A Survey on Cooperative Co-Evolutionary Algorithms

Xiaoliang Ma, Xiaodong Li, *Senior Member, IEEE*, Qingfu Zhang, *Fellow, IEEE*, Ke Tang, *Senior Member, IEEE*, Zhengping Liang, Weixin Xie, and Zexuan Zhu, *Member, IEEE*

***Abstract*—The first cooperative co-evolutionary algorithm (CCEA) was proposed by Potter and De Jong in 1994 and since then many CCEAs have been proposed and successfully applied to solving various complex optimization problems. In applying CCEAs, the complex optimization problem is decomposed into multiple subproblems, and each subproblem is solved with a separate subpopulation, evolved by an individual evolutionary algorithm (EA). Through cooperative co-evolution of multiple EA subpopulations, a complete problem solution is acquired by assembling the representative members from each subpopulation. The underlying divide-and-conquer and collaboration mechanisms enable CCEAs to tackle complex optimization problems efficiently, and hence CCEAs have been attracting wide attention in the EA community. This paper presents a comprehensive survey of these CCEAs, covering problem decomposition, collaborator selection, individual fitness evaluation, subproblem resource allocation, implementations, benchmark test problems, control parameters, theoretical analyses, and applications. The unsolved challenges and potential directions for their solutions are discussed.**

***Index Terms*—Cooperative co-evolutionary algorithm (CCEA), evolutionary algorithm (EA), genetic algorithm (GA).**

Manuscript received August 28, 2017; revised January 16, 2018, May 1, 2018, and July 1, 2018; accepted August 23, 2018. Date of publication September 4, 2018; date of current version May 29, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61471246, Grant 61603259, Grant 61871272, Grant 51405075, Grant 61672478, and Grant 61473241, in part by the ANR/RCC Joint Research Scheme sponsored by the Research Grants Council of the Hong Kong Special Administrative Region, China, and France National Research Agency under Grant A-CityU101/16, in part by the Guangdong Special Support Program of Top-Notch Young Professionals under Grant 2014TQ01X273, in part by the Fundamental Research Funds for the Central Universities under Grant GK201603014 and Grant GK201603082, in part by the Project of Department of Education of Guangdong Province under Grant 2016KTSCX121, and in part by the Shenzhen Fundamental Research Program under Grant JCYJ20150324141711587, Grant JCYJ20170302154328155, Grant JCYJ20170302154227954, and Grant JCGG20170414111229388. *(Corresponding author: Zexuan Zhu.)*

X. Ma, Z. Liang, and Z. Zhu are with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: maxiaoliang@yeah.net; liangzp@szu.edu.cn; zhuzx@szu.edu.cn).

X. Li is with the School of Science (Computer Science and Software Engineering), RMIT University, Melbourne, VIC 3001, Australia (e-mail: xiaodong.li@rmit.edu.au).

Q. Zhang is with the Department of Computer Science, City University of Hong Kong, Hong Kong, and also with the Shenzhen Research Institute, City University of Hong Kong, Hong Kong (e-mail: qingfu.zhang@cityu.edu.hk).

K. Tang is with the Shenzhen Key Laboratory of Computational Intelligence, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518000, China (e-mail: tangk3@sustc.edu.cn).

W. Xie is with the ATR National Key Laboratory of Defense Technology, Shenzhen University, Shenzhen 518060, China (e-mail: wxxie@szu.edu.cn).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org provided by the author. This material is a PDF file that provides a simple analysis of well-suited meta-heuristic algorithms for the competitions on large-scale global optimization, the explanation of how CCEAS with incorrect problem decomposition getting trapped into the pseudominima, the extra function evaluations for performing the variable grouping, some examples on collaborator selection strategies for MOP, the definition of CLOB problem, an example of the relative overgeneralization issue, and the applications of CCEAS. The total size of the file is 777 KB.

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

Digital Object Identifier 10.1109/TEVC.2018.2868770

I. Introduction

COOPERATIVE co-evolution is inspired by the ecological phenomenon of mutualism, wherein different species live together in a mutually beneficial relationship [16]. Mutualistic transversals can be considered a form of "biological barter" [121]. The coexisting species benefit from each other's evolution through three kinds of ecological interactions: 1) resource–resource interaction [121] (e.g., the relationship between cows and the rumen bacteria [121]); 2) resource–service interaction [176], (e.g., pollination in which pollen [food] are traded for dispersal [service]); and 3) service–service interaction [195] (e.g., the relationship between anemone fish and sea anemones).

The first cooperative co-evolutionary algorithm (CCEA), namely cooperative co-evolutionary genetic algorithm (CCGA), was proposed by Potter and De Jong in 1994 [156], and it evoked tremendous research interest by virtue of its ability solving various kinds of optimization problems. CCEAs are promising because they can decompose the original problem into a set of lower-dimensional and tractable subproblems, each of which can be solved in a separately evolving subpopulation. The evaluation of each individual in a subpopulation calls for cooperation with other subpopulations. Complete solutions are obtained by assembling the representative members from each subpopulation. The fitness of an individual in a subpopulation is evaluated in terms of that of the complete solution(s) in which this individual participates.

The advantages of CCEA over the traditional evolutionary algorithm (EA) result mainly from its divide-and-conquer decomposition strategy. The CCEA has mainly four advantages. First, decomposition of the problem allows parallelism to speed up the optimization process. Second, each subproblem is solved with a separate subpopulation, which maintains good solution diversity [27]. Third, decomposing a system into submodules increases the robustness against the modules' errors and failures, and thus enhances the reusability in dynamic environments [117]. Finally, the "curse of dimensionality," i.e.,

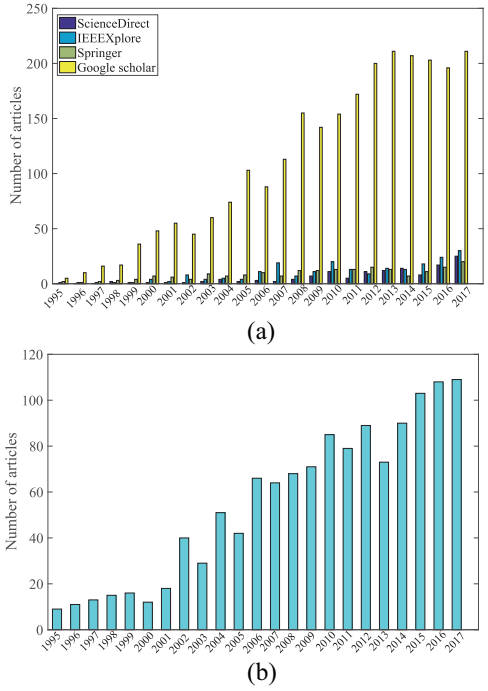

Fig. 1. (a) Numbers of yearly published articles with the term "cooperative co-evolution" in the abstract, title, and/or keywords in ScienceDirect, IEEE Xplore, Springer, and Google Scholar, from 1995 to 2017. (b) Yearly citations of the Potter and De Jong's paper [156] in Google Scholar.

the rapid deterioration in performance with increase in the number of decision variables, can be alleviated to some extent, if the problem is properly decomposed.

CCEAs have been successfully applied to a variety of optimization problems across a broad range of real-world application areas. The cooperative co-evolution framework has been adopted in conjunction with many other meta-heuristic algorithms. It thus injects vitality into the classic optimization solutions. There has been a rapid growth in research of CCEAs during the last two decades, as can be seen from the progressive increase in the number of articles published yearly on CCEAs and the yearly number of citations of Potter and De Jong's paper [156] (see Fig. 1).

This paper documents the recent advances in the studies on CCEAs. Different from the earlier surveys [133], [151], [221], this is a more comprehensive survey covering diverse topics, such as problem decomposition, collaborator selection, individual fitness evaluation, subproblem resource allocation, implementations, benchmark test problems, control parameters, theoretical analyses, and applications. This paper makes broadly four contributions to the field.

1) This paper is the first attempt to modularize CCEAs into five components, namely, problem decomposition, subproblem optimizer, collaboration selection, individual fitness assignment, and computing resource allocation of subproblem. The proposed modularization enables the reader to better understand the algorithm.
2) The decomposition of problem and variable interaction learning are first discussed here in six aspects, namely static variable grouping, random variable grouping, interaction learning-based variable grouping, domain knowledge-based variable grouping, overlap and hierarchy variable grouping, and hybrid variable grouping.
3) The benchmark test problems, control parameters, pathological issues, and real-world engineering applications of CCEAs are reviewed in this survey for the first time.
4) New challenges and future directions for dealing with them are provided.

The rest of this paper is organized as follows. Section II introduces some basic notions and the CCGA. Section III reviews the problem decomposition strategies and linkage learning methods. Section IV presents the collaborator selection strategies among subproblems. Section V discusses the approaches for individual fitness assignment and subproblem resource allocation. Section VI reviews the implementation of CCEAs. Section VII presents the benchmark test problems and the theoretical analyses of CCEAs. Section VII discusses the control parameters of CCEAs. Section VIII introduces the pathological issues of CCEAs. Section IX highlights the applications of CCEAs. Finally, Section X sums up the conclusions drawn from this paper and discusses possible directions for dealing with future challenges. For the convenience of the readers, the common notations and nomenclature, followed in this paper, are provided below.

$\mathbf{x} = (x_1, \ldots, x_n)$	n-dimensional decision vector.
Ω	n-dimensional feasible decision space.
$\triangle x_i$	Perturbation on x_i.
$\triangle_i f(\mathbf{x})$	Fitness change, subject to a perturbation on x_i.
$\triangle_{i,j} f(\mathbf{x})$	Fitness change by a perturbation on x_i and x_j.
m	Number of subcomponents.
N	Subpopulation size.
n	Number of decision variables.
$O_{i,j}$	jth individual in the ith subpopulation.
s	Number of variables in a subcomponent.
Context vector $\mathbf{b} = (b_1, \ldots, b_n)$	Complete problem solution composed of representative solutions from each subpopulation.
Subcomponent $\mathbf{x}_i$	Subset of the decision variables $\mathbf{x}$.
Subpopulation	Population of individuals used to optimize a subproblem.
Subproblem	Problem optimizing a corresponding subcomponent, i.e., optimizing the original problem considering a subset of $\mathbf{x}$ only.

II. Basic Notions and CCGA

Before reviewing the CCEAs, the basic concepts of problem separability and the first CCEA, i.e., CCGA, are introduced to facilitate the understanding of CCEAs.

A. Fully Separable, Additively Separable, and k-Nonseparable

In applying CCEAs, the solvability of an optimization problem relates to the separability of this problem. Here the definition of separability is differentiated in fully separable, fully nonseparable, k-nonseparable, and additively separable.

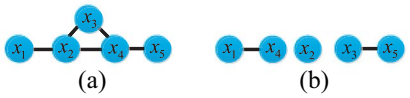

Fig. 2. (a) Fully nonseparable problem and (b) k-nonseparable ($k = 2$) problem. A node is a decision variable and an edge indicates a variable interaction.

Definition 1: An optimization problem $f(\mathbf{x}) : \Omega \rightarrow R$ is fully separable [123], [230] if and only if

$$\arg\min_{x_1,\ldots,x_n} f(\mathbf{x}) = \left(\arg\min_{x_1} f(x_1,\ldots), \ldots, \arg\min_{x_n} f(\ldots,x_n)\right)$$

where Ω is an n-dimensional decision space. Otherwise, $f(\mathbf{x})$ is called a nonseparable optimization problem.

According to Definition 1, a separable optimization problem can be solved by optimizing each decision variable independently [230]. Therefore, fully separable problems are easy to be solved by CCEAs equipped with the divide-and-conquer strategy.

Nonseparable optimization problems are more challenging to CCEAs. A nonseparable function is called fully nonseparable, if every two decision variables interact with each other, as shown in Fig. 2(a). From fully separable to fully nonseparable problems, there exist various partially separable problems, also called as k-nonseparable problems [33]. An optimization problem $f(\mathbf{x})$ is called k-nonseparable, if at the most k decision variables are interdependent [218]. For example, a 2-nonseparable problem, with three independent subcomponents, is shown in Fig. 2(b). Fewer k values indicate that the problems are easier to solve.

Additively separable problems are a type of k-nonseparable problems widely used in the literature to test CCEAs.

Definition 2: A problem $f(\mathbf{x})$ is additively separable if it can be formulated in the following form [32], [35], [68], [123], [181], [206]:

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

where $\mathbf{x}_1,\ldots,\mathbf{x}_m$ are disjoint subcomponents of $\mathbf{x}$, and m is the number of subcomponents. Each component problem $f_i(\cdot)$ is an optimization problem, depending on the part of the decision variables. An additively separable problem is fully separable if each of $\mathbf{x}_1,\ldots,\mathbf{x}_m$ contains only one decision variable. An additively separable problem is relatively easy to solve for CCEAs, because its variable interactions are easily identifiable [123].

B. CCGA

CCGA is the first CCEA proposed by Potter and De Jong [156]. In CCGA, an optimization problem of n variables is first decomposed into n 1-D subproblems. Each subproblem is then optimized by a separate population-based genetic algorithm (GA). A complete solution of the original problem is obtained by combining the representative solutions from each subproblem. The fitness of an individual in a particular subpopulation is estimated by the quality of the complete solution(s) in which the individual participates.

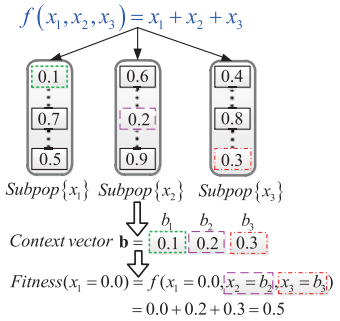

Fig. 3. Collaborator selection and individual fitness assignment in CCGA-1.

CCGA solves the subproblems in a round-Robin fashion, i.e., they are assigned the same computing resources.

Algorithm 1 outlines the procedure of CCGA. In step 1, 1-D decomposition is used to divide the original problem. The subpopulations and the context vector, i.e., a complete problem solution composed of the best individuals from each subpopulation, are randomly initialized in step 2. In step 3, each subcomponent is optimized in a round-Robin fashion, with a standard GA. Particularly, an offspring subpopulation is generated in step 3.1, with evolutionary operators for each subproblem. In step 3.2, the fitness of the jth offspring in the ith subpopulation, denoted as $O_{i,j}$, is estimated, using two schemes that result in two versions, namely CCGA-1 and CCGA-2, respectively. CCGA-1 evaluates the fitness of $O_{i,j}$ by calculating the fitness of the context vector with the corresponding part replaced by $O_{i,j}$ in Fig. 3. CCGA-2 evaluates the fitness of $O_{i,j}$ twice by combining it with the context vector and a solution vector that randomly selects individuals from the subpopulations. The better of the two resultant vectors is adopted as the final fitness of $O_{i,j}$. Step 3.3 updates the context vector. Finally, step 3.4 constructs new subpopulations based on both parent and offspring subpopulations using survival-of-the-fittest selection. For the sake of simplicity, the sizes of all subpopulations are kept the same.

To illustrate the problem decomposition and individual fitness evaluation in CCGA-1, a clearly toy example is provided in Fig. 3. Given an optimization problem $f(x_1, x_2, x_3) = x_1 + x_2 + x_3$, the decision vector is decomposed into three 1-D subcomponents $\{x_1\}$, $\{x_2\}$, and $\{x_3\}$, each of which is evolved in a subpopulation. In each generation, the context vector $\mathbf{b}$ is formed by selecting the best individuals from each subpopulation. Given $\mathbf{b} = (0.1, 0.2, 0.3)$ in this example, the fitness of a new individual in the first subpopulation, e.g., $x_1 = 0.0$, is evaluated by calculating $f(x_1 = 0.0, b_2 = 0.2, b_3 = 0.3)$.

CCGA statically decomposes the n-dimensional optimization problem into n 1-dimensional subproblems and the problem decomposition is fixed during the process of evolution. In the case of fully separable problems, this decomposition scheme offers significant advantages over classic GA [156], because CCGA reduces the search space from n-dimension to n 1-dimension. However, it fails to solve the nonseparable problems in which tight interactions exist among decision variables [156]. It is to be noted that real-world optimization problems are often

Algorithm 1 Procedure of CCGA

Require: n: the number of variables; N: the subpopulation size.
Ensure: $\mathbf{b} = (b_1, ..., b_n)$ and f_b: current optimal solution and its fitness found by the algorithm.
Step 1 Problem decomposition: n-dimensional decision vector is divided into n 1-dimensional subcomponents $\{x_1\}, \{x_2\}, ..., \{x_n\}$.
Step 2 Randomly initialize and evaluate subpopulations and the context vector b
Randomly initialize **b** and set $f_b = f(\mathbf{b})$.
For $i = 1$ to n **do** % initialize the i-th subpopulation
2.1 Randomly initialize the i-th subpopulation $\mathbf{P}_i$. % $\mathbf{P}_i = \{P_{i,1}, ..., P_{i,N}\}$
2.2 Individual fitness assignment: Evaluate the fitness of each individual in the i-th subpopulation $\mathbf{P}_i$
$f_{P_{i,j}} \leftarrow f(b_1, ..., b_{i-1}, P_{i,j}, b_{i+1}, ..., b_n), j = 1, ..., N.$
End of For
Step 3 Evolution
While stopping criterion is not met **do**
For $i = 1$ to n **do** % optimize the i-th subpopulation/subcomponent
3.1 Generate an offspring subpopulation: $\mathbf{O}_i \leftarrow GeneticOperators(\mathbf{P}_i)$.
3.2 Individual fitness assignment: Evaluate the fitness of each individual in offspring subpopulation $\mathbf{O}_i$:
$f_{O_{i,j}} \leftarrow f(b_1, ..., b_{i-1}, O_{i,j}, b_{i+1}, ..., b_n), j = 1, ..., N.$
If CCGA-2 is used
$f_{ro_{i,j}} = f(r_1, ..., r_{i-1}, O_{i,j}, r_{i+1}, ..., r_n), j = 1, ..., N.$ % r_k is a randomly selected individual from the k-th subpopulation
If $f_{ro_{i,j}} < f_{O_{i,j}}$, **then** $f_{O_{i,j}} = f_{ro_{i,j}}$.
End of If
3.3 Update the context vector: b
For $j = 1$ to N **do**
If $f_{O_{i,j}} < f_b$, **then** $b_i = O_{i,j}, f_b = f_{O_{i,j}}$.
End of for
3.4 Select the next parent subpopulation: $\mathbf{P}_i \leftarrow Select(\mathbf{P}_i, \mathbf{O}_i)$
End For
End of While

nonseparable [74], [107], [219], and they needs more research efforts to improve CCGA.

III. Problem Decomposition and Variable Interaction Learning

One important issue in CCEAs is the decomposition of decision vector regarding variable interactions. Ideally, the decomposition should be performed based on the variable interactions, so as to minimize the interactions between the subcomponents [123]. If the interacting variables are not grouped into the same subcomponent, CCEAs tend to be trapped in a pseudo-minimum [11], which is not the local minimum of the original problem but a local minimum introduced by incorrect problem decomposition (see Appendix B of the supplementary material for more details).

To deal with this issue, many variable grouping strategies have been proposed, including static variable grouping, random variable grouping, linkage learning-based variable grouping, overlap and hierarchical variable grouping, domain knowledge-based variable grouping, and their hybrids.

A. Static Variable Grouping

Static variable grouping methods do not rely on any systematic or intelligent procedure to discover the interdependence of the variables. Instead, it preliminarily decomposes a high-dimensional decision vector into a set of low-dimensional subcomponents, and fixes the variable grouping during the process of optimization.

Potter and De Jong [156] first suggested decomposing an n-dimensional problem into n 1-dimensional subproblems, each of which is optimized by a separated GA. This strategy has proved successful on fully separable optimization problems [92], [156], but not on nonseparable optimization problems. To solve this issue, Bergh and Engelbrecht [11] decomposed an n-dimensional decision vector into m s-dimensional subcomponents, as shown in Fig. 4(a), where $n = m * s$. They showed that the decomposition method performs better than 1-D decomposition on many benchmark problems. In [173], a group of static decompositions were designed to create new individuals by a random eligible combination of gene blocks. Cao *et al.* [24] introduced a static decomposition based on a sequential sliding window for large-scale optimization problems, in which the number of variables n is great than or equal to 1000. The static decomposition methods introduced in this section are summarized in Table I.

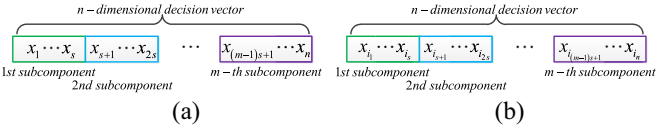

Fig. 4. (a) Sequential decomposition and (b) random decomposition of n-dimensional problem into m s-dimensional subproblems, where $i_1, \ldots, i_n$ is a permutation of the arrangement $1, \ldots, n$.

TABLE I
Summary of Studies on Static Grouping in CCEAs

Algorithm(s)	Static problem decomposition	Collaborator selection
CCGA [156]	1-D decomposition	Single best collaborator Best+random collaborator
AEGL [173]	A pool of static decompositions	Random collaborator
CC-GLS [24]	Sequential sliding window	Single best collaborator
CCPSO-S, CCPSO-S$_K$ [11]	m s-D subcomponents	
GCD [12]	Domain knowledge-based	

B. Random Variable Grouping

The main problem of static variable grouping is that if two interacting variables are placed in different groups, then there is no chance for them to be in the same group anymore. Random variable grouping or random grouping (RG) relieves the drawback of static grouping by randomly selecting decision variables for each subcomponent. The subcomponent size in RG strategy can be fixed or dynamically altered.

1) RG With Fixed Subcomponent Size: This strategy fixes the number of subcomponents m and randomly selects s exclusive variables for each subcomponent in every co-evolution cycle[1] [230] as shown in Fig. 4(b), where $n = m * s$. The purpose of RG is to increase the probability of putting interacting variables in the same subcomponent.

Yang *et al.* [229] proposed the first RG strategy in cooperative co-evolution with differential evolution (DECC-I), where RG was shown to outperform static variable grouping, especially on nonseparable problems. Later, they improved the algorithm by introducing adaptive weighting strategy [230] and proposed DECC-G, which outperforms the standard differential evolution (DE) and DE with static variable grouping. Li and Yao [82] applied RG with fixed subcomponent size in particle swarm optimization (PSO)-based cooperative co-evolution [125].

The advantage of RG strategy over static grouping strategy lies in the fact that the probability of assigning two interacting variables to the same subcomponent is relatively high, for at least two cycles [230]. Whereas, if the number of interacting variables is large, the probability of putting them together in the same subcomponent, for at least one cycle, is still very low [125].

2) RG With Dynamic Subcomponent Size: A common drawback of static variable grouping and RG with a fixed subcomponent size and static variable grouping is that prior knowledge of problem structure is required for the correct setting of subcomponent size s. A small s suits separable problems, whereas a large s increases the probability of retaining interacting variables in a group if the problems are nonseparable [197]. A dynamic strategy is therefore desirable for tuning s [83].

Yang *et al.* [229] proposed a dynamic RG strategy in DECC-II, by randomly tuning s in a predefined range, in each co-evolution cycle. A self-adaptive strategy was also introduced to set s in a multilevel cooperative co-evolution algorithm (MLCC) [231]. The probability of selecting s, from a set $S = \{s_1, s_2, \ldots, s_t\}$, is calculated based on its recent performance in improving the context vector $\mathbf{b}$. Multiple trajectory search (MTS) [200] uses a strategy with a set of predefined group sizes $S = \{1, n/4\}$. A value function method and a softmax selection rule in [128] were adopted to choose the subcomponent size. A few studies [83], [125], [126] first used a fixed s to optimize the problem until no fitness improvement can be achieved and then chose a new group size from $S = \{s_1, \ldots, s_t\}$ uniformly. The number of variables in each subcomponent s is gradually reduced during the optimization process [149]. CCFA [196] and CCAS [198], [199] simultaneously adapt the subcomponent size and the corresponding subpopulation size based on their recent performance in improving the context vector $\mathbf{b}$. Heuristic adjustment based on domain knowledge is also applicable to adjustment of the subcomponent size [63]. The summary of the studies on RG is presented in Table II.

[1]A co-evolution cycle refers to one iteration of the While loop in step 3 of Algorithm 1.

TABLE II
SUMMARY OF STUDIES ON RG IN CCEAS

Algorithm(s)	Random grouping strategy	Collaborator selection
MTS[200]	RG with dynamic group sizes	Location based collaborator
DECC-I[229] DECC-G[230] CCPSO[82]	RG with a fixed subcomponent size	Single best collaborator
DECC-II[229] CCPSO2[83] DECC-ML[125] DECC-DML[126]	RG with uniform selection of subcomponent sizes	
DECC-NW[125]	More frequent RG with a fixed subcomponent size	
MLCC[231]	RG with adaptive group sizes	
MLSoft[128]	Self-adaptive RG based on softmax action selection	
CCFA[196] CCAS[199]	Dynamic RG and subpopulation sizes	
CCPSO2-IP[149]	Automated iterative partitioning	
CCAS[198]	RG with dynamic group sizes	
CCDECD[63]	RG+heuristic information	

C. Variable Grouping Based on Interaction Learning

Static variable grouping and random variable grouping have two defects: 1) the user must predefine the size(s) of the subcomponents $\{s_1, \ldots, s_t\}$ ($t = 1$ for static grouping) and 2) once the group size is chosen, the decision variables get equally divided into a set of subcomponents, which is usually an unreasonable assumption. It is favorable that the variable grouping strategy can automatically determine the number of subcomponents and their corresponding sizes. Ideally, the subcomponents should be formed, based on variable interaction, so that no interactions take place between subcomponents [104], [123].

Variable interactions describe the structure of a problem. If the algorithm can learn the problem structure and decompose the problem accordingly, the difficulty in solving the problem can be reduced significantly [232]. Many approaches have been proposed to detect variable interactions, which are mainly based on perturbation, statistical model, distribution model, approximate model, and linkage adaptation [31], [123], [203], [232].

1) Perturbation: In this approach, the interaction is captured by perturbing decision variables and measuring the change in the fitness caused by the perturbations. The straightforward measurements of the fitness changes are defined as follows:

$$\begin{aligned}
\Delta_i f(\mathbf{x}) &= f\big(\ldots, x_i + \Delta x_i, \ldots, x_j, \ldots\big) - f(\ldots, x_i, \ldots, x_j, \ldots) \\
\Delta_j f(\mathbf{x}) &= f\big(\ldots, x_i, \ldots, x_j + \Delta x_j, \ldots\big) - f\big(\ldots, x_i, \ldots, x_j, \ldots\big) \\
\Delta_{i,j} f(\mathbf{x}) &= f\big(\ldots, x_i + \Delta x_i, \ldots, x_j + \Delta x_j, \ldots\big) \\
&\quad - f\big(\ldots, x_i, \ldots, x_j, \ldots\big)
\end{aligned}$$

where $\Delta_i f(\mathbf{x})$, $\Delta_j f(\mathbf{x})$, and $\Delta_{i,j} f(\mathbf{x})$ are the changes in fitness, caused by a perturbation on the ith variable, the jth variable,

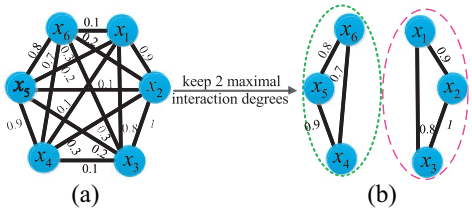

Fig. 5. (a) Example of interaction degree of variables. (b) Variable grouping where each variable is assigned two variable interactions, with maximal degrees.

and both variables, respectively. Whether or not the two variables interact with each other can be detected based on $\Delta_i f(\mathbf{x})$, $\Delta_j f(\mathbf{x})$, and $\Delta_{i,j} f(\mathbf{x})$.

Linkage identification by nonlinearity check (LINC) [113] uses bitwise perturbations to capture nonlinear linkages between x_i and x_j, if there exists a solution $\mathbf{x}$ in the current population, such that

$$\Delta_i f(\mathbf{x}) + \Delta_j f(\mathbf{x}) \neq \Delta_{i,j} f(\mathbf{x}). \tag{2}$$

Equation (2) implies that $\Delta_i f(\mathbf{x})$ is dependent on the value of x_j. Linkage identification with nonmonotonicity detection (LIMD) [112] checks the violation of monotonicity conditions, i.e., $0 < \Delta_i f(\mathbf{x}), \Delta_j f(\mathbf{x}) < \Delta_{i,j} f(\mathbf{x})$, to detect variable interaction. The works [110], [111] generalized the variable interaction conditions in LINC and LIMD to measure the degree of interaction. Each variable x_i interacts with other k variables having maximal interaction degrees, and the variables with dense interactions are grouped together [110], [111]. Interaction degree between every two decision variables can be estimated as follows:

$$\Delta_{i,j} = \left|\Delta_i f(\mathbf{x}) + \Delta_j f(\mathbf{x}) - \Delta_{i,j} f(\mathbf{x})\right|. \tag{3}$$

An example of interaction degree of variables based on (3) is shown in Fig. 5, where each variable is assigned two interactions with maximal degrees. The interacting variables are then grouped into $\{x_1, x_2, x_3\}$ and $\{x_4, x_5, x_6\}$.

The aforementioned methods were proposed for binary optimization problem. For continuous optimization, only a limited number of techniques are available. Two continuous variables are considered interdependent, if a new candidate solution achieves better fitness by changing both variables simultaneously than by changing one individually [217], i.e.,

$$\begin{aligned} \exists \Delta x_i, \Delta x_j, \text{ s.t. } & \Delta_{i,j} f(\mathbf{x}) < \Delta_i f(\mathbf{x})|_{\mathbf{x}=\mathbf{b}} \\ & \textbf{and } \Delta_{i,j} f(\mathbf{x}) < \Delta_j f(\mathbf{x})|_{\mathbf{x}=\mathbf{b}} \end{aligned} \tag{4}$$

where $\mathbf{b}$ is the context vector.

Cooperative co-evolution with variable interaction learning (CCVIL) [30] considers two variables x_i and x_j interdependent if

$$\begin{aligned} \exists \mathbf{x} = (x_1, \ldots, x_n) \in \Omega, \Delta x_i, \Delta x_j \\ \text{s.t. } \Delta_i f(\mathbf{x}) > 0 \textbf{ and } \Delta_{i,j} f(\mathbf{x}) - \Delta_j f(\mathbf{x}) < 0. \end{aligned} \tag{5}$$

Equation (5) is equivalent to the original definition in [30] by subtracting $f(\cdots, x_i, \ldots, x_j, \ldots)$ on both sides. It is trivial to prove the following equivalent form:

$$\begin{aligned} \exists \mathbf{x} = (x_1, \ldots, x_n) \in \Omega, \Delta x_i, \Delta x_j \\ \text{s.t. } \Delta_i f(\mathbf{x}) \cdot \left(\Delta_{i,j} f(\mathbf{x}) - \Delta_j f(\mathbf{x})\right) < 0. \end{aligned} \tag{6}$$

Equation (5) is widely used for both single-objective and multiobjective optimization [98], [209]. Based on (5), Sun *et al.* [183] introduced a statistical method to calculate the interaction probability as $P\{\Delta_i f(\mathbf{x}) \cdot (\Delta_{i,j} f(\mathbf{x}) - \Delta_j f(\mathbf{x})) < 0\}$.

LINC for real-coded GA (LINC-R) [193] advances two continuous variables interacting with each other, if the change in fitness, caused by a perturbation on one variable, is independent of the value of the other variable, i.e.,

$$\begin{aligned} \forall \mathbf{x} = (x_1, \ldots, x_n) \in \Omega, \Delta x_i \neq 0, \Delta x_j \neq 0 \\ \text{s.t. } \Delta_i f(\mathbf{x}) + \Delta_j f(\mathbf{x}) \neq \Delta_{i,j} f(\mathbf{x}). \end{aligned} \tag{7}$$

Unlike (2) developed for binary-coded EA, (7) is designed for real-coded EA and $\mathbf{x} \in \Omega$. Xu *et al.* [227] used (7) to uncover the interaction relationship between a decision variable and the environment variable for dynamic optimization.

Differential grouping (DG) method [123] provides a theoretical derivation of using (7) to recognize the interacting variables of additively separable problems. To reduce the number of function evaluations, DG does not check indirect variable interactions. Both direct and indirect variable interactions are considered simultaneously in the extended DG (XDG) [184]. An example of direct and indirect interactions is shown in Fig. 6(a), where x_1 has a direct variable interaction with x_2, and x_1 indirectly interacts with x_5 via x_2, x_3, x_4. To detect all variable interactions, the global DG [104] and DG2 [129] improve DG by considering all pair-wise interactions at the cost of $O(n^2)$ function evaluations on separable optimization problems [61]. Hu *et al.* [61] used a fixed number of function evaluations to recognize whether one variable is separable from others. Particularly, a modified version of (7) was used to capture the nonseparable relationship between one variable and a group of variables.

The two quantitative measures defined in (5) and (7), suffer from inaccurate prediction when the number of samples or the number of function evaluations is limited [123]. Combining these two measures could be a better option. Interaction learning method based on perturbation is a recent and open direction.

Most of the variable grouping strategies need no extra function evaluations, excepting the perturbation-based variable grouping strategy. The worst case requires $O(n^2)$ more function evaluations for most perturbation-based variable grouping strategies [61], [104], [123], [129], [183], [184], [193], [227], and $O(n)$ for CCVIL [30] (see Appendix C of the supplementary material for more details).

The experimental studies in [30], [123], and [217] showed that perturbation-based variable grouping could perform better than static grouping and random variable grouping on large-scale optimization problems.

2) Statistical Model: In the interaction learning based on statistic model, all variables and the objective function(s) are

considered as random variables. Statistical analysis on variables and/or objective function(s) are performed first and then the variables are grouped, based on the following metrics.

1) Variables dependency measure based on Pearson correlation coefficient (PCC) [159], [172], [205], [209], entropic epistasis [164], mutual information [182], etc.
2) Correction between variable and objective functions, including the cumulative PCC between them [161] and the maximal information coefficient between x_j and $\partial f(\mathbf{x})/\partial x_i$ [185].
3) The distribution of variable values, including the mean of variable values [126], the variance of variable values [90], [213], and the Kullback–Leibler distance between the bivariate joint distribution and the univariate distribution [232].

The superiorities of statistic model-based strategies over static grouping and random variable grouping on some specific problems have been shown in [90], [126], and [185].

3) Distribution Model: In distribution model, the set of promising solutions is first used to estimate the variable distributions and variable interactions, and then to generate new candidate solutions, based on the learned variable distributions and variable interactions [143]. Estimation of distribution algorithm (EDA) [109] is one most widely used representative of such methods. The simplest way in EDA is to consider each variable independently. This is the basic principle of univariate marginal distribution algorithm [109], the compact GA [57], and the population-based incremental learning algorithm (PBIL) [9]. However, these algorithms might not suit well to nonseparable problems.

Earlier attempts to solve nonseparable problems were based on pairwise interactions, e.g., the bivariate marginal distribution algorithm [144], the incremental algorithm using dependency trees to estimate the distribution [10], and the population-based MIMIC algorithm based on simple chain distributions [14]. Such methods are used to detect important building blocks of two variables and to prevent disruption in reproduction. Yet, they cannot handle complex nonseparable problems with high order interactions in which most decision variables interact with each other [145].

Factorized distribution algorithm (FDA) [108], [235] was proposed to deal with higher order interactions using a conditional distribution based on problem decomposition. FDA performs well on additively decomposable problems, but it relies on prior knowledge of problem structure and factorization with expensive computational cost. Bayesian optimization algorithm (BOA) is another solution introduced in [143] to discover higher order interactions using Bayesian networks with relatively expensive computational cost.

Considering linkage in a probability distribution manner, EDAs tend to ignore building blocks having a relatively low fitness contribution [203]. They might perform poorly on complex optimization problems [143].

4) Approximate Model: Many real-world optimization problems demand expensive physical simulation or computational cost to evaluate the candidate solutions. Efficient surrogate model is necessary to approximate the evaluation of the original objective function. For example, the fitness evaluation of a large-scale continuous optimization problem is converted to the evaluation of a simpler partially separable problem in [162]. The high dimensional model representation (HDMR) [79] uses an approximately quantitative model to capture the input–output system behavior of the objective function. As multiple input variables can affect the output independently and collectively, the model output $f(\mathbf{x})$ can be denoted as a sum of hierarchically correlated functions on the input variables

$$f(\mathbf{x}) = f_0 + \sum_i f_i(x_i) + \sum_{1\le i<j\le n} f_{ij}(x_i, x_j) + \sum_{1\le i<j<k\le n} f_{ijk}(x_i, x_j, x_k) + \cdots + f_{12\ldots n}(x_1, x_2, \ldots, x_n)$$

where the 0-order component problem f_0 is a constant reflecting the average response to $f(\mathbf{x})$, the 1-order component problem $f_i(x_i)$ represents the independent contribution of each single variable x_i to $f(\mathbf{x})$, the 2-order component problem $f_{ij}(x_i, x_j)$ indicates the cooperative contribution of two variables x_i and x_j to $f(\mathbf{x})$, and the rest may be deduced by analogy. The item $f_{12\ldots n}(x_1, \ldots, x_n)$ represents the residual n-order correlated contribution of n input variables to $f(\mathbf{x})$. Two variables are considered nonseparable if they have a cooperative effect on the approximated second-order HDMR model [100].

5) Linkage Adaption: Linkage adaption methods use specially designed evolution operators, representations, and mechanisms, to divide variables into groups. Schaffer and Morishima [163] attached a punctuation flag to each gene in the individual chromosome to indicate the crossover point. The genes between two adjacent crossover points are combined into the same group. Levenick [78] extended [163] by introducing additional bits to indicate the probability of choosing a position as a crossover point. Smith and Fogarty [174] presented the linkage evolving genetic operator (LEGO) to adjust the variable grouping in an adaptive way. They imposed two Boolean flags to each gene to indicate which of the neighbors, left neighbor or right neighbor, it interacts with on the chromosome. The interacting neighboring genes are then considered as part of the same group.

The CCEAs using variable grouping, based on interaction learning, are summarized in Table III.

D. Variable Grouping Based on Domain Knowledge

If prior domain knowledge is available, it is natural to exert it to learn the variable interactions. Before CCEAs kick in to solve specific real-world problems, domain knowledge can be harnessed to reduce the complexity of the problems. For example, the local property of reactive power and voltage relation were applied to divide a power system into several relatively interindependent subsystems in [85]. The conflicting probability of two flights was used in [53] to learn the variable interaction in solving flight conflicting avoidance problem. The proposed grouping strategy was shown to perform better than delta values-based, splitting-in-half, and dependency identification grouping strategies on flight conflicting avoidance problem. Multidepot vehicle routing problem can be

TABLE III
SUMMARY OF VARIABLE GROUPING BASED ON INTERACTION LEARNING IN CCEAS

Algorithm(s)	Grouping strategy	Collaborator selection
Perturbation		
LINC [114]	Nonlinearity check	Best problem solution
LIMD [113]	Non-monotonicity detection	-
LIEM [110, 100]	Epistasis measures	-
AE [217]	Epistatic links recognition	Single best collaborator
CCVIL [30]	Variable interaction learning	Single best collaborator
CPSO-SL [183] CDECC [45] CCC_DE [209] MOEA/DVA[98]	Variable interaction learning	Single best collaborator location collaborator
LINC-R [193]	Nonlinearity check	Single best collaborator
DNSGAII-CO, DMOPSO-CO[227]	Differential grouping	Random individuals of nondominated front
DECC-DG [123]	Differential grouping	Single best collaborator
DECC-XDG[184]	Extended differential grouping	Single best collaborator
RDG-MAENS[104]	Global differential grouping	Single best collaborator
DG2 [129]	Improved differential grouping	Single best collaborator
FII [61]	Fast interaction identification	Single best collaborator
Statistical model		
LIMS [205]	PCC of decision variables	-
CCEA-AVP [159, 172, 209]	PCC of decision variables	Single best collaborator
DSMGA [232]	Mutual information metric	-
DECC-CIG [182]	Mutual information and entropy	Single best collaborator
4CDE [161]	PCC of function and variable	Single best collaborator
MEE [185]	Maximal information coefficient	1-D and global search
CCME [117]	PCA-based decomposition	Single best collaborator
VP-DECC[213]	Variance of variable	Single best collaborator
CC-CMA-ES[90]	Min-variance decomposition, max-variance decomposition	Single best collaborator
DECC-D and DECC-DML[126]	Delta grouping	Single best collaborator
Distribution model		
UMDA [109]	Univariate marginal distribution	-
BUMDA [207]	Univariate marginal distribution	-
PBIL [9]	Population incremental learning	-
BMDA [144]	Bivariate marginal distribution	-
MIMIC [14]	Simple chain distributions	-
MIMIC+DT[10]	Dependency trees	-
FDA [108, 235]	Factorized distribution	-
BOA [143]	Bayesian networks	-
CCME [117]	Principal component analysis	-
Approximate model		
DIMA [162]	Partially separable function	Single best collaborator
DM-HDMR[100]	Second-order RBF-HDMR	Single best collaborator
Linkage adaption		
BTGA [163]	Punctuation marks encoding	-
Metabits[78]	Metabits encoding	-
LEGO [174]	flags based neighbor link	-

decomposed into multiple subproblems, based on the nearest neighbor relationship between the depot and the customer node [120]. In [89], [215], and [216], variable interactions were detected based on arithmetic operations ($+$, $-$, $*$, and $/$) and the composite operations of six basic elementary functions.

E. Overlap and Hierarchy Variable Grouping

Fully nonseparable problems, in which every two variables interact with each other, pose big challenges to the classical CCEAs [24]. Overlap variable grouping, as shown in Fig. 6(a), can be a potential solution to these problems, if the following three issues are properly addressed.

1) How to design an efficient variable grouping to minimize the interactions among subcomponents [180]?
2) How to construct the composite n-dimensional solution for individual evaluation [49], [50]?

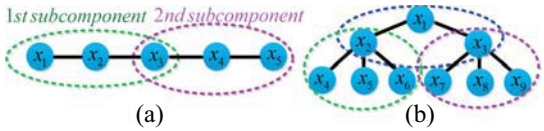

Fig. 6. (a) Example of overlap variable grouping. (b) Example of hierarchy variable interactions.

3) What kind of information can be exchanged among subcomponents, in the case two subcomponents share the same variable(s) [183], as shown in Fig. 6?

Goh *et al.* [49], [50] suggested assigning each variable to multiple subcomponents, each of which contains more than one variable. The subcomponent competes with each other in representing the shared variables. A subcomponent is designed to contain variable x_i and other variables that interact with x_i to a degree no less than a given threshold [183]. The composite solution, composed of the best solution of the ith subpopulation on x_i, competes with the current global best solution for information exchange. Factored EA [180] introduces a number of overlap grouping strategies, including random overlap grouping, neighbor overlap grouping, centered overlap grouping, parents overlap grouping, loci overlap grouping, Markov grouping, and Clique grouping.

The interactions of the variables could be more complex in a hierarchical structure [101], [232], as shown in Fig. 6(b). To deal with such problems, Yu *et al.* [232] proposed to decompose the problems properly at each level, such that the EA can effectively represent variable grouping in a lower level as an input variable of an upper-level and mix subsolutions. The upper level subproblems partly adjust the evolution process of its lower level subproblems [101], [102]. Bonson *et al.* [15] and Lichodzijewski and Heywood [87] presented reviews of task decomposition and diversity maintenance for the evolution of complex systems in genetic programming (GP). Zheng and Chen [241] adopted a hierarchical cooperative approach to decompose the original target problem into several low-dimensional subcomponents and then used PSO to solve the main problem and the subproblems.

F. Hybrid Variable Grouping

The combination of the above-mentioned variable grouping strategies could be an more effective way to solve variable decomposition problems. For example, MTS [200], [201] combines 1-D static decomposition with a random variable grouping. The dependency detection for distribution, derived from fitness differences (D^5) [194], [204], integrates a bitwise perturbation method with a probabilistic distribution model. A combination of RG, min-variance grouping, and max-variance grouping was proposed in [90]. Using hybrid grouping methods has been shown to perform better than using the counterpart member grouping strategies individually in some specific problems.

CCEAs based on the last three decomposition strategies, i.e., variable grouping based on domain knowledge, overlap and hierarchy variable grouping, and hybrid variable grouping, are summarized in Table IV.

TABLE IV
SUMMARY OF VARIABLE GROUPING BASED ON DOMAIN KNOWLEDGE, OVERLAP AND HIERARCHY VARIABLE GROUPING, AND HYBRID VARIABLE GROUPING IN CCEAS

Algorithm(s)	Variable grouping strategy	Collaborator selection
Variable grouping based on domain knowledge		
FBG-CMA [89] VGDE [215], CCVF [216]	Composite of arithmetic operations and base elementary problems	Single best collaborator
CCDE-PSD[85]	Local property of reactive power and voltage relation	Single best collaborator
CCDG [53]	Conflict time of two flights	Single best collaborator
CoES [120]	Neighbor of node and depot	Single random collaborator
Overlap and hierarchy variable grouping		
COEA [49], CCPSO [50]	Competition mechanism	Single best collaborator, Best+random competition
CPSO-SL[183]	Interaction degree	Single best collaborator
FEA [180]	Random, neighbor, loci, simple, parents overlap group	A shared memory iteratively optimized by subswarms
DSMGA [232]	Dependency matrix clustering	-
HCCE[110, 102]	Primary motor-premotor cortex	Appropriate individuals
Hybird variable grouping		
MTS [200, 201]	1-D and random grouping	Single best collaborator
D^5 [194, 204]	EDA and perturbation method	-
CCCMAES[90]	Random, min-variance, and max-variance grouping	Single best collaborator

IV. COLLABORATOR SELECTION STRATEGIES OF CCEA

In addition to the problem decomposition, the collaboration method among subproblems is also a key issue of CCEAs [169]. The fitness evaluation of each subpopulation individual requires the collaboration of other subpopulations as shown in Algorithm 1. The way the representative members are selected from other subpopulations to form the complete solution(s) plays an important role in the performance of CCEAs. The existing collaborator selection strategies in CCEAs mainly consider two factors.

1) Collaborator selection pressure, i.e., the greedy degree in choosing a representative member from a subpopulation to form the complete solution(s) of the original problem [223].
2) Collaborator pool size, i.e., the number of the complete problem solutions used to evaluate the fitness of an individual in a particular subpopulation [223].

The collaborator selection strategies, either individual-centric or population-centric [150], [151], are widely used to solve single objective optimization problems (SOPs) and/or multiobjective optimization problems (MOPs).

A. Collaborator Selection Strategies for SOPs

The existing collaborator selection strategies for SOPs construct single or multiple complete problem solutions to estimate the fitness of a subpopulation individual.

1) Collaborator Selection for Single Complete Solution: Many collaborator selection strategies were proposed in the literature to construct a single complete solution, including single best collaborator selection, single worst collaborator selection, random collaborator selection, elite collaborator selection, roulette/tournament-based collaborator selection, linked collaborator selection, and neighborhood-based collaborator selection.

Single best collaborator selection strategy [83], [123], [156], [230] is one of the most widely used strategies in CCEAs. In this strategy, the fitness of the jth individual $O_{i,j}$ in the ith subpopulation is evaluated by $f(b_1, \ldots, b_{i-1}, O_{i,j}, b_{i+1}, \ldots, b_n)$ where b_i is the best individual from the ith subpopulation, i.e., the complete solution is the context vector with the ith component replaced by $O_{i,j}$ [see Fig. 7(a)]. During the evolution of CCEAs, the context vector $\mathbf{b} = (b_1, \ldots, b_n)$ is iteratively optimized, as in the block coordinate descent (BCD) methods [12], [13], [160], [202] used in mathematical programming. This strategy is particularly suitable for optimizing fully separable problems, but not for nonseparable problems [156]. It may also result in Nash equilibria and local optima within large basins of attraction [86], [137].

Single worst collaborator selection, the opposite of single best collaborator selection, combines the worst individuals of the subpopulations in the estimation of the individual fitness [223] as shown in Fig. 7(b). This strategy offers an alternative solution, but it usually is not the optimal one.

Random collaborator selection randomly selects individuals from the subpopulations to form the complete problem solution [223] as shown in Fig. 7(d). For example, the works [96], [138] randomly shuffle the individuals of each subpopulation and then select the ith individual to form the complete solution for fitness evaluation. Random collaborator selection does well in maintaining the diversity of a subpopulation and preventing premature convergence [96], [120], [156], [188]. It outperforms the single best collaborator selection in dynamic optimization with a fast-changing environment, but its randomness can also slow down the local convergence of CCEAs [8]. Comparison studies of the first three strategies were conducted in [96], [152], and [223].

Elite collaborator selection strategy chooses the K best individuals from each subpopulation to form the collaborator pool and then randomly selects individuals from the pool to construct a complete solution [47], as shown in Fig. 7(c). It is equivalent to the single best collaborator selection strategy if K is set to one, and to random collaborator selection if K equals to the subpopulation size. CCEAs can start with a large K to enable a more diverse search and then gradually reduce K to perform a greedier search in the later phase. A comparison of single best, elite, best+random, and best+elite collaboration selection strategies is available in [188].

Roulette/tournament-based collaborator selection chooses representative cooperators from each subpopulation based on roulette or tournament mechanism to form the complete solution [120], [177]. Experimental studies in [1] and [179] showed that roulette-based collaborator selection performs better than single best and random collaborator selection strategies on low-epistasis optimization problems.

Linked collaborator selection [98], [165] combines the linked individuals in all subpopulations as shown in Fig. 7(e). This can relieve the suffering of co-evolution systems from oscillatory phenomenon, caused by the variance of collaborator in the evaluation environment [1]. Linked collaborator selection performs superiorly in optimization problems with high epistasis [1].

Neighborhood-based collaborator selection composes the complete solution of a subpopulation individual with its

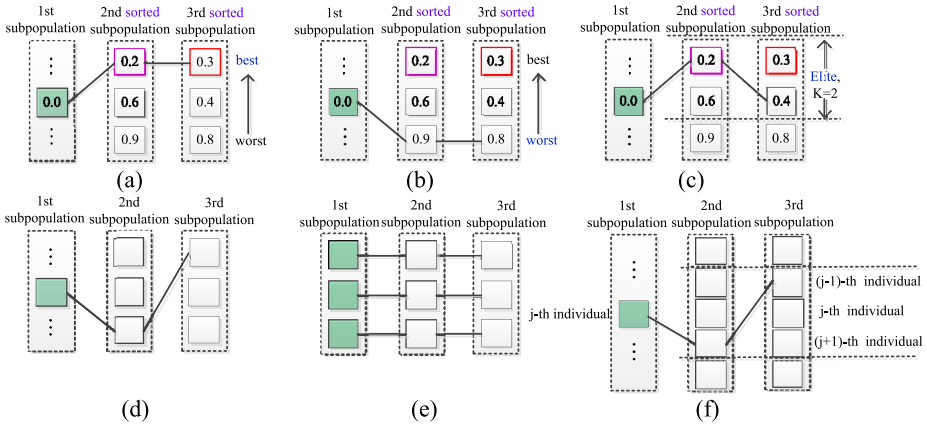

Fig. 7. Collaborator selection strategy for SOPs. (a) Single best collaborator selection. (b) Worst collaborator selection. (c) Elite collaborator selection. (d) Random collaborator selection. (e) Linked collaborator selection. (f) Neighborhood-based collaborator selection. Solid squares are individuals of the first subpopulation undergoing evaluation. White squares are collaborators selected from other subpopulations.

corresponding neighbors from other subpopulations [98]. For example, in Fig. 7(f), considering three neighbors for the evaluation of the fitness of $O_{i,j}$, the collaborator in the kth subpopulation can be $O_{k,j-1}$, $O_{k,j}$, or $O_{k,j+1}$, where $k \neq i$.

2) Collaborator Selection of Multiple Complete Solutions: The collaborator selection strategies can also construct multiple complete solutions to evaluate the fitness of a subpopulation individual. The representative strategies include complete collaborator selection, archive-based collaborator selection, and hybrid collaborator selection.

In complete collaborator selection strategy, each individual in a subpopulation is evaluated by combining it with all candidate combinations from other subpopulations [135], [137], [141]. If there are m subpopulations, and each subpopulation contains N individuals, then N^{m-1} fitness evaluations are required to evaluate the fitness of an individual in a subpopulation. The best fitness value of the N^{m-1} complete solutions is assigned to the individual. Complete collaborator selection can guarantee the convergence to the global optima, if the subpopulation size is large enough [141].

Archive-based collaborator selection was proposed in the Pareto dominance[2] based CCEA (pCCEA) [19]. In each generation, pCCEA evaluates each individual in a subpopulation with all individuals in the nondominated archive, i.e., N_a complete solutions are generated where N_a is the size of the nondominated archive in the subpopulation. The problem of pCCEA is that N_a tends to increase infinitely as the algorithm evolves. This issue is solved in iCCEA [138] by recording the "informative" collaborators, i.e., the individuals that can change the relative ranks of two individuals in the other subpopulation. The studies [168]–[170] put forward a sharing reference archive with a fixed size for all cooperating subpopulations.

Hybrid collaborator selection combines multiple collaborator selection strategies to form complete problem solutions. For example, the best+random collaborator selection is a hybrid of single best collaborator selection and random collaborator selection proposed in [147], [156], and [238]. The first complete solution in best+random collaborator selection combines an individual in a subpopulation with the best individuals of other subpopulations, and the remaining complete solutions successively combine the individual in a subpopulation with randomly selected individuals from other subpopulations [51], [62], [138], [147], [238]. Best+elite collaborator selection hybridizes the single best collaborator selection and the elite collaborator selection [188]. Two complete solutions are generated, i.e., the first one combines the individual in a subpopulation with the best individuals from other subpopulations, and the second one combines the individual with randomly selected elite individuals from other subpopulations. The better of the two completed solutions is reserved.

CCEAs can also benefit from a time-dependent/adaptive collaborator selection strategy by decreasing the number of collaborators over time. For example, in [136], ten complete solutions are used for the first five evolution generations to better explore the joint search space, and two complete solutions for the rest of the generations to speed up the convergence. Further to this, adaptive number of collaborators was introduced in [140] considering population diversity and operator success rate. Reviews and comparison studies of collaborator selection strategies are provided in [135] and [223].

B. Collaborator Selection Strategies for MOPs

Some of the previously described strategies for SOPs, like single best collaboration selection [186]–[188], best+random collaborator selection [73], [238], elite collaborator selection [188], roulette/tournament-based collaborator selection [44], [118], and linked collaborator selection [98] are applicable to MOPs too. Apart from these strategies, this section introduces three specialized strategies for MOPs.

The first strategy is a stochastic approach in which multiple nondominated solutions are randomly selected as the representative collaborators of subpopulations [3], [64], [239]. If multiple complete solutions are generated, the nondominated one among these is used to estimate the fitness

[2]Each subpopulation individual is evaluated with different context vectors in a single-objective optimization. An individual $O_{i,j}$ dominates another one $O_{i,k}$ if and only if each fitness of $O_{i,j}$ using a unique context vector is better than the corresponding fitness of $O_{i,k}$.

TABLE V
SUMMARY OF STUDIES ON COLLABORATOR SELECTION STRATEGIES IN CCEAS

Algorithm	Collaborator selection pressure	Collaboration pool size
SOPs		
CCPSO2[83],DCCEA[188], CCGA[156],DECC-DG[123]	Single best collaborator selection	1
CCA [223]	Single worst collaborator selection	1
rCCEA-Perm [138], CCA[96, 152, 223]	Random collaborator selection	1 or more
DCCEA[188],CCCDE[47]	Elite collaborator selection	1
GP-ADF[1, 177], CoES [120, 179]	Roulette/tournament based collaborator selection	1
MOEA/DVA [98] [165],GP-ADF[1]	Linked collaborator selection	1
cCCEA[141],iCCEA[137]	Complete collaborator selection	N^{m-1}
pCCEA[19],iCCEA[138] [168], Pareto CCGA [170]	Archive-based collaborator selection	N_a or more
DCCEA[188],mCCEA[147] CCGA[156],rCCEA [138] PMOCCA[238]	Hybrid collaborator selection	2 or more
CCEA[136, 140]	Time-dependent collaborator selection	2 or more
MOPs		
NSCCGA[64],CCGDE3[3], DNSGAII-CO[227]	Random member from the non-dominated solutions	1 or more
IBCCMOEA[5]	Individual with the largest hypervolume contribution	1
MOCCA-II[239], CEPSO[240]	Non-dominated solution with the better crowding distance	1 or 2
Bi-MOSU,Tri-MOSU[52]	Knee point from the non-dominated solutions	1
NashCC [4]	Individual whose objective values is a Nash equilibrium	1
COEA/dCOEA[49] CCPSO[50]	Competitive and cooperative mechanisms	1

of subpopulation individuals [227]. The second method is a preference-based approach. The nondominated individual, which can be the knee point [52], the solution of best hypervolume value [5], the Nash equilibrium individual [4], or the solution of the better crowding distance [239], [240], is chosen from each subpopulation as the representative to form a complete solution. The selected nondominated individual (the representative member of subpopulation) usually has the best diversity and convergence (more details are provided in Appendix D of the supplementary material). The third method, a competition-based approach, was proposed in [49] and [50] to select the representative subpopulation individual for cooperation. Each subpopulation competes with other subpopulations to represent the shared variables and the better subpopulation is given a higher probability.

A summary of the collaborator selection strategies is presented in Table V.

V. INDIVIDUAL FITNESS ASSIGNMENT AND SUBPROBLEM RESOURCE ALLOCATION

Individual fitness assignment and subproblem resource allocation are closely related to the collaborator selection strategies. The individual in a subpopulation serves as an essential component of the complete solution of the original problem. How to evaluate the fitness of an individual and how to reasonably allocate the computational resource for subproblems are two key issues to be solved in the design of efficient CCEAs [124], [127], [222]. This section reviews the existing methods of individual fitness assignment and subproblem resource allocation in CCEAs.

A. Individual Fitness Assignment

The fitness of a subpopulation individual can be estimated based on how well the individual cooperates with other subpopulations to form the complete solutions [156]. The common strategies of individual fitness assignment are based on one or multiple complete problem solutions. Constructing a complete problem solution to estimate the fitness of an individual in a subpopulation has a strong bias in the evolutionary process [17], [139], [140], [221]. An alternative is to enroll the individual in multiple complete solutions. The fitness values of these complete solutions can be aggregated into a single value to evaluate the fitness of the individual [150]. The aggregation can take the best [102], [156], [168], the average [107], the worse [223], the biasing technique introduced by Panait *et al.* [141], [142], or the tournament selection of the winner [223]. Wiegand *et al.* [223] provided a comparative study of these methods.

Other measures are also available for individual fitness assignment in CCEAs. For example, Pareto dominance techniques were used to assign a fitness to an individual in [19], [41], [69], and [168]. To save the computational resource, the fitness inheritance method [55], [56] uses a weighted average of parent individuals' fitness to estimate the fitness of some offspring individuals. The fitness of an individual in CCEAs can be estimated by niching methods in order to maintain the diversity of subpopulations [49], [64], [187], [239].

Heuristic estimation methods were studied in training a neural network with CCEAs [74], [107], [219]. For example, a heuristic estimation method introduced in [74] and [219] uses the weights of the neurons (individuals) to estimate their contribution/fitness. The neuron fitness can also be evaluated simply on the basis of the average fitness of the good networks in which the neuron participates [107], credit sharing along orthogonal dimensions [74], or weighted voting of network outputs [15].

B. Computational Resource Allocation of Subproblems

As fitness evaluation consumes most of the computational resources in CCEAs, reasonable computational resource allocation[3] for different subproblems should be ensured [127]. Round-Robin strategy, proposed in CCGA, is the most widely used way to assign the same amount of computational resource to each subproblem. The studies [54], [124], [199] proposed improved strategies considering the imbalanced contributions of subproblems. It is reasonable to invest more efforts in solving more influential subproblems [124].

The existing computational resource allocation methods for subproblems are mainly based on random strategy, the long-term contribution of subproblem, a recent contribution of subproblem, and different subpopulation sizes.

In random strategy [220], the computational resource allocated to a subproblem is based on the number of tags associated with the subproblem. The binary tags are mutated in the evolution process.

[3]The number of fitness evaluations used to evolve a subpopulation before switching to other subproblems.

The method based on long-term contribution estimates the credit of a subproblem by its accumulated improvement in solving the original problem. Two contribution-based cooperative co-evolutionary (CBCC) algorithms, i.e., CBCC1 and CBCC2, were proposed in [127]. They both optimize each subproblem with the same computational resource in the exploration phase. In the exploitation phase, CBCC1 optimizes the most contributing subproblem just one more time, whereas CBCC2 continuously optimizes the most contributing subproblem until it cannot be improved anymore. A sensitive analysis of CBCCs is available in [72].

The computational resource of subproblems can also be allocated based on their recent improvements [122], [127], [197]. For instance, CBCC3 [122], unlike CBCC1 and CBCC2, optimizes only the most contributing subproblem in the current exploitation phase. Cooperative coevolution with adaptive optimizer iterations [197] evolves each subproblem with computational overhead proportional to its recent contribution. CCFR [228] estimates the recent contribution of a subproblem, based on its average contribution value in the last two generations. In [158], the area under the curve measure was used to calculate the recent contribution of a subproblem and a dynamic multiarmed bandit was used to choose the most promising subproblem.

Different subproblems can use different subpopulation sizes lending to different fitness evaluations. To evaluate the performance of a CCEA, the effects of the subpopulation size and the number of subproblems were studied in [199]. In [153], the performance of CCEA was investigated with different subpopulation sizes on oneRidge and twoRidges problems.

The methods of individual fitness assignment and subproblem resource assignment are summarized in Table VI.

TABLE VI
SUMMARY OF STUDIES ON INDIVIDUAL FITNESS EVALUATION AND COMPUTATIONAL RESOURCE ASSIGNMENT OF SUBPROBLEMS IN CCEAS

Fitness/credit assignment	Problem decomposition	Collaboration strategy
Individual fitness assignment		
Best [136, 156, 188]	1-D decomposition, hierarchical grouping	Best+random, random collaborator
Average [107]	A neuron population, a network population	Proportional elite collaborator selection
Worst [223]	1-D decomposition	Best+random
Tournament [223]	1-D decomposition	Random
Fitness inheritance [55, 56]	Random grouping Delta grouping	Parents Single best collaborator
Dominate rank [239]	1-D decomposition	Best+random collaborator
Multi-fitness measure [19, 168, 170]	1-D decomposition	Reference sharing
Weights-based assignment [74, 219]	A neuron population, a network population	5 good related networks
Credit sharing [74, 219]	A neuron population, a network population	its participating networks
Computing resource allocation of subproblem		
Total improvements from the start[72, 127]	Delta group ideal decomposition	Single best collaborator
Recent contribution [122, 228]	Ideal decomposition	Single best collaborator
AUC measure[158]	Static group	Single best collaborator
Accumulating variable important degree [99]	Variable importance + K-means clustering	single best collaborator
The number of subcomponent tagging[220]	1-D decomposition	single best collaborator
Average utility of once evaluation [199]	Random grouping	Single best collaborator

VI. IMPLEMENTATION OF CCEAS

The common issues of the cooperation co-evolution paradigm design have been discussed in the previous sections. This section explores how CCEAs can be implemented with different meta-heuristic algorithms and parallelism.

A. CCEAs Implemented in Different Meta-Heuristic Algorithms

The main motivation to integrate cooperation co-evolution into meta-heuristic algorithms is to improve their performance by adopting the divide-and-conquer strategy. Most CCEAs are based on GA [28], [38], [49], [156], [188], [193], PSO [11], [50], [83], [180], [183], [186], DE [5], [30], [98], [123], [171], [230], GP [1], [15], [39], [87], [118], [175], evolution strategy [89], [90], [104], [120], artificial bee colony [80], [132], ant colony optimization (ACO) [37], [208], competitive co-evolution [26], [49], [50], artificial immune system [93], surrogate-assisted search [48], [130], [185], and memetic algorithm [24], [103]. Among them, DE and PSO are the two most widely used meta-heuristic algorithms for implementing CCEAs and better suit for continuous optimization [5], [11], [123], [186]. ACO base CCEAs are good at solving combinatorial optimization problems [37], [208]. Cooperation co-evolution with surrogate-assisted search is well-suited for expensive optimization [48], [130]. GP-based CCEAs, using a different representation from the general CCEAs, are developed to facilitate task decomposition under decision making tasks [15], [87], [175]. Memetic algorithms and hybrid methods are accomplished in taking advantages of different component methods to solve large-scale optimization [123], [191].

CCEAs and the conventional EAs can complement well with each other. CCEAs are easily trapped into suboptimal solution on fully nonseparable problems. Blending a CCEA with different meta-heuristic algorithms can avoid the premature convergence problem of CCEAs [26], [34]. More details of the CCEA implementations introduced in this section can be found in Appendix G of the supplementary material.

B. Parallel Implementation of CCEAs

It is known that the computational cost of evolutionary optimization increases is proportional to the size and complexity of the problem. Fortunately, the population-based search and divide-and-conquer strategy make CCEAs particularly suitable for parallel computing. For example, in [85], the optimal reactive power flow problem was decomposed into five subproblems based on the domain knowledge and solved by a three-level parallel computing topology on a PC-cluster. A parallel co-evolutionary immune PSO algorithm, based on GPU (G-PCIPSO) [94], was developed for permanent magnet synchronous machines problem. A parallel MOCCGA was implemented in [73]. In a distributed CCEA (DCCEA) [188], subpopulations are divided into groups based on the number of core processors available in each computer. Each core processor maintains its own representatives and archive, and the offspring solutions are generated through collaboration within

the same core processor. A parallel CCEA was proposed for multiobjective optimization in [38], where unidimensional decomposition and a cooperation strategy based on random nondominated solution were used. Another parallel CCEA was introduced in [212] to learn the ranking in Web search. Parallel CCEAs were also proposed to solve vehicle routing problems in [120]. A summary of the studies on the existing parallel CCEAs is presented in Table XXV of the supplementary material.

VII. Benchmark Test Problems, Theoretical Analyses, and Control Parameters of CCEAs

For a systematic comparison and evaluation of different CCEAs, it is beneficial to introduce benchmark functions of various features and different levels of difficulty, theoretical analyses, and the settings of control parameters.

A. Benchmark Test Problems

The earliest benchmark test functions of CCEAs are continuous functions, including Rastrigin, Schwefel, Griewangk, and Ackley problems [156]. To simplify the theoretical analyses of CCEAs and to test the CCEAs, pseudo-Boolean problems $f : \{0, 1\}^n \rightarrow R$, including OneMax, LeadingOnes, LeadingOnesBlocks (LOB), concatenated LOB (CLOB), Trap, and (m,s)-separable problems were introduced in [67]. In [139], the maximum of two quadratics problem was put forward to study the relative overgeneralization pathology of CCEAs. Multirobot systems [51], game theory [18], [148], [151], and NK-landscape problem [154] were also suggested in the benchmarks of CCEAs. Many challenging test problems, especially large-scale optimization problems, have also been proposed in special sessions and competitions of well-known evolutionary computation conferences [81], [190], [191] (see Appendix A of the supplementary material for more details).

B. Theoretical Analyses on CCEAs

The majority of CCEA-versus-EA comparisons were experimentally validated in [154], [221], and [224]. However, theoretical analyses on CCEAs are more favorable to understand the underlying behavior of the algorithms.

Many theoretical analyses of CCEAs were done on pseudo-Boolean problems, $f : \{0, 1\}^n \rightarrow R$ because it is more intuitive to see the effects of the genetic operators based on binary representation [221]. The algorithm performance of a CCEA can be theoretically analyzed based on the expected optimization time [221], i.e., the number of function evaluations spent by the CCEA on average before achieving the global optimum for the first time. As shown in [65], the expected optimization time of cooperative co-evolutionary $(1 + 1)$ EA is $O(n\log n)$ on OneMax problem and $\Theta(n)$ on LeadingOnes problem. It is much less than the expected optimization time of $(1+1)$ EA on the CLOB problem. Nevertheless, cooperative co-evolutionary $(1 + 1)$ EA fails to find the global optimum of Trap problem with a probability $1 - 2^{-\Omega(n)}$ [66]. Besides, the expected optimization time of cooperative co-evolutionary $(1+1)$ EA is more than that of $(1 + 1)$ EA on (k, l)-separable optimization problem [67]. The work [166] presents a novel framework to demonstrate that some classes of CCEAs do have free lunches, based on a weak preference relation.

The theoretical analyses of CCEAs also have gained significant research achievements on continuous optimization. Under some regularity conditions, Bezdek *et al.* [12] proved that the BCD algorithm can converge to a local minimum with linear complexity. Tseng [202] showed that the BCD algorithm can reach a stationary point and the coordinate-wise minimum point without oscillation. A randomized BCD method [160] was demonstrated to get a ϵ-accurate solution with the minimum probability of $1 - p$ using at most $O([m/\epsilon]\log[1/p])$ iterations. The work in [13] verified the global convergence of two BCD variants. Omidvar *et al.* [123] provided a theoretical foundation for identifying interacting variables on additively separable optimization problems. The FDA with truncation selection was proven to converge globally on additively decomposable optimization problems in [236]. Further, Zhang and Mühlenbein [237] proved that the FDA with proportional selection converges globally.

C. Control Parameters of CCEAs

The key parameters that control the performance of CCEAs are the number of subcomponents, subpopulation size, and computation resource allocation.

The setting of the number of subcomponents is usually dependent on problem and the type of variable grouping strategy used. In static variable grouping [11], [154] and random variable grouping [200], [230], the number of subcomponents is fixed in advance, whereas in other strategies, it could be implicitly controlled by other parameters. For example, in interaction learning-based variable grouping strategies, the number of subcomponents is decided by the number of interactions in LIEM [203], the number of neighbors in loci-based overlap grouping [180], the threshold of variable interaction in LINC-R [193], and DECC-DG [123], the threshold of PCC in CCEA-AVP [159], and the threshold of directed variable interaction in MEE [185].

Using different subpopulation sizes for different subproblems is relatively less investigated. The representative works include [54], [153], and [199]. Particularly, the works [153], [199] study the effects of the size of subpopulations and the number of subproblems on the performance of CCEAs. The work [54] suggested three rules to adjust the subpopulation size adaptively.

The representative parameters to configure reasonable allocation of computational resource for subproblems include the number of fitness evaluations [72], [127], [199], the control parameter U in CCFR [228], and the balance factor in multiarmed bandit selection [158].

VIII. Pathological Issues of CCEAs

In CCEAs, individual fitness in a subpopulation is evaluated based on collaborators. With incorrect problem decomposition, individual fitness estimation in CCEAs could suffer from pathological issues that are not encountered in traditional EAs [71], [221]. Typical issues among these are

Red Queen effect, relative overspecialization, and the loss of gradient [70], [221].

Red Queen effect is used to describe the phenomenon that some subpopulation individuals that are worse than the others could become better in the course of evaluation merely due to the change of collaborators [42], [150], [221]. The evolutionary change of two co-evolving subpopulations can be an oscillatory process [70], [131].

The relative overgeneralization issue [140], [142], [224] refers to propensity of CCEA to sample the collaborator at the wider peaks, which may bias the search in favor of suboptimal/robust/equilibrium solutions with a large attraction valley rather than global optimal solution (an example is provided in Appendix F of the supplementary material). Premature convergence of CCEAs to robust/equilibrium solutions has to be distinguished from the local convergence problems of noncoevolutionary algorithms [67], [134]. In dealing with this issue, Bucci and Pollack [19] proposed a Pareto coevolution mechanism to promote the discovery ability of subpopulations. The method introduced in [152] generates several random complete solutions and uses an optimistic reward scheme to bias co-evolution towards globally optimal solutions. Panait [134] demonstrated that this optimistic reward scheme can guarantee convergence to a global optimum if the subpopulation size and the number of collaborations are sufficiently large. Panait *et al.* [138] reduced the number of evaluations by maintaining an archive of informative collaborations. To prevent convergence to equilibrium states, novelty search was integrated with CCEAs by rewarding individuals that generate novel collaborations in [51].

Gradient loss means that the diversity of the subpopulations suddenly decreases such that the others search only a static projection and not the full problem space [150], [225]. Gradient loss is caused by the flat fitness distribution of at least one subpopulation, and this gives rise to genetic drift [214]. To deal with this issue, some locality constraints in selection and collaborator interactions were suggested in [119], and spatially distributed schemes to select interactions and parents were introduced in [105] and [225].

TABLE VII
SUMMARY OF APPLICATIONS OF CCEAS TO REAL WORLD PROBLEMS

Areas	Types of CCEA applied and references
Neural network training	[26, 27, 155],SANE[74, 107],CCME[116]
Game theory	[106],SBB GP[15],coevolutionary games[148]
Multi-agent learning	CCA[51, 136, 139–141],pCCEA[19],NEAT[178]
Scheduling problem	[2, 38, 119, 132],CCGA[2]
Route planning	[9],CCES[121],RDG-MAENS[103]
Feature selection	Two cooperative ant colonies[210], SBB GP[39]
Clustering and classification	[1, 39, 60, 75, 80, 179]
Image processing	MOSU[52],CCDE[21],MABC[80],[242]
Manufacturing	CCGA[28], CCABC[132]
Traffic engineering	CCDG[53], CoBRA[77]
Fuel and energy	CCDE[85], hybird CCGA[177]
Bioinformatics	CoCoMi[59]
Information retrieval	CCRank[212]
Multiplexer problem	Co-evolving GP [1]
Community detection	[59],CCDE[63]
Space structure design	CCGA[115]
Bi-level optimization	CoBRA[77]
Rubiks cubes	Coevolving deep hierarchies of programs [175]
Inventory control optimization	CCGA[40]
Time series prediction	Cooperative-competitive genetic evolution [219]
Predator-prey pursuit	Novelty-driven CCEA[51]
Cancer diagnosis	Fuzzy cooperative coevolution [146]
Satellite-module layout	Dual-system variable-grain CCEA [192]
Intrusion detection	CCAIS[29]
Waveform inversion	DE with CC and SA [210]
Wind turbine prototypes	CCEA with surrogate-assisted search [157]
Air vehicles design	Multiple-species coevolution [20]
Endurance World Championship	CCGA[25]
Finite-state machine	CCEA and model pepresentation [36]
Shortest common superstring	CCEA with co-puzzle algorithm [233]
RFID network planning	CABC[97]
Steelmaking casting	CCABC[132]
Welded beam design	CPSO[58]
Permanent magnet machines	G-PCIPSO[94]
Reliability optimization	CCDE with harmony search [211]
Decoupled optimal design	CCGA[226]
Brain-lesion modelling	Hierarchical coevolutionary method [101]
Assembly line balancing	CPSO[84]

IX. APPLICATIONS OF CCEAS

CCEAs have achieved a great success in the field of optimization domain and attracted a lot of attention from various science and engineering domains.

A. Applications of CCEAs on Continuous Optimization

To solve constrained optimization problems (COPs), CCEAs have to deal with the infeasible individuals. Most CCEAs deal with COPs based on penalty functions [58], [234], cooperative subpopulations [88], multiobjective optimization [43], Lagrangian multiplier method [46], [76], and preference to feasible solutions over infeasible solutions [2].

For dynamic optimization problems, the key requirement for CCEAs is to maintain the population diversity with efficient exploring ability in a dynamic environment. The related methods are mainly based on increasing diversity after a change [7], maintaining diversity throughout the whole evolution [8], memory-based methods [49], prediction-based methods [91], [227], and multipopulation methods [167].

Many CCEAs have been proposed to solve MOPs. Based on how they evaluate the fitness of an individual, these methods can be categorized into indicator-based method [5], decomposition-based method [98], and Pareto domination-based method [3], [49], [64], [73], [186], [188]. A different classification method of CCEAs for MOPs can be found in [6].

B. Applications of CCEAs on Combinatorial Optimization

CCEAs are also widely used to solve combinatorial optimization problems, including traveling salesman problem [9], vehicle routing problem [103], [120], clustering and classification problem [39], [60], [75], [80], job-shop scheduling problem [118], resource scheduling of multiple composite Web services [2], steelmaking-continuous casting scheduling [132], network optimization [53], bin-packing problem [9], coloring problem [14], layout design [192], supply chain design [28], checkerboard problem [10], Rubiks Cube problem [175], and assembly line balancing problem [84].

Owing to the space limit, detailed descriptions of the CCEAs on the aforementioned applications are provided in Appendix H of the supplementary material. A summary of the typical engineering applications of CCEAs is provided in Table VII.

X. Conclusion and Potential Directions

With the growing complexity of real-world optimization problems, CCEAs have received increasing attention for solving complex optimization problems. This paper attempts to provide a comprehensive survey of CCEAs, covering problem decomposition, collaborator selection, individual fitness evaluation, subproblem resource allocation, implementations, benchmark test problems, control parameters, theoretical analyses, and applications. The survey is expected to provide a general overview of the development of CCEAs and insights into the design of the algorithms. Notwithstanding the numerous publications that appeared on CCEAs in the last two decades, many important problems remain unsolved and new application areas of CCEAs keep continually emerging. Possible directions for solving some important problems of CCEAs are discussed below.

1) *Problem Decomposition and Variable Linkage Learning:*
   a) In perturbation-based variable grouping, it takes too many function evaluations $O(n^2)$ to detect variable linkages using pairwise fashion. Only a few efforts have been made to deal with this issue [61]. A potential possibility is to develop a fast interaction identification method for the nonseparable relationship between one variable and a group of variables, or between two groups of variables.
   b) Most of the existing overlap and hierarchy variable grouping are performed manually or based on domain knowledge. How to design an intelligent strategy for overlap and hierarchy variable grouping still remains an open issue.
   c) Although many variable grouping strategies have been proposed, efficient variable grouping strategies are still scarce, especially for combinatorial optimization, dynamic optimization, and robust optimization.
   d) Quantitative analyses on the deterioration in the performance of a CCEA under incorrect variable grouping are required.
2) *Cooperation Among Subproblems:*
   a) If interactions among subcomponents are inevitable, then there is a lack of studies on selecting the most suitable collaborator selection strategy and individual fitness assignment for different problems.
   b) Adaptive/time-dependent collaborator selection has not been well studied and it could be a candidate solution to deal with the linkage learning errors and pathological issues.
   c) More studies should be devoted to designing new collaborator selection strategies for MOPs.
3) *Individual Fitness Assignment and Subproblem Resource Allocation:*
   a) It is unclear as to when multiple fitness evaluations should be used for single individual and which individual deserves multiple fitness evaluations.
   b) Not enough studies were carried out on the effects of using different subpopulation sizes for different subproblems. Using adaptive subpopulation size is a potential candidate to assign the computation resource reasonably.
4) *Subproblem Optimizer:*
   a) LSGO problems with rotation transformation pose a challenge for CCEAs. Integrating covariance matrix adaptation-based mutations [89], [90], [104] with CCEAs and approximating the coordinate transformation [95] could be two promising solutions.
   b) Not much work was done on integrating adaptive selection of multiple subproblem optimizers into CCEAs.
5) BCD is a recent popular algorithm in mathematical programming and is very similar to CCEAs. However, it has not received the attention it deserves in the domain of CCEAs. Borrowing theories and techniques of BCD algorithms can help the development of CCEAs.
6) Theoretical analyses of CCEAs are still in their infancy. For most of the real-world application problems, there is no answer to "How long do CCEAs take to reach the global optimum" and "How likely are the CCEAs to get there?"
7) There is a lack of a clear mapping between problem features and the best-suited CCEAs.
8) The practitioners specifically need to provide a good understanding of what coevolution is good at, what it is not, and why.
9) Most CCEAs are based on decomposition of decision variables, but very few on decision space or objective space decomposition.
10) Studies extending CCEAs to many-objective optimization [23], [189] and multiobjective with large-scale decision variables [3], [22], [98] are very few.
11) There is a lack of CCEAs together with niching method.
12) Besides mutualism relationship, different species can have competitive and exploitative relationships in natural ecosystems. An interesting direction would be to model the coevolution of species having different types of ecological relationships.
13) Evolutionary game theory (EGT) is a potential but less studied tool to understand the dynamics of CCEAs in the process of evolution [224]. Using EGT for CCEAs is worth studying.

References

[1] M. Ahluwalia, L. Bull, and T. Fogarty, "Co-evolving functions in genetic programming: A comparison in ADF selection strategies," in *Proc. 2nd Annu. Conf. Genet. Program.*, 1997, pp. 3–8.
[2] L. F. Ai, M. L. Tang, and C. Fidge, "Resource allocation and scheduling of multiple composite Web services in cloud computing using cooperative coevolution," in *Proc. Int. Conf. Neural Inf. Process.*, Shanghai, China, 2011, pp. 258–267.
[3] L. M. Antonio and C. A. C. Coello, "Use of cooperative coevolution for solving large scale multiobjective optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, Cancún, Mexico, 2013, pp. 2758–2765.
[4] L. M. Antonio and C. A. C. Coello, "A non-cooperative game for faster convergence in cooperative coevolution for multi-objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, Sendai, Japan, 2015, pp. 109–116.

[5] L. M. Antonio and C. A. C. Coello, "Indicator-based cooperative coevolution for multi-objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2016, pp. 991–998.

[6] L. M. Antonio and C. A. C. Coello, "Coevolutionary multi-objective evolutionary algorithms: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, to be published, doi: 10.1109/TEVC.2017.2767023.

[7] C.-K. Au and H.-F. Leung, "On the behavior of cooperative coevolution in dynamic environments," in *Proc. IEEE Congr. Evol. Comput.*, Hong Kong, 2008, pp. 2827–2836.

[8] C.-K. Au and H.-F. Leung, "Investigating collaboration methods of random immigrant scheme in cooperative coevolution," in *Proc. IEEE Congr. Evol. Comput.*, Trondheim, Norway, 2009, pp. 2701–2707.

[9] S. Baluja, "Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning," School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, USA, Rep. CMU-CS-94-163, 1994.

[10] S. Baluja and S. Davies, "Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space," in *Proc. Int. Conf. Mach. Learn.*, 1997, pp. 30–38.

[11] F. V. D. Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 225–239, Jun. 2004.

[12] J. C. Bezdek, R. J. Hathaway, R. E. Howard, C. A. Wilson, and M. P. Windham, "Local convergence analysis of a grouped variable version of coordinate descent," *J. Optim. Theory Appl.*, vol. 54, no. 3, pp. 471–477, 1987.

[13] M. Blondel, K. Seki, and K. Uehara, "Block coordinate descent algorithms for large-scale sparse multiclass classification," *Mach. Learn.*, vol. 93, no. 1, pp. 31–52, 2013.

[14] J. S. D. Bonet, C. L. Isbell, Jr., and P. A. Viola, "MIMIC: Finding optima by estimating probability densities," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 1997, pp. 424–430.

[15] J. P. C. Bonson, S. Kelly, A. R. McIntyre, and M. I. Heywood, "On synergies between diversity and task decomposition in constructing complex systems with GP," in *Proc. Genet. Evol. Comput. Conf. Companion*, Denver, CO, USA, 2016, pp. 969–976.

[16] J. L. Bronstein, "Our current understanding of mutualism," *Quart. Rev. Biol.*, vol. 69, no. 1, pp. 31–51, 1994.

[17] A. Bucci, "Emergent geometric organization and informative dimensions in coevolutionary algorithms," Ph.D. dissertation, Michtom School Comput. Sci., Brandeis Univ., Waltham, MA, USA, 2007.

[18] A. Bucci and J. B. Pollack, "Focusing versus intransitivity geometrical aspects of co-evolution," in *Proc. Conf. Genet. Evol. Comput.*, Chicago, IL, USA, 2003, pp. 250–261.

[19] A. Bucci and J. B. Pollack, "On identifying global optima in cooperative coevolution," in *Proc. Conf. Genet. Evol. Comput.*, 2005, pp. 539–544.

[20] M. D. Bugajska and A. C. Schultz, "Coevolution of form and function in the design of micro air vehicles," in *Proc. NASA/DoD Conf. Evol. Hardw.*, Alexandria, VA, USA, 2002, pp. 154–163.

[21] Z.-Q. Cai, L. Lv, H. Huang, H. Hu, and Y.-H. Liang, "Improving sampling-based image matting with cooperative coevolution differential evolution algorithm," *Soft Comput.*, vol. 21, no. 15, pp. 4417–4430, 2017.

[22] B. Cao, J. Zhao, Z. Lv, and X. Liu, "A distributed parallel cooperative coevolutionary multiobjective evolutionary algorithm for large-scale optimization," *IEEE Trans. Ind. Informat.*, vol. 13, no. 4, pp. 2030–2038, Aug. 2017.

[23] B. Cao *et al.*, "Distributed parallel particle swarm optimization for multi-objective and many-objective large-scale optimization," *IEEE Access*, vol. 5, pp. 8214–8221, 2017.

[24] Z. Cao *et al.*, "An effective cooperative coevolution framework integrating global and local search for large scale optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, Sendai, Japan, 2015, pp. 1986–1993.

[25] L. Cardamone, D. Loiacono, and P. L. Lanzi, "Applying cooperative coevolution to compete in the 2009 TORCS endurance world championship," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spainm 2010, pp. 1–8.

[26] R. Chandra, "Competition and collaboration in cooperative coevolution of elman recurrent neural networks for time-series prediction," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 12, pp. 3123–3136, Dec. 2015.

[27] R. Chandra, M. Frean, and M. Zhang, "A memetic framework for cooperative coevolution of recurrent neural networks," in *Proc. Int. Joint Conf. Neural Netw.*, San Jose, CA, USA, 2011, pp. 673–680.

[28] Y.-H. Chang, "Adopting co-evolution and constraint-satisfaction concept on genetic algorithms to solve supply chain network design problems," *Expert Syst. Appl.*, vol. 37, no. 10, pp. 6919–6930, 2010.

[29] J. Chen and D. Yang, "A study of detector generation algorithms based on artificial immune in intrusion detection system," in *Proc. Int. Conf. Comput. Res. Develop.*, Shanghai, China, 2011, pp. 4–8.

[30] W. Chen, T. Weise, Z. Yang, and K. Tang, "Large-scale global optimization using cooperative coevolution with variable interaction learning," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, Kraków, Poland, 2010, pp. 300–309.

[31] Y. P. Chen and D. E. Goldberg, "Convergence time for the linkage learning genetic algorithm," *Evol. Comput.*, vol. 13, no. 3, pp. 279–302, 2005.

[32] K. H. Choi and C. G. Moon, "Generalized extreme value model and additively separable generator function," *J. Econometrics*, vol. 76, nos. 1–2, pp. 129–140, 1997.

[33] B. Colson and P. L. Toint, "Optimizing partially separable functions without derivatives," *Optim. Methods Softw.*, vol. 20, nos. 4–5, pp. 493–508, 2005.

[34] F.-Z. Cui, L. Wang, Z. Z. Xu, X. K. Wang, and H.-F. Teng, "Dual-system cooperative coevolutionary differential evolution algorithm for solving nonseparable function optimization," *Inf. Technol. J.*, vol. 12, no. 9, pp. 1796–1803, 2013.

[35] G. Debreu and T. C. Koopmans, "Additively decomposed quasiconvex functions," *Math. Program.*, vol. 24, no. 1, pp. 1–38, 1982.

[36] G. Dick and X. Yao, "Model representation and cooperative coevolution for finite-state machine evolution," in *Proc. IEEE Congr. Evol. Comput.*, Beijing, China, 2014, pp. 2700–2707.

[37] K. Doerner, R. F. Hartl, and M. Reimann, "Cooperative ant colonies for optimizing resource allocation in transportation," in *Proc. Workshops Appl. Evol. Comput.*, 2001, pp. 70–79.

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

[39] J. A. Doucette, A. R. Mcintyre, P. Lichodzijewski, and M. I. Heywood, "Symbiotic coevolutionary genetic programming: A benchmarking study under large attribute spaces," *Genet. Program. Evol. Mach.*, vol. 13, no. 1, pp. 71–101, 2012.

[40] R. Eriksson and B. Olsson, "Cooperative coevolution in inventory control optimisation," in *Artificial Neural Nets and Genetic Algorithms*. Vienna, Austria: Springer, 1998, pp. 583–587.

[41] S. G. Ficici and J. B. Pollack, "Pareto optimality in coevolutionary learning," in *Proc. 6th Eur. Conf. Adv. Artif. Life*, 2001, pp. 316–325.

[42] P. Funes and E. Pujals, "Intransitivity revisited coevolutionary dynamics of numbers games," in *Proc. Genet. Evol. Comput. Conf.*, 2005, pp. 515–521.

[43] W.-F. Gao, G. G. Yen, and S.-Y. Liu, "A dual-population differential evolution with coevolution for constrained optimization," *IEEE Trans. Cybern.*, vol. 45, no. 5, pp. 1094–1107, May 2015.

[44] N. Garcia-Pedrajas, C. Hervás-Martinez, and J. Munoz-Pérez, "Multi-objective cooperative coevolution of artificial neural networks (multi-objective cooperative networks)," *Neural Netw.*, vol. 15, no. 10, pp. 1259–1278, 2002.

[45] H. Ge, L. Sun, X. Yang, S. Yoshida, and Y. Liang, "Cooperative differential evolution with fast variable interdependence learning and cross-cluster mutation," *Appl. Soft Comput.*, vol. 36, pp. 300–314, Nov. 2015.

[46] B. Ghasemishabankareh, X. Li, and M. Ozlen, "Cooperative coevolutionary differential evolution with improved augmented Lagrangian to solve constrained optimisation problems," *Inf. Sci.*, vol. 369, pp. 441–456, Nov. 2016.

[47] E. Glorieux, B. Svensson, F. Danielsson, and B. Lennartson, "Improved constructive cooperative coevolutionary differential evolution for large-scale optimisation," in *Proc. IEEE Symp. Series Comput. Intell.*, Cape Town, South Africa, 2015, pp. 1703–1710.

[48] C. K. Goh, D. Lim, L. Ma, Y. S. Ong, and P. S. Dutta, "A surrogate-assisted memetic co-evolutionary algorithm for expensive constrained optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, New Orleans, LA, USA, 2011, pp. 744–749.

[49] C.-K. Goh and K. C. Tan, "A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 1, pp. 103–127, Feb. 2009.

[50] C. K. Goh, K. C. Tan, D. S. Liu, and S. C. Chiam, "A competitive and cooperative co-evolutionary approach to multi-objective particle swarm optimization algorithm design," *Eur. J. Oper. Res.*, vol. 202, no. 1, pp. 42–54, 2010.

[51] J. Gomes, P. Mariano, and A. L. Christensen, "Novelty-driven cooperative coevolution," *Evol. Comput.*, vol. 25, no. 2, pp. 275–307, 2015.

[52] M. G. Gong, H. Li, E. H. Luo, J. Liu, and J. Liu, "A multiobjective cooperative coevolutionary algorithm for hyperspectral sparse unmixing," *IEEE Trans. Evol. Comput.*, vol. 21, no. 2, pp. 234–248, Apr. 2017.

[53] X. Guan *et al.*, "A strategic conflict avoidance approach based on cooperative coevolutionary with the dynamic grouping strategy," *Int. J. Syst. Sci.*, vol. 47, no. 9, pp. 1995–2008, 2016.

[54] Y. Guo, X. Cao, H. Yin, and Z. Tang, "Coevolutionary optimization algorithm with dynamic sub-population size," *Int. J. Innov. Comput. Inf. Control*, vol. 3, no. 2, pp. 435–448, 2007.

[55] A. Hameed, D. Corne, D. Morgan, and A. Waldock, "Large-scale optimization: Are co-operative co-evolution and fitness inheritance additive?" in *Proc. IEEE Congr. Evol. Comput.*, 2013, pp. 104–111.

[56] A. Hameed, A. Kononova, and D. Corne, "Engineering fitness inheritance and co-operative evolution into state-of-the-art optimizers," in *Proc. IEEE Symp. Comput. Intell.*, Cape Town, South Africa, 2016, pp. 1695–1702.

[57] G. R. Harik, F. G. Lobo, and D. E. Goldberg, "The compact genetic algorithm," Illinois Genet. Algorithms Lab., Univ. Illinois at Urbana-Champaign, Urbana, IL, USA, Rep. IlliGAL-97006, 1997.

[58] Q. He and L. Wang, "An effective co-evolutionary particle swarm optimization for constrained engineering design problems," *Eng. Appl. Artif. Intell.*, vol. 20, no. 1, pp. 89–99, 2007.

[59] S. He *et al.*, "Cooperative co-evolutionary module identification with application to cancer disease module discovery," *IEEE Trans. Evol. Comput.*, vol. 20, no. 6, pp. 838–858, Dec. 2016.

[60] M. I. Heywood, "Evolutionary model building under streaming data for classification tasks: Opportunities and challenges," *Genet. Program. Evol. Mach.*, vol. 16, no. 3, pp. 283–326, 2014.

[61] X.-M. Hu, F.-L. He, W.-N. Chen, and J. Zhang, "Cooperation coevolution with fast interdependency identification for large scale optimization," *Inf. Sci.*, vol. 381, pp. 142–160, Mar. 2017.

[62] M. Huang, J. Chen, and B. Sun, "A new collaborator selection method of cooperative co-evolutionary genetic algorithm and its application," in *Proc. Int. Conf. Multisensor Fusion Inf. Integr. Intell. Syst.*, Beijing, China, 2014, pp. 1–6.

[63] Q. Huang *et al.*, "Community detection using cooperative co-evolutionary differential evolution," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, 2012, pp. 235–244.

[64] A. W. Iorio and X. Li, "A cooperative coevolutionary multiobjective algorithm using non-dominated sorting," in *Proc. Conf. Genet. Evol. Comput.*, Seattle, WA, USA, 2004, pp. 537–548.

[65] T. Jansen and R. P. Wiegand, "Exploring the explorative advantage of the cooperative coevolutionary (1+1) EA," in *Proc. Conf. Genet. Evol. Comput.*, Chicago, IL, USA, 2003, pp. 310–321.

[66] T. Jansen and R. P. Wiegand, "Sequential versus parallel cooperative coevolutionary (1+1) EAs," in *Proc. IEEE Congr. Evol. Comput.*, Canberra, ACT, Australia, 2003, pp. 30–37.

[67] T. Jansen and R. P. Wiegand, "The cooperative coevolutionary (1+1) EA," *Evol. Comput.*, vol. 12, no. 4, pp. 405–434, 2004.

[68] B. E. Jones and L. Stracca, *Are Money and Consumption Additively Separable in the Euro Area? A Non-Parametric Approach*. Frankfurt, Germany: Soc. Sci. Electron., 2006, pp. 295–303.

[69] E. D. D. Jong, "Towards a bounded Pareto-coevolution archive," in *Proc. IEEE Congr. Evol. Comput.*, Portland, OR, USA, 2004, pp. 2341–2348.

[70] E. D. D. Jong and J. B. Pollack, "Ideal evaluation from coevolution," *Evol. Comput.*, vol. 12, no. 2, pp. 159–192, 2004.

[71] E. D. D. Jong, K. O. Stanley, and R. P. Wiegand, "Introductory tutorial on coevolution," in *Proc. Conf. Companion Genet. Evol. Comput.*, 2007, pp. 3133–3157.

[72] B. Kazimipour, M. N. Omidvar, X. Li, and A. K. Qin, "A sensitivity analysis of contribution-based cooperative co-evolutionary algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Sendai, Japan, 2015, pp. 417–424.

[73] N. Keerativuttitumrong, N. Chaiyaratana, and V. Varavithy, "Multi-objective co-operative co-evolutionary genetic algorithm," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, Granada, Spain, 2002, pp. 288–297.

[74] V. R. Khare, X. Yao, and B. Sendhoff, "Credit assignment among neurons in co-evolving populations," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, Birmingham, U.K., 2004, pp. 882–891.

[75] K. Krawiec and B. Bhanu, "Coevolution and linear genetic programming for visual learning," in *Proc. Conf. Genet. Evol. Comput.*, Chicago, IL, USA, 2003, pp. 332–343.

[76] R. A. Krohling and L. D. S. Coelho, "Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36, no. 6, pp. 1407–1416, Dec. 2006.

[77] F. Legillon, A. Liefooghe, and E.-G. Talbi, "CoBRA: A cooperative coevolutionary algorithm for bi-level optimization," in *Proc. Evol. Comput.*, Brisbane, QLD, Australia, 2012, pp. 1–8.

[78] J. Levenick, "Metabits: Generic endogenous crossover control," in *Proc. Int. Conf. Genet. Algorithms*, 1995, pp. 88–95.

[79] G. Li, C. Rosenthal, and H. Rabitz, "High dimensional model representations," *J. Phys. Chem. A*, vol. 105, no. 33, pp. 7765–7777, 2001.

[80] J.-Y. Li, Y.-D. Zhao, J.-H. Li, and X.-J. Liu, "Artificial bee colony optimizer with bee-to-bee communication and multipopulation coevolution for multilevel threshold image segmentation," *Math. Problems Eng.*, vol. 2015, pp. 1–23, Jan. 2015.

[81] X. Li, K. Tang, M. N. Omidvar, Z. Yang, and K. Qin, "Benchmark functions for the CEC'2013 special session and competition on large-scale global optimization," Evol. Comput. Mach. Learn. Group, RMIT Univ., Melbourne, VIC, Australia, Rep., 2013.

[82] X. Li and X. Yao, "Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms," in *Proc. IEEE Congr. Evol. Comput.*, Trondheim, Norway, 2009, pp. 1546–1553.

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

[84] Z. Li, M. N. Janardhanan, Q. Tang, and P. Nielsen, "Co-evolutionary particle swarm optimization algorithm for two-sided robotic assembly line balancing problem," *Adv. Mech. Eng.*, vol. 8, no. 9, pp. 1–14, 2016.

[85] C. H. Liang, C. Y. Chung, K. P. Wong, and X. Z. Duan, "Parallel optimal reactive power flow based on cooperative co-evolutionary differential evolution and power system decomposition," *IEEE Trans. Power Syst.*, vol. 22, no. 1, pp. 249–257, Feb. 2007.

[86] M. I. Lichbach, *The Cooperator's Dilemma*. Ann Arbor, MI, USA: Univ. Michigan Press, 1996.

[87] P. Lichodzijewski and M. I. Heywood, "Managing team-based problem solving with symbiotic bid-based genetic programming," in *Proc. Genet. Evol. Comput. Conf.*, Atlanta, GA, USA, 2008, pp. 363–370.

[88] B. Liu, H. Ma, X. Zhang, and Y. Zhou, "A memetic co-evolutionary differential evolution algorithm for constrained optimization," in *Proc. IEEE Congr. Evol. Comput.*, Singapore, 2007, pp. 2996–3002.

[89] H. Liu, S. Guan, F. Liu, and Y. Wang, "Cooperative co-evolution with formula based grouping and cma for large scale optimization," in *Proc. Int. Conf. Comput. Intell. Security*, Shenzhen, China, pp. 282–285, 2015.

[90] J. Liu and K. Tang, "Scaling up covariance matrix adaptation evolution strategy using cooperative coevolution," in *Proc. Int. Conf. Intell. Data Eng. Autom. Learn.*, Hefei, China, 2013, pp. 350–357.

[91] R. Liu, Y. Chen, W. Ma, C. Mu, and L. Jiao, "A novel cooperative coevolutionary dynamic multi-objective optimization algorithm using a new predictive model," *Soft Comput.*, vol. 18, no. 10, pp. 1913–1929, 2014.

[92] Y. Liu, X. Yao, Q. Zhao, and T. Higuchi, "Scaling up fast evolutionary programming with cooperative coevolution," in *Proc. IEEE Congr. Evol. Comput.*, 2001, pp. 1101–1108.

[93] Z.-H. Liu, J. Zhang, S.-W. Zhou, X.-H. Li, and K. Liu, "Coevolutionary particle swarm optimization using AIS and its application in multiparameter estimation of PMSM," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 1921–1935, Dec. 2013.

[94] Z.-H. Liu, X.-H. Li, L.-H. Wu, S.-W. Zhou, and K. Liu, "GPU-accelerated parallel coevolutionary algorithm for parameters identification and temperature monitoring in permanent magnet synchronous machines," *IEEE Trans. Ind. Informat.*, vol. 11, no. 5, pp. 1220–1230, Oct. 2015.

[95] I. Loshchilov, M. Schoenauer, and M. Sebag, "Adaptive coordinate descent," in *Proc. Conf. Genet. Evol. Comput.*, 2011, pp. 885–892.

[96] S. Luke, K. Sullivan, and F. Abidi, "Large scale empirical analysis of cooperative coevolution," in *Proc. Conf. Genet. Evol. Comput.*, 2011, pp. 151–152.

[97] L. Ma, K. Hu, Y. Zhu, and H. Chen, "Cooperative artificial bee colony algorithm for multi-objective RFID network planning," *J. Netw. Comput. Appl.*, vol. 42, pp. 143–162, Jun. 2014.

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

[99] S. Mahdavi, S. Rahnamayan, and M. E. Shiri, "Multilevel framework for large-scale global optimization," *Soft Comput.*, vol. 21, no. 14, pp. 4111–4140, 2017.

[100] S. Mahdavi, M. E. Shiri, and S. Rahnamayan, "Cooperative co-evolution with a new decomposition method for large-scale optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2014, pp. 1285–1292.

[101] M. Maniadakis and P. Trahanias, "A hierarchical coevolutionary method to support brain-lesion modelling," in *Proc. Int. Joint Conf. Neural Netw.*, 2005, pp. 434–439.

[102] M. Maniadakis and P. Trahanias, "Assessing hierarchical cooperative coevolution," in *Proc. IEEE Int. Conf. Tools Artif. Intell.*, 2007, pp. 391–398.

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

[104] Y. Mei, M. N. Omidvar, X. Li, and X. Yao, "A competitive divide-and-conquer algorithm for unconstrained large-scale black-box optimization," *ACM Trans. Math. Softw.*, vol. 42, no. 2, pp. 1–13, 2016.

[105] M. Mitchell, M. D. Thomure, and N. L. Williams, "The role of space in the success of coevolutionary learning," in *Proc. 10th Int. Conf. Simulat. Synth. Living Syst. Artif. Life X*, 2006, pp. 118–124.

[106] N. Moran and J. Pollack, "Effects of cooperative and competitive coevolution on complexity in a linguistic prediction game," in *Proc. Eur. Conf. Artif. Life ECAL*, 2017, pp. 298–305.

[107] D. E. Moriarty and R. Miikkulainen, "Forming neural networks through efficient and adaptive coevolution," *Evol. Comput.*, vol. 5, no. 4, pp. 373–399, 1997.

[108] H. Mühlenbein and T. Mahnig, "FDA -A scalable evolutionary algorithm for the optimization of additively decomposed functions," *Evol. Comput.*, vol. 7, no. 4, pp. 353–376, Dec. 1999.

[109] H. Mühlenbein and G. Paaß, *From Recombination of Genes to the Estimation of Distributions I. Binary Parameters*. New York, NY, USA: Springer, 1996.

[110] M. Munetomo, "Linkage identification based on epistasis measures to realize efficient genetic algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, USA, 2002, pp. 445–452.

[111] M. Munetomo, "Linkage identification with epistasis measures considering monotonicity conditions," in *Proc. Asia–Pac. Conf. Simulat. Evol. Learn.*, 2002, pp. 231–238.

[112] M. Munetomo and D. E. Goldberg, "Linkage identification by non-monotonicity detection for overlapping functions," *Evol. Comput.*, vol. 7, no. 4, pp. 377–398, Dec. 1999.

[113] M. Munetomo and D. E. Goldberg, "A genetic algorithm using linkage identification by nonlinearity check," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, Tokyo, Japan, 1999, pp. 595–600.

[114] M. Munetomo and D. E. Goldberg, "Identifying linkage groups by nonlinearity/non-monotonicity detection," in *Proc. Conf. Genet. Evol. Comput.*, 1999, pp. 433–440.

[115] P. B. Nair and A. J. Keane, "Passive vibration suppression of flexible space structures via optimal geometric redesign," *AIAA J.*, vol. 39, no. 7, pp. 1338–1346, 2001.

[116] M. H. Nguyen, H. A. Abbass, and R. I. Mckay, "A novel mixture of experts model based on cooperative coevolution," *Neurocomputing*, vol. 70, nos. 1–3, pp. 155–163, 2006.

[117] M. H. Nguyen, H. A. Abbass, and R. I. Mckay, "Analysis of CCME: Coevolutionary dynamics, automatic problem decomposition, and regularization," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 38, no. 1, pp. 100–109, Jan. 2008.

[118] S. Nguyen, M. J. Zhang, M. Johnston, and K. C. Tan, "Automatic design of scheduling policies for dynamic multi-objective job shop scheduling via cooperative coevolution genetic programming," *IEEE Trans. Evol. Comput.*, vol. 18, no. 2, pp. 193–208, Apr. 2014.

[119] S. L. Nuismer, J. N. Thompson, and R. Gomulkiewicz, "Coevolution between hosts and parasites with partially overlapping geographic ranges," *J. Evol. Biol.*, vol. 16, no. 6, pp. 1337–1945, 2003.

[120] F. B. D. Oliveira, R. Enayatifar, H. J. Sadaei, F. G. Guimarães, and J. Y. Potvin, "A cooperative coevolutionary algorithm for the multi-depot vehicle routing problem," *Expert Syst. Appl.*, vol. 43, pp. 117–130, Jan. 2016.

[121] J. Ollerton, "'Biological barter?' Patterns of specialization compared across different mutualisms," in *Plant-Pollinator Interactions: From Specialization to Generalization*. Chicago, IL, USA: Univ. Chicago Press, 2006, pp. 411–435.

[122] M. N. Omidvar, B. Kazimipour, X. Li, and X. Yao, "CBCC3—A contribution-based cooperative co-evolutionary algorithm with improved exploration/exploitation balance," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2016, pp. 3541–3548.

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

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

[125] M. N. Omidvar, X. Li, Z. Yang, and X. Yao, "Cooperative co-evolution for large scale optimization through more frequent random grouping," in *Proc. IEEE Congr. Evol. Comput.*, 2010, pp. 1754–1761.

[126] M. N. Omidvar, X. Li, and X. Yao, "Cooperative co-evolution with delta grouping for large scale non-separable function optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2010, pp. 1762–1769.

[127] M. N. Omidvar, X. Li, and X. Yao, "Smart use of computational resources based on contribution for cooperative co-evolutionary algorithms," in *Proc. Conf. Genet. Evol. Comput.*, 2011, pp. 1115–1122.

[128] M. N. Omidvar, Y. Mei, and X. Li, "Effective decomposition of large-scale separable continuous functions for cooperative co-evolutionary algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Beijing, China, 2014, pp. 1305–1312.

[129] M. N. Omidvar, M. Yang, Y. Mei, X. Li, and X. Yao, "DG2: A faster and more accurate differential grouping for large-scale black-box optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 6, pp. 929–942, Dec. 2017.

[130] Y. S. Ong, A. J. Keane, and P. B. Nair, "Surrogate-assisted coevolutionary search," in *Proc. Int. Conf. Neural Inf. Process.*, Singapore, 2002, pp. 1140–1145.

[131] L. Pagie and P. Hogeweg, "Information integration and red queen dynamics in coevolutionary optimization," in *Proc. IEEE Congr. Evol. Comput.*, 2000, pp. 1260–1267.

[132] Q.-K. Pan, "An effective co-evolutionary artificial bee colony algorithm for steelmaking-continuous casting scheduling," *Eur. J. Oper. Res.*, vol. 250, no. 3, pp. 702–714, 2016.

[133] L. Panait, "The analysis and design of concurrent learning algorithms for cooperative multiagent systems," Ph.D. dissertation, Dept. Comput. Sci., George Mason Univ., Fairfax, VA, USA, 2006.

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

[135] L. Panait and S. Luke, "A comparison of two competitive fitness functions," in *Proc. Conf. Genet. Evol. Comput.*, 2002, pp. 503–511.

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

[137] L. Panait and S. Luke, "Selecting informative actions improves cooperative multiagent learning," in *Proc. Int. Joint Conf. Auton. Agents Multiagent Syst.*, 2006, pp. 760–766.

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

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

[140] L. Panait, K. Sullivan, and S. Luke, "Lenience towards teammates helps in cooperative multiagent learning," Dept. Comput. Sci., George Mason Univ., Fairfax, VA, USA, Rep. GMU-CS-TR-2013-2, 2008.

[141] L. Panait, R. P. Wiegand, and S. Luke, "Improving coevolutionary search for optimal multiagent behaviors," in *Proc. Int. Joint Conf. Artif. Intell.*, 2003, pp. 653–658.

[142] L. Panait, R. P. Wiegand, and S. Luke, "A visual demonstration of convergence properties of cooperative coevolution," in *Parallel Problem Solving From Nature*. Heidelberg, Germany: Springer, 2004, pp. 892–901.

[143] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz, "BOA: The Bayesian optimization algorithm," in *Proc. Conf. Genet. Evol. Comput.*, 1999, pp. 525–532.

[144] M. Pelikan and H. Mühlenbein, *The Bivariate Marginal Distribution Algorithm*. London, U.K.: Springer, 1999.

[145] M. Pelikan and H. Mühlenbein, "The bivariate marginal distribution algorithm," in *Advances in Soft Computing*. London, U.K.: Springer, 1999, pp. 521–535.
[146] C. A. Pena-Reyes and M. Sipper, "Applying fuzzy CoCo to breast cancer diagnosis," in *Proc. IEEE Conf. Evol. Comput.*, 2000, pp. 1168–1175.
[147] X. Peng, K. Liu, and Y. Jin, "A dynamic optimization approach to the design of cooperative co-evolutionary algorithms," *Knowl. Based Syst.*, vol. 109, pp. 174–186, Oct. 2016.
[148] M. Perc and A. Szolnoki, "Coevolutionary games—A mini review," *Bio Syst.*, vol. 99, no. 2, pp. 109–125, 2010.
[149] P. F. Perroni, D. Weingaertner, and M. R. Delgado, "Automated iterative partitioning for cooperatively coevolving particle swarms in large scale optimization," in *Proc. Braz. Conf. Intell. Syst.*, 2015, pp. 19–24.
[150] E. Popovici, "An analysis of two-population coevolutionary computation," Ph.D. dissertation, Dept. Comput. Sci., George Mason Univ., Fairfax, VA, USA, 2006.
[151] E. Popovici, A. Bucci, R. P. Wiegand, and E. D. D. Jong, "Coevolutionary principles," in *Handbook of Natural Computing*. Heidelberg, Germany: Springer, 2012, pp. 987–1033.
[152] E. Popovici and K. D. Jong, "A dynamical systems analysis of collaboration methods in cooperative co-evolution," in *Proc. AAAI Fall Symp. Coevolutionary Coadaptive Syst.*, 2005, pp. 1–8.
[153] E. Popovici and K. D. Jong, "Understanding cooperative coevolutionary dynamics via simple fitness landscapes," in *Proc. Conf. Genet. Evol. Comput.*, 2005, pp. 507–514.
[154] M. A. Potter, "The design and analysis of a computational model of cooperative coevolution," Ph.D. dissertation, Dept. Comput. Sci., George Mason Univ., Fairfax, VA, USA, 1997.
[155] M. A. Potter and K. A. De Jong, "Cooperative coevolution: An architecture for evolving coadapted subcomponents," *Evol. Comput.*, vol. 8, no. 1, pp. 1–29, 2000.
[156] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, 1994, pp. 249–257.
[157] R. J. Preen and L. Bull, "Toward the coevolution of novel vertical-axis wind turbines," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 284–294, Apr. 2013.
[158] F.-M. D. Rainville, M. Sebag, C. Gagné, M. Schoenauer, and D. Laurendeau, "Sustainable cooperative coevolution with a multi-armed bandit," in *Proc. Conf. Genet. Evol. Comput.*, 2013, pp. 1517–1524.
[159] T. Ray and X. Yao, "A cooperative coevolutionary algorithm with correlation based adaptive variable partitioning," in *Proc. IEEE Congr. Evol. Comput.*, 2009, pp. 983–989.
[160] P. Richtárik and M. Takáč, "Iteration complexity of randomized block-coordinate descent methods for minimizing a composite function," *Math. Program.*, vol. 144, nos. 1–2, pp. 1–38, 2014.
[161] Y. Rojas and R. Landa, "Towards the use of statistical information and differential evolution for large scale global optimization," in *Proc. Int. Conf. Elect. Eng. Comput. Sci. Autom. Control*, 2011, pp. 1–6.
[162] E. Sayed, D. Essam, and R. Sarker, "Dependency identification technique for large scale optimization problems," in *Proc. IEEE Congr. Evol. Comput.*, 2012, pp. 1–8.
[163] J. D. Schaffer and A. Morishima, "An adaptive crossover distribution mechanism for genetic algorithms," in *Proc. Int. Conf. Genet. Algorithms*, 1987, pp. 36–40.
[164] D. I. Seo and B. R. Moon, "An information-theoretic analysis on the interactions of variables in combinatorial optimization problems," *Evol. Comput.*, vol. 15, no. 2, pp. 169–198, 2007.
[165] F. Seredynski, "Loosely coupled distributed genetic algorithms," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, 1994, pp. 514–523.
[166] T. C. Service and D. R. Tauritz, "A no-free-lunch framework for coevolution," in *Proc. Conf. Genet. Evol. Comput.*, Atlanta, GA, USA, 2008, pp. 371–378.
[167] R. Shang, L. Jiao, Y. Ren, L. Li, and L. P. Wang, "Quantum immune clonal coevolutionary algorithm for dynamic multiobjective optimization," *Soft Comput.*, vol. 18, no. 4, pp. 743–756, 2014.
[168] M. Shi, "Comparison of sorting algorithms for multi-fitness measurement of cooperative coevolution," in *Proc. Conf. Genet. Evol. Comput.*, 2009, pp. 2583–2588.
[169] M. Shi and S. Gao, "Reference sharing: A new collaboration model for cooperative coevolution," *J. Heuristics*, vol. 23, no. 1, pp. 1–30, 2017.
[170] M. Shi and H. Wu, "Pareto cooperative coevolutionary genetic algorithm using reference sharing collaboration," in *Proc. Conf. Genet. Evol. Comput.*, Montreal, QC, Canada, 2009, pp. 867–874.
[171] Y.-J. Shi, H.-F. Teng, and Z.-Q. Li, "Cooperative co-evolutionary differential evolution for function optimization," in *Proc. Int. Conf. Natural Comput.*, Changsha, China, 2005, pp. 1080–1088.
[172] H. K. Singh and T. Ray, "Divide and conquer in coevolution: A difficult balancing act," in *Agent-Based Evolutionary Search*. Heidelberg, Germany: Springer, 2010, pp. 117–138.
[173] J. Smith and T. C. Fogarty, "An adaptive poly-parental recombination strategy," in *Proc. AISB Workshop Evol. Comput.*, 1995, pp. 48–61.
[174] J. Smith and T. C. Fogarty, "Recombination strategy adaptation via evolution of gene linkage," in *Proc. IEEE Int. Conf. Evol. Comput.*, Nagoya, Japan, 1996, pp. 826–831.
[175] R. J. Smith and M. I. Heywood, "Coevolving deep hierarchies of programs to solve complex tasks," in *Proc. Genet. Evol. Comput. Conf.*, Berlin, Germany, 2017, pp. 1009–1016.
[176] M. C. Soares, I. M. Côté, S. Cardoso, and R. Bshary, "The cleaning goby mutualism: A system without punishment, partner switching or tactile stimulation," *J. Zool.*, vol. 276, no. 3, pp. 306–312, 2008.
[177] Y. S. Son and R. Baldick, "Hybrid coevolutionary programming for Nash equilibrium search in games with local optima," *IEEE Trans. Evol. Comput.*, vol. 8, no. 4, pp. 305–315, Aug. 2004.
[178] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evol. Comput.*, vol. 10, no. 2, pp. 99–127, 2002.
[179] C. Stoean, "Various collaborator selection pressures for cooperative coevolution for classification," in *Proc. Int. Conf. Artif. Intell. Digit. Commun.*, 2006, pp. 45–53.
[180] S. Strasser, J. Sheppard, N. Fortier, and R. Goodman, "Factored evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 21, no. 2, pp. 281–293, Apr. 2017.
[181] M. J. Streeter, "Upper bounds on the time and space complexity of optimizing additively separable functions," in *Proc. Genet. Evol. Comput. Conf.*, Seattle, WA, USA, 2003, pp. 186–197.
[182] J. Sun and H. Dong, "Cooperative co-evolution with correlation identification grouping for large scale function optimization," in *Proc. Int. Conf. Inf. Sci. Technol.*, Yangzhou, China, 2013, pp. 889–893.
[183] L. Sun, S. Yoshida, X. Cheng, and Y. Liang, "A cooperative particle swarm optimizer with statistical variable interdependence learning," *Inf. Sci.*, vol. 186, no. 1, pp. 20–39, 2012.
[184] Y. Sun, M. Kirley, and S. K. Halgamuge, "Extended differential grouping for large scale global optimization with direct and indirect variable interactions," in *Proc. Conf. Genet. Evol. Comput.*, Madrid, Spain, 2015, pp. 313–320.
[185] Y. Sun, M. Kirley, and S. K. Halgamuge, "Quantifying variable interactions in continuous optimization problems," *IEEE Trans. Evol. Comput.*, vol. 21, no. 2, pp. 249–264, Apr. 2017.
[186] C. H. Tan, C. K. Goh, K. C. Tan, and A. Tay, "A cooperative coevolutionary algorithm for multiobjective particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, Singapore, 2007, pp. 3180–3186.
[187] K. C. Tan, T. H. Lee, Y. J. Yang, and D. S. Liu, "A cooperative coevolutionary algorithm for multiobjective optimization," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, 2004, pp. 1926–1931.
[188] K. C. Tan, Y. J. Yang, and C. K. Goh, "A distributed cooperative coevolutionary algorithm for multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 10, no. 5, pp. 527–549, Oct. 2006.
[189] T. G. Tan, J. Teo, and H. K. Lau, "Performance scalability of a cooperative coevolution multiobjective evolutionary algorithm," in *Proc. Int. Conf. Comput. Intell. Security*, Harbin, China, 2007, pp. 119–123.
[190] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC'2010 special session and competition on large-scale global optimization," Nat. Inspired Comput. Appl. Lab., Univ. Sci. Technol. China, Hefei, China, Rep., 2009.
[191] K. Tang *et al.*, "Benchmark function for the CEC'2008 special session and competition on large scale global optimization," Nat. Inspired Comput. Appl. Lab., Univ. Sci. Technol. China, Hefei, China, Rep., 2007.
[192] H.-F. Teng, Y. Chen, W. Zeng, Y.-J. Shi, and Q.-H. Hu, "A dual-system variable-grain cooperative coevolutionary algorithm: Satellite-module layout design," *IEEE Trans. Evol. Comput.*, vol. 14, no. 3, pp. 438–455, Jun. 2010.
[193] M. Tezuka, M. Munetomo, and K. Akama, "Linkage identification by nonlinearity check for real-coded genetic algorithms," in *Proc. Conf. Genet. Evol. Comput.*, 2004, pp. 222–233.

[194] C.-K. Ting, W.-M. Zeng, and T.-C. Lin, "Linkage discovery through data mining [research frontier]," *IEEE Comput. Intell. Mag.*, vol. 5, no. 1, pp. 10–13, Feb. 2010.

[195] C. R. Tosh, A. L. Jackson, and G. D. Ruxton, "Individuals from different-looking animal species May group together to confuse shared predators: Simulations with artificial neural networks," *Proc. Roy. Soc. London B Biol. Sci.*, vol. 274, no. 1611, pp. 827–832, 2007.

[196] G. Trunfio, "Enhancing the firefly algorithm through a cooperative coevolutionary approach: An empirical study on benchmark optimisation problems," *Int. J. Bio-Inspired Comput.*, vol. 6, no. 2, pp. 108–125, 2014.

[197] G. A. Trunfio, "Adaptation in cooperative coevolutionary optimization," in *Adaptation and Hybridization in Computational Intelligence*. Cham, Switzerland: Springer, 2015, pp. 91–109.

[198] G. A. Trunfio, "A cooperative coevolutionary differential evolution algorithm with adaptive subcomponents," *Procedia Comput. Sci.*, vol. 51, no. 1, pp. 834–844, 2015.

[199] G. A. Trunfio, P. Topa, and J. Was, "A new algorithm for adapting the configuration of subcomponents in large-scale optimization with cooperative coevolution," *Inf. Sci.*, vol. 372, pp. 773–795, Dec. 2016.

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

[201] L.-Y. Tseng and C. Chen, "Multiple trajectory search for unconstrained/constrained multi-objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, Trondheim, Norway, 2009, pp. 1951–1958.

[202] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *J. Optim. Theory Appl.*, vol. 109, no. 3, pp. 475–494, 2001.

[203] M. Tsuji and M. Munetomo, "Linkage analysis in genetic algorithms," in *Computational Intelligence Paradigms*. Heidelberg, Germany: Springer, 2008, pp. 251–279.

[204] M. Tsuji, M. Munetomo, and K. Akama, "Modeling dependencies of loci with string classification according to fitness differences," in *Proc. Conf. Genet. Evol. Comput.*, 2004, pp. 246–257.

[205] S. Tsutsui and D. E. Goldberg, "Simplex crossover and linkage identification: Single-stage evolution vs. multi-stage evolution," in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, USA, 2002, pp. 974–979.

[206] U. Segal, "A sufficient condition for additively separable functions," *J. Math. Econ.*, vol. 23, no. 3, pp. 295–303, 1994.

[207] S. I. Valdez, A. Hernández, and S. Botello, "A Boltzmann based estimation of distribution algorithm," *Inf. Sci.*, vol. 236, no. 1, pp. 126–137, Jul. 2013.

[208] S. M. Vieira, J. M. C. Sousa, and T. A. Runkler, "Two cooperative ant colonies for feature selection using fuzzy models," *Expert Syst. Appl.*, vol. 37, no. 4, pp. 2714–2723, 2010.

[209] S. Wan, "Differential evolution with clustering cooperative coevolution for high-dimensional problems," in *Proc. Int. Conf. Inf. Sci. Cloud Comput. Companion*, 2013, pp. 782–786.

[210] C. Wang and J. Gao, "High-dimensional waveform inversion with cooperative coevolutionary differential evolution algorithm," *IEEE Geosci. Remote Sens. Lett.*, vol. 9, no. 2, pp. 297–301, Mar. 2012.

[211] L. Wang and L.-P. Li, "A coevolutionary differential evolution with harmony search for reliability–redundancy optimization," *Expert Syst. Appl.*, vol. 39, no. 5, pp. 5271–5278, 2012.

[212] S. Wang *et al.*, "A cooperative coevolution framework for parallel learning to rank," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 12, pp. 3152–3165, Dec. 2015.

[213] Y. Wang, B. Li, and X. X. Lai, "Variance priority based cooperative co-evolution differential evolution for large scale global optimization," in *Proc. IEEE Congr. Evol. Comput.*, Trondheim, Norway, 2009, pp. 1232–1239.

[214] R. A. Watson and J. B. Pollack, "Coevolutionary dynamics in a minimal substrate," in *Proc. Conf. Genet. Evol. Comput.*, San Francisco, CA, USA, 2001, pp. 702–709.

[215] F. Wei, Y. Wang, and T. Zong, "A novel cooperative coevolution for large scale global optimization," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, San Diego, CA, USA, 2014, pp. 738–741.

[216] F. Wei, Y. Wang, and T. Zong, "Variable grouping based differential evolution using an auxiliary function for large scale global optimization," in *Proc. IEEE Congr. Evol. Comput.*, Beijing, China, 2014, pp. 1293–1298.

[217] K. Weicker and N. Weicker, "On the improvement of coevolutionary optimizers by learning variable interdependencies," in *Proc. IEEE Congr. Evol. Comput.*, Washington, DC, USA, 1999, pp. 1627–1632.

[218] T. Weise, R. Chiong, and K. Tang, "Evolutionary optimization: Pitfalls and booby traps," *J. Comput. Sci. Technol.*, vol. 27, no. 5, pp. 907–936, 2012.

[219] B. A. Whitehead and T. D. Choate, "Cooperative-competitive genetic evolution of radial basis function centers and widths for time series prediction," *IEEE Trans. Neural Netw.*, vol. 7, no. 4, pp. 869–880, Jul. 1996.

[220] R. P. Wiegand, "Applying diffusion to a cooperative coevolutionary model," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, Amsterdam, The Netherlands, 1998, pp. 560–569.

[221] R. P. Wiegand, "An analysis of cooperative coevolutionary algorithms," Ph.D. dissertation, Dept. Comput. Sci., George Mason Univ., Fairfax, VA, USA, 2004.

[222] R. P. Wiegand, W. C. Liles, and K. A. De Jong, "Analyzing cooperative coevolution with evolutionary game theory," in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, USA, 2002, pp. 1600–1605.

[223] R. P. Wiegand, W. C. Liles, and K. A. D. Jong, "An empirical analysis of collaboration methods in cooperative coevolutionary algorithms," in *Proc. Conf. Genet. Evol. Comput.*, San Francisco, CA, USA, 2001, pp. 1235–1242.

[224] R. P. Wiegand and M. A. Potter, "Robustness in cooperative coevolution," in *Proc. Genet. Evol. Comput. Conf.*, Seattle, WA, USA, 2006, pp. 369–376.

[225] R. P. Wiegand and J. Sartna, "Spatial embedding and loss of gradient in cooperative coevolutionary algorithms," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, Birmingham, U.K., 2004, pp. 912–921.

[226] A. Wu, J. Zhang, and H. Chung, "Decoupled optimal design for power electronic circuits with adaptive migration in coevolutionary environment," *Appl. Soft Comput.*, vol. 11, no. 1, pp. 23–31, 2011.

[227] B. Xu, Y. Zhang, D. Gong, Y. Guo, and R. Miao, "Environment sensitivity-based cooperative co-evolutionary algorithms for dynamic multi-objective optimization," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, to be published, doi: 10.1109/TCBB.2017.2652453.

[228] M. Yang *et al.*, "Efficient resource allocation in cooperative coevolution for large-scale global optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 4, pp. 493–505, Aug. 2017.

[229] Z. Yang, K. Tang, and X. Yao, "Differential evolution for high-dimensional function optimization," in *Proc. IEEE Congr. Evol. Comput.*, Singapore, 2007, pp. 3523–3530.

[230] Z. Yang, K. Tang, and X. Yao, "Large scale evolutionary optimization using cooperative coevolution," *Inf. Sci.*, vol. 178, no. 15, pp. 2986–2999, 2008.

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

[232] T.-L. Yu, D. E. Goldberg, K. Sastry, C. F. Lima, and M. Pelikan, "Dependency structure matrix, genetic algorithms, and effective recombination," *Evol. Comput.*, vol. 17, no. 4, pp. 595–626, 2009.

[233] A. Zaritsky and M. Sipper, "The preservation of favored building blocks in the struggle for fitness: The puzzle algorithm," *IEEE Trans. Evol. Comput.*, vol. 8, no. 5, pp. 443–455, Oct. 2004.

[234] H. Zhang, Y. Zhu, and X. Yan, "Multi-hive artificial bee colony algorithm for constrained multi-objective optimization," in *Proc. IEEE Congr. Evol. Comput.*, Brisbane, QLD, Australia, 2012, pp. 1–8.

[235] Q. Zhang, "On stability of fixed points of limit models of univariate marginal distribution algorithm and factorized distribution algorithm," *IEEE Trans. Evol. Comput.*, vol. 8, no. 1, pp. 80–93, Feb. 2004.

[236] Q. Zhang, "On the convergence of a factorized distribution algorithm with truncation selection," *Complexity*, vol. 9, no. 4, pp. 17–23, 2004.

[237] Q. Zhang and H. Mühlenbein, "On the convergence of a class of estimation of distribution algorithms," *IEEE Trans. Evol. Comput.*, vol. 8, no. 2, pp. 127–136, Apr. 2004.

[238] Y. Zhang, X.-B. Wu, Z.-Y. Xing, and W.-L. Hu, "On generating interpretable and precise fuzzy systems based on Pareto multi-objective cooperative co-evolutionary algorithm," *Appl. Soft Comput.*, vol. 11, no. 1, pp. 1284–1294, 2011.

[239] W. J. Zhao, S. Alam, and H. A. Abbass, "MOCCA-II: A multi-objective co-operative co-evolutionary algorithm," *Appl. Soft Comput.*, vol. 23, pp. 407–416, Oct. 2014.

[240] X. Zheng and H. Liu, "A scalable coevolutionary multi-objective particle swarm optimizer," *Int. J. Comput. Intell. Syst.*, vol. 3, no. 5, pp. 590–600, 2010.

[241] Y.-J. Zheng and S.-Y. Chen, "Cooperative particle swarm optimization for multiobjective transportation planning," *Appl. Intell.*, vol. 39, no. 1, pp. 202–216, 2013.
[242] Y. Zhong, A. Ma, Y. S. Ong, Z. Zhu, and L. Zhang, "Computational intelligence in optical remote sensing image processing," *Appl. Soft Comput.*, vol. 64, pp. 75–93, Mar. 2018.

Xiaoliang Ma received the B.S. degree in computing computer science and technology from Zhejiang Normal University, Jinhua, China, in 2006 and the Ph.D. degree from the School of Computing, Xidian University, Xi'an, China, in 2014.

He is currently an Assistant Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His current research interests include evolutionary computation, multiobjective optimization, cooperative coevolution, complex network, and bioinformatics.

Xiaodong Li (M'03–SM'07) received the B.Sc. degree from Xidian University, Xi'an, China, and the Ph.D. degree in information science from the University of Otago, Dunedin, New Zealand.

He is a Professor with the School of Science (Computer Science and Software Engineering), RMIT University, Melbourne, VIC, Australia. His current research interests include machine learning, evolutionary computation, neural networks, data analytics, multiobjective optimization, multimodal optimization, and swarm intelligence.

Dr. Li was a recipient of the 2013 ACM SIGEVO Impact Award and the 2017 IEEE CIS IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award. He serves as an Associate Editor for the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, *Swarm Intelligence* (Springer), and the *International Journal of Swarm Intelligence Research*.

Qingfu Zhang (M'01–SM'06–F'17) received the B.Sc. degree in mathematics from Shanxi University, Taiyuan, China, in 1984 and the M.Sc. degree in applied mathematics and the Ph.D. degree in information engineering from Xidian University, Xi'an, China, in 1991 and 1994, respectively.

He is a Professor with the Department of Computer Science, City University of Hong Kong, Hong Kong, and a Changjiang Visiting Chair Professor with Xidian University. His current research interests include evolutionary computation, optimization, neural networks, and data analysis and their applications.

Dr. Zhang was a recipient of the 2010 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award and the Highly Cited Researcher Award in Computer Science by Web of Science. He is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and the IEEE TRANSACTIONS ON CYBERNETICS.

Ke Tang (M'07–SM'13) received the B.Eng. degree from the Huazhong University of Science and Technology, Wuhan, China, in 2002 and the Ph.D. degree from Nanyang Technological University, Singapore, in 2007.

He is currently a Professor with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. His current research interests include evolutionary computation and machine learning and their applications.

Dr. Tang was a recipient of the Royal Society Newton Advanced Fellowship in 2015 and the 2018 IEEE Computational Intelligence Society Outstanding Early Career Award. He is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and served as a member of Editorial Boards for few other journals.

Zhengping Liang received the B.S. degree in computer science and technology from Hunan Normal University, Changsha, China, in 2001 and the Ph.D. degree in computer science and technology from Wuhan University, Wuhan, China, in 2006.

He is currently an Associate Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His current research interests include computational intelligence and multiobjective optimization.

Weixin Xie received the degree from Xidian University, Xi'an, China.

In 1965, he joined Xidian University as a Faculty Member. In 1989, he was invited to the University of Pennsylvania, Philadelphia, PA, USA, as a Visiting Professor. He is currently with the School of Information Engineering, Shenzhen University, Shenzhen, China. His current research interests include intelligent information processing, image processing, and pattern recognition.

Zexuan Zhu (M'12) received the B.S. degree in computer science and technology from Fudan University, Shanghai, China, in 2003 and the Ph.D. degree in computer engineering from Nanyang Technological University, Singapore, in 2008.

He is currently a Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His current research interests include computational intelligence, machine learning, and bioinformatics.

Dr. Zhu is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE. He is also the Chair of the IEEE CIS Emergent Technologies Task Force on Memetic Computing.