Information Sciences 295 (2015) 407–428

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Metaheuristics in large-scale global continues optimization: A survey

Sedigheh Mahdavi [a], Mohammad Ebrahim Shiri [a,*], Shahryar Rahnamayan [b]

[a] Department of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, Iran
[b] Department of Electrical, Computer, and Software Engineering, University of Ontario Institute of Technology (UOIT), 2000 Simcoe Street North, Oshawa, ON L1H 7K4, Canada

ARTICLE INFO

Article history:
Received 3 July 2014
Received in revised form 9 September 2014
Accepted 12 October 2014
Available online 23 October 2014

Keywords:
Large-Scale Global Optimization (LSGO)
Evolutionary Algorithms (EAs)
Cooperative Coevolution (CC)
Problem decomposition
High-dimension
Metaheuristic

ABSTRACT

Metaheuristic algorithms are extensively recognized as effective approaches for solving high-dimensional optimization problems. These algorithms provide effective tools with important applications in business, engineering, economics, and science. This paper surveys state-of-the-art metaheuristic algorithms and their current applications in the field of large-scale global optimization. The paper mainly covers the fundamental algorithmic frameworks such as decomposition and non-decomposition methods. More than 200 papers are carefully reviewed to prepare the current comprehensive survey.

1. Introduction

Most real-world optimization problems tackle with a large number of decision variables, known as Large Scale Global Optimization (LSGO) problems. Many science and engineering applications are formulated as LSGO problems such as designing large scale electronic systems, scheduling problems with large number of resources, vehicle routing in large scale traffic networks, gene recognition in bioinformatics, inverse problem chemical kinetics, etc. For example, inverse problems in the biological systems are a large-scale and highly time-consuming optimization problems [95,82,105,163]. A well-known model of biological systems is the S-system model which is a particular set of complex non-linear differential equations. The estimation of parameters in this model is generally described as a challenging optimization problem because a large number of parameters (i.e., $2N(N+1)$) should be determined simultaneously (e.g., if an S-system model involves 30 components; then the number of variables is equal to $2 \times 30 \times (30+1) = 1860$). These models are faced with some challenging characteristics such as strong interaction among parameters and high multimodality. Recently, LSGO has become a well-recognized field of research and various metaheuristic algorithms such as Simulated Annealing (SA) [16] and variant population-based algorithms, such as, Evolutionary Algorithms (EAs) [7], Genetic Algorithms (GA) [58], Estimation of Distribution Algorithms (EDAs) [91,122] and Differential Evolution (DE) Algorithm [141,161], Particle Swarm Optimization (PSO) [81,80], Ant Colony Optimization (ACO) [39,40], and Artificial Bee Colony (ABC) [75] have been applied to solve them. However, generally speaking, the standard metaheuristic algorithms for solving LSGO problems suffer from main deficiency,

* Corresponding author.
E-mail addresses: s_mahdavi@aut.ac.ir (S. Mahdavi), shiri@aut.ac.ir (M.E. Shiri), Shahryar.Rahnamayan@uoit.ca (S. Rahnamayan).

http://dx.doi.org/10.1016/j.ins.2014.10.042
0020-0255/

curse of dimensionality; i.e., the performance of these algorithms deteriorates when tackling the high dimensional problems [100,167,168]. There are two major reasons for the performance deterioration of these algorithms: The first, increasing size of the problem dimension increases its landscape complexity and characteristic alteration. The second, the search space exponentially increases with the problem size; so an optimization algorithm must be able to explore the entire search space efficiently; which is not a trivial task.

It is motivating to consider these reasons and difficulties to propose new approaches for tackling LSGO problems. Therefore, in recent years, many valuable attempts in utilizing metaheuristic algorithms have been conducted. Organizing special sessions in conferences, developing novel benchmark functions, LSGO-related web-sites, and establishing several relevant journal publications confirm importance of the mentioned research field. Research works in LSGO area were reported in the leading journals, such as, Information Sciences, Lecture Notes in Computing Science, IEEE Transactions on Evolutionary Computation, Applied Soft Computing, Soft Computing, European Journal of Operational Research, and Computers and Chemical Engineering, Computers and Mathematics with Applications, etc. Also, research works on LSGO have been presented during several international conferences and workshops, such as, Evolutionary Algorithms and other Metaheuristics for Continuous Optimization Problems, Special Session on Evolutionary Computation for Large Scale Global Optimization, Genetic and Evolutionary Computation Conference, IEEE World Congress on Computational Intelligence, IEEE Congress on Evolutionary Computation, etc.

Recently, twenty high-dimensional global optimization functions in CEC-2010 [167] and a set of fifteen scalable benchmark functions in CEC-2013 [100] are provided to evaluate the performance of LSGO methods. In the CEC'2013 benchmark functions, the CEC'2010 benchmark functions were extended to better simulate real-world problems by introducing imbalance between the contribution of various subcomponents, subcomponents with nonuniform sizes, and conforming and conflicting overlapping functions. An LSGO problem is defined as follow:

$$min/max\ F(\vec{x}) = f(x_1, x_2, \ldots, x_n),\ \vec{x} \in X \tag{1}$$

where $X \subseteq R^n$ denotes the decision space with n dimensions, $\vec{x} = (x_1, x_2, \ldots, x_n) \in R^n$ is the decision variable vector, $f : X \to R$ stands for a real-valued continuous nonlinear objective function for mapping from n dimensional space to one dimensional fitness value $F(\vec{x})$, and n is the number of variables in large scale setting (generally speaking, $n > 100$). Many optimization algorithms attempt to solve LSGO problems efficiently in a given number of fitness evaluation budget. The purpose of this paper is to present a comprehensive survey of the main contributions of metaheuristic algorithms to solve LSGO problems.

The remainder of this paper is organized as follows: Section 2 is a brief description of the LSGO tackling approaches. In Section 3, the benchmark problems and performance measures are explained. The summary of results for some research works are provided in Section 4. Section 5 presents the applications of LSGO. Finally, the work is concluded in Section 6.

2. LSGO tackling approaches

Several particular mechanisms have been proposed to handle LSGO problems, since large scale optimization is the essential challenge in the science and engineering fields. Basically, two main categorizes of approaches can be found, namely, Cooperative Coevolution (CC) algorithms with problem decomposition strategy [139,138], and non-decomposition based methods. Non-decomposition-based methods solve LSGO problems as a whole; and they are designed with the specific effective operators or they are combined with other optimization method to further enhance their performance to explore complex search spaces. The decomposition methods are based on divide-and-conquer approach which decomposes LSGO problems into single-variable or multiple low dimensional subcomponents [139,138]. The following subsections provide a brief overview of the LSGO techniques; Fig. 1 presents a hierarchical classification of these methods.

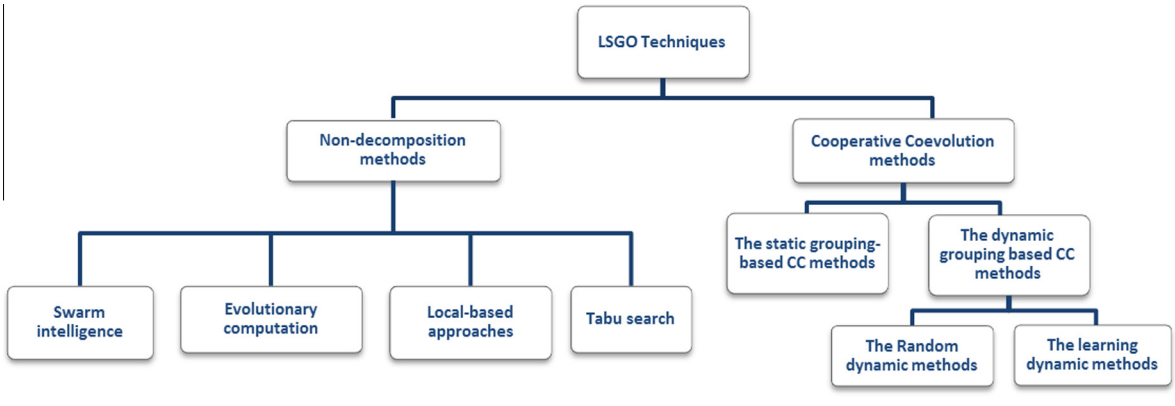

Fig. 1. A hierarchical classification of the LSGO techniques.

2.1. Cooperative Coevolution (CC) algorithms

The first effort in decomposition of a high dimensional problem was proposed by Potter and De Jong [139,138] in 1994, they designed a CC algorithm to improve the performance of the standard GA. The primary CC methods were one-dimensional based and also splitting-in-half strategies [140]. The one-dimension based method divides an n-dimensional problem into n one-subcomponents while the splitting-in-half method divides an n-dimensional problem into two $n/2$ subcomponents. The classical steps of CC framework are as follows:

Step 1: *Problem decomposition*: Decomposing a high-dimensional objective vector into some smaller non-overlapping subcomponents,
Step 2: *Subcomponent optimization*: Executing individually a traditional optimization algorithm to evolve each subcomponent for a predefined number of generations in a round-robin strategy,
Step 3: *Cooperative combination*: Merging the solutions of all subcomponents to construct the n-dimensional solution.

In subcomponent optimization step, the fitness of an individual in each subcomponent is computed as an n-dimensional vector which is constructed by integrating it with the selected individuals taken from other subcomponents. Two simple collaboration methods among subcomponents were suggested in [139], the best and random collaboration methods. The CCGA-1 algorithm employs the best collaboration method to compute the fitness of an individual by integrating it with the current best members of other components. The CCGA-2 algorithm employs the random collaboration method to compute the fitness of an individual by integrating it with the randomly selected members of other subcomponents. The CC method is applied in a wide range of real-world applications [18,54,133,135,116,8]. The CC method decomposes the decision vector into groups of variables and cooperatively optimizes them during predetermined cycles. Several studies have employed the CC methods to solve LSGO problems where they are divided to two general categories in term of variable grouping strategy: static and dynamic grouping methods. Next subsection focuses on the several CC based works which are significantly advanced well-known categories proposed in the last decade. Table 1 summarizes proposed major variants of the CC method.

2.1.1. The static grouping-based CC methods

Potter and De Jong proposed the great decomposition strategies for the CC methods [139]. The CCGA-1 and CCGA-2 methods were only tested on problems with maximum dimension of 30. The results have shown that the performance of CCGA-1 on separable problems is significantly better than a standard GA, while the performance of CCGA-1 alleviates on non-separable problems. The separable problems are referred to the problems with no interacting variables, i.e., the influence of each variable on the fitness value is independent of any other variables whereas problems with strong interacting variables are

Table 1
A summary of major variants of the CC method.

Author	Algorithm	Brief explanation
Omidvar et al. [130]	MLSoft [130]	A decomposition method based on reinforcement learning
Mahdavi et al. [111]	DM-HDMR [111]	A decomposition method based on high dimensional model representation method
liu and Tang [104]	CC-CMA-ES [104]	The CC framework was applied to CMA-ES for scaling up CMA-ES
Omidvar et al. [129]	DECC-DG [129]	DECC using automatic decomposition strategy (The differential grouping)
Ren and Wu [150]	CCOABC [150]	The orthogonal experimental design method was employed in cooperatively coevolving ABC
Li and Yao [99]	CCPSO2 [99]	The random grouping and adaptive weighting scheme were embedded in a cooperatively coevolving PSO
Sayed et al. [155]	HDIMA [155]	The hybrid dependency identification with the memetic algorithm
Sayed et al. [154]	DIMA [154]	The dependency identification with the memetic algorithm
Sun et al. [197]	CPSO-SL [197]	A cooperative particle swarm optimizer with statistical variable interdependence learning
Omidvar et al. [128]	CBCC [128]	A contribution based CC which selects the subcomponents based on their contributions to the global fitness
Omidvar et al. [126]	DECC-ML [126]	The uniform selection of subcomponent sizes along with more frequent random grouping
Chen et al. [23]	CCVIL [23]	A CC method is composed of two learning and optimization phases
Omidvar et al. [127]	[127]	A decomposition method based on a delta value
Ray. and Yao [158]	CCEA-AVP [158]	A Cooperative Co-evolutionary EA is developed with an adaptive partitioning
Yang et al. [200]	DECC-G [200]	A n-dimensional problem is partitioned randomly into several subcomponents with an adaptive weighting method
Yang et al. [201]	MLCC [201]	Using a decomposer pool to select the different group size based on the recorded performance of the decomposer
Van den Bergh and Engelbrecht [174]	CPSO-SK and CPSO-HK [174]	The CC method is integrated with PSO algorithm
Liu et al. [106]	FEPCC [106]	The CC framework with FEP (fast evolutionary programming)
Potter and De Jong [139]	CCGA-1 and CCGA-2 [139]	The first CC algorithm was proposed to solve LSGO problems

referred to non-separable [153] or epistasis [83] problems. Liu et al. [106] incorporated the CC framework with FEP (fast evolutionary programming) called FEPCC to solve benchmark functions with 100 to 1000 real-valued variables. FEPCC confirmed insufficiency of Potter and Jong's decomposition method to deal with non-separable problems.

The first attempt was made by Van den Bergh and Engelbrecht [174] to integrate the CC method with PSO algorithms for proposing two cooperative PSO algorithms, CPSO-SK and CPSO-HK. Main framework of CPSO-SK is based on the original decomposition method defined in [139] with a special difference where a vector was partitioned into K s-dimensional sub-problems (where $n = K * s$). A concatenation of all global best particles from all K swarms is called a context vector $\hat{y}$ which is used to compute the fitness of a particle in a swarm. In CPSO-HK, the standard PSO and CPSO-SK are incorporated such that the CPSO-SK is performed for one cycle, followed by the standard PSO in the next cycle. A simple information exchange method between CPSO-SK and the standard PSO is conducted as the context vector $\hat{y}$ in CPSO-SK section after one iteration is applied to replace a randomly selected particle in the standard PSO section, and in the same way, if the standard PSO discovers a new global best particle, this vector will be replaced by a randomly selected particle in the CPSO-SK section. Both CPSO-SK and CPSO-HK were tested on maximum dimension of 30. Also, the static grouping based CC method was applied using DE in [157] where the search space were divided into two $n/2$-dimensional subcomponents. Therefore, this method losses its effectiveness for handling LSGO problems with very large dimension.The same CC method was also applied in the Bacterial Foraging Optimization (BFO) [20]. Similar to CPSO-HK and CPSO-SK, two Cooperative co-evolutionary ABC algorithms, namely, CABC-S and CABC-H, were proposed in [44]. The static grouping based CC method is effective on low-dimensional (up to 100 D) problems. Therefore, several authors have been interested in developing the novel decomposing strategies to handle very large scale problems more efficiently.

2.1.2. The dynamic grouping based CC methods

Since the static grouping based CC methods are inefficient during handling non-separable problems, several works have been proposed new methods to detect variable interactions and assign interacting variables to the same subcomponent. In the static grouping based CC methods, a fixed value of the subcomponent size k is chosen for the subcomponent size and then s variables in each subcomponent remain in the same subcomponents over the optimization process while the dynamic grouping based CC methods dynamically change the grouping structure. The methods can be categorized into two classes based on how variables are placed in one subcomponent, namely, random and learning based methods.

2.1.2.1. The random dynamic grouping based CC methods. Recently, Yang et al. [200,199] introduced a DE-based CC method which utilizes the random grouping to solve LSGO problems with dimension of 500 and 1000. Their method is called DECC-G [200], which is based on the research work introduced in [199] for handling non-separable problems. An n-dimensional objective vector is partitioned randomly into multiple low dimensional subcomponents with respect to predetermined sizes; each subcomponent is evolved by a self-adaptive DE with the neighborhood search algorithm (namely SaNSDE) [202]. An adaptive weighting framework is proposed for further tuning of solutions; it assigns a weight to each of the subcomponents after each cycle. An optimization algorithm optimizes these weights in which the optimization problem has a much lower dimensionality than the original n-dimensional problem. The steps of DECC-G algorithm are summarized as follows:

1. *Set $i = 1$ to start a new cycle.*
2. *Based on random grouping method, an n-dimensional object vector is randomly divided into m s-dimensional subcomponents.*
3. *Evolve the ith subcomponent with a certain EA for a certain number of objective function evaluations.*
4. *If $i < m$then $i++$, and go to Step 3.*
5. *Assign a weight vector for each subcomponent and then optimize it via a certain EA for the best, worst, and random members of the current population.*
6. *Stop if termination criteria are satisfied; otherwise go to Step 1 for the next cycle.*

The results of the random grouping method show an efficient performance on scalable non-separable benchmark functions (up to 1000D); however, its performance becomes ineffective when the number of interacting variables grows [126].

A new multilevel CC algorithm (MLCC) was proposed to improve DECC-G [201]. Since an appropriate group size is highly related to the objective function, MLCC uses a decomposer pool. Each decomposer indicates a specific group size value to select different group sizes based on the recorded performance of the decomposer. MLCC updates the performance of the selected decomposer with respect to its current performance. The selection probability for each decomposer is computed with a self-adaptive method where is based on the performance history of the decomposer through the evolution process. In MLCC, when a group size value is picked, the objective vector is divided into a set of equally sized subcomponents while the most real-world problems have the different sizes of interacting groups [129]. Omidvar et al. [126] proposed a method to improve DECC-G and MLCC. They extended the predefined probability of the random grouping to each number of interacting variables. Also, they indicated that the use of adaptive weighting [200] in the random weighting grouping is not very efficient while more frequent random grouping is efficient without increasing the number of objective function evaluations, especially when the number of interacting variables is more than two. Moreover, a simple method for choosing a decomposer from the pool in MLCC was introduced. In [203], the DECC-G algorithm was modified with an improved adaptive weighting strategy. By using this strategy, the computational time and the number of objective function evaluations are reduced. Also,

JADE algorithm [210] is used as the subcomponent and weight vector optimizer algorithm to find more accurate solutions. In addition, this scheme was applied to other optimization techniques to enhance their performance on LSGO problems. In [98], a cooperatively coevolving PSO was proposed where the random grouping and the adaptive weighting schemes are embedded in the CCPSO algorithms proposed by Van den Bergh and Engelbrecht [174]. A new cooperative coevolving PSO (CCPSO2) was introduced in [99]; it was performed on problems with up to dimension 2000. It embeds the lbest ring topology, Cauchy, and Gaussian distributions in the standard PSO to create a new position for particles. Furthermore, a random grouping method, with a modified position updating rule, and a coevolving strategy to dynamically determine size of subcomponents are utilized. In the updating personal best stage, two matrices are defined to store all information of particles'current position and personal best in all K generations. Then, a random grouping method is applied which is randomly permutated the indices of all columns and thus a particle's position and personal best vectors in each swarm are formed by the new permutated dimension indices.

A CC Orthogonal ABC (CCOABC) algorithm was proposed in [150] where uses the Orthogonal Experimental Design (OED) method to improve its performance effectively. In OED, a factor denotes as a variable in the optimization problem and each factor has three levels, one level is the original one and two levels are calculated as follow:

$$V_{ij} = x_{ij} + R_{ij}(x_{ij} - x_{kj})$$
$$V_{ij} = x_{ij} - R_{ij}(x_{ij} - x_{kj})$$

where V_{ij} is a new location, R_{ij}(a random number) $\in [-1,1], i \in 1,2,\ldots,N, k \in 1,2,\ldots,N$ (N is the number of food sources) and $k \neq i, j \in 1,2,\ldots,N$ (D is the number of dimensions). In the CC stage, the random grouping strategy is used to split a problem into multiple low dimensional subcomponents and each subcomponent by a CCOABC algorithm is optimized. In [30], two synchronous and asynchronous co-evolutionary approaches were analyzed with a broad experiments on LSGO problems which extended the prior research work in [200]. In synchronous approach, the evaluation context, i.e., a representative vector, is updated after evolving all components while the evaluation context in asynchronous approach is updated after evolving each component.

2.1.2.2. The learning-based dynamic grouping. Since the appropriate grouping of the interacting variables requires the prior knowledge of a problem, the learning methods focus on other strategies instead of random groping strategy. In these methods, the identification of interacting variables is learnt by the obtained experiences of the problem characteristics either before or during optimization process. The learning methods were proposed in order to increase the placing chance of the interacting variables in the same subcomponent especially when the number of interacting variables becomes very large.

Ray and Yao introduced a CC Algorithm based on the correlation matrix [149]. In the first M cycles, there is one subcomponent which includes all variables therefore evolution process is similar to a standard EA. In the next cycles, the correlation matrix of top 50% solutions in the population is calculated and the decision variables are partitioned into several subcomponents based on the correlation coefficient value among them such that the variables with a correlation coefficient greater than the certain threshold value is placed in the same subcomponent. In [158], a CCEA with an adaptive partitioning- called CCEA-AVP- was introduced, inspired from Ray and Yao's earlier research work [149]. It has been shown that methods based on correlation coefficient use the huge computational resources but they do not recognize the nonlinear dependencies among the variables [155,125]. In the standard CC method [139], each of the subcomponents was optimized in a round-robin strategy. In this method, the computational budget was equally associated into all of the subcomponents. In [128], they recognized the imbalance among the fitness portions of different subcomponents. Hence, they suggested Contribution Based Cooperative Co-evolution algorithm (CBCC) in which the computational budget is associated to the subcomponents according to their contributions. They showed that if there is an imbalance among the separable and non-separable parts of the fitness value, the round-robin strategy loses its performance and uses the considerable amount of the computational budget. The method is capable to save the significant amount of computational time.

A simple method was proposed to recognize interactions among variables which is briefly described as follows [197]. It is assumed that the '**best**' is the best solution achieved so far, the '**new**' denotes the best individual when a CC optimizer algorithm has optimized for a dimension i, and a randomly selected individual from the population is called the '**rand**'. Two new individuals are generated according to these three vectors as following form:

$$x_j = \begin{cases} new_i & \text{if } j = i \\ best_j & \text{otherwise} \end{cases} \quad x'_j = \begin{cases} new_i & \text{if } j = i \\ rand_k & \text{if } j = k \\ best_j & \text{otherwise} \end{cases}$$

After that, if $f(x')$ is better than $f(x)$, the interaction probability between i and k is increased. Based on the mentioned research work [197], Chen et al. [23] proposed a CC method with Variable Interaction Learning (CCVIL) which is capable to adaptively change group sizes. This method includes two phases, namely, learning and optimization. In learning phase, the detection of interaction among variables is identified similar to the method proposed in [197]. In [127], a CC framework was introduced based on a delta value which is computed corresponding to the absolute amount of change in each dimension at two sequential cycles. The delta grouping is taken from a key feature of non-separable problems, i.e., there is the limited extent of interacting variables to optimize towards global optimum [153]. Next, the decision variables are sorted

corresponding on their delta values; then the grouping of variables is provided based on the sorted delta values. When there are several non-separable subcomponents, the performance of this method deteriorates [129].

According to [197], the variables x_i and x_j are dependent if the influence of a variable x_i on the fitness value depends on the variable x_j. A statistical model of the variable interdependence learning was introduced in [164] where is briefly described as follows. Let $\vec{\alpha} = (x_1, \ldots, x_i, \ldots, x_j, \ldots, x_n), \vec{\beta} = (x_1, \ldots, x'_i, \ldots, x_j, \ldots, x_n)$, and $f(\vec{\alpha}) \leqslant f(\vec{\beta})$, if by turning x_j value into x'_j, the inequality $f(\vec{\alpha}) \leqslant f(\vec{\beta})$ changes into the inequality $f(\vec{\alpha}) > f(\vec{\beta})$ then variable x_i is affected by variable x_j. A probability value for the influence of a variable x_i by turning x_j value is estimated by the statistical model on N statistical samples. A decomposition method was also suggested where divides large scale problems into overlapping small scale subcomponents according to the estimated dependence probability. In decomposition method, N subcomponents are created for each of variables; i.e., each subcomponent s_i is set to a variable x_i which is called the core of the subcomponent s_i. Then, for each subcomponent s_i with the core x_i, if the estimated probability for the core x_i with each of variables is no less than the predefined threshold value r, they are allocated to the subcomponent s_i. Finally, each obtained subcomponent is optimized by an individual PSO.

Omidvar et al. [129] introduced an automated decomposition approach (called DECC-DG), differential grouping, where is mathematically derived from the description of the partial separability. A Theorem was defined to recognize the interaction between two variables which is defined as follow.

Theorem 1. *Suppose the function $f(\vec{x})$ an additively separable function, If the following condition holds, then x_p and x_q are non-separable:*

$$\forall a, b_1 \neq b_2, \delta \in R, \delta \neq 0, \Delta_{\delta, x_p}[f](\vec{x})|_{x_p=a, x_q=b_1} \neq \Delta\delta, x_p[f](\vec{x})|_{x_p=a, x_q=b_2}$$

where $\Delta_{(\delta, x_p)}[f](\vec{x}) = f(\ldots, x_p + \delta, \ldots) - f(\ldots, x_p, \ldots)$ is the forward difference of f according to variable x_p with the interval δ. For all decision variables, the following process is repeated: The interaction between a decision variable i and all other variables is checked based on using the Theorem 1 in a pairwise way. And then, if the interaction is recognized, the interacting variables are located in the same subcomponent i. if any interaction is not recognized for a decision variable, this variable is recognized as separable variable. Furthermore, they shown how the linkage identification method (LINCR) [169] can be derived by the Theorem 1. In [155,154], the dependency identification (DI) technique for decomposing a LSGO problem was proposed; it is derived from the definition of problem separability provided in [121,162]. The problem separability is defined in [121] as following form:

$$F(x) = \sum_{i=1}^{n} f(x_i) \tag{2}$$

And a partially separable problem is defined in [149,162] as following equation:

$$F(x) = \sum_{k=1}^{m} f_k(x_v), \quad v = [1, V] \tag{3}$$

where the problem $F(x)$ is decomposed into m subcomponents which each subcomponent has V dependent variables. The DI technique decomposes the decision variables into several subgroups so that provides the least square difference sq_{diff} between $F(x)$ and $F(x) = \sum_{k=1}^{m} f_k(x_v), v = [1, V]$ which is calculated by the following equation:

$$sq_{diff} = \left[F(x) - \sum_{k=1}^{m} f_k(x_v) \right]^2, \quad v = [1, V]$$

$$x_{v,c_1} = \begin{cases} c_1 & \forall v = [1, V] \\ c_2 & \text{otherwise} \end{cases}, \quad x_{v,c_2} = \begin{cases} c_2 & \forall v = [1, V] \\ c_1 & \text{otherwise} \end{cases}$$

Then, the subgroups are optimized by applying a memetic algorithm (MA) with a self-directed local search. Moreover, new benchmark problems were developed to evaluate the performance of the DI method. In [192], a new variance priority strategy was developed to enhance the classical CC structure. It selects variables with larger variances to direct the current optimized sub-component. Statistical information based on linear correlation coefficient between each variable and the objective function was utilized; the DE algorithm is employed as an optimizer [151]. In the first generation, the linear correlation coefficient between each variable and the objective function in the current population is computed. In the other generations, a weighted sum of the correlation coefficient value of the previous generation and the calculated correlation coefficient for current population is utilized. A modification of the MLCC algorithm (MLSoft) based on reinforcement learning was proposed in [130] which identify adaptively the subcomponent size. They show that adaptively identifying the subcomponent size for fully separable problem by using reinforcement learning methods can improve the performance of MLCC.

In [111], a decomposition method based on high dimensional model representation method (DM-HDMR) was proposed. High dimensional model representation models the high dimensional input–output system behavior. First, the first-order RBF-HDMR model is computed according to the first-order RBF-HDMR introduced by Shan and Wang [156]. In this method,

two interacting variables are detected if a correlation relationship of two variables is identified according to the Shan and Wang's strategy [156] and then the separable and nonseparable subcomponents are constructed. A method (CC-CMA-ES) based on CMA-ES was proposed in [104] which uses a CC framework to CMA-ES for scaling up CMA-ES to solve large scale problems. Also, two decomposition strategies (Min-Variance decomposition strategy (MiVD) and Max-Variance decomposition strategy (MaVD)) based on the diagonal of the covariance matrix were introduced.

In addition, the study of the parameter settings for CC methods was conducted in [108]. A centralized cooperative strategy based on a rule-driven coordinator was proposed in [113] and a new analysis was presented to study the behavior of the variant coordination schemes. In [72], diversity measures for the cooperative PSO was proposed. The CC methods transform the search space into several separate subcomponents since a great value of information may be lost due to inappropriate aggregation strategy to compute the fitness of each individual in a subcomponent. A CC algorithm with global search was developed in [211] which combines the CC algorithm and the EDA based-on mixed Gaussian and Cauchy models [190] to overcome premature convergence in CC algorithms. Many research works were conducted to analyze the behaviors of CC methods [134,132,171,137,176]. In [171], a benchmark problem, the Lamps problem, was introduced to analyze the behavior of CC algorithms which models a search for the optimal placement of a set of lamps inside a room. The influence of problem decomposition on the performance of CC algorithms was analyzed in [22]. Empirical studies are conducted based on the prior knowledge of problems to uncover the advantage of detecting variable interactions. In [76], the impact of initial population was investigated on LSGO problems. They have shown that finding a proper initialization method could be valuable in solving LSGO problems efficiently. The versions of information exchange referred as update timing were analyzed in [137] which controls the optimization algorithm in CC framework evolves subcomponents sequentially or in parallel. Panait et al. [134] introduced a probabilistic biasing method to evaluate individual from a particular subcomponent by the linear combination of bias and collaborative fitness assessment. A master/slave model for CC algorithms was designed in [176] to parallelize these algorithms. In [206], a hybrid adaptive optimization strategy was proposed for solving LSGO problems which integrates JADE and SaNSDE as the subcomponent optimization algorithm of CC. The subcomponent optimization algorithm is changed among JADE and SaNSDE when the current optimization algorithm cannot any improvement in the fitness value.

2.2. Non-decomposition methods

Since the classical operators and the different stages in the metaheuristic algorithms were usually designed for low-dimensional problems and show the deficient performance during tackling high-dimensional problems, many modified algorithms have been proposed strategies on the classical operators and applied on different stages to enhance the performance of algorithms. The non-decomposition methods without divide-and-conquer strategy proposed methods with focus on the especial alteration such as defining new mutation, selection, and crossover operators, designing and using local search, opposition-based learning, sampling operator, hybridization, and incremental or reduction population size methods to significantly enhance their performance during exploration to handle LSGO problems. Major research works are summarized as in Table 2.

2.2.1. Swarm intelligence

Hsieh et al. [68] proposed a PSO algorithm with an Efficient Population Utilization strategy (EPUS-PSO). A population manager and solution sharing strategies were defined and embedded in PSO to remove redundant particles. A population manager strategy, based on the following three criteria, increases or decreases its population size:

1. When the *gbest* is not changed in k consecutive iterations, a new particle is inserted into the swarm.
2. If particles can find one or more solutions to update the *gbest* in k consecutive iterations, the redundant particles are removed from the swarm to enhance search progress.
3. When the global best solution is not changed and the population size is equal to the maximum size of population, a particle with the poor fitness value in the current swarm is removed in order to add a new potential particle.

In solution sharing strategy, the moving vector of the third item in the velocity update equation of standard PSO is changed in proportion to a certain probability to another particle's pbest. Also, a searching range sharing strategy was proposed for escaping from local minimum solutions.

In [53], a PSO algorithm with two strategies, namely, velocity modulation and restarting, was proposed. The velocity modulation controls the movement of particles so that they are directed within a limited area. A restarting strategy was used to prevent a premature convergence. If the standard deviation of the fitness values of particles in the whole swarm or the overall change in the objective function value is very low, the restarting strategy is executed. A modified PSO algorithm with a new velocity updating method, as the rotated particle swarm, was proposed in [62]. The velocity updating method uses a rotation matrix which has most zero and some non-zero elements related to rotation of limited number of pairs of randomly selected axes. In [24], a competitive PSO (CPSO) was introduced based on the random pairwise competition mechanism to solve LSGO problems (up to 5000D). At each cycle of the algorithm, particles are pairwise randomly selected from the current swarm to compete which the winner particle is passed into the next swarm and the next position and velocity of the loser particle are updated by learning from the winner particle. Fan et al. [48] proposed a PSO algorithm with dynamic neighborhood topology (FT-DNPSO) for tackling LSGO problems. FT-DNPSO incorporates the CC algorithm with kernel fuzzy

Table 2
A summary of the non-decomposition-based methods.

Author	Algorithm	Brief explanation
Cheng et al. [24]	CPSo [24]	A competitive PSO based on the random pairwise competition mechanism
Fan et al. [48]	FT-DNPSO [48]	A PSO algorithm with dynamic neighborhood topology
Dong et al. [38]	[38]	EDA frame-work based on weakly dependent variable identification and subspace modeling
Wang et al. [195]	EOEA [195]	A two-stage based ensemble optimization EA
Wang et al. [185]	GOjDE [185]	A parallel DE algorithm based on GPU
Chowdhury et al. [25]	LSCBO [25]	A large scale optimization based on coordinated bacterial dynamics and opposite numbers
Takahama and Sakai [165]	LMDEa [165]	A DE algorithm based on landscape modality detection, unimodal or multimodal, and a diversity archive
Wang and Gao [177]	[177]	A DE algorithm with a new selection operator based on the local fitness by using CC methods
Hedar and Ali [64]	TSVP [64]	Tabu search with multi-level neighborhood structures
Wang et al. [188]	OXDE [188]	A DE with quantization orthogonal crossover
Weber et al. [196]	SOUPDE [196]	A shuffle parallel DE based on multi-population
Garcá-Martńez et al. [52]	[52]	A differential evolution approach with the role differentiation and malleable mating mutation methods
Wang et al. [183]	GODE [183]	A DE based on Generalized Opposition-Based Learning
Wang et al. [184]	GOPSO [184]	A PSO integrated with GOBL and the Cauchy mutation
LaTorre et al. [92]	[92]	A memetic DE was combined with the multiple offspring sampling
Garcá-Nieto et al. [53]	RPSO-vm [53]	A PSO with two mechanisms; velocity modulation and restarting method
Montes de Oca et al. [36]	IPSOLS [36]	An incremental PSO for large-scale continuous optimization problems
Rahnamayan and Wang [144]	[144]	A novel center-based sampling method
Zhang and Sanderson [210]	JADE [210]	An improved DE algorithm with a new mutation operation as DE/current-to-pbest
Hsieh et al. [68]	EPUS-PSO [68]	A PSO with an efficient population utilization method
Korosec and Silc [53]	DASA [53]	An ACO-based algorithm based on a new concept as the variable offsets
Tseng and Chen [173]	MMTS [173]	The modified multiple trajectory search
Wang and Li [189]	LSEDA-gl [189]	A univariate EDA base on three methods: sampling under the mixed Gaussian and the Levy probability distribution, standard deviation control and restart strategy
Rahnamayan et al. [148]	ODE [148]	A DE algorithm with combing the concept of Opposition-Based Learning

clustering and variable trust region methods. The kernel fuzzy clustering method divides the high dimension problem into several low dimensional subgroups. The variable trust region learning method changes adaptably the independent variable ranges to improve the convergence speed. Also, a dynamic neighborhood topology is introduced which is incorporated with PSO to avoid premature convergence.

Montes de Oca et al. [35,36,120,34,37] proposed an Incremental Social Learning (ISL) framework which combines components of social and individual learning to increase learning rate. In [36], a modified IPSOLS algorithm was developed to solve LSGO problems. The original IPSOLS algorithm is a PSO algorithm which is hybridized with an incremental population size and a local search to improve some particles. The redesign method of IPSOLS algorithm consists of six phases, namely, selection of a local search method, alteration of calling and controlling the local search method, using vectorial PSO rules, penalizing bound constraints violation, and fighting stagnation with restarting. For automatically tuning parameters, iterated F-race [10,11] was applied which has the configuration generation, selection, and refinement stages iteratively. In [101], they also introduced an incremental ACO algorithm (IACOR-LS) with incremental solution archive which is combined with a local search method. In [86,88,90,89], an Ant-based algorithm, namely the differential ant-stigmergy, was proposed according to a new concept as the variable offsets for solving the continuous optimization problems. With utilizing discretized offsets, a real-parameter optimization problem is transformed to a graph-search problem. The parameter difference for new value x' by supposing current value x is defined as follows:

$$x_i = x'_i + \delta_i \text{ (For the } i\text{-th variable)}$$

where δ_i is called the parameter difference and is selected from the following set:

$$\Delta_i = \Delta_i^- \cup \{0\} \cup \Delta_i^+$$
$$\Delta_i^- = \delta_{i,k}^- | \delta_{i,k}^- = -b^{k+L_i-1}, k = 1, 2, \ldots, d_i$$
$$\Delta_i^+ = \delta_{i,k}^+ | \delta_{i,k}^+ = b^{k+L_i-1}, k = 1, 2, \ldots, d_i$$

where $d_i = U_i - L_i + 1, L_i = \lfloor \log_b(\epsilon_i) \rfloor, U_i = \lfloor \log_b(x_i^{max} - x_i^{min}) \rfloor$, and b is the discrete base. A large scale optimization algorithm based on coordinated bacterial dynamics and opposite numbers (LSCBO) was introduced in [25]. A population includes the three types of bacteria, namely, primary, associated bacteria and the opposite associated bacterium. The associated bacterium $\widetilde{X}$ is produced from the primary bacterium X by following formula:

$$\widetilde{X^k} = X^k + \delta D^k \text{ (For the } k\text{-th variable)}$$

where $\delta D^k = (0, 0, 0, \ldots\ldots\ldots, d_l^k, 0, 0, \ldots\ldots, 0), d_l^k = c_1 r_1 (UB - LB), r_1$ is a random number in interval $[-1, 1]$, and UB and LB are the upper and lower boundaries, respectively. The opposite associated bacterium is also generated from the primary bacterium by following formula:

$$\widetilde{X^*} = X^k - \delta D^k$$

Then, the primary bacterium moves towards the fittest bacterium of the population (FP). The velocity of primary bacterium calculated by:

$$v_i^k = FP_i^k - X_i^k \text{(For the } k\text{-th variable)}$$

where FP_i^k and X_i^k are the components of FP. Thus, the position of the primary bacterium is defined by:

$$X^{K+1} = X^k + V^K$$

A hybrid PSO algorithm [186] was proposed which is equipped with a new diversity method and neighborhood search strategies. In the diversity method, a trial particle for each particle is created by:

$$TX_{ij}(t+1) = \begin{cases} X_{ij}(t+1), & \text{if } rand_j(0,1) < p_r \\ X_{ij}(t) & \text{otherwise} \end{cases}$$
$$TV_{ij}(t+1) = V_{ij}(t+1)$$

A soft adaptive particle swarm algorithm was introduced in [9]. This method includes three stages: (1) a new level is defined to update the position of the particle based on the difference between its initial position and a random particle, (2) the inertia weight is changed and improved, (3) and an acceleration parameter is developed to normalize the current velocity. Chu et al. [26] analyzed and compared the aspects of the three boundary handling techniques, namely, the random, absorbing, and reflecting schemes in the high-dimensional complex problems and indicated insights and the specific information about the performance of PSO. In [32], a micro-Bacterial Foraging Algorithm (BFA) was proposed which uses a very small population and the preservation of the best bacterium. In [96], the scalability behavior of ACOR [159], an ant colony optimization algorithm for continuous domains, was studied and a simple method was introduced to deal with low diversity. Moreover, the modified ABC algorithm [4], a locust swarms [21], a particle swarm optimization based on utilizing guided re-initialization [17], an immune algorithm [31], and a hybrid BFA-PSO algorithm [29] were proposed to solve LSGO problems.

2.2.2. Evolutionary computation

In [110], an Unbiased Evolutionary Programming algorithm (UEP) was derived from a classical EP algorithm. In UEP, the mutation operation was designed which is different from a classical EP by adding angles in the encoding of an individual. Also, the effect of several parameters was analyzed where can significantly improve the performance of algorithm. A GA with a matrix-coding partitioning was proposed in [63]. Individuals in the population are coded as a matrix and then this matrix is divided into several sub-matrices at every generation. The crossover and mutation operations are used to evolve the sub-matrices. Additionally, a modified version of termination criteria which was introduced in [65] is used. In [27], a new evolutionary search strategy, SP-UCI, was developed which includes four modules, namely, the complex shuffling, population dimensionality monitoring and restoration, modified competitive complex evolution, and multi-normal resampling.

Wang and Li [189] proposed a univariate EDA (namely LSEDA-gl) base on the following three methods, sampling under the mixed Gaussian and the Levy probability distributions, standard deviation control, and restart strategy. In [190], Wang and Li analyzed the performance of univariate EDAs mixed with the different kernel probability densities according to the fitness landscape analysis and proposed a self-adaptive mixed distribution based on uni-variate EDA with the mixed kernels. In [38], a EDA framework was proposed which combines Weakly dependent variable Identification (WI) and Subspace Modeling (SM) steps to control the complexity of multivariate model on high dimensional problems. In WI strategy, the global correlation matrix is computed and then variables with the absolute values of correlation coefficients less than a certain threshold are identified as weakly dependent variables which a simple univariate model is estimated for them. In SM step, the n dimensional search space is partitioned into several subspaces, and then a multivariate model for each subspace is constructed. A two-stage based Ensemble Optimization Evolutionary Algorithm (EOEA) was introduced [195,191] where includes two sections; the global shrinking and the local exploration. In global shrinking section, an EDA based-on mixed

Gaussian and Cachy models is used to shrink the searching scope to the potential region. In the local exploration section, a CC based algorithm with tuning the size of each group adaptively is used. Each group is evolved via selected randomly algorithms from three candidate algorithms. After the certain number of iterations, a new group is created with some variables which have more effect on the fitness values. Garcá-Martńez and Lozano [51] developed a continuous variable neighborhood search algorithm which has three evolutionary metaheuristic components; generation, improvement, and shaking. These components employ Covariance Matrix Adaptation Evolution Strategy (CMA-ES), continuous local EA and Heterogeneous recombination, and Cataclysmic mutation (μCHC) as particular EA, respectively. Furthermore, a new assortative mating method was proposed which is incorporated in the continuous local EA. In the assortative mating method, after selecting first parent, the second parent is selected according to two criteria; better fitness value and high similarity to first parent. In [125], the performance of CMA-ES was considered on LSGO problems and compared with CC Algorithms (CCEAs). In [28], three shuffled complex evolutions (SCE-UA,an algorithm based on simplex scheme) [42,41], PSO, and DE algorithms were performed on high-dimensional benchmark functions. After a few evolution cycles, the population tends toward a subspace and the search region for discovering of global solution is limited to this subspace. This event is called "population degeneration" and thus Principal Components Analysis (PCA) is introduced to detect population degeneration and to alleviate its bad influences.

2.2.2.1. DE-based algorithm. Due to the simple concept of DE algorithm and the easy implementation, it has become popular in solving LSGO problems. In this subsection, several modified DE algorithms for handling LSGO problems are briefly described. An improved DE algorithm, JADE, was introduced in [210] which is incorporated with a new mutation operation as DE/current-to-pbest. The greedy mutation operations of DE are modified to develop a new mutation operation so that it utilizes the information of a specific set of top solutions (p% of top solutions). In this mutation operation, an optional external archive is applied to handle recorded information about success and failure. Let A is the archive of inferior solutions and P denotes the current population. The DE/current-to-pbest/1 without external archive is defined for the current X as follows:

$$\overrightarrow{V}_{i,G} = \overrightarrow{X}_{i,G} + F_i.\left(\overrightarrow{X}^{p}_{best,G} - \overrightarrow{X}_{i,G}\right) + F_i.\left(\overrightarrow{X}_{r^i_1,G} - \overrightarrow{X}_{r^i_2,G}\right) r_1, r_2 \in [1, NP]$$

where $\overrightarrow{X}^{p}_{(best,G)}$ is randomly selected as one of the top $p\%$ members of the current population with $p \in (0, 1]$. $\overrightarrow{X}_{r^i_1,G}$ and $X_{r^i_2,G}$ are selected from P. For each individual i, the mutation factor F_i, associated in each generation, is independently updated with corresponding to a Cauchy distribution by:

$$F_i = rand\ c_i(\mu_F, 0.1)\mu_F = (1-c).\mu_F + c.mean_L(S_F)$$

$$mean_L(S_F) = \frac{\sum_{F\in S_F} F^2}{\sum_{F\in S_F} F}$$

The location parameter μ_F and scale parameter are set to 0.1. The DE/current-to-pbest/1 with the external archive is defined:

$$\overrightarrow{V}_{i,G} = \overrightarrow{X}_{i,G} + F_i.\left(\overrightarrow{X}^{p}_{best,G} - \overrightarrow{X}_{i,G}\right) + F_i.\left(\overrightarrow{X}_{r^i_1,G} - \overrightarrow{\widetilde{X}}_{r^i_2,G}\right)$$

where $\overrightarrow{X}_{r^i_1,G}$ and $\overrightarrow{X}^{p}_{best,G}$ are selected from P and $\overrightarrow{\widetilde{X}}_{r^i_2,G}$ is randomly selected from $P \cup A$.

In [165], a DE algorithm based on landscape modality detection, i.e., unimodal or multimodal, was proposed. The landscape modality of the search points is recognized by the objective function values of a sampled point set on a line which is determined by the centroid of search points and the best search point. When the objective function value changes from decreasingly to increasingly, there is one valley. Therefore, if there is only one valley then the landscape is unimodal; otherwise the landscape is multimodal. Therefore, a method for choosing suitable value of the scaling factor was proposed based on this landscape modality detection. In [177], a DE algorithm with new selection operator based on the local fitness by using CC methods was proposed which is a modification of the research work referred in [178]. In DE or CC methods, a global fitness function is used to evaluate all the variables in an individual. However, a solution A can be worse than the solution B based on the global fitness function where some variables in solution A have great quality values. They introduced the concept of local fitness function which is motivated from the ideas of the genetic engineering and modern medicine to keep the good variables in a solution. Hence, the local selection operator splits a high scale problem into some subcomponents and associates a local fitness function to evaluate each subcomponent. The local and global fitness values are simultaneously used to evolve the population.

Brest et al. [15] was designed a self-adaptive DE where uses a population-size reduction method and a technique for the sign alteration of scale parameter F. The sign of scale parameter F is exchanged with a probability based on the fitness values of randomly selected vectors for the mutation operation of DE algorithm during search process. In [188], a DE with quantization orthogonal crossover (OXDE) was proposed. A search range of the two parents is quantized and then the levels of each factor according to quantization of search range are defined. When the dimension of a problem becomes much larger than the number of the levels, QOX is broken into some subvectors similar to the research work referred in [97]. QOX was used once at each generation to save the computational cost. In [196], a shuffle parallel DE based on multi-population was introduced where uses two randomized strategies. At the beginning of the algorithm, population is divided into several sub-populations which a scale factor is associated to each sub-population. The first strategy, i.e., shuffle strategy, repartitions

population into several sub-populations with a certain probability and the second strategy replaces all the scale factors of the sub-populations by randomly sampled numbers between 0.1 and 1. Zamuda et al. [208] proposed a cooperative coevolution DE by using log-normal self-adaptation to control parameters. Yang et al. [204] studied and analyzed previous adaptation methods. Then, a generalized parameter adaptation framework for DE was proposed which is incorporated with the advantages of other methods.

In [173,172], a Modified Multiple Trajectory Search (MMTS) is introduced which uses the simulated orthogonal array to make initial solutions. In MTS, the multiple agents are applied to exploration simultaneously the solution space while an iterated local search is performed by each agent. Also, three local search methods were applied to find different neighborhood structures. In [214], a self-adaptive DE as SaDE-MMTS was enhanced with JADE mutation strategy and MMTS. Trial vector generation strategies and control parameters are adapted in according to their prior performance and SaDE-MMTS also generates highly better solutions by using a MMTS. In [52], a DE algorithm based on two strategies, namely, role differentiation, and malleable mating mutation was proposed. Four different roles, namely, placing, leading, correcting and receiving vectors usually can be identified in the classic DE. In the classic DE, each individual of population are randomly selected to play one of these roles in the mutation and crossover operations. The role differentiation strategy provides the suitable selection of individuals for each role by generating four different groups. Furthermore, the malleable mating mutation adjusts the mating tendency of placing vectors to guarantee some similarity relations. Piotrowski et al. [136] designed a DE algorithm with the separated subpopulations in which population is divided into small groups and the information exchange rules among the individuals of the group were defined. In [71], a DE with a sampling method was introduced for generating more offspring of the crossover operation to increase diversity. In [59], a co-evolutionary chromosome encoding scheme to evolve individuals was introduced. In the CC stage, each subcomponent evolves its individuals which is collaborated with the different subcomponents by using the concept of friends introduced by Gomez et al. [60]. A friend concept for m subcomponents is a reference to an individual from other subcomponents. The definition of fitness function is extended by using the friend concept from considering just one individual to consider a group of them. The generation trial process is similar to the crossover operation of DE. A DE with a new mutation operator was proposed in [198] which generates an offspring based on the local best solution, the best vector among the randomly selected vectors, and global best solution. An adaptive DE with local search based on Cauchy mutation was developed in [131] which generates a neighbor solution around the global best individual to add population when the global best individual does not improves in certain generations. A MOS-based hybrid algorithms were developed in [94] which executes population-based and local algorithms in sequence and the participation of each algorithm is adjusted dynamically. Wang et al. [187] proposed a DE framework with multiobjective sorting-based mutation operator which can select parents in the DE mutation operator according to the fitness and diversity simultaneously. First, an Euclidean distance-based diversity metric for each individual is defined and then two measures, i.e., fitness and diversity information, are considered by multiobjective sorting to select the parents. Also, the first effort in scaling the Minimum Population Search (MPS) [12] was proposed on high dimensional problems in [13]. MPS can guarantee a full cover of the entire search space with a small population. By developing a new adaptive version of threshold convergence, MPS can control the balance among exploration and exploitation which is critical in LSGO.

Opposition-based approaches. Rahnamayan et al. [148,146,145,147] enhanced DE (ODE) by using the concept of Opposition-Based Learning (OBL) which was proposed by Tizhoosh [170]. The ODE used the basic concept as opposite numbers where is defined as follow.

Opposite Number [145]: supposing that $X = (x_1, x_2, \ldots, x_D)$ be a point in D-dimensional space such that $x_i \in [a_i, b_i]$ for $a_i < b_i \in R$ and $i = 1, \ldots\ldots, D$. The corresponding opposite point $\overset{\vee}{X}$ is defined by:

$$\overset{\vee}{X} = a_i + b_i - X$$

Two stages, namely, opposition-based population initialization and generation jumping are designed to be utilized in the DE algorithm. In the opposition-based population initialization stage, when the initialization population is generated, simultaneously the opposite of each individual in the initial population is created to construct the opposite population. Then, the fittest individuals are picked from the union of two populations (initial and opposite one). During search process, the generation jumping stage computes the opposite of each new individual according to a predetermined probability and then fittest individuals are selected such as the mentioned way. Also, instead of using the predefined boundaries, boundaries of individuals are determined dynamically based on minimum and maximum values of each variable of individuals in the current population.

In [142,143], Rahnamayan and Wang analyzed the performance of ODE on LSGO problems. They compared ODE with DE on seven large scale benchmark functions (D = 500 and 1000) and confirmed that in terms of convergence rate, robustness, and solution accuracy, ODE has better result than DE and ODE's performance gets better when the dimension increases. In [183], a new DE (GODE) based on Generalized Opposition-Based Learning (GOBL) was proposed. In [181,182], OBL was generalized which transforms the candidates in the current search space to a new search space. Let $\Delta = k(a + b)$, where k is a real number and the generalized OBL model is defined by the following form:

$$X^* = \Delta - X$$

The GODE employs a GOBL model with the random number k. The GODE uses two opposition-based population initialization and dynamic opposition strategies similar to ODE. In GODE, GOBL method is combined as another search component with DE

method where each method executes with respect to the certain probability. Gao et al. [49] proposed a hybrid optimization algorithm in which the OBL is integrated with the harmony search (HS) in mutation operation. In [184], an enhancing particle swarm optimization (called GOPSO) was integrated with GOBL and the Cauchy mutation [180]. GOBL is applied to the initialization population to select the appropriate candidate solutions at the beginning of the algorithm. Also, GOBL is used with respect to the certain probability in all algorithm steps. A parallel DE algorithm based on GPU (GOjDE) [185] was introduced which modifies the GODE algorithm [183]. In order to enhance the performance, the self-adapting control parameter method and GOBL were applied together in GOjDE. Additionally, GOjDE executes in parallel on the multiprocessors of GPU to decrease effectively the computational time. In [2], several versions of extended shuffled DE (SDE) were proposed based on the opposition-based population initialization and generation jumping methods. In the shuffled DE (SDE), the population is divided into several subsets, memeplexes, and each memeplex is improved by a DE. All algorithms use the opposition-based population initialization and different types of opposition-based generation jumping. In [79], a hybridization of OBL with CC framework was introduced to deal with LSGO problems. when the population of each subcomponent is evaluated, its opposite population is constructed for the associated dimension of each subcomponent and then the fittest individuals are picked from the union of two populations.

Rahnamayan and Wang [144] introduced the center-based sampling concept. They defined the probability of closeness to an unknown solution for the points of the search space in a black-box problem based on the Euclidean distance as the closeness measure. Monte-Carlo simulation is applied to compute this probability. They have shown that the probability of closeness to an unknown solution for the center point is higher than other points and by increasing the dimension of the problems, this probability value grows drastically. Due to the valuable feature of the center point, the generated samples close to this point increase the chance to be closer to an unknown solution. Furthermore, the center point properties were compared to the opposition points. In [47], a center-point-based SA was introduced in which the center point as an initial point is used to enhance the algorithm performance. Also, an enhancement to ODE based on Center-Point sampling concept was proposed in [46]. Recently, a survey of the population initialization methods for EA is provided in [78] which divides existing population initialization methods to three general categories, i.e., randomness, compositionality and generality. Also, the effects of various population initialization methods are studied for DE algorithm on LSGO problems in [77]. They conducted experiments including two parts. In the first part, the best parameter configuration of DE is obtained and the second part of experiments provides a comparison among the different population initialization methods to handle LSGO problems.

2.2.3. Local search-based approaches

In [212], a dynamic multi-swarm particle swarm optimizer was proposed as DMS-PSO which has dynamic and randomized neighborhood topologies. The population is adaptively divided into small subpopulations according to the alteration of the neighborhood structures. Each subpopulation finds the promising areas by applying its own particles. DMS-PSO is enhanced with the Quasi-Newton method to increase convergence speed. In [213], the DMS-PSO algorithm, a sub-regional HS (SHS), and Modified Multi-Trajectory Search (MTS) were combined to construct a hybridization algorithm as DMS-PSO-SHS. In [117,119,118], several memetic algorithms based on the local search chains were designed. In the local search chains, at each stage, the initial configuration of the local search operator is the final configuration reached by the previous local search. In the local search chains, the memetic algorithms, namely, the Solis and Wets's [160], the Nelder and Mead's simplex [123], the specific local MTS-LS1 search, the specific local MTS-LS2 search, the CMA-ES, and a new local search method (the Subgrouping Solis Wets' method) are used. In [194], a memetic DE scheme was proposed which the efficient evolutionary algorithms as the local search techniques are applied to the top solution within an adaptive computational budget. A hybrid approach combining the Solis & Wets algorithm [160] and the first of the local searches of the MTS algorithm [173] within the the Multiple Offspring Sampling (MOS) framework [33] is proposed in [93] and the behavior of the hybrid proposed algorithm is analyzed to solve LSGO. In [19], a DE with the scale factor local search was introduced. The local search for scale factor is described as minimization over the scale factor variable of fitness function f in the direction of two random solutions where finds a scale factor value to generate the high quality offspring.

In [50], a memetic DE with applying the simplex search method was proposed and the stochastic properties of chaotic system were used to generate the population. In [92], a memetic DE was combined with MOS framework [33] which can choose the best configuration according to the problem characteristics. A MOS-based algorithm simultaneously uses several offspring generating methods. A new quality function is defined to adapt the participation ratio of each method according to the average fitness increment and the number of improvements after the associated number of objective function evaluations. In each step, the participation ratio is applied to determine the number of objective function evaluations of each method in the next step. Hvattum and Glover [70] proposed a direct search/scatter search method where is derived from the incorporating traditional direct search method and a scatter search with the clustering based and a randomized subsets method. A simplex-based GA was developed in [67]. This method has the two reproduce operations, namely, an extremum mutation and directional reproduce. In the generation of offspring, an extremum mutation uses the best individual while the directional reproduce operations use all individuals except the best. A one-dimensional search algorithm was introduced in [56] which is combined the EUS (Enhanced Unidirectional Search) [55] and the 3-2-3 line search [57]. In [103], an adaptive local search depth strategy was introduced to arrange the computing resources among global search and local search according to the comparison of average fitness increment among them.

2.2.4. Tabu search

In [64], a Tabu search algorithm with multi-level neighborhood structures was proposed in which the decision variables are divided into small groups to create multi-level neighborhood structures. The groups are randomly chosen and modified to generate the trail solutions. A local search is also applied to enhance the best obtained solution. The neighborhood radius is updated according to the improvement of Tabu search process. Duarte and Marti [43] introduced an adaptive memory programming procedure based on the combination of the scatter and Tabu search methods.

3. Benchmark problems and performance measures

The seven benchmark test set functions provided in the CEC'08 special session on LSGO [58]. In these benchmark test set, functions G_1 (Shifted Sphere), G_4 (Shifted Rastrigin), and G_6 (Shifted Ackley) are separable and functions G_2 (Schwefel Problem), G_3 (Shifted Rosenbrock), G_5 (Shifted Griewank), and G_7 (Fast Fractal) are non-separable functions. The twenty well-known benchmark functions $F_1 - F_{20}$ were provided by the CEC'2010 Special Session and Competition on LSGO [167]. Functions $F_1 - F_6$ are the modified versions of the suite of benchmark functions developed in the CEC'2008 [166]. The four types of benchmark functions were designed including: separable, partially-separable in which most variables are independent while the rest of a small number of variables are dependent, partially-separable which includes several independent m-nonseparable subcomponents, and fully-nonseparable. The codes of the benchmark functions are available in [1]. Table 3 provides the descriptions of these benchmark functions. The specific definitions of 19 benchmark functions were also provided by the 2010 Special Issue of Soft Computing on Scalability of Evolutionary Algorithms and other Metaheuristics for Large Scale Continuous Optimization Problems [66]. Detailed mathematic formulas and descriptions of these benchmark functions can be found in [66]. The functions $H_1 - H_6$ are similar to the CEC'2008 functions $G_1 - G_6$. The functions $H_7 - H_{11}$ are H_7 (Shifted Schwefel'Problem 2.22), H_8 (Shifted Schwefel's Problem 1.2), H_9 (Shifted Extended f_{10}), H_{10} (Shifted Bohachevsky), and H_{11} (Shifted Schaffer). The hybrid composition functions $H_{12} - H_{19}$ are obtained by merging two functions from the set of functions $H_1 - H_{11}$. In addition, some non-linear transformations were applied on the CEC'2010 benchmark functions to more describe real-world large-scale optimization problems and evaluate LSGO algorithms in the report of CEC'2013 [100]. This report proposed 15 large-scale benchmark problems where are the extended versions of CEC'2010 benchmark. The performance measure of LSGO is the average function error of the obtained best solution of algorithms in 25 independent runs where the error is $f(x) - f(x^*)$. The maximal number of fitness evaluations usually are set to $5000 * D$.

4. Analysis of results

In this section, the summary of the quality of results are provided. We selected some algorithms which have been currently evaluated on CEC'2010 and CEC'08 benchmark functions and 19 benchmark functions of 2010 special issue. In [201], the MLCC method was tested on CEC'08 benchmark test functions (D = 100, 500, and 1000). The results of the MLCC algorithm are consistently good on G_1, G_4, G_5, and G_6. The performance of MLCC was compared with DECC-G and MLCC-R (D = 1000) where is a version of MLCC with using the random strategy instead of the self-adaptive strategy to adapt levels. MLCC outperforms DECC-G on $G_2 - G_6$ and both algorithms have the same results and the high performance on G_1. It was

Table 3
Summary of the CEC'2010 test functions.

Function	Name	Properties
F_1	Shifted Elliptic Function	Unimodal-Separable
F_2	Shifted Rastrigin's Function	Multimodal-Separable
F_3	Shifted Ackley's Function	Multimodal-Separable
F_4	Single-group Shifted 50-rotated Elliptic Function	Unimodal-Single-group m-nonseparable
F_5	Single-group Shifted 50-rotated Rastrigin's Function	Multimodal-Single-group m-nonseparable
F_6	Single-group Shifted 50-rotated Ackley's Function	Multimodal-Single-group m-nonseparable
F_7	Single-group Shifted 50-dimensional Schwefel's	Unimodal-Single-group m-nonseparable
F_8	Single-group Shifted 50-dimensional Rosenbrock's	Multimodal-Single-group m-nonseparable
F_9	10-group Shifted 50-rotated Elliptic Function	Unimodal-$\frac{D}{2m}$-group m-nonseparable
F_{10}	10-group Shifted 50-rotated Rastrigin Function	Multimodal-$\frac{D}{2m}$-group m-nonseparable
F_{11}	10-group Shifted 50-rotated Ackley Function	Multimodal-$\frac{D}{2m}$-group m-nonseparable
F_{12}	10-group Shifted 50-dimensional Schwefel's	Unimodal-$\frac{D}{2m}$-group m-nonseparable
F_{13}	10-group Shifted 50-dimensional Rosenbrock's	Multimodal-$\frac{D}{2m}$-group m-nonseparable
F_{14}	20-group Shifted 50-rotated Elliptic Function	Unimodal-$\frac{D}{m}$-group m-nonseparable
F_{15}	20-group Shifted 50-rotated Rastrigin's Function	Multimodal-$\frac{D}{m}$-group m-nonseparable
F_{16}	20-group Shifted 50-rotated Ackley Function	Unimodal-$\frac{D}{m}$-group m-nonseparable
F_{17}	20-group Shifted 50-rotated Schwefel's Function	Multimodal-$\frac{D}{m}$-group m-nonseparable
F_{18}	20-group Shifted 50-rotated Rosenbrock's Function	Unimodal-$\frac{D}{m}$-group m-nonseparable
F_{19}	Shifted Schwefel's Function 1.2	Multimodal-Fully nonseparable
F_{20}	Shifted Rosenbrock's Function	Multimodal-Fully nonseparable

remarked that MLCC is better than MLCC-R on $G_2 - G_7$. In [204], GaDE was executed on 19 benchmark functions of 2010 special issue with $D = 50$, 100, 200, 500, and 1000. Also, a comparison among GaDE, DE, CHC (Cross generational elitist selection, Heterogeneous recombination, and Cataclysmic mutation) [45], and G-CMA-ES (Restart Covariant Matrix Evolutionary Strategy) [6] was provided. GaDE on $H_1, H_4 - H_7, H_9 - H_{12}, H_{14} - H_{16}, H_{18}$ and H_{19} has the impressive performance. GaDE outperforms DE on $H_4, H_6, H_8, H_9, H_{11}, H_{12}, H_{14}, H_{16}$ and H_{18} while the results of DE are significantly better than GaDE on H_2, H_{10}, and H_{19} and both algorithms have the same results on other functions. GaDE also outperforms G-CMA-ES on most of the functions while G-CMA-ES has the efficient performance on H_2 and H_8. The results showed the performance of CHC is worse than other algorithms. Furthermore, efficacy of parameter adaptation and setting and computational running time were analyzed.

In [99], CCPSO2 was analyzed on the CEC'08 benchmark test functions. It was realized that for the separable functions G_1 and $G_4 - G_6$ ($D = 500$), the performance of CCPSO2 becomes significantly better by selecting a combination of small and large group sizes at the different steps of run while for non-separable functions G_2, G_3, and G_7, its performance is enhanced by selecting a small group size. The CCPSO2 algorithm was compared with sep-CMA-ES method [152,61] on CEC'08 benchmark test functions and four functions G_{3r}, G_{4r}, G_{5r}, and G_{6r} ($D = 100$, 500, and 1000) which are rotated versions of G_3, G_4, G_5, and G_6. They presented that CCPSO2 outperforms sep-CMA-ES on high-dimensional multimodal functions with a complex fitness landscape. sep-CMA-ES outperforms CCPSO2 on G_{4r}, G_1, and G_3 but on other functions both methods have the similar behavior. Then, the CCPSO2 were compared with EPUS-PSO, DMS-PSO, and MLCC methods on CEC'08 benchmark test functions ($D = 1000$). In terms of results, CCPSO2 performs reasonably better than EPUS-PSO on six CEC'08 test functions, except G_2. CCPSO2 has better results than DMS-PSO on five functions, except G_1 and G_5. The results of CCPSO2 are better than MLCC on G_1, G_2, and G_3 and not different on G_5 and G_6. Finally, they provided results on G_1, G_3, and G_7 up to 2000 dimensions.

In [196], the results of SOUPDE were provided on 19 benchmark functions of 2010 special issue with $D = 50$, 100, 200, 500, and 1000 and SOUPDE was compared with DE, CHC, and G-CMA-ES. The results based on Wilcoxon Signed-Rank test have shown SOUPDE outperforms CHC and G-CMA-ES. Comparing to DE, SOUPDE has the same results on most of the functions with $D = 50$ and 100 while SOUPDE performs better than DE on the most of the functions with $D = 100$, 500, and 1000. In addition, the Holm procedure was performed on the results and they indicated that both tests are similar for CHC and G-CMA-ES while the Holm procedure does not confirm that the results of SOUPDE are considerably better than DE. In [36], the performance of IPSOLS was analyzed on 19 benchmark functions of 2010 special issue with $D = 50$, 100, 200, 500, and 1000 and compared with DE, CHC, and G-CMA-ES. The results showed that the results of IPSOLS are better than CHC and G-CMA-ES. The performance of IPSOLS with the growth of dimensional increases in comparison to CHC and G-CMA-ES for $D = 500$ and 1000. Also, IPSOLS obtains the good results on H_1, H_3, H_4, H_5, H_6, and H_{10}.

In [185], GOjDE was executed on 19 benchmark functions of 2010 special issue with $D = 100$, 200, 500 and 1000. GOjDE was compared with DE, GODE, CHC, and G-CMA-ES. Also, the computational time of GOjDE on CPU and GPU was analyzed. The results were confirmed that the results of GOjDE on $H_1, H_5 - H_7, H_9 - H_{12}, H_{15}, H_{16}$ and H_{19} are significant while DE and GODE have the better performance on $H_1, H_5 - H_7, H_{10}, H_{12}, H_{15}, H_{16}$ and H_{19}. The performance of GOjDE is better on $H_1, H_2, H_5 - H_{19}$ than DE with $D = 100$, 200, and 500 while the performance of both algorithms is the same on $D = 1000$. GOjDE outperforms GODE and G-CMA-ES on all functions, except H_4, and all functions, except H_2, H_3 , and H_8, respectively. It should be noted that in [181,182], GODE was executed on 19 high-dimensional problems for $D = 50$, 100, 200, 500, and 1000 and was compared with DE, CHC and G-CMA-ES and it was concluded that GODE is significantly better than CHC and G-CMA-ES on all dimensions. Based on the mentioned notice in [174], they shown that CHC is not suitable for solving LSGO problems. Furthermore, GOjDE was compared with two other DE algorithms, namely, SOUPDE and GaDE and the results were reported. The results ($D = 1000$) showed that the performance of GaDE is better than GOjDE on 6 functions, while GOjDE gained much better results than GaDE on 9 functions. For $D = 1000$, GaDE achieves better results than GOjDE on 6 functions, while GOjDE outperforms GaDE on 9 functions. They achieved the same results on F_1 and F_5. Moreover, GOjDE performs better than SOUPDE on 7 functions, while SOUPDE outperforms GOjDE on 7 functions. They presented that the computational time of GOjDE by using GPU is effectively decreased.

In [118], MA-SW-Chains was performed on CEC'2010 benchmark test functions ($D = 1000$) and compared with DECC-G and MLCC. The results confirm that although the performance of MA-SW-Chains is significant on non-separable functions, its performance deteriorates significantly on F_9 and F_{14}, or F_{13} and F_{18}. Also, MA-SW-Chains outperforms DECC-G and MLCC on most of function except separable functions. MA-SW-Chains obtains the remarkable results on F_7 and F_{12} in compassion to other algorithms. In [23], CCVIL was performed on CEC'2010 benchmark test functions ($D = 1000$). From results, it was inferred that if the number of fitness evaluations in the learning stage is not sufficient to discover all interacting variables of a group, CCVL obtains more number of groups than the real groups. CCVIL determines more groups than the real group on F_3, F_6, and F_{11}. On F_8, F_{13}, F_{18} and F_{20}, CCVIL achieved good results since it detects them as separable problems. Furthermore, they compared CCVIL with DECC-G, MLCC, and JADE. The performance of CCVIL is better than DECC-G and MLCC on $F_2, F_4 - F_6, F_9 - F_{17}$, and $F_{19} - F_{20}$ while DECC-G and MLCC performed significantly better than CCVIL on F_3. The performance of CCVIL is better than JADE on $F_1 - F_3, F_{10} - F_{11}, F_{13}, F_{15} - F_{17}$, and F_{20} and JADE obtains much superior results than CCVIL on $F_4 - F_5, F_8 - F_9, F_{14}$, and F_{18}. In [129], the DECC-DG method was performed on CEC'2010 benchmark test functions. First, the details and results of identifying subcomponents were described. The grouping accuracy of this method according to the detection of the grouping of non-separable subcomponents is 100% on $F_1 - F_6, F_9 - F_{10}, F_{12}, F_{14} - F_{15}, F_{17}$, and F_{19}. It was noted that the performance of this method is related to the required number of fitness evaluations to identify subcomponents. This method requires a numerous of fitness evaluations to identify the groups on fully separable functions and the

Table 4
GOJDE's number of wins, losses, and ties against SOUPDE and GADE (on 19 benchmark functions of 2010 special issue).

Algorithm	Wins	Loses	Ties
v.s. SOUPDE	7	9	3
v.s. GaDE	9	8	2

Table 5
CCPSO2's number of wins, losses, and ties of CCPSO2 against MLCC (on the CEC'08 benchmark test functions).

Algorithm	Wins	Loses	Ties
v.s. MLCC	3	2	2

Table 6
DECC-DG's number of wins, losses, and ties against other compared algorithms (on CEC'2010 benchmark test functions).

Algorithm	Wins	Loses	Ties
v.s. MA-SW-CHAINS	4	14	2
v.s. MLCC	10	9	1
v.s. DECC-D	10	7	3
v.s. DECC-DML	9	8	3

Table 7
Summary of applications of LSGO.

Algorithm applied and references	Application
CC with weighted random grouping (CCWR) [116]	Large-scale crossing waypoints locating in air route network
A monopopulation CC genetic programming [8]	A problem related to the modeling of a cheese ripening process
The differential Ant-Stigmergy algorithm [87]	Electric motor power losses minimization, turbo-compressor aerodynamic power maximization, An electric motor casing stiffness maximization
Cooperative Co-evolution DE [178]	Seismic waveform inversion
The three algorithms; SCE-UA, PSO, and DE [28]	The parameter estimation and calibration of the highly nonlinear hydrologic models used for flood forecasting
An adaptive heuristic-based EA [112]	Large scale set covering problems with application to airline crew scheduling
Simple scheduled memetic algorithm [85]	An inverse problem of chemical kinetics
Parallel PSO [84]	Large-scale human movement problems
Cooperative Co-evolutionary DE [69]	Detecting communities in complex networks
Estimation of distribution and DE cooperation(ED-DE) [193]	Large scale economic load dispatch optimization of power systems
Three hybrid metaheuristic algorithms [207]	Engineering design optimization
Cooperative Enhanced Scatter Search (CeSS) [175]	Parameter estimation in large scale systems biology models
Two-stage based ensemble optimization EA (EOEA) [64]	The parameter calibration problem of water pipeline system
Genetic Programming [109]	The RoboCup soccer server domain
A novel cooperative co-evolutionary MOEA CCGDE3 [5]	Large scale multiobjective optimization problems
A Cooperative Coevolutionary Genetic Algorithm CCGA [3]	Resource allocation and scheduling of multiple composite web services in cloud computing
Cooperative Coevolutionary DE Algorithm [179]	High-dimensional waveform inversion
Genetic algorithms [107]	Optimal selection of wavelengths in multi-component Analysis
The Cooperative Article Bee Colony [216]	Clustering problems
A cooperative coevolutionary particle swarm optimization algorithm [74]	Flow shop production scheduling with uncertainty
Improved Self-Adaptive Differential Evolution With Neighborhood Search(SaNSDE+) and SaNSDE [205]	Building thermal model parameter identification problem
An enhanced cooperative coevolution genetic algorithm(ECCGA) [215]	The application domain of pattern classification
The Self-Adaptive DE(jDE) [209]	Reconstruction of procedural three dimensional models of woody plants
A cooperative coevolution of genetic local search with distance independent diversity control [82]	Inferring S-system models of large-scale genetic networks
Cooperative coevolution Genetic Programming [124]	The modeling of an industrial agrifood process
Cooperative coevolutionary GA(CCGA) [14]	Vibration-based damage detection
Cooperative bare-bone particle swarm optimization [73]	Clustering high-dimensional data
Variable neighborhood decomposition method [114]	Large scale capacitated arc routing Problems
A hybrid GA-PSO algorithm [102]	High-dimensional subspace clustering problem
Cooperative co-evolution with route distance grouping [115]	Large-Scale capacitated arc routing problems

number of fitness evaluations slightly decrease according to the growth of non-separable subcomponents. It was compared with CCVIL and results shown that the DECC-DG method constructs subcomponents with more accurate than CCVIL and uses the fewer number of fitness evaluations on most of the functions, except F_1, F_2, and F_7. On fully separable subcomponents, CCVIL is able to identify subcomponents with a low number of fitness evaluations while DECC-DG uses approximately one million fitness evaluations to identify them. Since the differential grouping method use the approximation of the gradient to recognize interacting subcomponents, it has the great performance on the non-separable functions. Also, the results of sensitivity analysis showed that the performance of DECC-DG does not depend too much to the parameter ϵ. Moreover, it was compared with delta grouping (DECC-D) [149], MLCC, a method as DECC-DML which is similar to DECC-D but it uses a set of potential group sizes like MLCC, MA-SW-Chains [94], DECC-G, a method as CBCC-DG which is contribution based CC [128] with differential grouping, and an ideal grouping (DECC-I) is a DECC which manually uses the knowledge of the benchmark functions to make groups. The results of all algorithms were reported and it was seen from results that DECC-DG outperforms other algorithms.

Table 4 shows the summary of the comparison results between algorithms for $D = 1000$ on 19 benchmark functions of 2010 special issue. We have included the number of wins, losses and ties for algorithm GOjDE against SOUPDE and GaDE have been included in according to the mentioned mean result referred in [185] in Table 4. Results of GaDE are taken from Table 1 in [204]. Results of SOUPDE are taken from Tables 15 and 17 in [196]. Results of GOjDE are taken from Table 9 in [185]. The comparison results between CCPSO2 and MLCC summarized in Table 5 based on two-tailed t-tests conducted in [99] among two algorithm on the CEC'08 benchmark test functions for $D = 1000$.

We provide a two-tailed t-test among the results of DECC-DG and four algorithms MA-SW-CHAINS, MLCC, DECC-D, and DECC-DML on CEC'2010 benchmark test functions. The results are taken from Tables VI and V in [129]. This information is summarized in Table 6.

5. Applications

Research works from several domains of science and engineering have been applying LSGO algorithms to solve optimization problems arising in their own fields. Table 7 summarizes several applications of LSGO.

6. Conclusions and future directions

The real world optimization problems often appear with high complexity and dimension, known as LSGO problems which have attracted much interest among research works from various fields. Over the last decade, large numbers of metaheuristic algorithms or their modifications have been developed to improve significantly the performance of the algorithm for tackling LSGO problems. In this paper, a general overview of research works to solve LSGO problems has been presented. It includes the advances in the decomposition and non-decomposition methods, benchmark functions, performance measures, major results, and some applications. The metaheuristic algorithm challenges to solve LSGO problems derive from various main aspects: (1) in many problems, the search space increases exponentially with the number of decision variables, (2) the features of an optimization problem may deteriorate with increase dimension of the problem, e.g. a unimodal function transforms into a multimodal function, and (3) the metaheuristic algorithms in handling with LSGO problems usually needs very large computational cost, (4) another measure of the complicate of LSGO problems is the non-separable properties of problem which the identification of the interacting variables is a main challenge especially in the CC methods. We describe below some gaps which have been identified and define future directions of research in the area of LSGO.

- The optimal decomposition: The major challenge of CC algorithms is grouping variables into some subcomponents near optimal decomposition to significantly improve their performance. Most of the current decomposition methods have focused on constructing the non-separable subcomponents. Much more effort is needed to fully develop decomposition methods with the high performance and great accuracy on both non-separable and separable subcomponents. Also, in most real-world problems, some imbalance will exist between various subcomponents so the strengths and weaknesses of these decomposition methods should be investigated thoroughly on imbalanced problems [100]. The design of new decomposition methods becomes an essential requirement for imbalanced problems.
- The CC Framework characteristics: Although a number of research works have been developed to design decomposition methods for CC algorithms, efficient approaches are still greatly needed to investigate other characteristics of CC algorithms. The collaboration method, choosing a representing solution for collaborative fitness assessment, dividing the computational budget among subcomponents according to their contributions, and investigating impact of the initial population and population size have a significant effect on the performance of CC algorithms. The theoretical studies on these characteristics are quite limited so far. Thus, it would be interesting to investigate how use other characteristics of CC algorithms to improve their performance.
- Fully separable or fully non-separable problems with a single non-separable group: The decomposition methods decompose all variables of these problems into one subcomponent therefore this subcomponent is still a LSGO problem and the performance of CC algorithms deteriorates on such problems. In fully separable, some decomposition methods can divide all variable into some subcomponents. The problem is how to design a decomposition method to automatically achieve a

suitable decomposition. The decomposition methods need to develop for such problems. In fully non-separable problems, it is desirable to design new algorithms with exploring capabilities for these problems. Some issues e.g., parallel implementations of well-known non-decomposition methods, hybrid between the non-decomposition methods and local search methods can be investigated in future.

- The imbalanced LSGO benchmark functions: The real-world optimization problems have an important characteristic, imbalance in the contribution of subcomponents, the CEC'2013 benchmark functions were developed to better represent this characteristic. The theoretical studies are still quite infant on the imbalanced LSGO benchmark functions and the CC algorithm and non-decomposition methods have not developed to solve them. Further research can be directed to extend the CC algorithm and non-decomposition methods for tackling the imbalanced LSGO functions.
- The real-world problems: Most of the LSGO methods have been evaluated only on LSGO benchmark test functions. There is a fundamental question: how these common LSGO benchmark test set and evaluation criteria reflect the characteristics of real-world problems. Researchers in the LSGO of field need to consider and model a real-world benchmark set, i.e., a challenging task, and apply the CC and non-decomposing methods to solve them in the future. Also, the studies should investigate the characteristics of real-world problems and show how they relate to the characteristics of current benchmark problems.
- The higher dimensional problems: Little efforts have been made to further improve the scalability of the LSGO methods for tackling LSGO benchmark test set with dimension greater than 1000. Scalability of LSGO methods becomes an essential requirement which is a great potential to future research works.

References

[1] http://nical.ustc.edu.cn/cec10ss.php.
[2] Morteza Alinia Ahandani, Hosein Alavi-Rad, Opposition-based learning in the shuffled differential evolution algorithm, Soft Comput. 16 (8) (2012) 1303–1337.
[3] Lifeng Ai, Maolin Tang, Colin Fidge, Resource allocation and scheduling of multiple composite web services in cloud computing using cooperative coevolution genetic algorithm, in: Neural Information Processing, Springer, 2011, pp. 258–267.
[4] Bahriye Akay, Dervis Karaboga, Artificial bee colony algorithm for large-scale problems and engineering design optimization, J. Intell. Manuf. 23 (4) (2012) 1001–1014.
[5] Luis Miguel Antonio, C.A. Coello Coello, Use of cooperative coevolution for solving large scale multiobjective optimization problems, in: 2013 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2013, pp. 2758–2765.
[6] Anne Auger, Nikolaus Hansen, A restart cma evolution strategy with increasing population size, The 2005 IEEE Congress on Evolutionary Computation, 2005, vol. 2, IEEE, 2005, pp. 1769–1776.
[7] Thomas Bäck, Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms, Oxford University Press, 1996.
[8] Olivier Barrière, Evelyne Lutton, Experimental analysis of a variable size mono-population cooperative-coevolution strategy, in: Nature Inspired Cooperative Strategies for Optimization (NICSO 2008), Springer, 2009, pp. 139–152.
[9] Y.M. Ben Ali, Soft adaptive particle swarm algorithm for large scale optimization, in: 2010 IEEE Fifth International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA), IEEE, 2010, pp. 1658–1662.
[10] Mauro Birattari, Tuning Metaheuristics: A Machine Learning Perspective, vol. 197, Springer, 2009.
[11] Mauro Birattari, Thomas Stützle, Luis Paquete, Klaus Varrentrapp, et al., A racing algorithm for configuring metaheuristics, in: GECCO, vol. 2. Citeseer, 2002, pp. 11–18.
[12] A. Bolufe-Rohler, Sheng Chen, Minimum population search-lessons from building a heuristic technique with two population members, in: 2013 IEEE Congress on Evolutionary Computation (CEC), IEE, 2013, pp. 2061–2068.
[13] A. Bolufe-Rohler, Sheng Chen, Extending minimum population search towards large scale global optimization, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 845–852.
[14] Kittipong Boonlong, Vibration-based damage detection in beams by cooperative coevolutionary genetic algorithm, Adv. Mech. Eng. 2014 (2014).
[15] Janez Brest, Ales Zamuda, Borko Boskovic, Mirjam Sepesy Maucec, Viljem Zumer, High-dimensional real-parameter optimization using self-adaptive differential evolution algorithm with population size reduction, in: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 2032–2039.
[16] S.P. Brooks, B.J.T. Morgan, Optimization using simulated annealing, The Statistician (1995) 241–257.
[17] Karan Kumar Budhraja, Ashutosh Singh, Gaurav Dubey, Arun Khosla, Exploration enhanced particle swarm optimization using guided re-initialization, in: Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012), Springer, 2013, pp. 403–416.
[18] Xian-Bin Cao, Hong Qiao, John Keane, A low-cost pedestrian-detection system with a single optical camera, IEEE Trans. Intell. Transport. Syst. 9 (1) (2008) 58–67.
[19] Andrea Caponio, Anna V. Kononova, Ferrante Neri, Differential evolution with scale factor local search for large scale problems, in: Computational Intelligence in Expensive Optimization Problems, Springer, 2010, pp. 297–323.
[20] Hanning Chen, Yunlong Zhu, Kunyuan Hu, Xiaoxian He, Ben Niu, Cooperative approaches to bacterial foraging optimization, in: Advanced Intelligent Computing Theories and Applications: With Aspects of Artificial Intelligence, Springer, 2008, pp. 541–548.
[21] Stephen Chen, An analysis of locust swarms on large scale global optimization problems, in: Artificial Life: Borrowing from Biology, Springer, 2009, pp. 211–220.
[22] Wenxiang Chen, Ke Tang, Impact of problem decomposition on cooperative coevolution, in: 2013 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2013, pp. 733–740.
[23] Wenxiang Chen, Thomas Weise, Zhenyu Yang, Ke Tang, Large-scale global optimization using cooperative coevolution with variable interaction learning, in: Parallel Problem Solving from Nature, PPSN XI, Springer, 2010, pp. 300–309.
[24] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Trans. Cybernet. PP (99) (2014). 1–1.
[25] Jaydeep Ghosh Chowdhury, Aritra Chowdhury, Arghya Sur, Large scale optimization based on co-ordinated bacterial dynamics and opposite numbers, in: Swarm, Evolutionary, and Memetic Computing, Springer, 2012, pp. 770–777.
[26] Wei Chu, Xiaogang Gao, Soroosh Sorooshian, Handling boundary constraints for particle swarm optimization in high-dimensional search space, Inform. Sci. 181 (20) (2011) 4569–4581.
[27] Wei Chu, Xiaogang Gao, Soroosh Sorooshian, A new evolutionary search strategy for global optimization of high-dimensional problems, Inform. Sci. 181 (22) (2011) 4909–4927.

[28] Wei Chu, Xiaogang Gao, Soroosh Sorooshian, A solution to the crucial problem of population degeneration in high-dimensional evolutionary optimization, IEEE Syst. J. 5 (3) (2011) 362–373.
[29] Ying Chu, Hua Mi, Huilian Liao, Zhen Ji, Q.H. Wu, A fast bacterial swarming algorithm for high-dimensional function optimization, in: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 3135–3140.
[30] Ciprian Crăciun, Monica Nicoară, Daniela Zaharie, Enhancing the scalability of metaheuristics by cooperative coevolution, in: Large-Scale Scientific Computing, Springer, 2010, pp. 310–317.
[31] Vincenzo Cutello, Natalio Krasnogor, Giuseppe Nicosia, Mario Pavone, Immune algorithm versus differential evolution: a comparative case study using high dimensional function optimization, in: Adaptive and Natural Computing Algorithms, Springer, 2007, pp. 93–101.
[32] Sambarta Dasgupta, Arijit Biswas, Swagatam Das, Bijaya K. Panigrahi, Ajith Abraham, A micro-bacterial foraging algorithm for high-dimensional optimization, in: IEEE Congress on Evolutionary Computation, 2009. CEC'09, IEEE, 2009, pp. 785–792.
[33] Antonio LaTorre de la Fuente, José María Peña Sánchez, A framework for hybrid dynamic evolutionary algorithms: multiple offspring sampling (mos), 2009.
[34] Marco A. Montes de Oca, Incremental social learning in swarm intelligence algorithms for continuous optimization, in: Computational Intelligence, Springer, 2013, pp. 31–45.
[35] Marco A. Montes De Oca, Ken Van den Enden, Thomas Stützle, Incremental particle swarm-guided local search for continuous optimization, in: Hybrid Metaheuristics, Springer, 2008, pp. 72–86.
[36] Marco A. Montes de Oca, Doğan Aydın, Thomas Stützle, An incremental particle swarm for large-scale continuous optimization problems: an example of tuning-in-the-loop (re)design of optimization algorithms, Soft Comput. 15 (11) (2011) 2233–2255.
[37] Marco Antonio Montes de Oca, Thomas Stutzle, Ken Van den Enden, Marco Dorigo, Incremental social learning in particle swarms, IEEE Trans. Syst. Man Cybernet. Part B: Cybernet. 41 (2) (2011) 368–384.
[38] Weishan Dong, Tianshi Chen, Peter Tino, Xin Yao, Scaling up estimation of distribution algorithms for continuous optimization, IEEE Trans. Evol. Comput. 17 (6) (2013) 797–822.
[39] Marco Dorigo, Ant colony optimization: a new meta-heuristic, in: Proceedings of the Congress on Evolutionary Computation, IEEE Press, 1999, pp. 1470–1477.
[40] Marco Dorigo, Vittorio Maniezzo, Alberto Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Trans. Syst. Man Cybernet. Part B: Cybernet. 26 (1) (1996) 29–41.
[41] Qingyun Duan, Soroosh Sorooshian, Vijai Gupta, Effective and efficient global optimization for conceptual rainfall-runoff models, Water Resour. Res. 28 (4) (1992) 1015–1031.
[42] Q.Y. Duan, Vijai K. Gupta, Soroosh Sorooshian, Shuffled complex evolution approach for effective and efficient global minimization, J. Optim. Theory Appl. 76 (3) (1993) 501–521.
[43] Abraham Duarte, Rafael Martí, Fred Glover, An adaptive memory procedure for continuous optimization, in: Ninth International Conference on Intelligent Systems Design and Applications, 2009. ISDA'09, IEEE, 2009, pp. 1085–1089.
[44] Mohammed El-Abd, A cooperative approach to the artificial bee colony algorithm, in: 2010 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2010, pp. 1–5.
[45] Larry J. Eshelman, J. David Schaffer, Real-coded genetic algorithms and interval-schemata, in: L. Darrell Whitley (Ed.), FOGA, Morgan Kaufmann, 1992, pp. 187–202. ISBN: 1-55860-263-1. <http://dblp.uni-trier.de/db/conf/foga/foga1992.html#EshelmanS92>.
[46] Ali Esmailzadeh, Shahryar Rahnamayan, Enhanced differential evolution using center-based sampling, in: 2011 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2011, pp. 2641–2648.
[47] Ali Esmailzadeh, Shahryar Rahnamayan, Center-point-based simulated annealing, in: 2012 25th IEEE Canadian Conference on Electrical & Computer Engineering (CCECE), IEEE, 2012, pp. 1–4.
[48] J. Fan, J. Wang, M. Han, Cooperative coevolution for large-scale optimization based on kernel fuzzy clustering and variable trust region methods, IEEE Trans. Fuzzy Syst. 22 (4) (2014) 829–839. ISSN: 1063-6706.
[49] X.Z. Gao, X. Wang, S.J. Ovaska, K. Zenger, A hybrid optimization method of harmony search and opposition-based learning, Eng. Optim. 44 (8) (2012) 895–914.
[50] Yu Gao, Yong-Jun Wang, A memetic differential evolutionary algorithm for high dimensional functions' optimization, Third International Conference on Natural Computation, 2007. ICNC 2007, vol. 4, IEEE, 2007, pp. 188–192.
[51] Carlos García-Martínez, Manuel Lozano, Continuous variable neighbourhood search algorithm based on evolutionary metaheuristic components: a scalability test, in: Ninth International Conference on Intelligent Systems Design and Applications, 2009. ISDA'09, IEEE, 2009, pp. 1074–1079.
[52] Carlos García-Martínez, Francisco J. Rodríguez, Manuel Lozano, Role differentiation and malleable mating for differential evolution: an analysis on large-scale optimisation, Soft Comput. 15 (11) (2011) 2109–2126.
[53] José García-Nieto, Enrique Alba, Restart particle swarm optimization with velocity modulation: a scalability test, Soft Comput. 15 (11) (2011) 2221–2232.
[54] Nicolás García-Pedrajas, César Hervás-Martínez, José Muñoz-Pérez, Covnet: a cooperative coevolutionary model for evolving artificial neural networks, IEEE Trans. Neural Netw. 14 (3) (2003) 575–596.
[55] Vincent Gardeux, Rachid Chelouah, Patrick Siarry, Fred Glover, Unidimensional search for solving continuous high-dimensional optimization problems, in: Ninth International Conference on Intelligent Systems Design and Applications, 2009. ISDA'09, IEEE, 2009, pp. 1096–1101.
[56] Vincent Gardeux, Rachid Chelouah, Patrick Siarry, Fred Glover, Em323: a line search based algorithm for solving high-dimensional continuous non-linear optimization problems, Soft Comput. 15 (11) (2011) 2275–2285.
[57] Fred Glover, The 3-2-3, stratified split and nested interval line search algorithms, Technical report, Research report, OptTek Systems, Boulder, CO, 2010.
[58] David E. Goldberg, John H. Holland, Genetic algorithms and machine learning, Machine Learn. 3 (2) (1988) 95–99.
[59] Jonatan Gomez, Elizabeth Leon, A coevolutionary chromosome encoding scheme for high dimensional search spaces, in: 2010 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2010, pp. 1–8.
[60] Jonatan Gómez, Arturo Garcia, Camilo Silva, Cofre: a fuzzy rule coevolutionary approach for multiclass classification problems, The 2005 IEEE Congress on Evolutionary Computation, 2005, vol. 2, IEEE, 2005, pp. 1637–1644.
[61] N. Hansen, Cma evolution strategy source code, <https://www.lri.fr/hansen/cmaes_inmatlab.html>.
[62] Toshiharu Hatanaka, Takeshi Korenaga, Nobuhiko Kondo, Katsuji Uosaki, Search performance improvement for pso in high dimensional space, Particle Swarm Optim. (2009) 249–260.
[63] A-R. Hedar, Ahmed Fouad Ali, Genetic algorithm with population partitioning and space reduction for high dimensional problems, in: International Conference on Computer Engineering & Systems, 2009. ICCES 2009, IEEE, 2009, pp. 151–156.
[64] Abdel-Rahman Hedar, Ahmed Fouad Ali, Tabu search with multi-level neighborhood structures for high dimensional problems, Appl. Intell. 37 (2) (2012) 189–206.
[65] Abdel-Rahman Hedar, Bun Theang Ong, Masao Fukushima, Genetic algorithms with automatic accelerated termination, Department of Applied Mathematics and Physics, Kyoto University, Tech. Rep, 2, 2007.
[66] F. Herrera, M. Lozano, D. Molina, Test suite for the special issue of soft computing on scalability of evolutionary algorithms and other metaheuristics for large scale continuous optimization problems, Last accessed: July, 2010.
[67] Xiao Hongfeng, Tan Guanzheng, Huang Jingui, Large scale function optimization or high-dimension function optimization in large using simplex-based genetic algorithm, in: Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation, ACM, 2009, pp. 209–216.

[68] Sheng-Ta Hsieh, Tsung-Ying Sun, Chan-Cheng Liu, Shang-Jeng Tsai, Solving large scale global optimization using improved particle swarm optimizer, in: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 1777–1784.
[69] Qiang Huang, Thomas White, Guanbo Jia, Mirco Musolesi, Nil Turan, Ke Tang, Shan He, John K. Heath, Xin Yao, Community detection using cooperative co-evolutionary differential evolution, in: Parallel Problem Solving from Nature-PPSN XII, Springer, 2012, pp. 235–244.
[70] Lars Magnus Hvattum, Fred Glover, Finding local optima of high-dimensional functions using direct search methods, Eur. J. Oper. Res. 195 (1) (2009) 31–45.
[71] Antony W. Iorio, Xiaodong Li, Improving the performance and scalability of differential evolution, in: Simulated Evolution and Learning, Springer, 2008, pp. 131–140.
[72] Adiel Ismail, Andries P. Engelbrecht, Measuring diversity in the cooperative particle swarm optimizer, in: Swarm Intelligence, Springer, 2012, pp. 97–108.
[73] Bo Jiang, Ning Wang, Cooperative bare-bone particle swarm optimization for data clustering, Soft Comput. 18 (6) (2014) 1079–1091.
[74] Bin Jiao, Qunxian Chen, Shaobin Yan, A cooperative co-evolution pso for flow shop scheduling problem with uncertainty, J. Comput. 6 (9) (2011) 1955–1961.
[75] Dervis Karaboga, An idea based on honey bee swarm for numerical optimization, Technical report, Technical report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.
[76] Borhan Kazimipour, Xiaodong Li, A.K. Qin, Initialization methods for large scale global optimization, in: 2013 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2013, pp. 2750–2757.
[77] Borhan Kazimipour, Xiaodong Li, A.K. Qin, Effects of population initialization on differential evolution for large scale optimization, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEE, 2014, pp. 2404–2411.
[78] Borhan Kazimipour, Xiaodong Li, A.K. Qin, A review of population initialization techniques for evolutionary algorithms, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 2585–2592.
[79] Borhan Kazimipour, Mohammad Nabi Omidvar, Xiaodong Li, A.K. Qin, A novel hybridization of opposition-based learning and cooperative co-evolutionary for large-scale optimization, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 2833–2840.
[80] James Kennedy, Russell Eberhart, et al., Particle swarm optimization, in: Proceedings of IEEE International Conference on Neural Networks, vol. 4. Perth, Australia, 1995, pp. 1942–1948.
[81] James F. Kennedy, James Kennedy, Russell C. Eberhart, Swarm Intelligence, Morgan Kaufmann, 2001.
[82] Shuhei Kimura, Kaori Ide, Aiko Kashihara, Makoto Kano, Mariko Hatakeyama, Ryoji Masui, Noriko Nakagawa, Shigeyuki Yokoyama, Seiki Kuramitsu, Akihiko Konagaya, Inference of s-system models of genetic networks using a cooperative coevolutionary algorithm, Bioinformatics 21 (7) (2005) 1154–1163.
[83] William S Klug, Michael R Cummings, et al, Concepts of Genetics, seventh ed., Pearson Education, Inc., 2003.
[84] Byung-Il Koh, Jeffrey A. Reinbolt, Alan D. George, Raphael T. Haftka, Benjamin J. Fregly, Limitations of parallel global optimization for large-scale human movement problems, Med. Eng. Phys. 31 (5) (2009) 515–521.
[85] Anna V. Kononova, Derek B. Ingham, Mohamed Pourkashanian, Simple scheduled memetic algorithm for inverse problems in higher dimensions: application to chemical kinetics, in: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 3905–3912.
[86] Peter Korošec, Jurij Šilc, The Differential Ant-stigmergy Algorithm for Large Scale Real-parameter Optimization, Springer, 2008.
[87] Peter Korosec, Jurij Silc, Applications of the differential ant-stigmergy algorithm on real-world continuous optimization problems, 2009.
[88] Peter Korošec, Jurij Šilc, High-dimensional real-parameter optimization using the differential ant-stigmergy algorithm, Int. J. Intell. Comput. Cybernet. 2 (1) (2009) 34–51.
[89] Peter Korosec, Katerina Tashkova, Jurij Silc, The differential ant-stigmergy algorithm for large-scale global optimization, in: 2010 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2010, pp. 1–8.
[90] Peter Korošec, Jurij Šilc, Bogdan Filipič, The differential ant-stigmergy algorithm, Inform. Sci. 192 (2012) 82–97.
[91] Pedro Larrañaga, Jose A. Lozano, Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, vol. 2, Springer, 2002.
[92] Antonio LaTorre, Santiago Muelas, José-María Peña, A mos-based dynamic memetic differential evolution algorithm for continuous optimization: a scalability test, Soft Comput. 15 (11) (2011) 2187–2199.
[93] Antonio LaTorre, Santiago Muelas, J-M Pena, Multiple offspring sampling in large scale global optimization, in: 2012 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2012, pp. 1–8.
[94] Antonio LaTorre, Santiago Muelas, J.-M. Pena, Large scale global optimization: experimental results with mos-based hybrid algorithms, in: 2013 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2013, pp. 2742–2749.
[95] Wei-Po Lee, Yu-Ting Hsiao, Inferring gene regulatory networks using a hybrid ga–pso approach with numerical constraints and network decomposition, Inform. Sci. 188 (2012) 80–99.
[96] Guillermo Leguizamón, CA Coello Coello, A study of the scalability of acor for continuous optimization problems, Departamento de Computación, CINVESTAV-IPN, Mexico, Mexico, Tech. Rep. EVOCINV-01-2010, 2010.
[97] Y.-W. Leung, Yuping Wang, An orthogonal genetic algorithm with quantization for global numerical optimization, IEEE Trans. Evol. Comput. 5 (1) (2001) 41–53.
[98] Xiaodong Li, Xin Yao, Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms, in: IEEE Congress on Evolutionary Computation, 2009. CEC'09, IEEE, 2009, pp. 1546–1553.
[99] Xiaodong Li, Xin Yao, Cooperatively coevolving particle swarms for large scale optimization, IEEE Trans. Evol. Comput. 16 (2) (2012) 210–224.
[100] Xiaodong Li, Ke Tang, Mohammad N. Omidvar, Zhenyu Yang, Kai Qin, Hefei China, Benchmark functions for the cec 2013 special session and competition on large-scale global optimization, Gene 7 (2013) 33.
[101] Tianjun Liao, Marco A Montes de Oca, Dogan Aydin, Thomas Stützle, Marco Dorigo, An incremental ant colony algorithm with local search for continuous optimization, in: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, ACM, 2011, pp. 125–132.
[102] Lin Lin, Mitsuo Gen, Yan Liang, A hybrid ea for high-dimensional subspace clustering problem, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 2855–2860.
[103] Can Liu, Bin Li, Memetic algorithm with adaptive local search depth for large scale global optimization, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 82–88.
[104] Jinpeng Liu, Ke Tang, Scaling up covariance matrix adaptation evolution strategy using cooperative coevolution, in: Intelligent Data Engineering and Automated Learning–IDEAL 2013, Springer, 2013, pp. 350–357.
[105] Li-Zhi Liu, Fang-Xiang Wu, Wen-Jun Zhang, Inference of biological s-system using the separable estimation method and the genetic algorithm, IEEE/ACM Trans. Comput. Biol. Bioinform. (TCBB) 9 (4) (2012) 955–965.
[106] Yong Liu, Xin Yao, Qiangfu Zhao, Tetsuya Higuchi, Scaling up fast evolutionary programming with cooperative coevolution, Proceedings of the 2001 Congress on Evolutionary Computation, 2001, vol. 2, IEEE, 2001, pp. 1101–1108.
[107] Carlos B. Lucasius, Gerrit Kateman, Genetic algorithms for large-scale optimization in chemometrics: an application, TrAC Trends Anal. Chem. 10 (8) (1991) 254–261.
[108] Sean Luke, Keith Sullivan, Faisal Abidi, Large scale empirical analysis of cooperative coevolution, in: Proceedings of the 13th Annual Conference Companion on Genetic and Evolutionary Computation, ACM, 2011, pp. 151–152.
[109] Sean Luke et al, Genetic programming produced competitive soccer softbot teams for robocup97, Genet. Program. 1998 (1998) 214–222.
[110] Cara MacNish, Xin Yao, Direction matters in high-dimensional optimisation, in: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 2372–2379.

[111] Sedigheh Mahdavi, Mohammad Ebrahim Shiri, Shahryar Rahnamayan, Cooperative co-evolution with a new decomposition method for large-scale optimization, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 1285–1292.
[112] Elena Marchiori, Adri Steenbeek, An evolutionary algorithm for large scale set covering problems with application to airline crew scheduling, in: Real-World Applications of Evolutionary Computing, Springer, 2000, pp. 370–384.
[113] Antonio David Masegosa, David Alejandro Pelta, José Luis Verdegay, A centralised cooperative strategy for continuous optimisation: the influence of cooperation in performance and behaviour, Inform. Sci. 219 (2013) 73–92.
[114] Yi Mei, Xiaodong Li, Xin Yao, Variable neighborhood decomposition for large scale capacitated arc routing problem, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 1313–1320.
[115] Yi Mei, Xiaodong Li, Xin Yao, Cooperative coevolution with route distance grouping for large-scale capacitated arc routing problems, IEEE Trans. Evol. Comput. 18 (3) (2014) 435–449. ISSN: 1089-778X.
[116] Xiao Mingming, Zhang Jun, Cai Kaiquan, Cao Xianbin, Tang Ke, Cooperative co-evolution with weighted random grouping for large-scale crossing waypoints locating in air route network, in: 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI), 2011, IEEE, 2011, pp. 215–222.
[117] Daniel Molina, Manuel Lozano, Francisco Herrera, Memetic algorithm with local search chaining for large scale continuous optimization problems, in: IEEE Congress on Evolutionary Computation, 2009. CEC'09, IEEE, 2009, pp. 830–837.
[118] Daniel Molina, Manuel Lozano, Francisco Herrera, Ma-sw-chains: memetic algorithm based on local search chains for large scale continuous global optimization, in: 2010 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2010, pp. 1–8.
[119] Daniel Molina, Manuel Lozano, Ana M Sánchez, Francisco Herrera, Memetic algorithms based on local search chains for large scale continuous optimisation problems: Ma-ssw-chains, Soft Comput. 15 (11) (2011) 2201–2220.
[120] Marco Montes de Oca, Thomas Stutzle, Ken Van den Enden, Marco Dorigo, Incremental social learning in particle swarms, Technical Report TR/IRIDIA/2009-002, IRIDIA, Facult des Sciences Appliques, Universit Libre de Bruxelles, 2009.
[121] Damon Mosk-Aoyama, Devavrat Shah, Fast distributed algorithms for computing separable functions, IEEE Trans. Inform. Theory 54 (7) (2008) 2997–3007.
[122] Heinz Mühlenbein, Jürgen Bendisch, H.-M. Voigt, From recombination of genes to the estimation of distributions ii. Continuous parameters, in: Parallel Problem Solving from Nature PPSN IV, Springer, 1996, pp. 188–197.
[123] John A. Nelder, Roger Mead, A simplex method for function minimization, The Comput. J. 7 (4) (1965) 308–313.
[124] Barrière Olivier, Evelyne Lutton, Pierre-Henri Wuillemin, Cédric Baudrit, Mariette Sicard, Bruno Pinaud, Nathalie Perrot, et al., Modeling an agrifood industrial process using cooperative coevolution algorithms, 2009.
[125] Mohammad Nabi Omidvar, Xiaodong Li, A comparative study of cma-es on large scale global optimisation, in: AI 2010: Advances in Artificial Intelligence, Springer, 2011, pp. 303–312.
[126] Mohammad Nabi Omidvar, Xiaodong Li, Zhenyu Yang, Xin Yao, Cooperative co-evolution for large scale optimization through more frequent random grouping, in: 2010 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2010, pp. 1–8.
[127] Mohammad Nabi Omidvar, Xiaodong Li, Xin Yao, Cooperative co-evolution with delta grouping for large scale non-separable function optimization, in: 2010 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2010, pp. 1–8.
[128] Mohammad Nabi Omidvar, Xiaodong Li, Xin Yao, Smart use of computational resources based on contribution for cooperative co-evolutionary algorithms, in: Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, ACM, 2011, pp. 1115–1122.
[129] Mohammad Nabi Omidvar, Xiaodong Li, Yi Mei, Xin Yao, Cooperative co-evolution with differential grouping for large scale optimization, IEEE Trans. Evol. Comput. 18 (3) (2014) 378–393.
[130] Mohammad Nabi Omidvar, Yi Mei, Xiaodong Li, Effective decomposition of large-scale separable continuous functions for cooperative co-evolutionary algorithms, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 1305–1312.
[131] Xiuqin Pan, Yue Zhao, Xiaona Xu, Adaptive differential evolution with local search for solving large-scale optimization problems, 2012.
[132] Liviu Panait, Theoretical convergence guarantees for cooperative coevolutionary algorithms, Evol. Comput. 18 (4) (2010) 581–615.
[133] Liviu Panait, Sean Luke, Cooperative multi-agent learning: the state of the art, Autonomous Agents Multi-Agent Syst. 11 (3) (2005) 387–434.
[134] Liviu Panait, R Paul Wiegand, Sean Luke, A sensitivity analysis of a cooperative coevolutionary algorithm biased for optimization, in: Genetic and Evolutionary Computation–GECCO 2004, Springer, 2004, pp. 573–584.
[135] Liviu Panait, Sean Luke, R. Paul Wiegand, Biasing coevolutionary search for optimal multiagent behaviors, IEEE Trans. Evol. Comput. 10 (6) (2006) 629–645.
[136] Adam P. Piotrowski, Jaroslaw J. Napiorkowski, Adam Kiczko, Differential evolution algorithm with separated groups for multi-dimensional optimization problems, Eur. J. Oper. Res. 216 (1) (2012) 33–46.
[137] Elena Popovici, Kenneth De Jong, Sequential versus parallel cooperative coevolutionary algorithms for optimization, in: IEEE Congress on Evolutionary Computation, 2006. CEC 2006, IEEE, 2006, pp. 1610–1617.
[138] Mitchell A. Potter, The design and analysis of a computational model of cooperative coevolution, PhD thesis, Citeseer, 1997.
[139] Mitchell A. Potter, Kenneth A. De Jong, A cooperative coevolutionary approach to function optimization, in: Parallel Problem Solving from Nature PPSN III, Springer, 1994, pp. 249–257.
[140] Mitchell A. Potter, Kenneth A. De Jong, Cooperative coevolution: an architecture for evolving coadapted subcomponents, Evol. Comput. 8 (1) (2000) 1–29.
[141] Kenneth V. Price, An introduction to differential evolution, in: New Ideas in Optimization, McGraw-Hill Ltd., UK, 1999, pp. 79–108.
[142] Shahryar Rahnamayan, G. Gary Wang, Investigating in scalability of opposition-based differential evolution, WSEAS Trans. Comput. 7 (2008) 1792–1804.
[143] Shahryar Rahnamayan, G. Gary Wang, Solving large scale optimization problems by opposition-based differential evolution (ode), WSEAS Trans. Comput. 7 (10) (2008) 1792–1804.
[144] Shahryar Rahnamayan, G. Gary Wang, Center-based sampling for population-based algorithms, in: IEEE Congress on Evolutionary Computation, 2009. CEC'09, IEEE, 2009, pp. 933–938.
[145] Shahryar Rahnamayan, Hamid R. Tizhoosh, Magdy M.A. Salama, Opposition-based differential evolution algorithms, in: IEEE Congress on Evolutionary Computation, 2006. CEC 2006, IEEE, 2006, pp. 2010–2017.
[146] Shahryar Rahnamayan, Hamid R. Tizhoosh, Magdy Salama, A novel population initialization method for accelerating evolutionary algorithms, Comput. Math. Appl. 53 (10) (2007) 1605–1614.
[147] Shahryar Rahnamayan, Hamid R. Tizhoosh, Magdy M.A. Salama, Opposition-based differential evolution (ode) with variable jumping rate, in: IEEE Symposium on Foundations of Computational Intelligence, 2007. FOCI 2007, IEEE, 2007, pp. 81–88.
[148] Shahryar Rahnamayan, Hamid R. Tizhoosh, Magdy M.A. Salama, Opposition-based differential evolution, IEEE Trans. Evol. Comput. 12 (1) (2008) 64–79.
[149] Tapabrata Ray, Xin Yao, A cooperative coevolutionary algorithm with correlation based adaptive variable partitioning, in: IEEE Congress on Evolutionary Computation, 2009. CEC'09, IEEE, 2009, pp. 983–989.
[150] Yuanfang Ren, Yan Wu, An efficient algorithm for high-dimensional function optimization, Soft Comput. 17 (6) (2013) 995–1004.
[151] Yazmin Rojas, Ricardo Landa, Towards the use of statistical information and differential evolution for large scale global optimization, in: 2011 8th International Conference on Electrical Engineering Computing Science and Automatic Control (CCE), IEEE, 2011, pp. 1–6.
[152] Raymond Ros, Nikolaus Hansen, A simple modification in cma-es achieving linear time and space complexity, in: Parallel Problem Solving from Nature–PPSN X, Springer, 2008, pp. 296–305.

[153] Ralf Salomon, Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions: a survey of some theoretical and practical aspects of genetic algorithms, BioSystems 39 (3) (1996) 263–278.
[154] Eman Sayed, Daryl Essam, Ruhul Sarker, Dependency identification technique for large scale optimization problems, in: 2012 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2012, pp. 1–8.
[155] Eman Sayed, Daryl Essam, Ruhul Sarker, Using hybrid dependency identification with a memetic algorithm for large scale optimization problems, in: Simulated Evolution and Learning, Springer, 2012, pp. 168–177.
[156] Songqing Shan, G. Gary Wang, Metamodeling for high dimensional simulation-based design problems, J. Mech. Des. 132 (5) (2010) 051009.
[157] Yan-jun Shi, Hong-fei Teng, Zi-qiang Li, Cooperative co-evolutionary differential evolution for function optimization, in: Advances in Natural Computation, Springer, 2005, pp. 1080–1088.
[158] Hemant Kumar Singh, Tapabrata Ray, Divide and conquer in coevolution: a difficult balancing act, in: Agent-Based Evolutionary Search, Springer, 2010, pp. 117–138.
[159] Krzysztof Socha, Marco Dorigo, Ant colony optimization for continuous domains, Eur. J. Oper. Res. 185 (3) (2008) 1155–1173.
[160] Francisco J. Solis, Roger J.-B. Wets, Minimization by random search techniques, Math. Oper. Res. 6 (1) (1981) 19–30.
[161] Rainer Storn, Kenneth Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359.
[162] Matthew J. Streeter, Upper bounds on the time and space complexity of optimizing additively separable functions, in: Genetic and Evolutionary Computation–GECCO 2004, Springer, 2004, pp. 186–197.
[163] Jianyong Sun, Jonathan M. Garibaldi, Charlie Hodgman, Parameter estimation using metaheuristics in systems biology: a comprehensive review, IEEE/ACM Trans. Comput. Biol. Bioinformat. 9 (1) (2012) 185–202.
[164] Liang Sun, Shinichi Yoshida, Xiaochun Cheng, Yanchun Liang, A cooperative particle swarm optimizer with statistical variable interdependence learning, Inform. Sci. 186 (1) (2012) 20–39.
[165] Tetsuyuki Takahama, Setsuko Sakai, Large scale optimization by differential evolution with landscape modality detection and a diversity archive, in: IEEE Congress on Evolutionary Computation (CEC), 2012, IEEE, 2012, pp. 1–8.
[166] Ke Tang, Xın Yáo, Ponnuthurai Nagaratnam Suganthan, Cara MacNish, Ying-Ping Chen, Chih-Ming Chen, Zhenyu Yang, Benchmark functions for the cec2008 special session and competition on large scale global optimization, Nature Inspired Computation and Applications Laboratory, USTC, China, 2007.
[167] Ke Tang, Xiaodong Li, Ponnuthurai Nagaratnam Suganthan, Zhenyu Yang, Thomas Weise, Benchmark Functions for the CEC'2010 Special Session and Competition on Large-Scale Global Optimization, Technical report, Nature Inspired Computation and Applications Laboratory (NICAL), USTC, China, 2010. <http://www.it-weise.de/documents/files/TLSYW2009BFFTCSSACOLSGO.pdf>.
[168] Yoel Tenne, Chi-Keong Goh, Computational Intelligence in Expensive Optimization Problems, vol. 2, Springer, 2010.
[169] Masaru Tezuka, Masaharu Munetomo, Kiyoshi Akama, Linkage identification by nonlinearity check for real-coded genetic algorithms, in: Genetic and Evolutionary Computation–GECCO 2004, Springer, 2004, pp. 222–233.
[170] Hamid R. Tizhoosh, Opposition-based learning: a new scheme for machine intelligence, in: CIMCA/IAWTIC, 2005, pp. 695–701.
[171] Alberto Tonda, Evelyne Lutton, Giovanni Squillero, A benchmark for cooperative coevolution, Memetic Comput. 4 (4) (2012) 263–277.
[172] Lin-Yu Tseng, Chun Chen, Multiple trajectory search for multiobjective optimization, in: IEEE Congress on Evolutionary Computation, 2007. CEC 2007, IEEE, 2007, pp. 3609–3616.
[173] Lin-Yu Tseng, Chun Chen, Multiple trajectory search for large scale global optimization, in: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 3052–3059.
[174] Frans Van den Bergh, Andries Petrus Engelbrecht, A cooperative approach to particle swarm optimization, IEEE Trans. Evol. Comput. 8 (3) (2004) 225–239.
[175] Alejandro F Villaverde, Jose A Egea, Julio R Banga, A cooperative strategy for parameter estimation in large scale systems biology models, BMC Syst. Biol. 6 (1) (2012) 75.
[176] Chi Cuong Vu, Huu Hung Nguyen, Lam Thu Bui, A parallel cooperative coevolution evolutionary algorithm, in: Third International Conference on Knowledge and Systems Engineering (KSE), 2011, IEEE, 2011, pp. 48–53.
[177] Chao Wang, J-H Gao, A differential evolution algorithm with cooperative coevolutionary selection operation for high-dimensional optimization, Optim. Lett. (2012) 1–16.
[178] Chao Wang, Jinghuai Gao, A new differential evolution algorithm with cooperative coevolutionary selection operator for waveform inversion, in: Geoscience and Remote Sensing Symposium (IGARSS), 2010 IEEE International, IEEE, 2010, pp. 688–690.
[179] Chao Wang, Jinghuai Gao, High-dimensional waveform inversion with cooperative coevolutionary differential evolution algorithm, Geosci. Remote Sensing Lett., IEEE 9 (2) (2012) 297–301.
[180] Hui Wang, Yong Liu, Sanyou Zeng, A hybrid particle swarm algorithm with cauchy mutation, in: IEEE Swarm Intelligence Symposium, 2007. SIS 2007, IEEE, 2007, pp. 356–360.
[181] Hui Wang, Zhijian Wu, Yong Liu, Jing Wang, Dazhi Jiang, Lili Chen, Space transformation search: a new evolutionary technique, in: Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation, ACM, 2009, pp. 537–544.
[182] Hui Wang, Zhijian Wu, Shahryar Rahnamayan, Lishan Kang, A scalability test for accelerated de using generalized opposition-based learning, in: Ninth International Conference on Intelligent Systems Design and Applications, ISDA'09 2009, IEEE, 2009, pp. 1090–1095.
[183] Hui Wang, Zhijian Wu, Shahryar Rahnamayan, Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems, Soft Comput. 15 (11) (2011) 2127–2140.
[184] Hui Wang, Zhijian Wu, Shahryar Rahnamayan, Yong Liu, Mario Ventresca, Enhancing particle swarm optimization using generalized opposition-based learning, Inform. Sci. 181 (20) (2011) 4699–4714.
[185] Hui Wang, Shahryar Rahnamayan, Zhijian Wu, Parallel differential evolution with self-adapting control parameters and generalized opposition-based learning for solving high-dimensional optimization problems, J. Parallel Distrib. Comput. 73 (1) (2013) 62–73.
[186] Hui Wang, Hui Sun, Changhe Li, Shahryar Rahnamayan, Jeng-shyang Pan, Diversity enhanced particle swarm optimization with neighborhood search, Inform. Sci. 223 (2013) 119–135.
[187] J. Wang, J. Liao, Y. Zhou, Y. Cai, Differential evolution enhanced with multiobjective sorting-based mutation operators, 2014.
[188] Yong Wang, Zixing Cai, Qingfu Zhang, Enhancing the search ability of differential evolution through orthogonal crossover, Inform. Sci. 185 (1) (2012) 153–177.
[189] Yu Wang, Bin Li, A restart univariate estimation of distribution algorithm: sampling under mixed gaussian and lévy probability distribution, in: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 3917–3924.
[190] Yu Wang, Bin Li, A self-adaptive mixed distribution based uni-variate estimation of distribution algorithm for large scale global optimization, in: Nature-Inspired Algorithms for Optimisation, Springer, 2009, pp. 171–198.
[191] Yu Wang, Bin Li, Two-stage based ensemble optimization for large-scale global optimization, in: IEEE Congress on Evolutionary Computation (CEC), 2010, IEEE, 2010, pp. 1–8.
[192] Yu Wang, Bin Li, Xuexiao Lai, Variance priority based cooperative co-evolution differential evolution for large scale global optimization, in: IEEE Congress on Evolutionary Computation, 2009. CEC'09, IEEE, 2009, pp. 1232–1239.
[193] Yu Wang, Bin Li, Thomas Weise, Estimation of distribution and differential evolution cooperation for large scale economic load dispatch optimization of power systems, Inform. Sci. 180 (12) (2010) 2405–2420.
[194] Yu Wang, Bin Li, Zhen He, Enhancing differential evolution with effective evolutionary local search in memetic framework, in: IEEE Congress on Evolutionary Computation (CEC), 2011, IEEE, 2011, pp. 2457–2464.

[195] Yu Wang, Jin Huang, Wei Shan Dong, Jun Chi Yan, Chun Hua Tian, Min Li, Wen Ting Mo, Two-stage based ensemble optimization framework for large-scale global optimization, Eur. J. Oper. Res. 228 (2) (2013) 308–320.
[196] Matthieu Weber, Ferrante Neri, Ville Tirronen, Shuffle or update parallel differential evolution for large-scale optimization, Soft Comput. 15 (11) (2011) 2089–2107.
[197] Karsten Weicker, Nicole Weicker, On the improvement of coevolutionary optimizers by learning variable interdependencies, Proceedings of the 1999 Congress on Evolutionary Computation, 1999. CEC 99, vol. 3, IEEE, 1999.
[198] YOU Xuemei, Differential evolution with a new mutation operator for solving high dimensional continuous optimization problems, J. Comput. Inform. Syst. 6 (9) (2010) 3033–3039.
[199] Zhenyu Yang, Ke Tang, Xin Yao, Differential evolution for high-dimensional function optimization, in: IEEE Congress on Evolutionary Computation, 2007. CEC 2007, IEEE, 2007, pp. 3523–3530.
[200] Zhenyu Yang, Ke Tang, Xin Yao, Large scale evolutionary optimization using cooperative coevolution, Inform. Sci. 178 (15) (2008) 2985–2999.
[201] Zhenyu Yang, Ke Tang, Xin Yao, Multilevel cooperative coevolution for large scale optimization, in: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 1663–1670.
[202] Zhenyu Yang, Ke Tang, Xin Yao, Self-adaptive differential evolution with neighborhood search, in: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 1110–1116.
[203] Zhenyu Yang, Jingqiao Zhang, Ke Tang, Xin Yao, Arthur C. Sanderson, An adaptive coevolutionary differential evolution algorithm for large-scale optimization, in: IEEE Congress on Evolutionary Computation, 2009. CEC'09, IEEE, 2009, pp. 102–109.
[204] Zhenyu Yang, Ke Tang, Xin Yao, Scalability of generalized adaptive differential evolution for large-scale continuous optimization, Soft Comput. 15 (11) (2011) 2141–2155.
[205] Zhenyu Yang, Xiaoli Li, Chris P Bowers, Thorsten Schnier, Ke Tang, Xin Yao, An efficient evolutionary approach to parameter identification in a building thermal model, IEEE Trans. Syst. Man Cybernet. Part C: Appl. Rev. 42 (6) (2012) 957–969.
[206] Sishi Ye, Guangming Dai, Lei Peng, Maocai Wang, Effective decomposition of large-scale separable continuous functions for cooperative co-evolutionary algorithms, in: IEEE Congress on Evolutionary Computation (CEC), 2014, IEEE, 2014, pp. 1305–1312.
[207] Huizhi Yi, Qinglin Duan, T. Warren Liao, Three improved hybrid metaheuristic algorithms for engineering design optimization, Appl. Soft Comput. 13 (5) (2013) 2433–2444.
[208] Ales Zamuda, Janez Brest, Borko Boskovic, Viljem Zumer, Large scale global optimization using differential evolution with self-adaptation and cooperative co-evolution, in: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 3718–3725.
[209] Aleš Zamuda, Janez Brest, Borko Bošković, Viljem Žumer, Differential evolution for parameterized procedural woody plant models reconstruction, Appl. Soft Comput. 11 (8) (2011) 4904–4912.
[210] Jingqiao Zhang, Arthur C. Sanderson, Jade: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009) 945–958.
[211] Kaibo Zhang, Bin Li, Cooperative coevolution with global search for large scale global optimization, in: IEEE Congress on Evolutionary Computation (CEC), 2012, IEEE, 2012, pp. 1–7.
[212] Shi-Zheng Zhao, Jing J. Liang, Ponnuthurai N. Suganthan, Mehmet Fatih Tasgetiren, Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization, in: IEEE Congress on Evolutionary Computation, 2008. CEC 2008 (IEEE World Congress on Computational Intelligence), IEEE, 2008, pp. 3845–3852.
[213] Shi-Zheng Zhao, Ponnuthurai Nagaratnam Suganthan, Swagatam Das, Dynamic multi-swarm particle swarm optimizer with sub-regional harmony search, in: IEEE Congress on Evolutionary Computation (CEC), 2010, IEEE, 2010, pp. 1–8.
[214] Shi-Zheng Zhao, Ponnuthurai Nagaratnam Suganthan, Swagatam Das, Self-adaptive differential evolution with multi-trajectory search for large-scale optimization, Soft Comput. 15 (11) (2011) 2175–2185.
[215] Fangming Zhu, Sheng-Uei Guan, Cooperative co-evolution of ga-based classifiers based on input increments, Eng. Appl. Artif. Intell. (2007).
[216] Wenping Zou, Yunlong Zhu, Hanning Chen, Xin Sui, A clustering approach using cooperative artificial bee colony algorithm, Discrete Dynam. Nature Soc. 10 (2010).