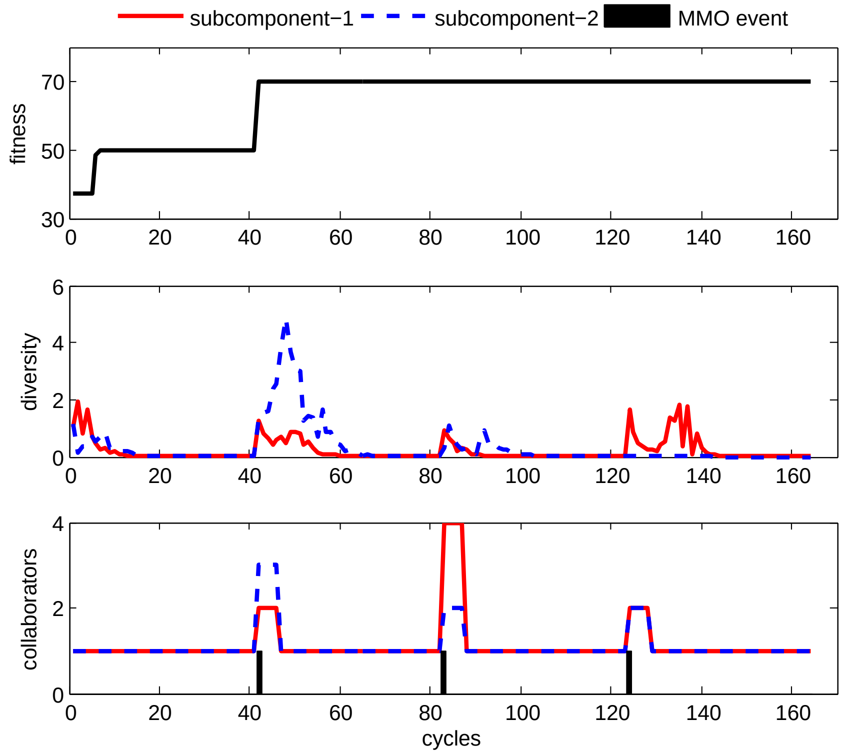

Fig. S- 1. Dynamics of the MMO-CC on MTQ function with $H2 = 70$.

TABLE S- II

DESCRIPTION OF THE CEC'2010 BENCHMARK SUITE.

Category	**Functions**	**Description**	
C1: Fully separable 1000D-separable variables	F1	Shifted Elliptic Function	Unimodal
	F2	Shifted Rastrigins Function	Multimodal
	F3	Shifted Ackleys Function	Multimodal
C2: Partially-separable Single 50D-nonseparable group 950D-separable variables	F4	Shifted and rotated Elliptic Function	Unimodal
	F5	Shifted and rotated Rastrigins Function	Multimodal
	F6	Shifted and rotated Ackleys Function	Multimodal
	F7	Shifted and Schwefels Problem 1.2	Unimodal
	F8	Shifted and Rosenbrocks Function	Multimodal
C3: Partially-separable Ten 50D-nonseparable groups 500D-separable variables	F9	Shifted and rotated Elliptic Function	Unimodal
	F10	Shifted and rotated Rastrigins Function	Multimodal
	F11	Shifted and rotated Ackleys Function	Multimodal
	F12	Shifted and Schwefels Problem 1.2	Unimodal
	F13	Shifted and Rosenbrocks Function	Multimodal
C4: Partially-separable Twenty 50D-nonseparable groups	F14	Shifted and rotated Elliptic Function	Unimodal
	F15	Shifted and rotated Rastrigins Function	Multimodal
	F16	Shifted and rotated Ackleys Function	Multimodal
	F17	Shifted and Schwefels Problem 1.2	Unimodal
	F18	Shifted and Rosenbrocks Function	Multimodal
C5: Fully-nonseparable 1000D-nonseparable variables	F19	Shifted Schwefels Problem 1.2	Unimodal
	F20	Shifted Rosenbrocks Function	Multimodal

TABLE S- III

DESCRIPTION OF VARIOUS CC ALGORITHMS THAT ARE USED TO COMPARE WITH MMO-CC.

Algorithm		Optimizer	Description
CC with differential grouping	DECC-DG	SaNSDE	Differential grouping, traditional CC in a round-robin fashion.
	DECC-IG	SaNSDE	Ideal grouping, traditional CC in a round-robin fashion.
	CBCC1-DG	SaNSDE	Differential grouping, contribution based CC.
	CBCC2-DG	SaNSDE	Differential grouping, contribution based CC.
	CC-GDG-CMAES	Modified CMA-ES	Global differential grouping, traditional CC in a round-robin fashion.
	CC-IG-CMAES	Modified CMA-ES	Ideal grouping, traditional CC in a round-robin fashion.
CC without differential grouping	DECC-D	SaNSDE	Delta grouping.
	DECC-G	SaNSDE	Random grouping.
	MLCC	SaNSDE	A pool of potential subcomponent sizes.
	CC-CMA-ES	CMA-ES	A strategies pool (3 strategies) with adaptive manner.
	DECC-DML	SaNSDE	Delta grouping with a pool of potential subcomponent sizes.

TABLE S- IV
SUMMARY OF THE COMPARED NON-CC METAHEURISTIC ALGORITHMS.

Algorithm	Description
2S-Ensemble	A two-stage based ensemble optimization evolutionary algorithm. The search procedure is divided into two stages: 1) global shrinking with EDA based-on mixed Gaussian and Cachy models; 2) local exploration with three randomly selected optimizers.
DECC-G	CC with random grouping and SaNSDE optimizer. Baseline algorithm.
GaDE	DE with generalized parameter adaptation scheme.
jDElscop	A self-adaptive differential evolution algorithm that employs three mutation strategies and a population size reduction mechanism.
MA-SW-Chains	Winner in CEC'2010 competition on large-scale global optimization. It adopts a concept of local search chain in which each individual a local search intensity so that the most promising individuals may exploit with higher local search intensity. The Solis & Wets algorithm (SW) is adopted as the local search optimizer.
MOS-CEC2012	Winner in CEC'2012 competition on large-scale global optimization. It is a hybrid approach combining the SW and the first local search method of the multiple trajectory search algorithm (MTS-LS1) within a multiple offspring sampling (MOS) framework.
MOS-CEC2013	Winner in CEC'2013, CEC'2015 competitions on large-scale global optimization. Similar to MOS-CEC2012, but GA, SW and a variant of MTS-LS1, named MTS-LS1-Reduced, are combined within the framework of MOS.

TABLE S- V
STATISTICAL VALIDATION WITH ADJUSTED p-VALUES TO ACCOUNT FOR THE FAMILY-WISE ERROR (MMO-CC IS THE CONTROL ALGORITHM).

MMO-CC vs.	Unadjusted p-value	Holm p-value	Hochberg p-value
jDElscop	3.66E-01	$1.00E+00$	$1.00E+00$
MOS-CEC2012	5.40E-01	$1.00E+00$	$1.00E+00$
MOS-CEC2013	1.65E+00	$1.00E+00$	$1.65E+00$
MA-SW-Chains	6.75E-01	$1.00E+00$	$1.00E+00$
2S-Ensemble	4.39E-01	$1.00E+00$	$1.00E+00$
DECC-G	1.08E-04	$7.53E-04\surd$	$7.53E-04\surd$
GaDE	1.74E-03	$1.05E-02\surd$	$1.05E-02\surd$

$\surd$ means that there are statistical differences with significance level $\alpha = 0.05$

TABLE S- VI
STATISTICAL VALIDATION WITH ADJUSTED p-VALUES TO ACCOUNT FOR THE FAMILY-WISE ERROR (MOS-CEC2013 IS THE CONTROL ALGORITHM).

MOS-CEC2013 vs.	Unadjusted p-value	Holm p-value	Hochberg p-value
jDElscop	6.58E-02	3.29E-01	3.29E-01
MOS-CEC2012	1.21E-01	3.64E-01	3.49E-01
MMO-CC	3.49E-01	3.64E-01	3.49E-01
MA-SW-Chains	1.75E-01	3.64E-01	3.49E-01
2S-Ensemble	8.72E-02	3.49E-01	3.49E-01
DECC-G	1.52E-06	$1.06E-05\surd$	$1.06E-05\surd$
GaDE	4.77E-05	$2.86E-04\surd$	$2.86E-04\surd$

$\surd$ means that there are statistical differences with significance level $\alpha = 0.05$

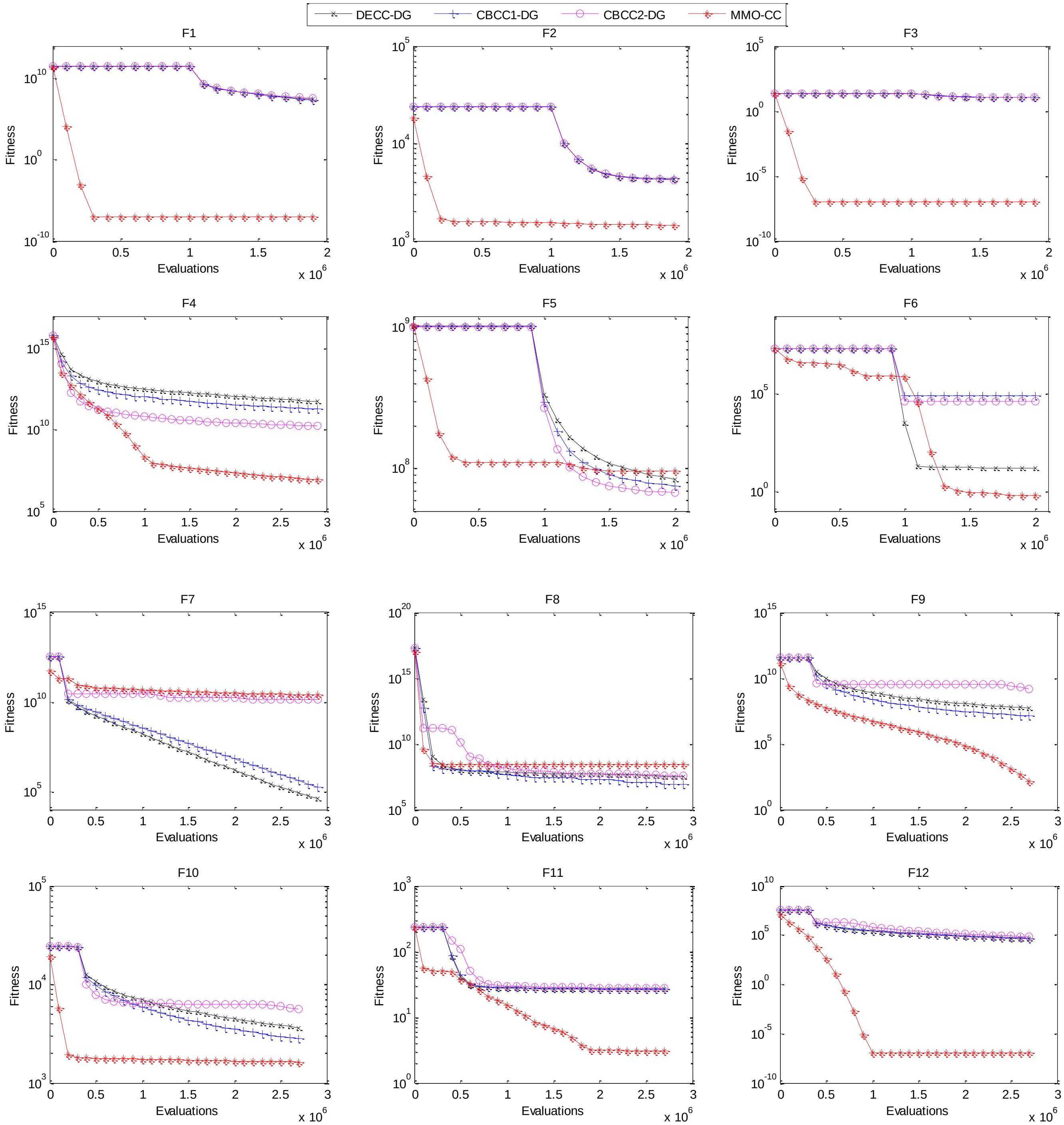

Fig. S- 2. Average fitness values (semi-logarithmic plots) of *differential grouping* based CC algorithms on CEC2010 benchmark functions ($F1 \sim F12$). The plots are generated from 25 independent runs and the number of fitness evaluations of each algorithm is the remaining number after grouping.

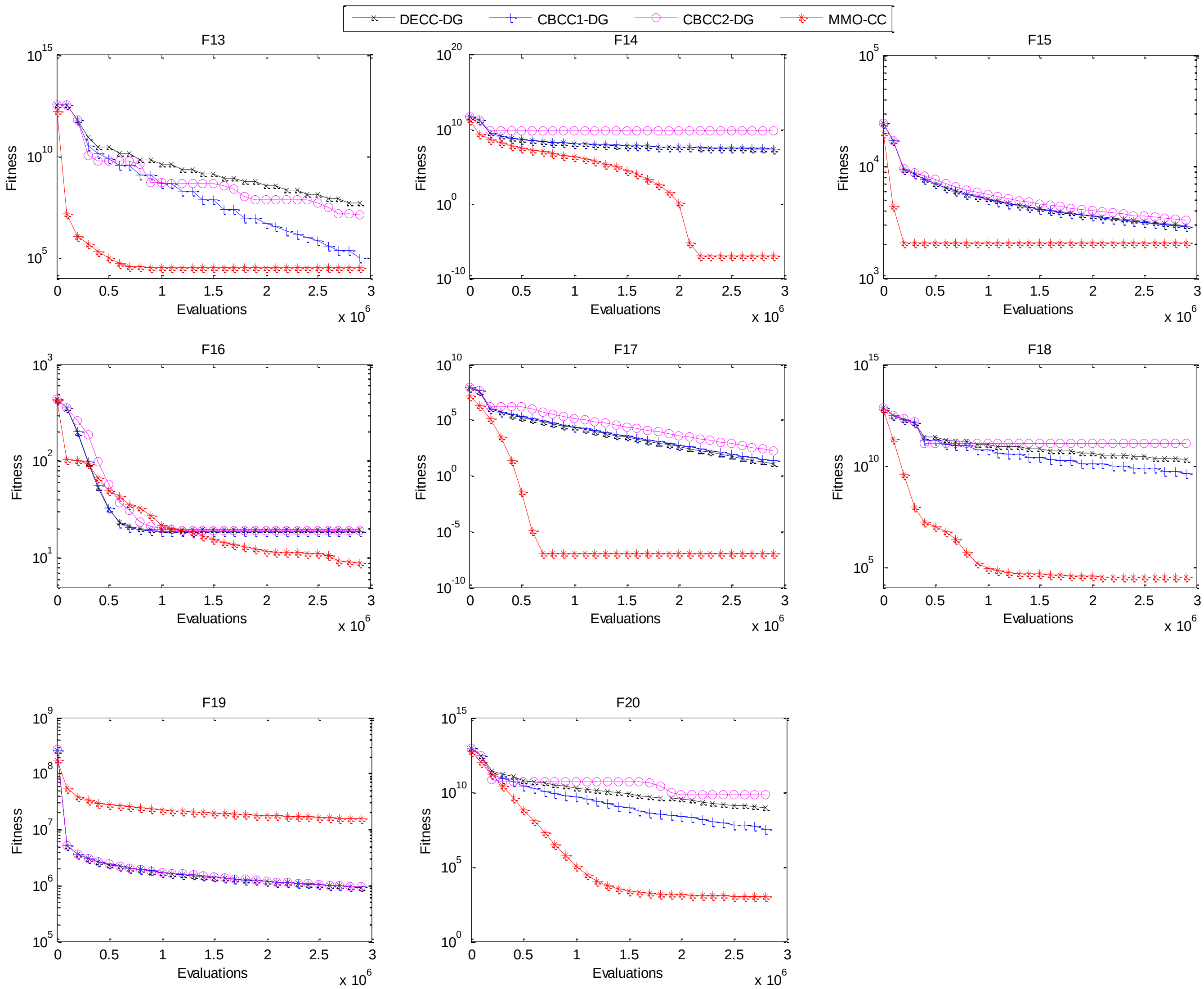

Fig. S- 3. Average fitness values (semi-logarithmic plots) of *differential grouping* based CC algorithms on CEC2010 benchmark functions ($F13 \sim F20$). The plots are generated from 25 independent runs and the number of fitness evaluations of each algorithm is the remaining number after grouping.

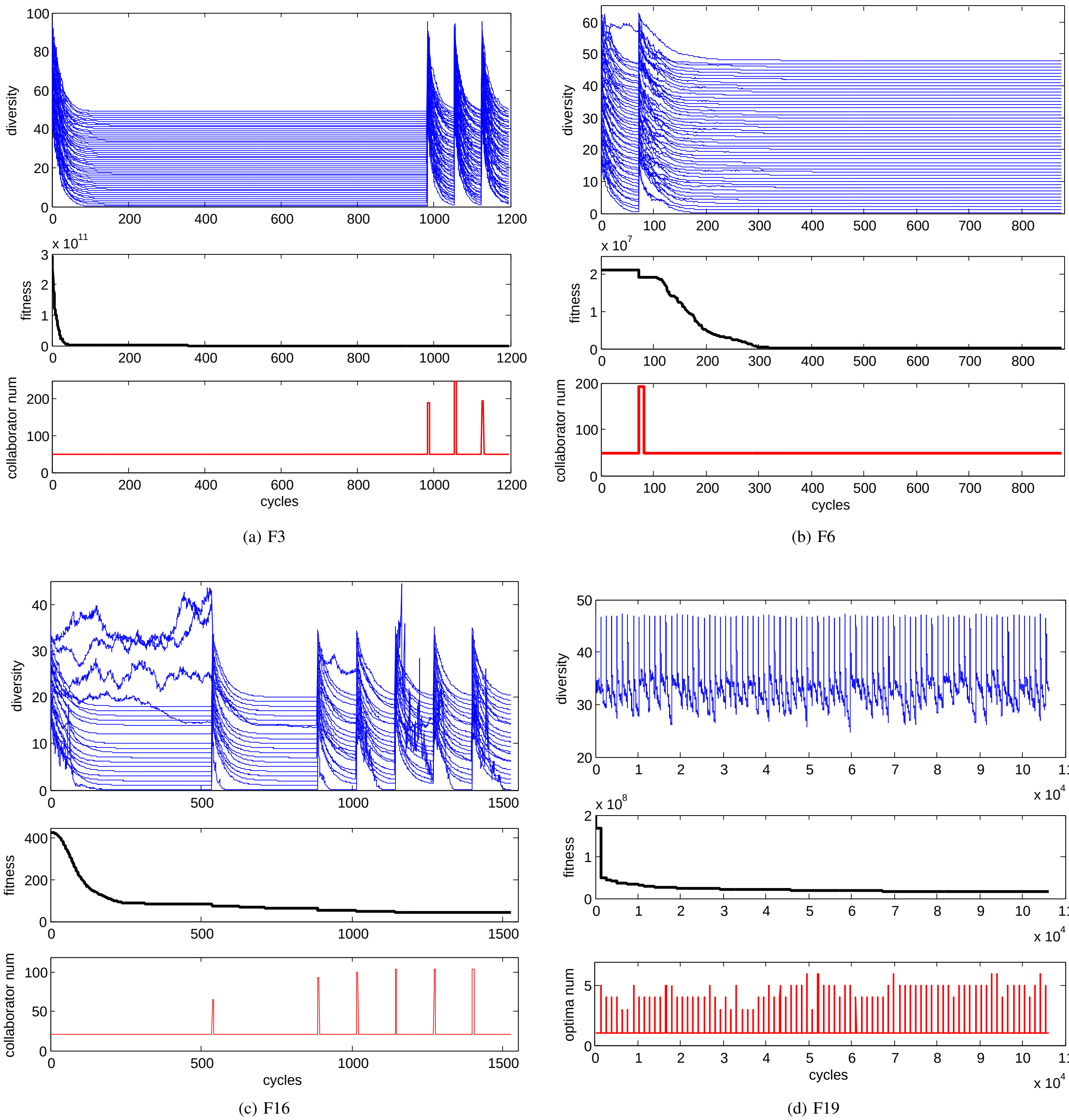

Fig. S- 4. Dynamics of the MMO-CC on $F3$, $F6$, $F16$, $F19$.

TABLE S- VII

PERFORMANCE COMPARISONS BETWEEN MMO-CC AND NON-DG BASED CC ALGORITHMS. STATISTICAL RESULTS ARE OBTAINED BY 25 INDEPENDENT RUNS ON CEC'2010 BENCHMARK SUITE ACCORDING TO WILCOXON SIGNED RANK TEST WITH A SIGNIFICANT LEVEL OF 0.05.

Functions		DECC-D	DECC-G	MLCC	CC-CMAES	DECC-DML	MMO-CC
F1	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Std	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.00E-06	0.00E+00
F2	Mean	6.52E+01	1.31E+03	**2.61E-03**	1.46E+03	1.04E+01	1.43E+03
	Std	4.47E+01	3.26E+01	5.34E-03	2.43E+02	2.24E+01	8.43E+01
F3	Mean	2.29E+00	1.39E+00	1.27E-02	**0.00E+00**	2.57E-01	**0.00E+00**
	Std	1.75E-01	9.73E-02	2.65E-02	0.00E+00	7.06E-01	0.00E+00
F4	Mean	2.98E+12	1.70E+13	1.17E+14	1.08E+12	1.18E+14	**7.64E+06**
	Std	9.35E+11	5.37E+12	4.12E+13	5.37E+11	1.69E+14	1.31E+06
F5	Mean	2.86E+08	2.63E+08	5.04E+08	**1.78E+08**	4.99E+08	3.34E+08
	Std	1.08E+08	8.44E+07	1.36E+08	9.92E+07	1.28E+08	1.54E+08
F6	Mean	5.89E+06	4.96E+06	1.90E+07	1.03E+06	1.68E+07	**5.77E-01**
	Std	5.43E+06	8.02E+05	2.12E+06	1.42E+06	6.08E+06	1.32E+00
F7	Mean	1.47E+05	1.63E+08	4.88E+10	**1.60E-02**	3.42E+10	2.41E+10
	Std	2.47E+05	1.37E+08	1.64E+10	4.41E-02	5.19E+10	6.26E+09
F8	Mean	1.27E+08	**6.44E+07**	8.23E+08	6.93E+07	3.10E+10	2.63E+08
	Std	1.52E+08	2.89E+07	1.92E+08	8.28E+07	6.90E+10	5.29E+08
F9	Mean	1.01E+08	3.21E+08	1.69E+09	2.75E+07	1.05E+09	**8.99E+01**
	Std	9.09E+06	3.38E+07	2.54E+08	2.86E+06	1.13E+09	4.64E+01
F10	Mean	4.07E+03	1.06E+04	5.19E+03	3.54E+03	4.30E+03	**1.63E+03**
	Std	1.26E+03	2.95E+02	1.72E+03	2.49E+02	1.77E+03	9.10E+01
F11	Mean	9.98E+01	2.34E+01	2.00E+02	1.07E+02	1.91E+02	**2.99E+00**
	Std	1.01E+02	1.78E+00	2.24E+00	5.22E+01	3.56E+01	3.97E+00
F12	Mean	9.14E+03	8.93E+04	8.68E+05	5.33E-01	4.76E+05	**0.00E+00**
	Std	1.08E+03	6.87E+03	1.24E+05	2.29E-01	4.69E+05	0.00E+00
F13	Mean	5.44E+03	5.12E+03	3.24E+04	**1.83E+03**	8.62E+04	3.05E+04
	Std	2.76E+03	3.95E+03	2.61E+04	1.16E+03	1.95E+05	9.43E+04
F14	Mean	3.00E+08	8.08E+08	3.62E+09	6.70E+07	2.22E+09	**0.00E+00**
	Std	2.19E+07	6.07E+07	5.43E+08	5.11E+06	2.04E+09	0.00E+00
F15	Mean	1.30E+04	1.22E+04	1.17E+04	4.99E+03	1.10E+04	**2.05E+03**
	Std	2.18E+02	8.97E+02	2.05E+03	4.98E+02	2.77E+03	9.39E+01
F16	Mean	2.02E+02	7.66E+01	3.99E+02	2.33E+02	3.62E+02	**8.87E+00**
	Std	1.58E+02	8.14E+00	3.43E+00	7.02E+01	1.09E+02	9.25E+00
F17	Mean	7.47E+04	2.87E+05	1.79E+06	4.19E+01	9.71E+05	**0.00E+00**
	Std	4.72E+03	1.98E+04	1.78E+05	7.12E+00	1.05E+06	0.00E+00
F18	Mean	1.44E+04	2.46E+04	1.07E+05	**8.29E+03**	7.77E+04	3.37E+04
	Std	1.27E+04	1.05E+04	2.68E+04	3.79E+03	1.75E+05	2.75E+04
F19	Mean	**1.59E+06**	**1.11E+06**	2.96E+06	5.26E+06	2.70E+06	1.54E+07
	Std	1.32E+06	5.15E+04	4.29E+05	1.26E+06	3.37E+06	1.59E+06
F20	Mean	2.27E+03	4.06E+03	1.75E+05	**9.86E+02**	5.42E+03	1.10E+03
	Std	2.44E+02	3.66E+02	2.08E+05	2.10E+01	1.46E+04	1.51E+02
No. Best		2	3	2	7	1	14
MMO-CC vs.		15 vs 3	14 vs 4	17 vs 2	11 vs 3	15 vs 4	/

TABLE S- VIII

AVERAGE FRIEDMAN RANKINGS OF THE COMPARED CC ALGORITHMS ON DIFFERENT CATEGORIES OF TEST FUNCTIONS.

Algorithm	f_separable F1–F3	p_separable F4–F18	n_separable F19–F20	Overall F1–F20
DECC-D	3.00	5.00	**3.50**	4.55
MLCC	2.67	7.27	5.50	6.40
DECC-DML	3.00	6.60	4.50	5.85
DECC-DG	6.33	3.67	4.00	4.10
CBCC-1	7.00	3.40	4.5	4.05
CBCC-2	6.67	4.93	5.00	5.20
CC-GDG-CMAES	5.00	2.87	4.00	3.30
MMO-CC	**2.33**	**2.67**	5.00	**2.55**

TABLE S- IX
AVERAGE FRIEDMAN RANKINGS OF THE COMPARED ALGORITHMS ACCORDING TO DIFFERENT CHARACTERS OF THE TEST FUNCTIONS.

Algorithm	Fully separable F1–F3	Partially separable F4–F18	Non-separable F19–F20
2S-Ensemble	4.33	3.93	6.50
DECC-G	7.67	6.33	7.50
GaDE	**2.33**	7.07	4.25
jDElscop	3.33	4.60	4.00
MA-SW-Chains	6.00	3.60	3.75
MMO-CC	4.00	3.13	7.00
MOS-CEC2012	5.00	4.37	**1.00**
MOS-CEC2013	3.33	**2.97**	2.00

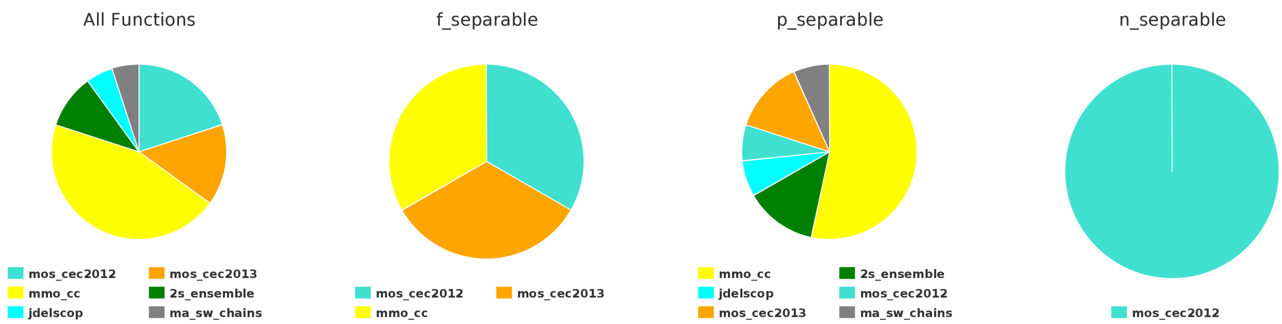

Fig. S- 5. Pie charts that describe the proportion of functions for which each algorithm obtains the best results in terms of the mean error.

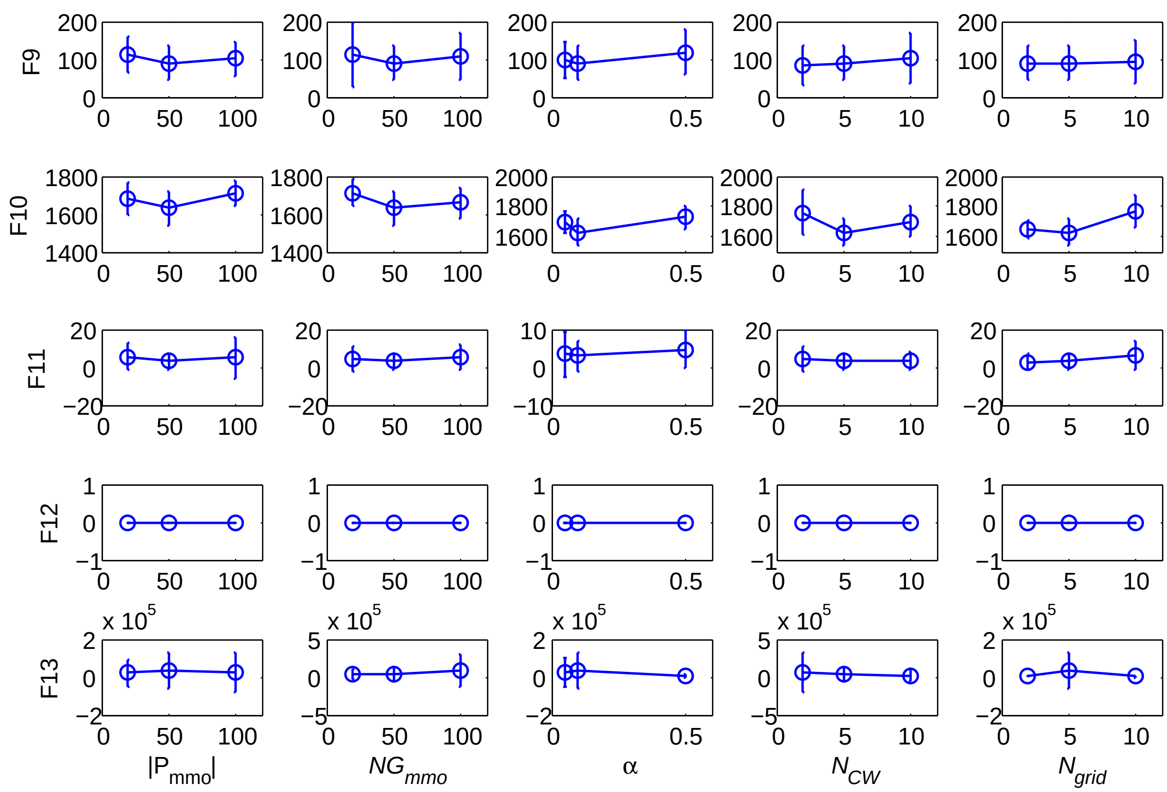

Fig. S- 6. Performance of the MMO-CC with different parameter settings on $F9 - F13$.

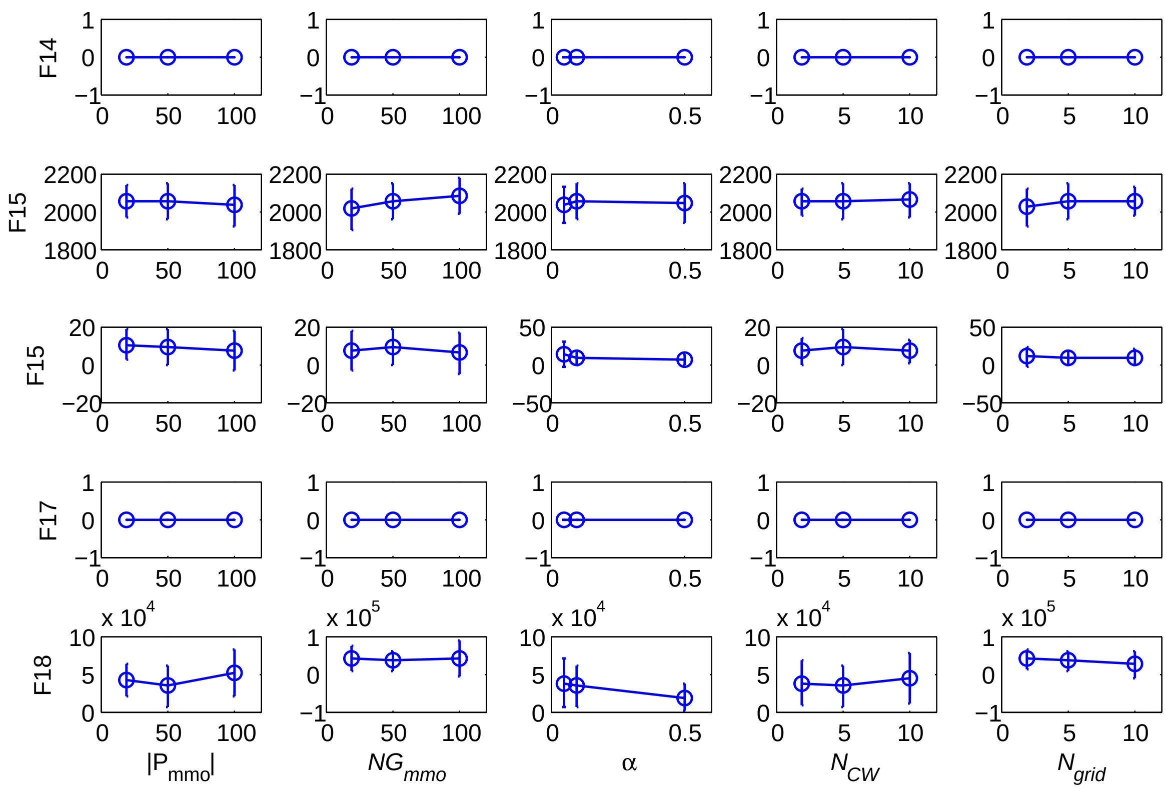

Fig. S- 7. Performance of the MMO-CC with different parameter settings on $F14 - F18$.