# A review of population-based metaheuristics for large-scale black-box global optimization: Part B

Mohammad Nabi Omidvar, *Senior Member, IEEE,* Xiaodong Li, *Fellow, IEEE,* and Xin Yao, *Fellow, IEEE*

*Abstract*—This paper is the second part of a two-part survey series on large-scale global optimization. The first part covered two major algorithmic approaches to large-scale optimization, namely decomposition methods and hybridization methods such as memetic algorithms and local search. In this part we focus on sampling and variation operators, approximation and surrogate modeling, initialization methods, and parallelization. We also cover a range of problem areas in relation to large-scale global optimization, such as multi-objective optimization, constraint handling, overlapping components, the component imbalance issue, and benchmarks, and applications. The paper also includes a discussion on pitfalls and challenges of current research and identifies several potential areas of future research.

*Index Terms*—large-scale global optimization, black-box optimization, metaheuristics, evolutionary optimization

## I. INTRODUCTION

The first part of this two-part survey series covered decomposition methods and hybrid methods as two most widely investigated approaches in the literature. Figure 1 depicts a high-level structure of the main topics covered across both parts. In this part, we review more approaches to large-scale global optimization and also address several problem areas including multi-objective optimization and constraint-handling. Section II covers sampling mechanism and variation operators of two well-known algorithms, particle swarm algorithm [1] and differential evolution [2], and how they are modified to solve large-scale problems. Section III covers the algorithms which rely on some form of approximation to cope with the challenges of high-dimensionality. Section IV covers population initialization methods and their significance in large-scale global optimization. Section V addresses the role of parallel algorithms to address the issue of scalability.

In addition to the algorithmic approaches to large-scale optimization, the paper also addresses a range of problem areas

Mohammad Nabi Omidvar is with the School of Computing, University of Leeds, and Leeds University Business School, Leeds LS2 9JT, UK (email: m.n.omidvar@leeds.ac.uk). He is also the current chair of the IEEE Taskforce on Large-Scale Global Optimization.

Xiaodong Li is with the School of Computing Technologies, RMIT University, Melbourne, Australia (email: xiaodong.li@rmit.edu.au).

Xin Yao is with the Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China. Xin Yao is also with the School of Computer Science, University of Birmingham, Birmingham B15 2TT, UK (email: x.yao@cs.bham.ac.uk; xiny@sustech.edu.cn).
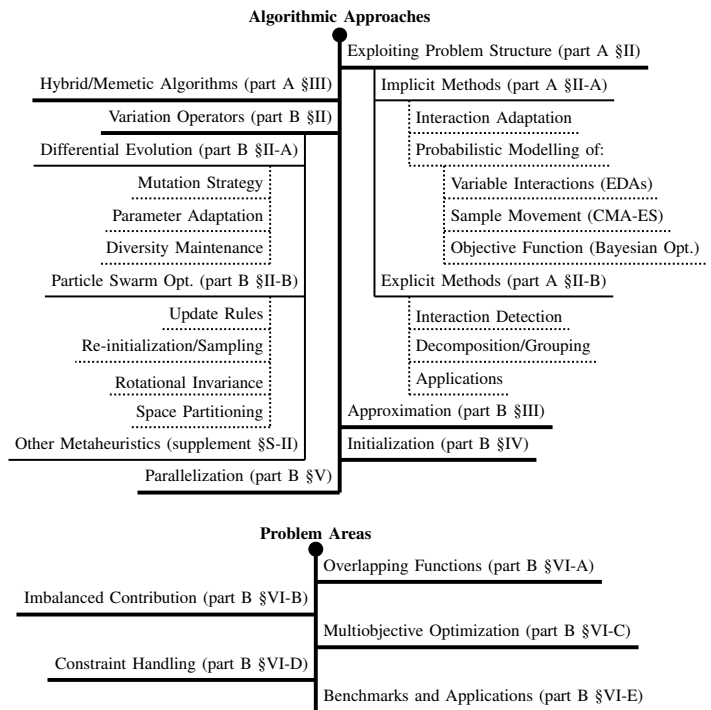
Fig. 1: Outline of the topics covered in the two parts of this survey series on large-scale global optimization.

pertaining to large-scale global optimization. These include: 1) scalability of multi-objective optimization algorithms with respect to their decision space; 2) challenges of constraint-handling in the context of large-scale optimization; 3) challenges in dealing with problems with overlapping components and the issue of exploitable structure; 4) resource allocation and the problem of imbalanced contribution; 5) benchmarking and application areas.

The paper also features a section on pitfalls and challenges of the field and potential areas of future research.

## II. SAMPLING AND VARIATION OPERATORS

In part A of this survey, we have seen that many optimizers such as estimation of distribution algorithms (EDA), differential evolution (DE), and particle swarm optimization (PSO), can be used as component optimizers in decomposition-based frameworks (part A §II-B), and as explorative agents in memetic algorithms (part A §III). In this section, we focus on algorithm-specific aspects such as parameter adaptation, modification of variation operators or design of new ones, diversity maintenance mechanisms, etc. In what follows we
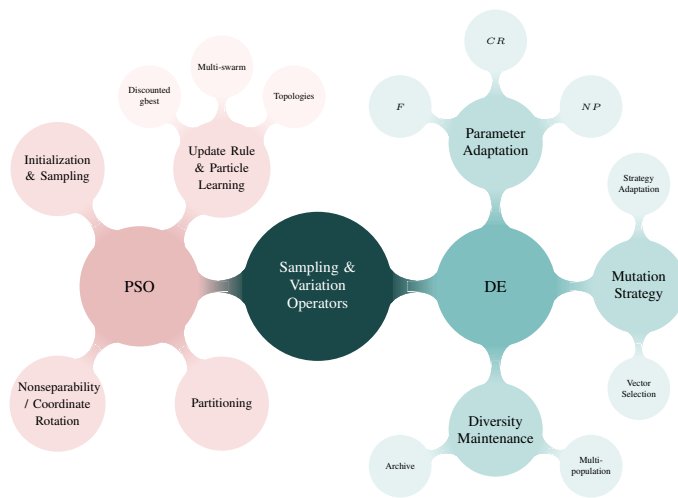
Fig. 2: DE and PSO are two widely used metaheuristic algorithms used in large-scale global optimization. This figure shows major aspects of these algorithms studied for large-scale optimization problems.

cover DE and PSO in more detail and other metaheuristics are covered in Section S-II of the supplementary document. EDAs were covered in Part A §II-A due to their focus on modeling variable interactions. Figure 2 shows major aspects of PSO and DE, which have been studied under high-dimensional settings.

### A. Differential Evolution

Due to its versatility, ease of implementation and simplicity, differential evolution (DE) [2] has become a widely used optimization algorithm for global optimization [3]. Consequently, many variants of DE have been developed for large-scale global optimization [4] from which the most popular ones are briefly reviewed in this section. Most the DE variants proposed for large-scale optimization are centered around maintaining population diversity, which is done by various means such as parameter adaptation, modification of DE mutation strategy, and diversity maintenance mechanisms.

*1) Mutation Strategy:* Mutation strategy is DE's central variation operator and has been subject to extensive investigation in the literature [3]. Several attempts have been made to improve DE for large-scale optimization by adapting or hybridizing several mutation strategies or by proposing new ones [5].

*a) Adaptation of Mutation Strategy:* Different mutation strategies exhibit various degrees of explorative/exploitative power each being suitable for certain problem types [5]. In the context of LSGO, several attempts have been made to use several mutation strategies to improve the convergence properties of DE in high-dimensional spaces. These methods are either based on adaptively applying a set of strategies to a single population or using a multi-population approach where each is evolved using its own mutation strategy. Ali et al. [6] proposed a multi-population DE where each subpopulation has its own mutation strategy. Banitalebi et al. [7] proposed a binary DE which adaptively selects the mutation strategy for generating trial vectors and also adapts the scaling factor

and crossover rate using a chaotic process (also see the section on parameter adaptation later in this section). Kushida et al. [8] proposed a rank-based mechanism for selecting the mutation strategies. Wang et al. [9] proposed to adaptively switch between DE/rand/1 and DE/current_to_best/1 mutation strategies. There are also approaches that switch between DE/rand/1/bin and a newly proposed strategy based on a uniform distribution [10, 11].

*b) Vector Selection:* Canonical DE uses random individuals in the mutation process to generate a scaled difference vector to be applied to a base vector and generate a new solution. The choice of the vectors participating in the mutation procedure plays a crucial role in DE's convergence behavior [5]. Ge et al. [12] analyzed different DE strategies and observed that those using the best individual are exploitative while those using random individuals are more explorative. They argue that instead of randomly selecting the participating vectors, it is better to systematically choose a vector close to the best solution to favor exploitation and far from the mutant to favor exploration. Inspired by PSO personal and global best particles, Wang et al. [13] proposed to generate trial vectors by including the global best and personal best individuals in the mutation strategy. The authors claim that this process is akin to neighborhood search and improves convergence. García-Martínez et al. [14] associate four basic roles – placing, leading, correcting and receiving – to each vector (solution), and the vector selection for mutation is performed based on these four basic roles. The vector selection strategy proposed by Ali et al. [6] is a function of the rank of a solution in the population, favoring higher quality solutions to participate in the mutation. Zhang and Sanderson [15] proposed a generalization of the classic DE/current-to-best mutation operator, DE/current-to-$p$best, which uses the top $p\%$ best solutions to balance the greediness level of the algorithm and to maintain better diversity in the population. Some other studies also proposed several mutation strategies in which the solution quality is taken into account in the vector selection process [10, 11]. Yang et al. [16] proposed to use multiple such difference vectors which are scaled differently to generate trial vectors.

The choice of the base vector to which the mutation is applied is also important in DE's convergence behavior. Ali et al. [6] proposed a new mutation strategy by selecting the base individual to be a convex combination of randomly chosen individuals from the population. Wang et al. [17] proposed an enhanced opposition-based differential evolution in which the candidate solutions are translated into a so-called *opposite space* using the definition of opposite numbers [18]. Wang et al. [17] argue that by evaluation of the candidate solutions and their translated counterparts in the opposite space, the probability of finding better solutions increases. This hypothesis is backed up by a set of empirical results on a set of 19 high-dimensional benchmark functions [19]. Hiba et al. [20] proposed a center-based mutation strategy which uses the center of three randomly chosen solutions as the base vector.

*2) Parameter Adaptation:* Population size ($NP$), crossover rate ($CR$), and the scaling factor ($F$) are DE's major parameters affecting its convergence properties on different problem

types. To eliminate the need for practitioners to set these hard-to-tweak parameters, several attempts have been made to adaptively set these parameters in the course of optimization [3]. In this section, we review some of these adaptation methods pertaining to large-scale global optimization.

Scaling factor and crossover rate are the two most studied parameters. Most of these attempts use some form of probability distribution from which the parameters are sampled. Brest et al. [21] dynamically change $F$ and $CR$ using a uniform distribution. Wang et al. [22] used a similar adaptation mechanism except that they restricted the range of $CR$ values. Brest et al. [23] introduced a sign changing mechanism to $F$ in addition to sampling of $CR$ and $F$ values from a uniform distribution. To improve the current best, it uses smaller F values in the second half of the optimization process. Weber et al. [24] use a multi-population approach each having its own scaling factor which are regenerated in a probabilistic way. Zamuda et al. [25] proposed to adapt $CR$ and $F$ using a log-normal distribution. Improving upon self-adaptive DE (SaDE) [26], Yang et al. [27] use a Gaussian distribution to generate $F$ and $CR$ for each individual and update the mean of the Gaussian based on the parameter values succeeding in generating surviving offsprings. Zhang and Sanderson [15] proposed JADE which randomly generates $F$ and $CR$ at every generation using Cauchy and Gaussian distributions respectively whose parameters are adapted in the course of optimization. Yang et al. [28] attempted to generalize the attempts by its predecessors such as JADE, SaDE, and SaNSDE into a unified mechanism.

There are also alternative approaches which are not based on sampling from probability distributions. For example, some studies propose to change the crossover rate and the scaling factor using a chaotic process [7, 9]. Takahama and Sakai [29] proposed a DE variant in which the scaling factor is adapted according to modality feature of the search space. Kushida et al. [8] improves upon the works of Takahama and Sakai [29] by adding a rank-based mechanism for setting the scaling factor and crossover rate as well as the mutation strategies.

The attempts for adaptation of population size in large-scale optimization are ad hoc and limited. Brest et al. [23] proposed to gradually reduce the population size in the course of optimization. Wang et al. [30] adaptively changes the population size by adding or removing solutions based on their performance. Tanabe and Fukunaga [31] linearly decrease the population size.

*3) Diversity Maintenance:* Loss of population diversity is central to DE's deficiency in high dimensional spaces. This is typically avoided in lower dimensions by means of increasing the population size [32]. However, in high dimensional spaces, large population size hinders convergence [33]. Adaptation of the mutation scaling factor, hybridizing an array of mutation strategies, or designing new ones are all attempts to improve the population diversity, which were addressed in the previous sections. Other approaches to diversification are *multi-population* approaches, either in the form of several islands searching the original search space or by means of partitioning and coevolution, or maintaining an *archive* of solutions (Fig. 2).

Weber et al. [24] proposed a multi-population strategy simultaneously searches different parts of the search space. This algorithm randomly rearranges the individuals across the subpopulations with the aim of maintaining diversity among the solutions. Ge et al. [34] also proposed a multi-population DE, which maintains diversity through migration of similar or diverse individuals. This mechanism controls the balance between exploration and exploitation. Ge et al. [35] use a multi-population approach with automatic merge and split operations to improve population diversity. Ali et al. [6] used a multi-population DE with each population having its own mutation strategy to maintain population diversity. Information exchange between populations helps with balancing exploration and exploitation. Parsopoulos [36] uses cooperative coevolution to partition the search space into smaller regions and optimizes them with a micro DE. Micro DE is prone to losing diversity and getting trapped in local optima; however, CC helps DE to focus the search with its micro population on smaller regions. Ge et al. [12] also use CC with cross-cluster mutation to promote exploration.

Maintaining an archive of solutions in the course of optimization is another means of maintaining diversity. Takahama and Sakai [29] proposed a DE variant with an archive of old solutions to help with diversification in the mating process, especially when the population size is small. Yang et al. [16] also proposed a DE variant which keeps an archive of failed trial vectors with the hope of preserving good genetic material. Zhang and Sanderson [15] proposed JADE which maintains an external archive of inferior solutions to estimate the possible improvement directions. The external archive of JADE proved to be beneficial, especially on relatively high dimensional problems with up to 100 dimensions.

### B. Particle Swarm Optimization

Particle swarm optimization (PSO) [1, 37] is known to be susceptible to premature convergence, which is magnified on high-dimensional problems [38, 39]. Most approaches to handle large-scale optimization problems are centered around increasing diversity to improve exploration. In some cases however, extreme exploration and exploitation can co-exist [40]. The common remedies to PSO's premature convergence in large-scale global optimization literature are population re-initialization, complementary sampling mechanisms, population size adaptation, space partitioning (by means of cooperative coevolution or otherwise), improving PSO's update rule and particle learning mechanisms, and mechanism to deal with variable interaction and nonseparable problems.

*a) PSO update rule and particle learning:* Excessive reliance on the global best particle can result in premature convergence. Many attempts to avoid premature convergence revolve around reducing the influence of global best. Cheng and Jin [41] proposed a PSO variant, named CSO, which does not use personal or global best solutions to update the position of the particles. Instead random pairs are chosen to compete and the winner returns directly to the swarm while the loser is updated by learning from the winner. CSO maintains a better diversity than PSO and is more explorative,

making it better suited for large-scale global optimization. Tian et al. [42] proposed a variant of CSO based on a two-stage update rule for position of particles and applied it to solving multiobjective problems. Naderi et al. [43] proposed a fuzzy adaptive system to adjust the inertia weight. This eliminates the use of global best in the velocity update rule to avoid premature convergence. Tang et al. [44] used a new update rule to exploit four best positions via Gaussian sampling to reduce the influence of global best and promote exploration. Pluhacek et al. [45] changes the velocity update rule such that with some probability the velocity is either zero, or is updated by taking either a random particle, personal best, or the global best into account.

Controlling information exchange among particles by means of population topologies or multi-population structures are other ways of enhancing the swarm diversity [46]. Fan et al. [47] proposed a PSO variant which builds a dynamic neighborhood topology for PSO by performing clustering on the population. The neighbors of the particles are chosen from the same cluster. It also chooses a distant neighbor for particles through random selection. Zhang et al. [48] improves CSO by applying Cauchy and Gaussian updates on the winner particles and uses a ring topology to enhance the swarm diversity. Distributed multi-population schemes [49–51] also promote controlled information exchange among particles which can improve population diversity.

In addition to the above, several other modifications to PSO's update rule have been suggested in the context of large-scale global optimization. Arasomwan and Adewumi [52] found that inertia weight, acceleration coefficients, and random factors were not of significance in velocity update for obtaining global solutions. They proposed to adaptively update particles' velocity based on Euclidean distance between particles and the global best. It also introduces chaotic behavior into the particle position update rule. The notion of social learning has been proposed to reduce the adverse effect of isolated asocial learning [53, 54]. Yang et al. [55] proposed to group particles into several levels based on their fitness. Two predominant particles from two different higher levels are chosen to guide the learning of particles. This has shown to improve diversity. Convergence speed controller was proposed as an independent operator to respond to premature or slow convergence [56]. Cheng et al. [57] proposed a mutation operator based on the Alpha-stable distribution to enhance the swarm diversity and avoid premature convergence. Li et al. [58] changed the particles' velocity update rule to decouple exploration and exploitation. Xue et al. [59] used multiple velocity and position update rules which are chosen probabilistically whose parameters are adapted according to the effectiveness of each strategy.

*b) Re-initialization, sampling, and population size control:* Hsieh et al. [60] proposed a PSO variant with dynamic swarm size which increases or decreases the swarm size based on the status of particles. In general, if the global best of the swarm is not updated for several consecutive iterations, new particles are generated by applying a crossover-like operator on the best solutions that were obtained in the past. Conversely, if the information content of the swarm is rich enough to allow frequent and robust updating of the global best, some of the poor quality solutions might be removed from the swarm. There are some other mechanisms in place to avoid the growth of the swarm size beyond bounds. de Oca et al. [61] proposed an improved version of incremental particle swarm-guided local search [62] which incrementally increases the population size to solve large-scale continuous optimization problems.

Garcí-Nieto and Alba [63] proposed restart particle swarm optimization with velocity modulation. Velocity modulation is the process by which the particles are guided within a region of interest. Additionally, a restart mechanism is devised to avoid premature convergence of the algorithm. Cheng et al. [64] proposed partial re-initialization to improve exploration. They partition the search space and count the number of particles in each partition and abandon the low activity areas. It also subdivides and reinitializes particles in higher activity regions. This mechanism has shown to improve exploitation. Zhou et al. [65] introduced opposition-based sampling into CSO.

*c) Nonseparability and coordinate rotation:* The update equations of PSO are dimension-wise which makes it suitable for separable functions. Hendtlass [66] use dynamic momentum values to enable PSO to better handle nonseparability, making it suitable for functions with interacting dimensions. Korenaga et al. [67] introduce coordinate rotation into the velocity update rule which makes it possible to consider the information of other coordinates when calculating the velocity of a component. This can improve population diversity and has shown to be beneficial for large-scale optimization. Chu et al. [68] uses PCA to parts of the space not spanned by the current population resulting in improved exploration. In a similar way, Chu et al. [69] also uses PCA to find lost dimensions and promote search in the less explored areas due to lost dimensions.

*d) Space partitioning:* Zhang et al. [70] partition the space by grouping the dimensions into segments. Then some newly designed operators are assigned to each segment to update those segments. These operators are designed to improve population diversity and avoid premature convergence. Zhao et al. [71] proposed to form subswarms which work independently during the search. Then the subswarms are randomly changed to enlarge their neighborhood and promote information exchange and population diversity. This improves exploration but may deteriorate exploitation which is compensated for by means of local search. Cheng et al. [64] proposed space partitioning with the aim of identifying and abandoning low activity areas and focus the search effort in high activity areas.

Balancing exploration and exploitation and maintaining population diversity are generally central to the design of effective optimizers for large-scale optimization. In this section, we have seen that this plays a crucial role in both DE and PSO. Novel means of using population topologies [46], sampling methods, parameter adaptation, and space partitioning are needed to further improve these algorithms for large-scale optimization.
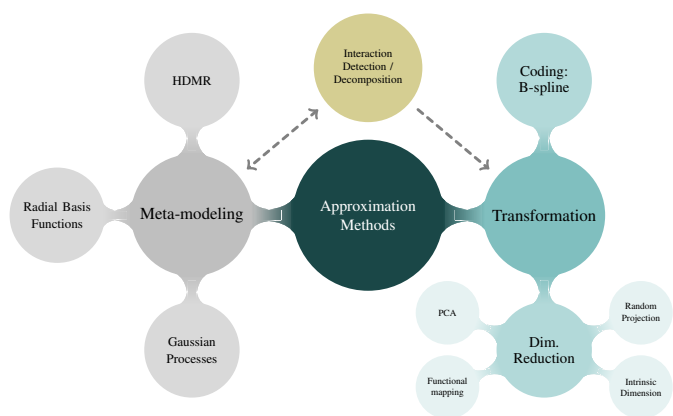
Fig. 3: Problem approximation/simplification methods used for large-scale global optimization.

## III. APPROXIMATION AND SURROGATE MODELING

Solving an approximation of a problem can potentially be more viable than obtaining a solution for its original high-fidelity model. In other words, the aim of approximation is to *simplify*. In optimization, this simplification is either achieved by means of applying some *transformation* to the objective function (often to reduce the dimensions), or by building a model of the objective function or its constraints [72], i.e., a *meta-model* to act as a surrogate to the original complex problem (Figure 3). Meta-models or surrogates are used to reduce the computational overhead of optimizing expensive objective functions. In recent years, one approach to large-scale global optimization is to treat it as an expensive optimization problem and use meta-models to solve it [73].

Meta-models are built and refined based on sampling of the objective function. In high-dimensional spaces, the accuracy of the model drops significantly due to the limited sample size upon which the model is built. To alleviate this problem, several studies use some form of problem decomposition to break the problem into a set of lower dimensional subproblems each of which is approximated using a meta-modeling technique such as radial basis functions or the Gaussian processes. Due to problem decomposition, it is clear that problem structure and variable interaction plays an important role in building an ensemble of surrogates. In some cases the meta-modeling itself is used to identify separable and nonseparable components of a problem [74]. For example, Li et al. [75] used cut-HDMR to detect the components [75]. Then a multi-surrogate strategy is used to model the nonseparable components. In other cases, variable interaction analysis algorithms such as differential grouping [76] is used to decompose the problem and the subsequent subproblems are then approximated using meta-modeling techniques [77]. Werth et al. [78] also proposed a sliding window approach over the decision vectors to reduce the dimensionality of the problem. They use LINC-R to discover variable interaction structure of a subset of the decision variables falling within the sliding window. Then a surrogate-assisted algorithm is used to optimize over those variables.

Meta-modelling and problem decomposition are mutually benefiting approaches to solve large-scale optimization problems. Problem decomposition makes it possible to build more accurate meta-models given the limited samples, while meta-models can help with the issue of evaluating partial solutions in a divide-and-conquer paradigm. In cooperative coevolution and other divide-and-conquer paradigms, partial solutions need to be evaluated in the context of other partial solutions to form a complete solution. The issue of estimating the fitness of partial solutions was studied by Wang and Gao [79] using fixed auxiliary functions with no dynamic meta-modeling mechanism. Several studies use meta-modeling as a means of reducing the overhead of matching partial solutions and their re-evaluation for cooperative coevolution [77, 80, 81] and other divide-and-conquer paradigms, such as random projections [73].

In addition to modeling of subproblems, meta-models have also been used to balance the global search (exploration) and local search (exploitation) efforts. Meta-models have been used in competitive swarm optimizer [41] to approximate the fitness of neighboring particles of a particle with a known fitness [82]. Sun et al. [83] proposed to balance the exploration and exploitation efforts by means of building local and global surrogates. For the exploration part, they use social learning PSO [53] which has good global optimization properties in conjunction with radial basis functions capable of capturing the global profile of the objective function. For exploitation, they use a variant of the fitness approximation method proposed by Sun et al. [82] in conjunction with PSO for local search.

Beside building an explicit model of the objective function, as is the case with meta-modeling, approximation can be built into problem representation [84] or be achieved by means of dimensionality reduction through transformation [85–91] or finding intrinsic dimensions of a problem [92]. Wang et al. [93] combined the benefit of meta-models and dimensionality reduction by using auto-encoders to find lower dimensional features of graph embedding problems and use them to construct a surrogate model to approximate the robustness value of large-scale graph networks. Principal component analysis has also been used to identify a lower-dimensional representation of the probabilistic Gaussian models of EDAs [88], and the convergence variables on multiobjective problems [89]. Variable reduction strategy is another means of representing the decision variables of an objective function or its constraints based on a smaller subset of *core* decision variables [90, 91]. Random projection theory, which was covered in part A of the survey as an implicit way of exploiting problem structure, can be used for dimensionality reduction in the context of EDAs. Weighted optimization framework, inspired from adaptive weighting by Yang et al. [94], is another way of transforming the problem into a lower dimensional one (See Section VI-C).

## IV. INITIALIZATION METHODS

Random initialization of a set of candidate solutions is at the heart of existing metaheuristic algorithms. The aim of initialization methods is to make the best use of random number generators or other sampling techniques to cover the vast search space more uniformly. This section covers

the studies on random initialization for large-scale global optimization.

A wide range of population initialization methods have been employed by evolutionary algorithms [95, 96]. There are various conclusions, conflicting at times, on the effect of initialization methods on large-scale optimization [97–99]. Kazimipour et al. [97] studied the effect of advanced initialization methods on large-scale optimization. The study suggested that EAs are more sensitive to initialization in high-dimensional spaces than in lower-dimensional ones regardless of the population size. They also reported that pseudo-random number generator is inferior to advanced initialization methods in high dimensions. A follow-up study showed that the effect of advanced initialization methods becomes marginal when the parameters of the optimizer is properly set [98]. A systematic study of advanced initialization methods on a DE variant, DE/rand/1/bin [3], showed that when the parameters of the algorithm is set close to their optimal, the statistical difference between random number generators and other initialization methods becomes insignificant.

Segredo et al. [100] showed that although overall distinction between random number generators and advanced initialization methods fades away in high-dimensional spaces, there is still a significant difference between them when best-case and worst-case performances are taken into account. They therefore concluded that the choice of the initialization method is of crucial importance, especially when a limited number of runs is allowed.

Kazimipour et al. [99] used centered $L_2$ discrepancy to measure population uniformity as a function of population size and the dimensionality of the space. They reported that the loss of population uniformity (hence diversity) due to curse-of-dimensionality is the dominant factor in the performance degradation of optimization algorithms, regardless of the choice of the initialization method. Putting differently, it is the geometric peculiarities of high-dimensional spaces that affect all initialization methods. For example, it is well-known that the contrast in distance between randomly chosen points diminish as the dimensionality of the space increases [101, 102], which has serious implications on various initialization/sampling techniques. Consequently, Kazimipour et al. [99] recommended the use of advanced initialization methods only when the population size and the problem dimensionality are low.

## V. Parallelization

In this section we review the algorithms that rely on CPU and GPU parallelization to improve solving large-scale global optimization problems.

*a) The Historical Context:* Cantú-Paz and Goldberg [103] study the scalability of parallel single- and multi-population GAs. Their goal was to find the optimal number of processors that minimizes runtime. Their analysis showed that the number of processors that minimizes the execution time is proportional to the square root of the population size and the objective function evaluation time. Munetomo et al. [104] also investigate the use of parallel processing for linkage learning

and proposed a parallel implementation of the LINC linkage learning algorithm called pLINC (see part A of the survey). They also proposed a two-level GA with a series of intra-GAs operating on the linkage groups identified by pLINC, and an inter-GA which operates at a higher level and treats the linkage groups as a whole.

*b) Large-Scale Cases:* In the context of large-scale optimization, two types of parallelization are common. The first type is specific to a particular EA, and the second type is a generic framework applicable to a wide range of EAs most of which are based on a divide-and-conquer paradigm by means of problem decomposition.

In the algorithm specific department, Mendiburu et al. [105] proposed a parallel master-slave implementation of several binary and continuous EDAs based on a Bayesian network model using MPI and POSIX threads. They parallelized the learning phase or the model building process, which often takes the maximum proportion of the execution time and tested their algorithm on 500 dimensional binary problems and 1500 dimensional continuous problems. A drawback of this study is the use of very simple benchmark problems such as OneMax for the binary case and the sphere function, which is fully separable, for the continuous case. Wang et al. [22] proposed a parallelized version of DE based on GPU parallelization and tackled continuous problems of up to 1000 dimensions. They also observed that with a fixed population size the speed-up rate decreases as the dimensionality of the problem increases. Iturriaga and Nesmachnow [106] proposed a parallel version of compact GA for CPU/GPU architectures and tested it on OneMax and noisy OneMax with up to one billion variables. They also proposed an asynchronous model on GPU which is only suitable for large-scale separable problems. More recently, Duan et al. [107] proposed a spark-based software framework for parallelization of various PSO implementations. Their proposed parallel PSO showed a super-linear speedup and was tested on continuous benchmark functions with up to $10^5$ dimensions as well as on expensive functions. Cao et al. [108] proposed a parallel quantum-enhanced DE by parallelizing the fitness evaluation of individuals.

Lastra et al. [109] proposed a GPU-based MA-SW-Chains [110], a memetic algorithm for large-scale global optimization (see part A of the survey), by parallelizing all major components of the algorithm such as fitness function evaluation, crossover, local search, random number generation, and population sorting. In another study, Cano and García-Martínez [111] proposed an improved GPU-based model for MA-SW-Chain and tested it on a scaled version of the CEC'2013 large-scale benchmarking suite on functions with up to 100 million decision variables. Cano et al. [112] proposed a parallel MA-SW-Chains by adapting it to the MapReduce framework to tackle problems with up to 10 million decision variables. The local search component of the algorithm is done using a divide-and-conquer strategy by performing local search for a subset of the decision variables.

Multi-population algorithms are common in parallelizing many metaheuristics for large-scale optimization. Ge et al. [35] proposed a multi-population topology-based island model with a master-slave paradigm to solve large-scale problems. Wang

et al. [50] proposed a distributed PSO based on randomly formed equally-size subpopulations which are co-evolved using a master-slave paradigm. Yang et al. [49] proposed a distributed swarm optimizer based multi-population master-slave model where the elites of each subpopulation are used in the velocity update rule. Su et al. [113] proposed a parallel multiobjective algorithm for community detection. They first identify the key nodes in the network graph and the communities associated with the key nodes are then detected in parallel using a multi-population model.

Problem decomposition into lower dimensional subproblems is central in several recent parallelization frameworks. Among them, Cao et al. [114] proposed a distributed parallel cooperative coevolutionary algorithm for solving large-scale multiobjective problems. They used a variant of differential grouping [115] to decompose the decision space into smaller components which are optimized in parallel using a two-level parallelization structure based on the message passing interface (MPI). The experimental results are based on 1000 dimensional DTLZ and WFG test functions. De Falco et al. [80] proposed a decomposition-based parallel model for solving expensive large-scale continuous optimization problems. They use the random grouping decomposition method and build a separate surrogate (meta-model) for each component of the problem. The components are then solved in parallel using cooperative coevolution. The proposed algorithm was tested on problems with up to 1000 dimensions. Yang et al. [116] argue that decomposition-based methods, despite their modular nature, cannot be readily parallelized due to defects in how partial solutions are evaluated. They show that the objective function used to assign a fitness to a partial solution is not consistent with the ideal fitness assignment. They address the problem of fitness assignment to partial solutions for divide-and-conquer methods by appealing to a parallel framework called naturally parallelizable divide-and-conquer.

Decomposition-based and multi-population parallelization methods can also be combined to solve large-scale problems. Hybrid two-way parallelism has also been used for large-scale optimization. These often combine: 1) Parallelism by means of problems decomposition, i.e, the problem is decomposed into several lower-dimensional subproblems, and; 2) Parallelism by means of population distributions, i.e., a distributed pool model [117]. Jia et al. [118] proposed such a two-way algorithm which controls the resource allocation by adapting the subpopulation sizes as well as the number of iterations a particular component (subproblem) is optimized. Another two-way parallelism decomposes the problem into several components based on variable interaction analysis. Each component is subsequently divided into subpopulations each receiving a processor for optimization. A resource allocator then prioritizes processor allocation as a function of components' contribution towards the overall improvement of the objective function.

## VI. RELATED RESEARCH TOPICS

In the previous sections, we reviewed common *approaches* to large-scale global optimization. In this section however,

we take a problem oriented perspective and discuss several problem areas arising in the context of large-scale global optimization, such as the problem of overlapping components (§VI-A), resource allocation and the imbalance problem (§VI-B), decision space scalability of multiobjective optimization (§VI-C), and the scalability of constrained optimization problems (§VI-D). The section concludes with a discussion on benchmarking large-scale optimization algorithms and a brief review of their real-world applications (§VI-E).

### A. Overlapping Problems

The importance of problem structure and how it can be exploited in various ways was discussed in part A of the survey. The decomposition approaches covered in part A of the series are mostly suited to partially separable problems, i.e., those with distinct independent lower-dimensional components. However, there are problems with sparse variable interaction structure which are not partially decomposable. These problems, which we refer to as *overlapping* problems, occur in many application areas such as multidisciplinary design optimization [119] and concurrent engineering [120]. Multiobjective optimization problems can also be seen as overlapping problems due to shared decision variables among the objective functions [121]. This is particularly the case when a scalarization technique is used to convert them to a series of single objective optimization problems. Constrained optimization problems may have overlapping interaction structures [122] or may become overlapping depending on the constraint-handling techniques used to handle them. The variable interaction structure of the overlapping problems can be discovered using the methods outlined in part A of the survey. However, exploiting the structure is a more challenging task as compared to partially separable problems. Despite the importance of overlapping problems, very few works have been dedicated to large-scale overlapping problems [78, 123, 124]. This section reviews some of such techniques that can help with the scalability of algorithms for large-scale global optimization.

Some approaches rely on breaking selected interactions with the aim of converting the overlapping problems into partially separable ones [125–127], or by means of special crossover operators that take the overlapping nature of the problem into account [128, 129]. Munetomo and Goldberg [125] use monotonicity checking to identify the variable interaction structure of the objective function and propose a metric to measure the linkage tightness with the aim of breaking weak interactions. Yu et al. [130] use an information-theoretic interaction detection mechanism to detect problem structure and use an entropy-based measure of a component or building-block to identify and break weak interactions. Yu et al. [131] assumes that the interaction structure is given and designs a crossover operator that partitions the interaction graph into two sub-graphs such that the disruption of overlapping components is minimized. Sun et al. [126] proposed a variation of recursive differential grouping [132], RDG3, which limits the dimensionality of components forcing some interactions to be broken as a consequence. Li et al. [127] proposed a decomposition

method based on spectral clustering, which takes the strength of interactions into account and breaks some weak interactions such that inter-group interactions are minimized and the intra-group interactions are maximized.

Thierens [128] proposed to use hierarchical clustering to represent the interactions using a linkage tree which is subsequently used to perform recombination. Bosman and Thierens [129] proposed an improved version of linkage tree recombination to eliminate superfluous hierarchical linkage relations. Experimental results have shown that this type of recombination performs well on overlapping problems.

In the context of cooperative coevolution, overlapping components or groups with shared decision variables is a common way of solving overlapping problems. Sun et al. [123] used monotonicity detection to identify the structure of an $n$-dimensional problem and form $n$ groups, one for each decision variable containing all other variables it interacts with. These $n$ groups, which are not necessarily mutually exclusive, are optimized in a round-robin fashion within a CC framework. Due to non-exclusive nature of the groups, this algorithm takes, to some extent, the overlapping nature of the problem into account. Jia et al. [133] use variable interaction analysis to identify the underlying components of the objective function and their shared variables. They then use a contribution-based mechanism to promote components with a higher contribution towards improving the objective function and assign the shared variables the components with large contributions.

Werth et al. [78] used a sliding window mechanism and optimizes the variables inside the window to reduce the dimension of the problem. The problem structure is taken into account by iteratively constructing the interaction matrix of the problem using LINC-R; however, instead of finding the entire matrix, only the interactions within a given sliding window are considered at each iteration. To deal with overlap, the Cuthill-McKee algorithm is used to reduce the bandwidth of the matrix, which places interacting variables close to each other. Song et al. [124] proposed overlapped cooperative coevolution which uses a mechanism called delta disturbance to find the most influential variables and distribute them among the existing components. Although this algorithm was not intended for overlapping problems, replication of influential variables within all components can have a positive effect on solving overlapping problems. However, this has not been verified empirically on overlapping problems. Song et al. [134] used a similar mechanism and identify important variables which can participate in multiple components to solve large-scale virtual network embedding problems.

Factored EA [135] is another framework with the capacity to decompose a problem into a set of lower dimensional subproblems. It can mimic CC as its special case and has the capacity to define overlapping components suitable for solving overlapping problems. FEA can be an effective method for solving overlapping problem if the problem structure is known *a priori*. The performance of FEA remains to be checked on large-scale overlapping problems such as those proposed in the CEC'2013 large-scale benchmark suite.

Bayesian optimization algorithm (BOA), uses Bayesian networks to represent problem structure, which is capable of capturing overlapping components [136]. Although modeling Bayesian network is computationally expensive, their flexibility make them a good choice for solving overlapping problems. Empirical evidence suggests that BOA performs better than tightness detection [137]. Clustered EDAs are also among the implicit methods that improve the identification of problem structures as compared to canonical EDAs. Emmendorfer and Pozo [138] proposed a cluster based EDA which uses the notion of concept-guided combination to better capture and exploit problem structure. This technique was shown to outperform models based on Bayesian networks.

### B. Resource Allocation and the Imbalance Problem

Efficient use of computational resources is of significant importance in large-scale global optimization. The contribution of a decision variable or a group of decision variables, belonging to an underlying subfunction within the objective function, can have a varying degree of influence on the function output. This characteristic, which is often called the *imbalance* problem, can have a detrimental effect on the optimization performance if not handled properly. For example, Chuang and Chen [139] showed that the model building process of EDAs is affected by the imbalance problem. They reported that the selected individuals based on which the probabilistic model of EDAs is updated lacks the necessary information about the linkage structure of some parts of the problem. Omidvar et al. [140] also showed that the imbalance problem renders the round-robin optimization policy of cooperative coevolution suboptimal.

Although the imbalance issue has implications in many areas such as multiobjective optimization [141, 142], constraint-handling [122], and dynamic optimization Yazdani et al. [143], it has mostly been studied in the context of optimal component selection policy of cooperative coevolution. Contribution-based cooperative coevolution (CBCC) [140] is the first algorithm of this kind. To deal with the imbalance problem, CBCC and other contribution-aware algorithms first need to estimate the contribution of components, and secondly devise an exploration/exploitation policy to update the contribution of components and to optimize the influential components longer. The algorithms which will be reviewed in the rest of this section differ in the way they handle these two aspects.

*1) Problem Decomposition and Quantifying Contributions:* Problem decomposition and quantification of contributions are important prerequisites of an effective resource allocation policy. Under the black-box assumption, all we can observe is the objective value and how it changes over time. In the literature, the improvement on optimizing a subset of the decision variables (a component) can have on the objective function value is taken as a unit of improvement or *contribution*. This value can be quantified for an arbitrary subset of the decision variables irrespective of whether or not they belong to an underlying subfunction. For a partially separable problem, if the ideal decomposition is known, one can study the effect of a single component on the objective value by freezing all other components. However, this may not be possible for an overlapping problem (see Section VI-A) where an optimal

decomposition may not be known. It is therefore apparent that problem decomposition and estimation of contributions are closely interconnected.

Most algorithms define the contribution of a component to be a function of the improvement it makes on the overall objective value when it is optimized for a predetermined number of iterations while all other components are kept constant. Let $\delta_k^{(t)}$ be the improvement we get at time $t$ when the $k$th component is optimized for $\tau$ iterations. Based on this definition, the two original versions of the CBCC algorithms (called CBCC1 and CBCC2), define the contribution of the $k$th components to be $\frac{1}{T}\sum_{t=1}^{T}\delta_k^{(t)}$, i.e., the average of all previous improvements up to the current iteration. Due to non-stationary nature of the underlying distribution of the contributions, some algorithms defined the contribution as a moving average over the last $L$ iterations. CBCC3 [144] used the extreme case of $L = 1$, while in the case of CCFR [145] $L = 2$. CCFR2 [146] improves upon CCFR by averaging the improvements per function evaluations to account for unequal subpopulations. CCFR2 [146] and some other algorithms [133, 142] exponentially decay the effect of historical contributions. Ren et al. [147] define the contribution as a function of both $\delta_k^{(t)}$ and the standard deviation across all components. Some authors suggested various normalization of $\delta_k^{(t)}$ [118, 133, 142, 148, 149] as the contribution of a component.

In addition to $\delta_k^{(t)}$ defined previously, other measures of contributions have also been proposed. Global sensitivity analysis techniques such as Morris screening is used in several works as the contribution measure for individual variables as well as components [150–152]. Delta disturbance is a perturbation method proposed to measure the contribution of individual variables and finding the most influential variables for further optimization. Using the plain fitness of a component as the contribution of a component has also been suggested [153].

*2) Resource Allocation Policies:* A simple allocation policy, the variations of which are used by many algorithms, is to complement the round-robin policy of canonical CC by one or more iterations of optimizing the highest contributing component (as measured by the methods outlined in §VI-B1 and Table I). CBCC1 [140] is the most conservative approaches in which the round-robin is followed by only one episode of optimizing the highest contributing component (also employed by SACC1 [152]. CBCC2 [140] is greedier and exploits the highest contributing component until no improvement is observed (also employed by SACC2 [152]). This is shown to be an unstable policy because the contribution of initially best component may not remain the best for the rest of a run. CBCC3 [144] addresses this issue by optimizing the highest contributing component until its contribution drops below the second best. This has the effect of equalizing contributions. CBCC3 also randomly enters an exploration phase where all components get a chance of updating their contributions. CCFR [145] and CCFR2 [146] use a similar equalization strategy and includes a stagnation detection mechanism to avoid optimizing stagnant components. FCRACC [147] is similar to CBCC3 in that it exploits the best component, but their method

of quantifying contributions differs (see §VI-B1). Meselhi et al. [149] use round-robin followed by a fuzzy rule-based allocation based on the contribution and population diversity. Shen et al. [142] use a roulette-wheel selection mechanism based on the contributions. Jia et al. [133] optimize all the components whose contribution is more than half the best contribution. Although not specifically designed to address the imbalance problem, overlapped CC [124] replicates influential decision variables in more than one component, which has the effect of optimizing influential variables more often.

In addition to the heuristics described above, some algorithms define the resource allocation policy as a function of the estimated contributions. SACC3 [152] determines the number of times a component is optimized ($\tau$) as a function of its effect as measured by Morris screening. CCAOI [148] normalizes the contributions according to Gini index and allocates the computational resources accordingly. DCCA [118] uses a distributed model and allocates more CPU instances to better contributing components. The population size and $\tau$ are then defined to be functions of the number of CPUs assigned to a component. BBCC [154] uses multi-arm bandit approaches such as $\epsilon$-greedy, SoftMax, or Upper Confidence Bound (UCB) to select the components based on their contributions aiming at balancing exploration and exploitation in a more systematic way.

*3) Component Selection as a Multi-Armed Bandit Problem:* Despite being effective in outperforming the canonical CC, the contribution-aware algorithms discussed so far are based on a set of heuristics derived form empirical observations with minimal theoretical basis. Some authors proposed that the component selection policy of CC can be treated as a multi-armed bandit (MAB) problem [154, 155]. Among these, Kazimipour et al. [154] defined and mapped the building blocks of a contribution-aware CC into a MAB framework. This framework, bandit-based CC (BBCC), has the flexibility to mimic the previously described algorithms as a special case. BBCC treats the contribution of a component as the utility or the value function in reinforcement learning. This estimated contribution or long-term utility is defined to be a function of immediate improvements or rewards measured by quantities such as $\delta_k^{(t)}$. Given this interpretation, a wide range of contribution estimators such as moving average (simple, weighted, or exponential), rank-based, or hybrid estimators can be used. The component selection policy is also responsible for balancing between exploration and exploitation, which can be done using a wide range of existing selectors such as $\epsilon$-greedy, $\epsilon$-first, GreedyMix, LeastTaken, SoftMax, UCB, and other similar selectors widely used in the multi-arm bandit and reinforcement learning literature. The simplest instance of BBCC with a normalized $\delta_k^{(t)}$, and simple averaging as the contribution estimator, and $\epsilon$-greedy has shown to outperform all CBCC family of algorithms, as well as CCFR and MOF-BVE on the CEC'2013 large-scale benchmark suite.

*C. Multiobjective Optimization*

Multiobjective optimization problems are prevalent in a wide range of application areas [156]. As the name implies,

TABLE I: List of contribution-aware algorithms with a short description of their method of estimating the contribution of components as well as their resource allocation policy.

| Algorithm | Estimating Contribution | Resource Allocation Policy |
|---|---|---|
| CBCC1 [140] | Moving average: all previous contributions $\sum \delta_k^{(t)}$ | Round-robin followed by the best component (once) |
| CBCC2 [140] | see CBCC1 | Round-robin followed by the best until stagnation |
| CBCC3 [144] | the most recent contribution ($\delta_k^{(t)}$) | The best component while better than the second best with occasional round-robin |
| CCFR [145] | Average of two most recent contributions | Equalize contributions then round-robin. It also has stagnation detection |
| BBCC [154] | A value function based on improvements | Multi-Arm Bandit: $\epsilon$-greedy; SoftMax; UCB, etc |
| SACC1 [152] | Sensitivity analysis: Morris screening | See CBCC1 |
| SACC2 [152] | Sensitivity analysis: Morris screening | See CBCC2 |
| SACC3 [152] | Sensitivity analysis: Morris screening | Defining $\tau$ as a function of sensitivity analysis mechanism (Moris Screening) |
| OCC [124] | Delta disturbance | Replication of important decision variables in multiple components |
| CCAOI [148] | Normalized contributions using Gini index | The allocated resource is a function of Gini's index |
| FCRACC [147] | Weighted average of contributions and their st. dev | Optimizes the highest contributing component |
| DCCA [118] | see CBCC1 | Distributed model (CPUs per component); Population size and $\tau$ are functions of # of CPUs |
| IMMO-CC [153] | Bi-objective: fitness and diversity | Components in the first front as determined by non-dominated sorting |
| CCFR [145] | Weighted average of two most recent contributions (normalized) | Equalize contributions then round-robin with stagnation detection |
| CBCCO [133] | Weighted average of all previous contributions | Round-robin followed by all components with contributions higher than half of the best |
| ICRA [142] | Weighted average of all previous contributions | Probabilistic roulette-wheel based on the contribution |
| F3C [149] | Most recent normalized contribution | Round-robin followed by fuzzy rule-based allocation based on contribution and diversity |

these problems have two or more conflicting objectives causing them to have multiple trade-off solutions known as the Pareto-optimal solutions. Scalability of multiobjective problems can be studied in either the objective space [157] or the decision space. The former refers to the effect of the number of objective functions on the performance of the algorithms [157], while the latter is concerned with the scalability of each objective function with respect to its number of decision variables. In this section, we address the scalability of multiobjective problems in the decision space, which has attracted attention in the last decade [158–160].

Large-scale multiobjective approaches can be seen in four major categories. 1) Problem decomposition where the aim is to break the problem into a set of lower dimensional subproblems in the decision space. 2) Problem transformation where the aim is to reformulate the original problem into a single lower dimensional problem. 3) Operator design where the aim is to devise more efficient solution generation mechanisms. 4) Problem reduction and intrinsic dimensions where the aim is to find a lower dimensional latent space embedded in a higher dimensional sparse space.

Problem decomposition approaches, not to be confused with scalarization techniques such as Tchebychev [161], operate in the decision space and aim at forming a set of smaller and more manageable subproblems. Such decompositions are often done by considering the interaction structure of the decision variables [115], or by grouping the variables based on their effect in the objective space, i.e., those pertaining to the convergence of solutions towards the Pareto-optimal front, and those pertaining to diversity of the solutions on the Pareto-optimal front [162, 163]. Problem decomposition by means of variable interaction detection is a challenging task in large-scale multiobjective optimization for the following major reasons: 1) Lack of a consistent partially separable grouping across all objectives; 2) The effect of problem formulation on variable interaction [164]. For example use of scalarization techniques may result in an sparse but overlapping interaction structure due to shared decision variables among several objectives (see Section VI-A for more information about overlapping problems); and 3) Cost of variable interaction detection

for several objectives. Cooperative coevolution has been used with several decomposition strategies such as random grouping [165, 166], multi-level random grouping [167], differential grouping [114, 168], and monotonicity detection [57, 162] to solve large-scale multiobjective problems. In some cases several variable decomposition methods are combined [142], or completely new ones are proposed [169]. For instance, Ma et al. [170] proposed to use a convergence relevance degree to decompose the problem and, Wang et al. [169] used a tensor canonical polyadic decomposition to divide the $d$-order tensor of decision variables into a set of uncorrelated components. Some other decomposition techniques [114, 162, 163, 168, 171] divide the decision variables based on their dominant role in the optimization problems: convergence of solutions on the Pareto-optimal front and diversity of solution on the Pareto-optimal front. In such techniques it is customary to further divide the convergence matrix using variable interaction methods described before.

An alternative to problem decomposition is to *transform* the original large-scale problem into a single low-dimensional problem. This transformation can be implemented using well-known methods such as PCA to obtain a lower-dimensional representation of the convergence variables [89], regression analysis to represent a subset of variables as a function another subset [172], or other reformulation techniques [86, 173]. One such reformulation which has gained attraction in recent years and has shown to outperform some state-of-the-art methods [174], is the weighted optimization framework (WOF) [86, 175]. Inspired by the notion of *adaptive weighting* [94], WOF reduces the dimensionality of the decision space by forming several groups of the decision variables, often using some variable grouping method [176], and transforming them using a given transformation function parameterized by a weight vector.

Problem reduction is another approach where the algorithm attempts to find the intrinsic dimensionality of the problem, which is often substantially lower than its nominal dimensions. Sparse multiobjective problems arise in many practical application areas where the decision value of many Pareto-optimal solutions are zero. The essence of these algorithms is to find

the sparse distribution of the decision variables using techniques such as auto-encoders [177] or pattern mining [178], and devising mechanisms capable of taking the sparsity into account [179, 180].

The previous approaches stated above are all based on operating in a lower-dimensional space. There are also a range of algorithm-specific ways traversing the search space more effectively. For instance, Yi et al. [181] studied and improved the crossover operator of NSGA-III [182]. Some studies proposed new update strategies for PSO to improve its convergence and diversity properties for large-scale multiobjective problems[42, 57, 183]. Similarly, quantum-enhanced DE [108] and variable-importance-based DE [141] have also been proposed to tackle high-dimensional multiobjective problems. Designing better local and neighborhood search mechanisms [113, 184, 185] and better solution selection mechanisms [186, 187] are other ways of improving the overall search efficiency.

Multiobjective algorithms can be categorized into dominance-based, decomposition-based, or indicator-based approaches based on how they handle the multiplicity of objective functions. Table II summarizes the large-scale multiobjective algorithms based on these three approaches. Figure 4 shows the percentage of each approach used in the large-scale multiobjective literature. As can be seen, dominance- and decomposition-based approaches are the most dominant and indicator-based approaches are the least explored. Here generic refers to the frameworks which are neutral to the choice of optimizer.

TABLE II: Categorization of large-scale algorithms with respect to their strategy of handling multiple objectives.

| Type | Reference |
|---|---|
| Dominance | Cheng et al. [57], Shen et al. [142], Zhang et al. [163], Antonio and Coello [165], Wang et al. [169], Chen et al. [171], Liu et al. [176], Tian et al. [177, 178, 179, 180], Yin et al. [183], Qin et al. [185], Xiao et al. [188], Zhang et al. [189, 190], Ho et al. [191], Gong et al. [192] |
| Decomposition | Tian et al. [42], Liu et al. [89], Su et al. [113], Cao et al. [114], Liu et al. [141], Antonio and Coello [166], Song et al. [167], Ma et al. [170], Yi et al. [181], Cota et al. [184], Zhang et al. [186], Wang et al. [193], Shang et al. [194] |
| Indicator | He et al. [173], Hong et al. [187] |
| Generic | Zille et al. [86], Brownlee et al. [164], Cao et al. [168], Zille et al. [175] |

### D. Constraint Handling

Constraints are indispensable part of many real-world optimization problems [195]. As a result, a wide range of constraint-handling methods has been proposed for evolutionary algorithms and nature-inspired metaheuristics [196]. Despite the plethora of constraint-handling techniques, they suffer from the curse of dimensionality and very few studies have been dedicated to the topic of scalability in constrained optimization. Constraint-handling methods have the following scalability challenges: 1) High dimensional objective and/or constraint functions; 2) Dependence between the number of constraints and the number of decision variables. For a large-scale problem this may result in a highly constrained problem; 3) Complex problem structure due to shared decision variables among the objective function and the constraints; and 4) The
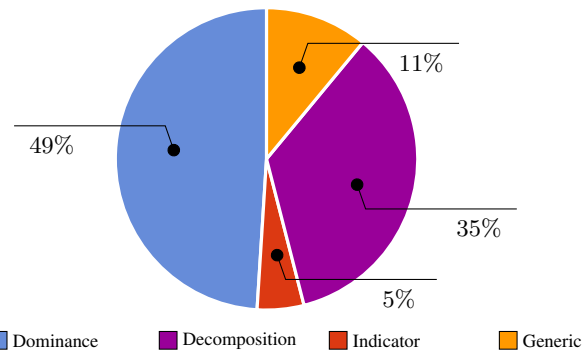


Fig. 4: Proportion of papers according to Table II using different approaches to multiobjective optimization.

effect of constraint-handling method on problem structure and variable interaction. For example, a simple penalty method can convert a partially separable problem into an overlapping one (see Section VI-A).

Constraint-handling techniques used in large-scale global optimization are mostly based on problem decomposition and variable interaction analysis. Accurate decomposition has been shown to have a significant impact on reducing the number of constraint violations [197]. Fitness difference minimization [198] and the differential grouping family [115] are two of the most widely used decomposition methods in large-scale constrained optimization.

Sayed et al. [198] used a fitness difference minimization approach to analyze the interaction structure of the objective and constraint functions and decompose them into a set of smaller subproblems. Aguilar-Justo and Mezura-Montes [199] improved upon [198] and used an aggregate function of all constraint violations, rather than the individual constraints, for interaction analysis and problem decomposition. A problem of fitness difference minimization methods is the need for specifying the number or the size of components. To fix this, Aguilar-Justo et al. [200] proposed to evolve the best arrangement of the decision variables as well as the number of components using GA to solve large-scale constrained problems.

Finite difference decomposition methods (part A §II-B) are generally more accurate than fitness difference minimization. A study shows that one such algorithm, differential grouping version 2 [115], is more robust and therefore better suited to complex highly constrained problems [201]. Blanchard et al. [202] used a variant of differential grouping, IDG [203], to form an interaction structure matrix for the objective function and the constraints. The aggregate interaction matrix of interactions are then used to decompose the objective as well as the constraints. Xu et al. [122] also proposed a coevolutionary algorithm based on differential grouping and used a contribution-based mechanism to allocate resources to components based on their contributions and degree of constraint violations.

In addition to problem decomposition, other approaches

such as surrogate modelling [72], memetic algorithms [201], offspring generation [204], and boundary constraints violations [69, 205] are also studied in the context of large-scale constrained optimization. Approximation and variable reduction techniques also appear to be promising approaches for solving large-scale constrained problems [90, 91] requiring further investigation.

### E. Benchmarks and Applications

Although the ultimate goal of designing efficient algorithms is to solve real-world problems, their sheer complexity due to entanglement of various aspects such as constraint-handling and dealing with mixed variable types, limits one's ability to conduct a focused study of a particular aspect of a problem common to a wider range of problems. Benchmark problems address this issue by capturing practical aspects such as dimensionality, modality, structure, constraints, variable types, noise, etc. into a set of tunable well-defined functions [206]. In the context of large-scale global optimization, the IEEE CEC large-scale global optimization benchmark suites [207–209] are the most widely used. All the benchmarks are based on a set of base functions popular in numerical optimization [210], such as Rastrigin, Rosenbrock, Ackley, Schwefel, Sphere, Elliptical, Griewank, and many more.

The CEC'2008 large-scale suite [207] contains a set of seven functions which are tested in 100, 500, and 1000D dimensions. This is a very small set and lacks modularity, i.e., systematic control over how decision variables are linked. The CEC'2005 suite, though not labeled as "large-scale" is scalable but used mostly in low dimensional ($\approx$30D) analyses. It introduces composite functions[1] through weighted sum of a series of base functions, which can potentially cause partial interaction between decision variables. However, this is not implemented in a systematic way to give full control over problem structure. Herrera et al. [19] used a similar approach to propose a set of 19 functions used in the special issue of Soft Computing Journal on scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems [212].

To facilitate the study of variable interaction and decomposition-based algorithms, the CEC'2010 large-scale suite [208] introduced modularity into the benchmarks where the number of component functions and their participating decision variables are known and controllable. The benchmark contains a set of 20 1000-dimensional functions in three categories: fully separable, two types of partially separable functions with and without a fully separable component, and fully nonseparable. The CEC'2013 large-scale suite [209] addressed the shortcomings of its predecessor by introducing nonuniform component sizes, imbalance among the contributions of components, and overlapping functions the components of which may not be disjoint. It also includes transformations such as symmetry breaking, ill-conditioning, and local irregularities proposed in black-box optimization

---

[1]Composition used by Suganthan et al. [211] does not refer to composite functions generally defined as $f(g(x)) \equiv (f \circ g)(x)$. It refers to weighted sum of a set of subfunctions.

benchmark (BBOB) suite [213]. Sun et al. [214] extended the CEC'2013 suite to make it more tunable. Other studies which paid attention to problems with overlapping components are conducted by Sayed et al. [215] and Werth et al. [78].

There are other scalable benchmarks that have been used to study various other aspects of large-scale optimization algorithms. COPS 3.0 represents a repertoire of scalable and constrained problems found in various areas of engineering and sciences. Goh et al. [216] use EEG big data optimization problem and propose BigOpt2015 benchmark suite for large-scale multiobjective optimization. Cheng et al. [217] adopted the ideas proposed in [208, 218] to propose a set of large-scale multi- and many-objective benchmark functions. Scalable benchmark functions for constrained optimization problems are very limited. Beside the constrained problems compiled by Dolan et al. [219], which are not widely used in the EC community, Sayed et al. [198, 215] also proposed a set of artificial benchmark functions for constrained problems. Recent studies also attempted to extend standard dynamic optimization benchmarks to study large-scale dynamic optimization problems [143, 220]

*Applications:* Benchmarks are used to ultimately help with designing and evaluating efficient algorithms for solving real-world problems. Some studies directly used real-world problem instances, in addition to artificial benchmarks, to compare algorithms. Table S-III of the supplementary document shows a set of high-dimensional optimization problems in a wide range of application areas. The problem types include real-valued, integer/binary, mixed integer, and combinatorial, and about half of these include constraints. Among the approaches, divide-and-conquer methods such as problem decomposition and coevolutionary algorithms are the most popular followed by hybrid methods (memetic algorithm, local search hybridization, and ensembles), which is consistent with our observations in part A of this survey series. Other approaches include parallelization, approximation and encoding schemes, and algorithm specific sampling and variation operator design. Curse of dimensionality is the predominant source of difficulty among these application areas and the existence of other factors such as constraints, noisy and non-smooth objective functions add to the complexity. For the continuous decomposition-based approaches, noise and non-smooth functions appear to be an obstacle to an accurate variable interaction analysis. In the case of combinatorial problems, finding an effective decomposition is a major challenge in its own right. A challenge for approximation and encoding schemes is model resolution or granularity which mediates between the accuracy of the model and its complexity. Among the constrained problems, handling of infeasible solutions and the interplay between dimensionality and the number of constraints are two of the most important challenges.

*Large-Scale Global Optimization Competitions:* In this section, we review the results of large-scale global competitions since 2008 when the first IEEE CEC competition on large-scale global optimization was held. Table III lists the winners and runner-ups. Based on the approaches to large-scale global optimization we outlined in this paper, high performing algorithms almost exclusively belong to either memetic algorithms

TABLE III: Winning and runner-up algorithms of large-scale global optimization competitions since 2008.

| Year | Competition | Algorithms | |
|------|-------------|------------|------|
| | | Winner | Runner-up |
| 2008 | IEEE CEC'2008 | MTS [221] | LSEDA-gl [222] |
| 2010 | IEEE CEC'2010 | MA-SW-Chains [110] | EOEA [223] |
| 2011 | Soft Computing Special Issue | MOS [224] | – |
| 2013 | IEEE CEC'2013 | MOS [225] | DECC-G [94] |
| 2015 | IEEE CEC'2015 | MOS [225] | IHDELS [226] |
| 2018 | IEEE CEC'2018 | SHADEILS [227] | LSHADE-SPA [228] |
| 2019 | IEEE CEC'2019 | CC-RDG3 [126] | MPS [229] |

(part A §III) or decomposition-based algorithms (part A §II) with memetic algorithms and local search dominating the competition.

It is interesting to note that the algorithmic philosophy of these two approaches are orthogonal and complementary to each other. The premise of memetic algorithms is to balance exploration and exploitation by means of combining global and local search operators. Whereas the main premise of decomposition methods is interaction-aware space partitioning and dimensionality reduction. Consequently, memetic algorithms lack an intrinsic mechanism for dealing with variable interactions and systematic space partitioning, and decomposition methods lack an intrinsic mechanism for balancing exploration and exploitation. It is therefore not surprising to see that the effort of hybrid algorithms is focused on proposing novel ways of balancing exploration and exploitation, and the effort of decomposition methods is centered around finding accurate interaction detection principles and effective grouping. This leaves both approaches with major blind spots. The absence of interaction detection mechanism in hybrid frameworks puts an extra burden on the exploration process by searching the regions which could have been avoided through partitioning, and reduces the efficiency of dimension-wise local search due to ignored interactions. Most decomposition-based methods also discount the role of component optimizer in balancing exploration and exploitation. This deficiency is partially compensated for by contribution-aware algorithms (see §VI-B) which balance exploration and exploitation at the component level.

The design biases of hybrid and decomposition-based methods stated above can also be seen among the competition winners and runner-ups listed in Table III. MTS [221] uses orthogonal arrays combined with a mixture of local operators. Orthogonal arrays give an expansive initial coverage of the search space which forms the basis for the subsequent local search process. MA-SW-Chains [110], IHDELS [226], and SHADEILS [227] combine a global search algorithm with a chain of iterated local search attempts to balance exploration and exploitation. MOS [224, 225], despite being a framework capable of hybridizing any set of operators, uses a mixture of global and local operators to control exploration and exploitation. Even the algorithms such as MPS [229] and LSEDA-gl [222], which are not memetic by definition, have explicit elements of balancing the exploration and exploitation forces. MPS [229], for instance, proposes a mechanism to disentangle the exploration and exploitation mechanisms with the aim

of minimizing failed and deceptive exploration attempts and maximize the successful ones. LSEDA-gl [222] also hybridizes heavy-tailed distributions such as Lévy to promote exploration with the classic Gaussian distribution to promote exploitation.

Among the competition winners and runner-ups listed in Table III, LSHADE-SPA [228] and EOEA [230] are the only hybrid algorithms that consider both exploration/exploitation balance as well as problem decomposition. On the decomposition side however, they both fall short of using an effective variable interaction detection method. LSHADE-SPA [228] uses the outdated random grouping [94] known to perform poorly on partially separable problems (see §II of part A) and EOEA [230] uses a grouping strategy incapable of an effective space partitioning. Due to their emphasis on accurate interaction analysis, decomposition methods are historically not systematic with their choice of component optimizer, which limits their ability in maintaining good exploration/exploitation balance within the lower-dimensional subspaces. This is perhaps why hybrid algorithms are dominant in competitions [231, 232]. As a matter of fact, decomposition-based algorithms are capable of using *any* component optimizer, including memetic algorithms and other hybrids, to further improve their scalability. This is why the combination of accurate problem decomposition (RDG3 [126]) and good component optimizer (CMA-ES) resulted in superior performance by CC-RDG3 in 2019. Further evidence also suggests that several decomposition-based algorithms not present in competitions can outperform competition winners. For example, the decomposition-based algorithm proposed by Mei et al. [233] outperformed MA-SW-Chains [110], the winner of CEC'2010 competition on both CEC'2008 can CEC'2010 LSGO benchmark suites, and recursive differential grouping [132] outperformed MA-SW-Chains [110] and MOS [225] on the CEC'2010 and CEC'2013 LSGO benchmark suites. It is often stated that the cost of decomposition prohibits their use in large-scale settings. However, recent advances in variable interaction and grouping methods allows this to be achieved in $\mathcal{O}(n \log n)$ in the general case and $\mathcal{O}(n)$ on separable functions (see part A §II-B for more details).

## VII. CONCLUDING REMARKS

In the two parts of this survey, we reviewed a wide range of population-based metaheuristics for large-scale global optimization in six major categories: problem decomposition, hybridization and memetic algorithms, sampling and variation operators, approximation and surrogate modeling, initialization methods, and parallelization. We reported on the state-of-the-art and what has been achieved over the last decade. In this section, we change perspective and try to touch upon two major issues pertaining to the future of the field: 1) Where do we stand as a field and what are the potential pitfalls and challenges hindering the progress of the field? 2) Where to go next? What are the pressing open questions and where more focus is needed?

### A. Large-Scale Global Optimization: Pitfalls and Challenges

*The big picture:* Despite the advances in various areas of large-scale global optimization, sometimes their relation to

the bigger picture is unclear. This is partly due to lack of a clear measure of the progress in the field and lack of clarity about its *grand challenges*. The bulk of the research in the field is currently driven by showing statistically significant improvements over existing results with minimal reference to whether these so-called *significant* results are actually meaningful in real-world settings. There is also an overemphasis of the *success* stories, rather than giving insights into *why* an algorithm *fails* on a particular problem or a class of problems. This is partly due to departure from the scientific method in conducting research in favor of an engineering approach where comparison with the state-of-the-art is encouraged.

The lack of a big picture may cause the field to focus too much on *nice-to-have* incremental research rather than addressing the core issues of large-scale global optimization. This deficiency currently manifests itself in at least three forms:

- Emergence of new 'metaphor'-based algorithms: a wide array of metaphor-based metaheuristics have been proposed in recent years. These "novel" algorithms are often a marginal variation of an existing algorithm under the disguise of new terminology. In large-scale global optimization, some works claim novelty by simply applying one such new metaheuristic to solve some standard large-scale benchmark suite. This is very detrimental to the field and "take the field of metaheuristics a step backward rather than forward" [234].
- Ad hoc improvements of algorithms with marginal scientific or practical significance: This type of work often present a minor variation of an existing algorithm, which *statistically* improves upon the previous results despite the magnitude of the difference being negligible for any practical purpose. As an example, applying a known parameter adaptation technique to dynamically control the parameters of a new metaheuristic algorithm falls short of addressing major challenges of the field.
- The theory-practice gap: given the ever growing need for scalable optimization algorithms in a wide range of application areas, the gap between theory and practice in terms of the problem sizes currently being tackled is widening. In other words, the large-scale problems being studied now using the standard benchmarks is far from the large-scale problems faced in practice.

To avoid falling prey to these defects, we need to check where we stand as a community in relation to identifying and addressing the grand challenges of the field and bridging the gap between theory and practice. This perhaps requires a separate quantitative in-depth investigation of the large body of reported results, which is outside the scope of this paper.

*Comparison:* Despite the availability of relatively standardized and widely used benchmark suites, it is still hard to compare the reported results across a wider body of publications to be able to see the major patterns and trends. Despite some attempts to develop automated comparison tools such as the Toolkit for Automatic Comparison of Optimizers (TACO)[2], we currently do not know the answer to questions

such as the following: Given a specific function or family of functions, which algorithm or class of algorithms perform the best and why? In continuous problems, especially due to the imbalance effect, the overall solution quality may seem poor, but the solution may indeed be close to the global optimum. Based on the reported results, we currently do not know how far the solutions are from the global optimum.

The large-scale global optimization competitions also acted as a venue to compare a wider range of algorithms, but their conclusions remain limited due to the absence of several state-of-the-art algorithms from the competitions. We do not know to what extent does the competition outcomes depend on the allotted number of objective function evaluations. Do the conclusions change if the algorithms are given less or more resources?

Answering some of the questions stated above can help in finding the recurring issues and bottlenecks and help with shaping the big picture and identifying the core issues of the field.

*Adoption:* Lack of streamlined and easy-to-use software packages make the adoption of the recent developments very difficult for practitioners or other researchers outside the field. The most recent algorithms are often sophisticated and hard to implement which is a stumbling block in the way of their wider adoption.

### B. Potential Areas for Future Research

*a) The synergy between optimization and learning:* Deep learning problems are in essence high-dimensional problems with the potential to contain millions or billions of decision variables. Although evolutionary algorithms have shown competitive results on high-dimensional learning problems [235, 236], research on devising population-based algorithms to tackle large-scale learning problems is scarce [237–242]. Population-based metaheuristics in general, and evolutionary algorithms in particular, are suited for environments that require hard exploration. As a result they can be competitive in areas such as neural architecture search [243], training of deep neural networks [236], and reinforcement learning [244]. In reinforcement learning for instance, evolution strategies has shown to perform better than policy gradient on Atari 2600 games. These methods are particularly suitable when the effective number of time steps is long, the actions have long-lasting effects, and no good value function estimator is available [244].

Conversely, machine learning algorithms can be used in conjunction with large-scale global optimization algorithms to improve their scalability. For example, machine learning can be used as a general approach to learn and discover a problem's structural information from available data [245]. The learned model can then be applied to unseen data for the purpose of classification or time-series prediction. An optimization process such as branch-and-bound can be modelled as a decision making process, hence a machine learning model can be applied, to learn the most efficient and effective way [246]. This will go a long way in helping an algorithm's ability to scale to higher dimensional problems. Similarly,

---

[2]https://tacolab.org/

machine learning algorithms can be used to reduce the size of an optimization problem before being tackled. For examples, some recent work on employing machine learning techniques to learn from known instances that contain optimal solutions in order to reduce the problem first, without losing the optimal solutions [247, 248].

*b) The synergy between metaheuristics and classic mathematical programming:* Several classic derivative-free optimization algorithms have been successfully used as local search operators in the context of memetic algorithms (c.f. part A §III). Other promising areas of research at the intersection of population-based optimization algorithms and classic mathematical programming are as follows:

- Problem decomposition: In classic mathematical programming, there exist some decomposition methods such as Column Generation, Bender's cut, and Dantzig-Wolfe decomposition[249]. These techniques can be effective under certain assumptions such convexity or linearity. An important question to ask is how one would combine the merits of metaheuristics with these decomposition methods to tackle real-world LSGO problems that are often non-convex and nonlinear? As an example, machine learning has been used to learn when Dantzig-Wolf decomposition is effective on mixed-integer linear programming problems [250].

- Variation operators: Many large-scale real-world optimization problems are combinatorial by nature, e.g., either discrete, binary, or mixed types. Examples include large-scale traveling salesman problems or other graph-based optimization problems. Designing metaheuristic variation operators (such as crossover and mutation) in order to produce new solutions from the existing ones can be a significant challenge. Most existing successful methods are conventional mathematical programming methods such as branch-and-bound and branch-and-cut methods. Hybrid methods that combine the merits of meta-heuristics and exact methods (e.g., taking advantage of the "shared information" in a meta-heuristic population) is a promising direction [251–254].

*c) Exploiting problem structure:* Exploiting problem structure and grey-box optimization has shown to be effective ways of solving large-scale problems (part A §II). These structural information can be used in the form of explicit decomposition or implicitly through model building. The challenge of explicit methods is the cost of offline variable interaction learning, which requires objective function evaluations and causes an overhead on the overall optimization cost. Another issue is that a crisp decomposition is sometimes impractical due to various forms of couplings caused by the existence of multiple objectives, overlapping components (shared variables among subfunctions), or coupling through constraints. Implicit methods also suffer from the accuracy of capturing problem structure, especially when the problem size grows in size. Finding more efficient and effective ways of exploiting structural information, such as overlap, can have a significant impact on improving the scalability of optimization algorithms.

*d) Noise, dynamism, and uncertainty:* Scalability of optimization algorithms in the presence of noise, dynamical changes of the landscape, and uncertainty has scarcely been studied with only few papers addressing these issues [143, 220]. These problem types pose a range of new challenges to the existing approaches to large-scale optimization presented before. For example, variable interaction analysis methods are designed based on the assumption that the objective function is noiseless. The dynamical changes of the landscape can change the structural properties of the objective function which makes it difficult for the explicit and implicit methods to exploit these information in an efficient manner.

*e) Constraint handling:* Constraints are indispensable part of most real-world optimization problems; however, very limited studies have been dedicated to the effect of problem dimensionality on constraints [198, 202] (also see §VI-D). There is still a lack of efficient constraint handling tools to cope with high-dimensional constraint functions, or the cases where the number of constraints is a function of the dimensionality of the objective function [255, 256]. In the later case, the problem may contain a large number of low dimensional constraints. The field currently lacks scalable and controllable constrained benchmark problems either synthetic or based on real-world problems [195].

## REFERENCES

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. of IEEE International Conference on Neural Networks*, vol. 4, 1995, pp. 1942–1948.

[2] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[3] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE transactions on evolutionary computation*, vol. 15, no. 1, pp. 4–31, 2010.

[4] M. Sepesy Maučec and J. Brest, "A review of the recent use of Differential Evolution for Large-Scale Global Optimization: An analysis of selected algorithms on the CEC 2013 LSGO benchmark suite," *Swarm and Evolutionary Computation*, vol. 50, p. 100428, Nov. 2019.

[5] A. W. Mohamed, A. A. Hadi, and A. K. Mohamed, "Differential Evolution Mutations: Taxonomy, Comparison and Convergence Analysis," *IEEE Access*, vol. 9, pp. 68 629–68 662, 2021.

[6] M. Z. Ali, N. H. Awad, and P. N. Suganthan, "Multi-population differential evolution with balanced ensemble of mutation strategies for large-scale global optimization," *Applied Soft Computing*, vol. 33, pp. 304–327, 2015.

[7] A. Banitalebi, M. I. A. Aziz, and Z. A. Aziz, "A self-adaptive binary differential evolution algorithm for large scale binary optimization problems," *Information Sciences*, vol. 367, pp. 487–511, 2016.

[8] J.-i. Kushida, A. Hara, and T. Takahama, "Rank-based differential evolution with multiple mutation strategies for large scale global optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2015, pp. 353–360.

[9] Y. Wang, B. Li, and X. Lai, "Variance priority based cooperative co-evolution differential evolution for large scale global optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2009, pp. 1232–1239.

[10] A. W. Mohamed and A. S. Almazyad, "Differential evolution with novel mutation and adaptive crossover strategies for solving large scale global optimization problems," *Applied Computational Intelligence and Soft Computing*, vol. 2017, 2017.

[11] A. W. Mohamed, "Solving large-scale global optimization problems using enhanced adaptive differential evolution algorithm," *Complex & Intelligent Systems*, vol. 3, no. 4, pp. 205–231, Dec. 2017.

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

[13] H. Wang, Z. Wu, S. Rahnamayan, and D. Jiang, "Sequential de enhanced by neighborhood search for large scale global optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2010, pp. 1–7.

[14] C. García-Martínez, F. J. Rodríguez, and M. Lozano, "Role differentiation and malleable mating for differential evolution: an analysis on large-scale optimisation," *Soft Computing*, vol. 15, no. 11, pp. 2109–2126, 2011.

[15] J. Zhang and A. C. Sanderson, "Jade: adaptive differential evolution with optional external archive," *IEEE Transactions on evolutionary computation*, vol. 13, no. 5, pp. 945–958, 2009.

[16] Q. Yang, H.-Y. Xie, W.-N. Chen, and J. Zhang, "Multiple parents guided differential evolution for large scale optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2016, pp. 3549–3556.

[17] H. Wang, Z. Wu, and S. Rahnamayan, "Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems," *Soft Computing*, vol. 15, no. 11, pp. 2127–2140, 2011.

[18] S. Rahnamayan, H. R. Tizhoosh, and M. Salama, "Opposition-based differential evolution algorithms," in *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*. IEEE, 2006, pp. 2010–2017.

[19] F. Herrera, M. Lozano, and D. Molina, "Test suite for the special issue of soft computing on scalability of evolutionary algorithms and other metaheuristics for large scale continuous optimization problems," 2009.

[20] H. Hiba, S. Mahdavi, and S. Rahnamayan, "Differential evolution with center-based mutation for large-scale optimization," in *IEEE Symposium Series on Computational Intelligence*. IEEE, 2017, pp. 1–8.

[21] J. Brest, A. Zamuda, B. Boskovic, M. S. Maucec, and V. Zumer, "High-dimensional real-parameter optimization using self-adaptive differential evolution algorithm with population size reduction," in *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 2032–2039.

[22] H. Wang, S. Rahnamayan, and Z. Wu, "Parallel differential evolution with self-adapting control parameters and generalized opposition-based learning for solving high-dimensional optimization problems," *Journal of Parallel and Distributed Computing*, vol. 73, no. 1, pp. 62–73, 2013.

[23] J. Brest, A. Zamuda, I. Fister, and M. S. Maučec, "Large scale global optimization using self-adaptive differential evolution algorithm," in *IEEE Congress on Evolutionary Computation*. IEEE, 2010, pp. 1–8.

[24] M. Weber, F. Neri, and V. Tirronen, "Shuffle or update parallel differential evolution for large-scale optimization," *Soft Computing*, vol. 15, no. 11, pp. 2089–2107, 2011.

[25] A. Zamuda, J. Brest, B. Boskovic, and V. Zumer, "Large scale global optimization using differential evolution with self-adaptation and cooperative co-evolution," in *IEEE Congress on Evolutionary Computation*. IEEE, 2008, pp. 3718–3725.

[26] A. Qin and P. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Proc. of IEEE Congress on Evolutionary Computation*, vol. 2, 2005, pp. 1785–1791.

[27] Z. Yang, K. Tang, and X. Yao, "Self-adaptive differential evolution with neighborhood search," in *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 1110–1116.

[28] ——, "Scalability of generalized adaptive differential evolution for large-scale continuous optimization," *Soft Computing*, vol. 15, no. 11, pp. 2141–2155, 2011.

[29] T. Takahama and S. Sakai, "Large scale optimization by differential evolution with landscape modality detection and a diversity archive," in *IEEE Congress on Evolutionary Computation*. IEEE, 2012, pp. 1–8.

[30] H. Wang, S. Rahnamayan, and Z. Wu, "Adaptive differential evolution with variable population size for solving high-dimensional problems," in *IEEE Congress on Evolutionary Computation*. IEEE, 2011, pp. 2626–2632.

[31] R. Tanabe and A. S. Fukunaga, "Improving the search performance of shade using linear population size reduction," in *2014 IEEE congress on evolutionary computation (CEC)*. IEEE, 2014, pp. 1658–1665.

[32] J. Lampinen, I. Zelinka *et al.*, "On stagnation of the differential evolution algorithm," in *Proceedings of MENDEL*, 2000, pp. 76–83.

[33] C. Segura, C. A. C. Coello, and A. G. Hernández-Díaz, "Improving the vector generation strategy of differential evolution for large-scale optimization," *Information Sciences*, vol. 323, pp. 106–129, 2015.

[34] Y.-F. Ge, W.-J. Yu, and J. Zhang, "Diversity-based multi-population differential evolution for large-scale optimization," in *Genetic and Evolutionary Computation Conference*. ACM, 2016, pp. 31–32.

[35] Y.-F. Ge, W.-J. Yu, Y. Lin, Y.-J. Gong, Z.-H. Zhan, W.-N. Chen, and J. Zhang, "Distributed Differential Evolution Based on Adaptive Mergence and Split for Large-Scale Optimization," *IEEE Transactions on Cybernetics*, vol. 48, no. 7, pp. 2166–2180, Jul. 2018.

[36] K. E. Parsopoulos, "Cooperative micro-differential evolution for high-dimensional problems," in *Proc. of Genetic and Evolutionary Computation Conference*. ACM, 2009, pp. 531–538.

[37] E. H. Houssein, A. G. Gad, K. Hussain, and P. N. Suganthan, "Major Advances in Particle Swarm Optimization: Theory, Analysis, and Application," *Swarm and Evolutionary Computation*, vol. 63, p. 100868, Jun. 2021.

[38] J. Vesterstrom and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No. 04TH8753)*, vol. 2. IEEE, 2004, pp. 1980–1987.

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

[40] N. Lynn and P. N. Suganthan, "Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation," *Swarm and Evolutionary Computation*, vol. 24, pp. 11–24, Oct. 2015.

[41] R. Cheng and Y. Jin, "A competitive swarm optimizer for large scale optimization," *IEEE Transactions on Cybernetics*, vol. 45, no. 2, pp. 191–204, 2015.

[42] Y. Tian, X. Zheng, X. Zhang, and Y. Jin, "Efficient Large-Scale Multiobjective Optimization Based on a Competitive Swarm Optimizer," *IEEE Transactions on Cybernetics*, vol. 50, no. 8, pp. 3696–3708, Aug. 2020.

[43] E. Naderi, H. Narimani, M. Fathi, and M. R. Narimani, "A novel fuzzy adaptive configuration of particle swarm optimization to solve large-scale optimal reactive power dispatch," *Applied Soft Computing*, 2017.

[44] R.-L. Tang, Z. Wu, and Y.-J. Fang, "Adaptive multi-context cooperatively coevolving particle swarm optimization for large-scale problems," *Soft Computing*, pp. 1–20, 2016.

[45] M. Pluhacek, R. Senkerik, and I. Zelinka, "Investigation on the performance of a new multiple choice strategy for PSO algorithm in the task of large scale optimization problems," in *IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 2007–2011.

[46] N. Lynn, M. Z. Ali, and P. N. Suganthan, "Population topologies for particle swarm optimization and differential evolution," *Swarm and Evolutionary Computation*, vol. 39, pp. 24–35, Apr. 2018.

[47] J. Fan, J. Wang, and M. Han, "Cooperative coevolution for large-scale optimization based on kernel fuzzy clustering and variable trust region methods," *IEEE Transactions on Fuzzy Systems*, vol. 22, no. 4, pp. 829–839, 2014.

[48] Q. Zhang, H. Cheng, Z. Ye, and Z. Wang, "A competitive swarm optimizer integrated with cauchy and gaussian mutation for large scale optimization," in *Chinese Control Conference*. IEEE, 2017, pp. 9829–9834.

[49] Q. Yang, W.-N. Chen, T. Gu, H. Zhang, H. Yuan, S. Kwong, and J. Zhang, "A Distributed Swarm Optimizer With Adaptive Communication for Large-Scale Optimization," *IEEE Transactions on Cybernetics*, vol. 50, no. 7, pp. 3393–3408, Jul. 2020.

[50] Z.-J. Wang, Z.-H. Zhan, W.-J. Yu, Y. Lin, J. Zhang, T.-L. Gu, and J. Zhang, "Dynamic Group Learning Distributed Particle Swarm Optimization for Large-Scale Optimization and Its Application in Cloud Workflow Scheduling," *IEEE Transactions on Cybernetics*, vol. 50, no. 6, pp. 2715–2729, Jun. 2020.

[51] Z.-J. Wang, Z.-H. Zhan, S. Kwong, H. Jin, and J. Zhang, "Adaptive Granularity Learning Distributed Particle Swarm Optimization for Large-Scale Optimization," *IEEE Transactions on Cybernetics*, vol. 51, no. 3, pp. 1175–1188, Mar. 2021.

[52] M. Arasomwan and A. O. Adewumi, "An adaptive velocity particle swarm optimization for high-dimensional function optimization," in *2013 IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 2352–2359.

[53] R. Cheng and Y. Jin, "A social learning particle swarm optimization algorithm for scalable optimization," *Information Sciences*, vol. 291, pp. 43–60, 2015.

[54] J.-R. Jian, Z.-G. Chen, Z.-H. Zhan, and J. Zhang, "Region Encoding Helps Evolutionary Computation Evolve Faster: A New Solution Encoding Scheme in Particle Swarm for Large-Scale Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 4, pp. 779–793, Aug. 2021.

[55] Q. Yang, W.-N. Chen, J. Da Deng, Y. Li, T. Gu, and J. Zhang, "A level-based learning swarm optimizer for large scale optimization," *IEEE*

*Transactions on Evolutionary Computation*, 2017.

[56] H. Huang, L. Lv, S. Ye, and Z. Hao, "Particle swarm optimization with convergence speed controller for large-scale numerical optimization," *Soft Computing*, vol. 23, no. 12, pp. 4421–4437, Jun. 2019.

[57] S. Cheng, H. Zhan, H. Yao, H. Fan, and Y. Liu, "Large-scale many-objective particle swarm optimizer with fast convergence based on Alpha-stable mutation and Logistic function," *Applied Soft Computing*, vol. 99, p. 106947, Feb. 2021.

[58] D. Li, W. Guo, A. Lerch, Y. Li, L. Wang, and Q. Wu, "An adaptive particle swarm optimizer with decoupled exploration and exploitation for large scale optimization," *Swarm and Evolutionary Computation*, vol. 60, p. 100789, Feb. 2021.

[59] Y. Xue, T. Tang, W. Pang, and A. X. Liu, "Self-adaptive parameter and strategy based particle swarm optimization for large-scale feature selection problems with multiple classifiers," *Applied Soft Computing*, vol. 88, p. 106031, Mar. 2020.

[60] S.-T. Hsieh, T.-Y. Sun, C.-C. Liu, and S.-J. Tsai, "Solving large scale global optimization using improved particle swarm optimizer," in *IEEE Congress on Evolutionary Computation*. IEEE, 2008, pp. 1777–1784.

[61] M. A. M. de Oca, D. Aydın, and T. Stützle, "An incremental particle swarm for large-scale continuous optimization problems: an example of tuning-in-the-loop (re)design of optimization algorithms," *Soft Computing*, vol. 15, no. 11, pp. 2233–2255, 2011.

[62] M. A. M. De Oca, K. Van den Enden, and T. Stützle, "Incremental particle swarm-guided local search for continuous optimization," in *International Workshop on Hybrid Metaheuristics*. Springer, 2008, pp. 72–86.

[63] J. Garcí-Nieto and E. Alba, "Restart particle swarm optimization with velocity modulation: a scalability test," *Soft Computing*, vol. 15, no. 11, pp. 2221–2232, 2011.

[64] S. Cheng, Y. Shi, and Q. Qin, "Dynamical exploitation space reduction in particle swarm optimization for solving large scale problems," in *IEEE Congress on Evolutionary Computation*. IEEE, 2012, pp. 1–8.

[65] J. Zhou, W. Fang, X. Wu, J. Sun, and S. Cheng, "An opposition-based learning competitive particle swarm optimizer," in *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2016, pp. 515–521.

[66] T. Hendtlass, "Particle swarm optimisation and high dimensional problem spaces," in *IEEE Congress on Evolutionary Computation*. IEEE, 2009, pp. 1988–1994.

[67] T. Korenaga, T. Hatanaka, and K. Uosaki, "Performance improvement of particle swarm optimization for high-dimensional function optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2007, pp. 3288–3293.

[68] W. Chu, X. Gao, and S. Sorooshian, "A new evolutionary search strategy for global optimization of high-dimensional problems," *Information Sciences*, vol. 181, no. 22, pp. 4909–4927, 2011.

[69] ——, "Fortify particle swam optimizer (pso) with principal components analysis: A case study in improving bound-handling for optimizing high-dimensional and complex problems," in *IEEE Congress on Evolutionary Computation*. IEEE, 2011, pp. 1644–1648.

[70] Q. Zhang, W. Liu, X. Meng, B. Yang, and A. V. Vasilakos, "Vector co-evolving particle swarm optimization algorithm," *Information Sciences*, vol. 394, pp. 273–298, 2017.

[71] S.-Z. Zhao, J. J. Liang, P. N. Suganthan, and M. F. Tasgetiren, "Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2008, pp. 3845–3852.

[72] R. G. Regis, "Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 326–347, 2014.

[73] P. Yang, K. Tang, and X. Yao, "Turning high-dimensional optimization into computationally expensive optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 143–156, 2018.

[74] S. Mahdavi, M. E. Shiri, and S. Rahnamayan, "Cooperative co-evolution with a new decomposition method for large-scale optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2014, pp. 1285–1292.

[75] E. Li, H. Wang, and F. Ye, "Two-level multi-surrogate assisted optimization method for high dimensional nonlinear problems," *Applied Soft Computing*, vol. 46, pp. 26–36, 2016.

[76] M. N. Omidvar, X. Li, Y. Mei, and X. Yao, "Cooperative co-evolution with differential grouping for large scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 3, pp. 378–393, 2014.

[77] B. Pang, Z. Ren, Y. Liang, and A. Chen, "Enhancing cooperative coevolution for large scale optimization by adaptively constructing surrogate models," *arXiv preprint*, 2018.

[78] B. Werth, E. Pitzer, and M. Affenzeller, "Enabling high-dimensional surrogate-assisted optimization by using sliding windows," in *Genetic and Evolutionary Computation Conference Companion*. ACM, 2017, pp. 1630–1637.

[79] C. Wang and J.-H. Gao, "A differential evolution algorithm with cooperative coevolutionary selection operation for high-dimensional optimization," *Optimization Letters*, vol. 8, no. 2, pp. 477–492, 2014.

[80] I. De Falco, A. D. Cioppa, and G. A. Trunfio, "Large scale optimization of computationally expensive functions: an approach based on parallel cooperative coevolution and fitness metamodeling," in *Genetic and Evolutionary Computation Conference Companion*. ACM, 2017, pp. 1788–1795.

[81] Z. Ren, B. Pang, Y. Liang, A. Chen, and Y. Zhang, "Surrogate model assisted cooperative coevolution for large scale optimization," *arXiv preprint*, 2018.

[82] C. Sun, Y. Jin, J. Ding, and J. Zeng, "Fitness estimation strategy assisted competitive swarm optimizer for high dimensional expensive problems," in *Genetic and Evolutionary Computation Conference*. ACM, 2016, pp. 1277–1278.

[83] C. Sun, Y. Jin, R. Cheng, J. Ding, and J. Zeng, "Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 4, pp. 644–660, 2017.

[84] Z. Yang, B. Sendhoff, K. Tang, and X. Yao, "Target shape design optimization by evolving b-splines with cooperative coevolution," *Applied Soft Computing*, vol. 48, pp. 672–682, 2016.

[85] A. Tiwari, R. Roy, G. Jared, and O. Munaux, "Interaction and multi-objective optimisation," in *Genetic and Evolutionary Computation Conference*. Morgan Kaufmann Publishers Inc., 2001, pp. 671–678.

[86] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "A framework for large-scale multi-objective optimization based on problem transformation," *IEEE Transactions on Evolutionary Computation*, 2017.

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

[88] W. Dong, Y. Wang, and M. Zhou, "A latent space-based estimation of distribution algorithm for large-scale global optimization," *Soft Computing*, vol. 23, no. 13, pp. 4593–4615, Jul. 2019.

[89] R. Liu, R. Ren, J. Liu, and J. Liu, "A clustering and dimensionality reduction based evolutionary algorithm for large-scale multi-objective problems," *Applied Soft Computing*, vol. 89, p. 106120, Apr. 2020.

[90] G. Wu, W. Pedrycz, P. N. Suganthan, and R. Mallipeddi, "A variable reduction strategy for evolutionary algorithms handling equality constraints," *Applied Soft Computing*, vol. 37, pp. 774–786, Dec. 2015.

[91] G. Wu, W. Pedrycz, P. N. Suganthan, and H. Li, "Using variable reduction strategy to accelerate evolutionary optimization," *Applied Soft Computing*, vol. 61, pp. 283–293, Dec. 2017.

[92] M. L. Sanyang and A. Kabán, "REMEDA: Random embedding EDA for optimising functions with intrinsic dimension," in *Parallel Problem Solving from Nature*. Springer, 2016, pp. 859–868.

[93] S. Wang, J. Liu, and Y. Jin, "Surrogate-Assisted Robust Optimization of Large-Scale Networks Based on Graph Embedding," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 4, pp. 735–749, Aug. 2020.

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

[95] B. Kazimipour, X. Li, and A. K. Qin, "A review of population initialization techniques for evolutionary algorithms," in *IEEE Congress on Evolutionary Computation*. IEEE, 2014, pp. 2585–2592.

[96] S. Mahdavi, S. Rahnamayan, and K. Deb, "Center-based initialization of cooperative co-evolutionary algorithm for large-scale optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2016, pp. 3557–3565.

[97] B. Kazimipour, X. Li, and A. K. Qin, "Initialization methods for large scale global optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 2750–2757.

[98] ——, "Effects of population initialization on differential evolution for large scale optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2014, pp. 2404–2411.

[99] B. Kazimipour, X. Li, and A. Qin, "Why advanced population initialization techniques perform poorly in high dimension?" in *Simulated Evolution and Learning*, ser. Lecture Notes in Computer Science. Springer International Publishing, 2014, vol. 8886, pp. 479–490.

[100] E. Segredo, B. Paechter, C. Segura, and C. I. González-Vila, "On the comparison of initialisation strategies in differential evolution for large scale optimisation," *Optimization Letters*, vol. 12, no. 1, pp. 221–234,

2018.

[101] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is 'nearest neighbor' meaningful?" in *International Conference on Database Theory*. Springer, 1999, pp. 217–235.

[102] R. J. Durrant and A. Kabán, "When is 'nearest neighbour' meaningful: A converse theorem and implications," *Journal of Complexity*, vol. 25, no. 4, pp. 385–397, 2009.

[103] E. Cantú-Paz and D. E. Goldberg, "On the scalability of parallel genetic algorithms," *Evolutionary computation*, vol. 7, no. 4, pp. 429–449, 1999.

[104] M. Munetomo, N. Murao, and K. Akama, "Empirical investigations on parallelized linkage identification," in *Parallel Problem Solving from Nature*. Springer, 2004, pp. 322–331.

[105] A. Mendiburu, J. A. Lozano, and J. Miguel-Alonso, "Parallel implementation of edas based on probabilistic graphical models," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 4, pp. 406–423, 2005.

[106] S. Iturriaga and S. Nesmachnow, "Solving very large optimization problems (up to one billion variables) with a parallel evolutionary algorithm in cpu and gpu," in *Seventh International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*. IEEE, 2012, pp. 267–272.

[107] Q. Duan, L. Sun, and Y. Shi, "Spark clustering computing platform based parallel particle swarm optimizers for computationally expensive global optimization," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2018, pp. 424–435.

[108] B. Cao, S. Fan, J. Zhao, P. Yang, K. Muhammad, and M. Tanveer, "Quantum-enhanced multiobjective large-scale optimization via parallelism," *Swarm and Evolutionary Computation*, vol. 57, p. 100697, Sep. 2020.

[109] M. Lastra, D. Molina, and J. M. Benítez, "A high performance memetic algorithm for extremely high-dimensional problems," *Information Sciences*, vol. 293, pp. 35–58, 2015.

[110] D. Molina, M. Lozano, and F. Herrera, "MA-SW-Chains: Memetic algorithm based on local search chains for large scale continuous global optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2010, pp. 3153–3160.

[111] A. Cano and C. García-Martínez, "100 million dimensions large-scale global optimization using distributed gpu computing," in *IEEE Congress on Evolutionary Computation*. IEEE, 2016, pp. 3566–3573.

[112] A. Cano, C. García-Martínez, and S. Ventura, "Extremely high-dimensional optimization with mapreduce: scaling functions and algorithm," *Information Sciences*, vol. 415, pp. 110–127, 2017.

[113] Y. Su, K. Zhou, X. Zhang, R. Cheng, and C. Zheng, "A parallel multiobjective evolutionary algorithm for community detection in large-scale complex networks," *Information Sciences*, vol. 576, pp. 374–392, Oct. 2021.

[114] B. Cao, J. Zhao, Z. Lv, and X. Liu, "A distributed parallel cooperative coevolutionary multiobjective evolutionary algorithm for large-scale optimization," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 2030–2038, 2017.

[115] M. N. Omidvar, M. Yang, Y. Mei, X. Li, and X. Yao, "Dg2: A faster and more accurate differential grouping for large-scale black-box optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 6, pp. 929–942, 2017.

[116] P. Yang, K. Tang, and X. Yao, "A parallel divide-and-conquer based evolutionary algorithm for large-scale optimization," *arXiv preprint arXiv:1812.02500*, 2018.

[117] G. Roy, H. Lee, J. L. Welch, Y. Zhao, V. Pandey, and D. Thurston, "A distributed pool architecture for genetic algorithms," in *2009 IEEE Congress on Evolutionary Computation*. IEEE, 2009, pp. 1177–1184.

[118] Y.-H. Jia, W.-N. Chen, T. Gu, H. Zhang, H. Yuan, S. Kwong, and J. Zhang, "Distributed cooperative co-evolution with adaptive computing resource allocation for large scale optimization," *IEEE Transactions on Evolutionary Computation*, 2018.

[119] J. R. Martins and A. B. Lambe, "Multidisciplinary design optimization: a survey of architectures," *AIAA journal*, vol. 51, no. 9, pp. 2049–2075, 2013.

[120] A. Yassine and D. Braha, "Complex concurrent engineering and the design structure matrix method," *Concurrent Engineering*, vol. 11, no. 3, pp. 165–176, 2003.

[121] K. Li, M. N. Omidvar, K. Deb, and X. Yao, "Variable interaction in multi-objective optimization problems," in *Parallel Problem Solving from Nature*. Springer International Publishing, 2016, pp. 399–409.

[122] P. Xu, W. Luo, X. Lin, J. Zhang, Y. Qiao, and X. Wang, "Constraint-Objective Cooperative Coevolution for Large-scale Constrained Optimization," *ACM Transactions on Evolutionary Learning and Optimiza-

[123] tion*, vol. 1, no. 3, pp. 12:1–12:26, Aug. 2021.

[123] L. Sun, S. Yoshida, X. Cheng, and Y. Liang, "A cooperative particle swarm optimizer with statistical variable interdependence learning," *Information Sciences*, vol. 186, no. 1, pp. 20–39, 2012.

[124] A. Song, W.-N. Chen, P.-T. Luo, Y.-J. Gong, and J. Zhang, "Overlapped cooperative co-evolution for large scale optimization," in *IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2017, pp. 3689–3694.

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

[126] Y. Sun, X. Li, A. Ernst, and M. N. Omidvar, "Decomposition for large-scale optimization problems with overlapping components," in *IEEE congress on evolutionary computation*, 2019.

[127] L. Li, W. Fang, Y. Mei, and Q. Wang, "Cooperative coevolution for large-scale global optimization based on fuzzy decomposition," *Soft Computing*, vol. 25, no. 5, pp. 3593–3608, Mar. 2021.

[128] D. Thierens, "The linkage tree genetic algorithm," in *Parallel Problem Solving from Nature*. Springer, 2010, pp. 264–273.

[129] P. A. Bosman and D. Thierens, "More concise and robust linkage learning by filtering and combining linkage hierarchies," in *Genetic and Evolutionary Computation Conference*. ACM, 2013, pp. 359–366.

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

[131] T.-L. Yu, K. Sastry, and D. E. Goldberg, "Linkage learning, overlapping building blocks, and systematic strategy for scalable recombination," in *Genetic and evolutionary computation conference*. ACM, 2005, pp. 1217–1224.

[132] Y. Sun, M. Kirley, and S. K. Halgamuge, "A recursive decomposition method for large scale continuous optimization," *IEEE Transactions on Evolutionary Computation*, 2017.

[133] Y.-H. Jia, Y. Mei, and M. Zhang, "Contribution-Based Cooperative Co-Evolution for Nonseparable Large-Scale Problems With Overlapping Subcomponents," *IEEE Transactions on Cybernetics*, pp. 1–14, 2020.

[134] A. Song, W.-N. Chen, Y.-J. Gong, X. Luo, and J. Zhang, "A Divide-and-Conquer Evolutionary Algorithm for Large-Scale Virtual Network Embedding," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 3, pp. 566–580, Jun. 2020.

[135] S. Strasser, J. Sheppard, N. Fortier, and R. Goodman, "Factored evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 281–293, 2017.

[136] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz, "BOA: The bayesian optimization algorithm," in *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation-Volume 1*. Morgan Kaufmann Publishers Inc., 1999, pp. 525–532.

[137] M. Tsuji, M. Munetomo, and K. Akama, "Linkage identification by fitness difference clustering," *Evolutionary Computation*, vol. 14, no. 4, pp. 383–409, 2006.

[138] L. R. Emmendorfer and A. T. R. Pozo, "Effective linkage learning using low-order statistics and clustering," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 6, pp. 1233–1246, 2009.

[139] C.-Y. Chuang and Y.-p. Chen, "Sensibility of linkage information and effectiveness of estimated distributions," *Evolutionary Computation*, vol. 18, no. 4, pp. 547–579, 2010.

[140] M. N. Omidvar, X. Li, and X. Yao, "Smart use of computational resources based on contribution for cooperative co-evolutionary algorithms," in *Genetic and evolutionary computation conference*. ACM, 2011, pp. 1115–1122.

[141] S. Liu, Q. Lin, Y. Tian, and K. C. Tan, "A Variable Importance-Based Differential Evolution for Large-Scale Multiobjective Optimization," *IEEE Transactions on Cybernetics*, pp. 1–15, 2021.

[142] X. Shen, Y. Guo, and A. Li, "Cooperative coevolution with an improved resource allocation for large-scale multi-objective software project scheduling," *Applied Soft Computing*, vol. 88, p. 106059, Mar. 2020.

[143] D. Yazdani, M. N. Omidvar, J. Branke, T. T. Nguyen, and X. Yao, "Scaling up dynamic optimization problems: A divide-and-conquer approach," *IEEE Transactions on Evolutionary Computation*, 2019.

[144] M. N. Omidvar, B. Kazimipour, X. Li, and X. Yao, "Cbcc3âĂŤa contribution-based cooperative co-evolutionary algorithm with improved exploration/exploitation balance," in *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2016, pp. 3541–3548.

[145] M. Yang, M. N. Omidvar, C. Li, X. Li, Z. Cai, B. Kazimipour, and X. Yao, "Efficient resource allocation in cooperative co-evolution for large-scale global optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 4, pp. 493–505, 2017.

[146] M. Yang, A. Zhou, C. Li, J. Guan, and X. Yan, "CCFR2: A more efficient cooperative co-evolutionary framework for large-scale global optimization," *Information Sciences*, vol. 512, pp. 64–79, Feb. 2020.

[147] Z. Ren, Y. Liang, A. Zhang, Y. Yang, Z. Feng, and L. Wang, "Boosting cooperative coevolution for large scale optimization with a fine-grained computation resource allocation strategy," *arXiv preprint*, 2018.

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

[149] M. A. Meselhi, S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Contribution Based Co-Evolutionary Algorithm for Large-Scale Optimization Problems," *IEEE Access*, vol. 8, pp. 203 369–203 381, 2020.

[150] S. Mahdavi, S. Rahnamayan, and M. E. Shiri, "Multilevel framework for large-scale global optimization," *Soft Computing*, pp. 1–30, 2016.

[151] ——, "Incremental cooperative coevolution for large-scale global optimization," *Soft Computing*, pp. 1–20, 2016.

[152] ——, "Cooperative co-evolution with sensitivity analysis-based budget assignment strategy for large-scale global optimization," *Applied Intelligence*, vol. 47, no. 3, pp. 888–913, 2017.

[153] X. Peng and Y. Wu, "Large-scale cooperative co-evolution with bi-objective selection based imbalanced multi-modal optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2017, pp. 1527–1532.

[154] B. Kazimipour, M. N. Omidvar, A. K. Qin, X. Li, and X. Yao, "Bandit-based cooperative coevolution for tackling contribution imbalance in large-scale optimization problems," *Applied Soft Computing*, vol. 76, pp. 265–281, 2019.

[155] D. Rainville, M. Sebag, C. Gagné, M. Schoenauer, D. Laurendeau *et al.*, "Sustainable cooperative coevolution with a multi-armed bandit," in *Proceedings of the 15th annual conference on Genetic and evolutionary computation*. ACM, 2013, pp. 1517–1524.

[156] C. A. C. Coello and G. B. Lamont, *Applications of multi-objective evolutionary algorithms*. World Scientific, 2004, vol. 1.

[157] B. Li, J. Li, K. Tang, and X. Yao, "Many-objective evolutionary algorithms: A survey," *ACM Computing Surveys (CSUR)*, vol. 48, no. 1, p. 13, 2015.

[158] J. J. Durillo, A. J. Nebro, C. A. C. Coello, F. Luna, and E. Alba, "A comparative study of the effect of parameter scalability in multi-objective metaheuristics," in *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*. IEEE, 2008, pp. 1893–1900.

[159] J. J. Durillo, A. J. Nebro, C. A. C. Coello, J. García-Nieto, F. Luna, and E. Alba, "A study of multiobjective metaheuristics when solving parameter scalable problems," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 4, pp. 618–635, 2010.

[160] S. Liu, Q. Lin, K.-C. Wong, Q. Li, and K. C. Tan, "Evolutionary Large-Scale Multiobjective Optimization: Benchmarks and Algorithms," *IEEE Transactions on Evolutionary Computation*, pp. 1–1, 2021.

[161] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731, 2007.

[162] X. Ma, F. Liu, Y. Qi, X. Wang, L. Li, L. Jiao, M. Yin, and M. Gong, "A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 2, pp. 275–298, 2016.

[163] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization," *IEEE Transactions on Evolutionary Computation*, 2016.

[164] A. E. I. Brownlee, J. A. Wright, M. He, T. Lee, and P. McMenemy, "A novel encoding for separable large-scale multi-objective problems and its application to the optimisation of housing stock improvements," *Applied Soft Computing*, vol. 96, p. 106650, Nov. 2020.

[165] L. M. Antonio and C. A. C. Coello, "Use of cooperative coevolution for solving large scale multiobjective optimization problems," in *IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 2758–2765.

[166] ——, "Decomposition-based approach for solving large scale multi-objective problems," in *Parallel Problem Solving from Nature*. Springer, 2016, pp. 525–534.

[167] A. Song, Q. Yang, W.-N. Chen, and J. Zhang, "A random-based dynamic grouping strategy for large scale multi-objective optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2016, pp. 468–475.

[168] B. Cao, J. Zhao, Y. Gu, Y. Ling, and X. Ma, "Applying graph-based differential grouping for multiobjective large-scale optimization," *Swarm and Evolutionary Computation*, vol. 53, p. 100626, Mar. 2020.

[169] Q. Wang, L. Zhang, S. Wei, and B. Li, "Tensor decomposition-based alternate sub-population evolution for large-scale many-objective optimization," *Information Sciences*, vol. 569, pp. 376–399, Aug. 2021.

[170] L. Ma, M. Huang, S. Yang, R. Wang, and X. Wang, "An Adaptive Localized Decision Variable Analysis Approach to Large-Scale Multiobjective and Many-Objective Optimization," *IEEE Transactions on Cybernetics*, pp. 1–13, 2021.

[171] H. Chen, R. Cheng, J. Wen, H. Li, and J. Weng, "Solving large-scale many-objective optimization problems by covariance matrix adaptation evolution strategy with scalable small subpopulations," *Information Sciences*, vol. 509, pp. 457–469, Jan. 2020.

[172] A. Tiwari and R. Roy, "Variable dependence interaction and multi-objective optimisation," in *Genetic and Evolutionary Computation Conference*. Morgan Kaufmann Publishers Inc., 2002, pp. 602–609.

[173] C. He, L. Li, Y. Tian, X. Zhang, R. Cheng, Y. Jin, and X. Yao, "Accelerating Large-Scale Multiobjective Optimization via Problem Reformulation," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 6, pp. 949–961, Dec. 2019.

[174] H. Zille and S. Mostaghim, "Comparison study of large-scale optimisation techniques on the lsmop benchmark functions," in *IEEE Symposium Series on Computational Intelligence*. IEEE, 2017, pp. 1–8.

[175] H. Zille, H. Ishibuchi, S. Mostaghim, and Y. Nojima, "Weighted optimization framework for large-scale multi-objective optimization," in *Genetic and Evolutionary Computation Conference*. ACM, 2016, pp. 83–84.

[176] R. Liu, J. Liu, Y. Li, and J. Liu, "A random dynamic grouping based weight optimization framework for large-scale multi-objective optimization problems," *Swarm and Evolutionary Computation*, vol. 55, p. 100684, Jun. 2020.

[177] Y. Tian, C. Lu, X. Zhang, K. C. Tan, and Y. Jin, "Solving Large-Scale Multiobjective Optimization Problems With Sparse Optimal Solutions via Unsupervised Neural Networks," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3115–3128, Jun. 2021.

[178] Y. Tian, C. Lu, X. Zhang, F. Cheng, and Y. Jin, "A Pattern Mining-Based Evolutionary Algorithm for Large-Scale Sparse Multiobjective Optimization Problems," *IEEE Transactions on Cybernetics*, pp. 1–14, 2020.

[179] Y. Tian, X. Zhang, C. Wang, and Y. Jin, "An Evolutionary Algorithm for Large-Scale Sparse Multiobjective Optimization Problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 2, pp. 380–393, Apr. 2020.

[180] Y. Tian, R. Liu, X. Zhang, H. Ma, K. C. Tan, and Y. Jin, "A Multipopulation Evolutionary Algorithm for Solving Large-Scale Multimodal Multiobjective Optimization Problems," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 3, pp. 405–418, Jun. 2021.

[181] J.-H. Yi, L.-N. Xing, G.-G. Wang, J. Dong, A. V. Vasilakos, A. H. Alavi, and L. Wang, "Behavior of crossover operators in NSGA-III for large-scale optimization problems," *Information Sciences*, vol. 509, pp. 470–487, Jan. 2020.

[182] K. Deb and H. Jain, "An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems With Box Constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, Aug. 2014.

[183] Y. Yin, Y. Zhao, H. Li, and X. Dong, "Multi-objective evolutionary clustering for large-scale dynamic community detection," *Information Sciences*, vol. 549, pp. 269–287, Mar. 2021.

[184] L. P. Cota, F. G. Guimarães, R. G. Ribeiro, I. R. Meneghini, F. B. de Oliveira, M. J. F. Souza, and P. Siarry, "An adaptive multi-objective algorithm based on decomposition and large neighborhood search for a green machine scheduling problem," *Swarm and Evolutionary Computation*, vol. 51, p. 100601, Dec. 2019.

[185] S. Qin, C. Sun, Y. Jin, Y. Tan, and J. Fieldsend, "Large-Scale Evolutionary Multiobjective Optimization Assisted by Directed Sampling," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 4, pp. 724–738, Aug. 2021.

[186] Y. Zhang, G.-G. Wang, K. Li, W.-C. Yeh, M. Jian, and J. Dong, "Enhancing MOEA/D with information feedback models for large-scale many-objective optimization," *Information Sciences*, vol. 522, pp. 1–16, Jun. 2020.

[187] W. Hong, K. Tang, A. Zhou, H. Ishibuchi, and X. Yao, "A Scalable Indicator-Based Evolutionary Algorithm for Large-Scale Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 3, pp. 525–537, Jun. 2019.

[188] J. Xiao, T. Zhang, J. Du, and X. Zhang, "An Evolutionary Multi-objective Route Grouping-Based Heuristic Algorithm for Large-Scale

Capacitated Vehicle Routing Problems," *IEEE Transactions on Cybernetics*, vol. 51, no. 8, pp. 4173–4186, Aug. 2021.

[189] X. Zhang, K. Zhou, H. Pan, L. Zhang, X. Zeng, and Y. Jin, "A Network Reduction-Based Multiobjective Evolutionary Algorithm for Community Detection in Large-Scale Complex Networks," *IEEE Transactions on Cybernetics*, vol. 50, no. 2, pp. 703–716, Feb. 2020.

[190] J. Zhang, L. Xing, G. Peng, F. Yao, and C. Chen, "A large-scale multiobjective satellite data transmission scheduling algorithm based on SVM+NSGA-II," *Swarm and Evolutionary Computation*, vol. 50, p. 100560, Nov. 2019.

[191] S.-Y. Ho, L.-S. Shu, and J.-H. Chen, "Intelligent evolutionary algorithms for large parameter optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 6, pp. 522–541, 2004.

[192] M. Gong, H. Li, E. Luo, J. Liu, and J. Liu, "A multiobjective cooperative coevolutionary algorithm for hyperspectral sparse unmixing," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 234–248, 2017.

[193] Y. Wang, B. Li, and T. Weise, "Estimation of distribution and differential evolution cooperation for large scale economic load dispatch optimization of power systems," *Information Sciences*, vol. 180, no. 12, pp. 2405–2420, 2010.

[194] R. Shang, K. Dai, L. Jiao, and R. Stolkin, "Improved memetic algorithm based on route distance grouping for multiobjective large scale capacitated arc routing problems," *IEEE Transactions on Cybernetics*, vol. 46, no. 4, pp. 1000–1013, 2016.

[195] A. Kumar, G. Wu, M. Z. Ali, R. Mallipeddi, P. N. Suganthan, and S. Das, "A test-suite of non-convex constrained optimization problems from the real-world and some baseline results," *Swarm and Evolutionary Computation*, vol. 56, p. 100693, Aug. 2020.

[196] E. Mezura-Montes and C. A. C. Coello, "Constraint-handling in nature-inspired numerical optimization: past, present and future," *Swarm and Evolutionary Computation*, vol. 1, no. 4, pp. 173–194, 2011.

[197] C. Peng and Q. Hui, "Comparison of differential grouping and random grouping methods on $\epsilon$CCPSO for large-scale constrained optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2016, pp. 2057–2063.

[198] E. Sayed, D. Essam, R. Sarker, and S. Elsayed, "Decomposition-based evolutionary algorithm for large scale constrained problems," *Information Sciences*, vol. 316, pp. 457–486, 2015.

[199] A. E. Aguilar-Justo and E. Mezura-Montes, "Towards an improvement of variable interaction identification for large-scale constrained problems," in *IEEE Congress on Evolutionary Computation*. IEEE, 2016, pp. 4167–4174.

[200] A. E. Aguilar-Justo, E. Mezura-Montes, S. M. Elsayed, and R. A. Sarker, "Decomposition of large-scale constrained problems using a genetic-based search," in *2016 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*. IEEE, 2016, pp. 1–6.

[201] A. E. Aguilar-Justo and E. Mezura-Montes, "A local cooperative approach to solve large-scale constrained optimization problems," *Swarm and Evolutionary Computation*, vol. 51, p. 100577, Dec. 2019.

[202] J. Blanchard, C. Beauthier, and T. Carletti, "A cooperative coevolutionary algorithm for solving large-scale constrained problems with interaction detection," in *Genetic and Evolutionary Computation Conference*. ACM, 2017, pp. 697–704.

[203] M. N. Omidvar, *IDG: A faster and more accurate differential grouping algorithm*. School of Computer Science, University of Birmingham, 2015.

[204] C. He, R. Cheng, Y. Tian, X. Zhang, K. C. Tan, and Y. Jin, "Paired Offspring Generation for Constrained Large-Scale Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 3, pp. 448–462, Jun. 2021.

[205] W. Chu, X. Gao, and S. Sorooshian, "Handling boundary constraints for particle swarm optimization in high-dimensional search space," *Information Sciences*, vol. 181, no. 20, pp. 4569–4581, 2011.

[206] O. A. Elhara, "Stochastic black-box optimization and benchmarking in large dimensions," Ph.D. dissertation, Université Paris-Saclay, 2017.

[207] K. Tang, X. Yao, P. N. Suganthan, C. MacNish, Y.-P. Chen, C.-M. Chen, and Z. Yang, "Benchmark functions for the cecâĂŹ2008 special session and competition on large scale global optimization," Nature Inspired Computation and Applications Laboratory, USTC, China, Tech. Rep., 2007.

[208] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC'2010 special session and competition on large-scale global optimization," Nature Inspired Computation and Applications Laboratory, USTC, China, Tech. Rep., 2009, http://nical.ustc.edu.cn/cec10ss.php.

[209] X. Li, K. Tang, M. N. Omidvar, Z. Yang, and K. Qin, "Benchmark functions for the CEC'2013 special session and competition on large-scale global optimization," RMIT University, Melbourne, Australia, Tech. Rep., 2013, http://goanna.cs.rmit.edu.au/ xiaodong/cec13-lsgo.

[210] M. Jamil and X.-S. Yang, "A literature survey of benchmark functions for global optimization problems," *arXiv preprint arXiv:1308.4008*, 2013.

[211] P. Suganthan, N. Hansen, J. Liang, K. Deb, Y. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Nanyang Technological University, Singapore, Tech. Rep., 2005, http://www.ntu.edu.sg/home/EPNSugan.

[212] M. Lozano, D. Molina, and F. Herrera, "Editorial scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems," *Soft Computing*, vol. 15, no. 11, pp. 2085–2087, 2011.

[213] N. Hansen, S. Finck, R. Ros, and A. Auger, "Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions," Tech. Rep., 2009.

[214] Y. Sun, M. Kirley, and S. K. Halgamuge, "Quantifying variable interactions in continuous optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 249–264, 2017.

[215] E. Sayed, D. Essam, and R. Sarker, "Dependency identification technique for large scale optimization problems," in *IEEE Congress on Evolutionary Computation*. IEEE, 2012, pp. 1–8.

[216] S. K. Goh, K. C. Tan, A. Al-Mamun, and H. A. Abbass, "Evolutionary big optimization (BigOpt) of signals," in *IEEE Congress on Evolutionary Computation*. IEEE, 2015, pp. 3332–3339.

[217] R. Cheng, Y. Jin, M. Olhofer *et al.*, "Test problems for large-scale multiobjective and many-objective optimization," *IEEE Transactions on Cybernetics*, vol. 47, no. 12, pp. 4108–4121, 2017.

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

[219] E. D. Dolan, J. J. More, and T. S. Munson, "Benchmarking optimization software with COPS 3.0," Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, Illinois 60439, Tech. Rep., 2004.

[220] W. Luo, B. Yang, C. Bu, and X. Lin, "A hybrid particle swarm optimization for high-dimensional dynamic optimization," in *Asia-Pacific Conference on Simulated Evolution and Learning*. Springer, 2017, pp. 981–993.

[221] L.-Y. Tseng and C. Chen, "Multiple trajectory search for large scale global optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2008, pp. 3052–3059.

[222] Y. Wang and B. Li, "A restart univariate estimation of distribution algorithm: sampling under mixed gaussian and lévy probability distribution," in *IEEE Congress on Evolutionary Computation*. IEEE, 2008, pp. 3917–3924.

[223] ——, "Two-stage based ensemble optimization for large-scale global optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2010, pp. 1–8.

[224] A. LaTorre, S. Muelas, and J.-M. Peña, "A mos-based dynamic memetic differential evolution algorithm for continuous optimization: a scalability test," *Soft Computing*, vol. 15, no. 11, pp. 2187–2199, 2011.

[225] ——, "Large scale global optimization: Experimental results with MOS-based hybrid algorithms," in *IEEE Congress on Evolutionary Computation*. IEEE, 2013, pp. 2742–2749.

[226] D. Molina and F. Herrera, "Iterative hybridization of de with local search for the CEC'2015 special session on large scale global optimization," in *IEEE Congress on Evolutionary Computation*. IEEE, 2015, pp. 1974–1978.

[227] D. Molina, A. LaTorre, and F. Herrera, "SHADE with Iterative Local Search for Large-Scale Global Optimization," in *2018 IEEE Congress on Evolutionary Computation (CEC)*, Jul. 2018, pp. 1–8.

[228] A. A. Hadi, A. W. Mohamed, and K. M. Jambi, "Lshade-spa memetic framework for solving large-scale optimization problems," *Complex & Intelligent Systems*, vol. 5, no. 1, pp. 25–40, 2019.

[229] A. Bolufé-Röhler, S. Chen, and D. Tamayo-Vera, "An analysis of minimum population search on large scale global optimization," in *2019 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2019, pp. 1228–1235.

[230] Y. Wang, J. Huang, W. S. Dong, J. C. Yan, C. H. Tian, M. Li, and W. T. Mo, "Two-stage based ensemble optimization framework for large-scale global optimization," *European Journal of Operational Research*, vol. 228, no. 2, pp. 308–320, 2013.

[231] D. Molina, A. LaTorre, and F. Herrera, "An insight into bio-inspired and evolutionary algorithms for global optimization: review, analysis, and

lessons learnt over a decade of competitions," *Cognitive Computation*, vol. 10, no. 4, pp. 517–544, 2018.

[232] D. M. Cabrera, "Evolutionary algorithms for large-scale global optimisation: a snapshot, trends and challenges," *Progress in Artificial Intelligence*, vol. 5, no. 2, pp. 85–89, 2016.

[233] Y. Mei, M. N. Omidvar, X. Li, and X. Yao, "A competitive divide-and-conquer algorithm for unconstrained large-scale black-box optimization," *ACM Transactions on Mathematical Software*, vol. 42, no. 2, p. 13, 2016.

[234] K. Sörensen, "MetaheuristicsâĂŤthe metaphor exposed," *International Transactions in Operational Research*, vol. 22, no. 1, pp. 3–18, 2015.

[235] K. O. Stanley, J. Clune, J. Lehman, and R. Miikkulainen, "Designing neural networks through neuroevolution," *Nature Machine Intelligence*, vol. 1, no. 1, pp. 24–35, 2019.

[236] F. P. Such, V. Madhavan, E. Conti, J. Lehman, K. O. Stanley, and J. Clune, "Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning," *arXiv preprint arXiv:1712.06567*, 2018.

[237] S. Fujino, T. Hatanaka, N. Mori, and K. Matsumoto, "The evolutionary deep learning based on deep convolutional neural network for the anime storyboard recognition," in *International Symposium on Distributed Computing and Artificial Intelligence*. Springer, 2017, pp. 278–285.

[238] G. Morse and K. O. Stanley, "Simple evolutionary optimization can rival stochastic gradient descent in neural networks," in *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference*. ACM, 2016, pp. 477–484.

[239] E. Real, S. Moore, A. Selle, S. Saxena, Y. L. Suematsu, J. Tan, Q. V. Le, and A. Kurakin, "Large-scale evolution of image classifiers," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 2902–2911.

[240] A. Yaman, D. C. Mocanu, G. Iacca, G. Fletcher, and M. Pechenizkiy, "Limited evaluation cooperative co-evolutionary differential evolution for large-scale neuroevolution," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2018, pp. 569–576.

[241] X. Cui, W. Zhang, Z. Tüske, and M. Picheny, "Evolutionary stochastic gradient descent for optimization of deep neural networks," in *Advances in neural information processing systems*, 2018, pp. 6048–6058.

[242] M. Jaderberg, V. Dalibard, S. Osindero, W. M. Czarnecki, J. Donahue, A. Razavi, O. Vinyals, T. Green, I. Dunning, K. Simonyan *et al.*, "Population based training of neural networks," *arXiv preprint arXiv:1711.09846*, 2017.

[243] T. Elsken, J. H. Metzen, and F. Hutter, "Neural Architecture Search: A Survey," *Journal of Machine Learning Research*, p. 21, 2019.

[244] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, "Evolution Strategies as a Scalable Alternative to Reinforcement Learning," *arXiv:1703.03864 [cs, stat]*, Sep. 2017.

[245] Y. Bengio, A. Lodi, and A. Prouvost, "Machine learning for combinatorial optimization: a methodological tour d'horizon," *European Journal of Operational Research*, vol. 290, no. 2, pp. 405–421, 2021.

[246] A. Lodi and G. Zarpellon, "On learning and branching: a survey," *Top*, vol. 25, no. 2, pp. 207–236, 2017.

[247] Y. Sun, X. Li, and A. Ernst, "Using statistical measures and machine learning for graph reduction to solve maximum weight clique problems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 5, pp. 1746–1760, 2019.

[248] Y. Sun, A. Ernst, X. Li, and J. Weiner, "Generalization of machine learning for problem reduction: a case study on travelling salesman problems," *OR Spectrum*, vol. 43, no. 3, pp. 607–633, 2021.

[249] D. Bertsimas and J. N. Tsitsiklis, *Introduction to linear optimization*. Athena Scientific Belmont, MA, 1997.

[250] M. Kruber, M. E. Lübbecke, and A. Parmentier, "Learning When to Use a Decomposition," in *Integration of AI and OR Techniques in Constraint Programming*, ser. Lecture Notes in Computer Science, D. Salvagnin and M. Lombardi, Eds. Cham: Springer International Publishing, 2017, pp. 202–210.

[251] C. Blum and G. R. Raidl, *Hybrid Metaheuristics: Powerful Tools for Optimization*. Springer, 2016.

[252] A. Kenny, X. Li, A. T. Ernst, and D. Thiruvady, "Towards solving large-scale precedence constrained production scheduling problems in mining," in *Genetic and Evolutionary Computation Conference*. ACM, 2017, pp. 1137–1144.

[253] A. Kenny, X. Li, and A. T. Ernst, "A merge search algorithm and its application to the constrained pit problem in mining," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2018, pp. 316–323.

[254] A. Kenny, X. Li, A. T. Ernst, and Y. Sun, "An improved merge search algorithm for the constrained pit problem in open-pit mining," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2019, pp. 294–302.

[255] S. Elsayed, R. Sarker, D. Essam, and C. A. Coello Coello, "Evolutionary approach for large-Scale mine scheduling," *Information Sciences*, vol. 523, pp. 77–90, Jun. 2020.

[256] C. He, R. Cheng, C. Zhang, Y. Tian, Q. Chen, and X. Yao, "Evolutionary Large-Scale Multiobjective Optimization for Ratio Error Estimation of Voltage Transformers," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 5, pp. 868–881, Oct. 2020.

**Mohammad Nabi Omidvar** (M09, SM'20) received the first bachelors (First Class Hons.) degree in computer science, the second bachelors degree in applied mathematics, and the Ph.D. degree in computer science from RMIT University, Melbourne, VIC, Australia, in 2010, 2014, and 2016, respectively. He is currently an Assistant Professor of Artificial Intelligence in Financial Services affiliated with Leeds University Business School and School of Computing, University of Leeds, Leeds, U.K and the chair of IEEE Computational Intelligence Taskforce on Large-Scale Global Optimization. Prior to that he was a Research Fellow with the School of Computer Science, University of Birmingham, Birmingham, U.K. Dr. Omidvar is the winner of IEEE CEC Large-Scale Global Optimization Competition in 2019, and a recipient of the IEEE Transaction on Evolutionary Computation Outstanding Paper Award for his research on large-scale global optimization in 2017, the Australian Postgraduate Award in 2010, and the Best Computer Science Honours Thesis Award from the School of Computer Science and IT, RMIT University. His current research interests include large-scale global optimization, high-dimensional machine learning, and AI for financial services.

**Xiaodong Li** (Fellow, IEEE) 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 Computing Technologies, RMIT University, Melbourne, VIC, Australia. His research interests include machine learning, evolutionary computation, neural networks, data analytics, multiobjective optimization, multimodal optimization, and swarm intelligence. Prof. Li is a recipient of 2013 ACM SIGEVO Impact Award and 2017 IEEE CIS "IEEE Transactions on Evolutionary Computation Outstanding Paper Award." He serves as an Associate Editor for the IEEE Transactions on Evolutionary Computation and Swarm Intelligence (Springer). He is a former Vice-Chair of the IEEE Task Force on Multimodal Optimization and a Former Chair of the IEEE CIS Task Force on Large Scale Global Optimization.
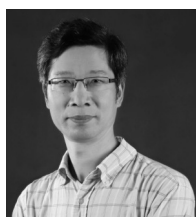
**Xin Yao** obtained his Ph.D. in 1990 from the University of Science and Technology of China (USTC), MSc in 1985 from North China Institute of Computing Technologies and BSc in 1982 from USTC. He is a Chair Professor of Computer Science at the Southern University of Science and Technology, Shenzhen, China, and a part-time Professor of Computer Science at the University of Birmingham, UK. He is an IEEE Fellow and was a Distinguished Lecturer of the IEEE Computational Intelligence Society (CIS). He served as the President (2014-15) of IEEE CIS and the Editor-in-Chief (2003-08) of IEEE Transactions on Evolutionary Computation. His major research interests include evolutionary computation, ensemble learning, and their applications to software engineering. His work won the 2001 IEEE Donald G. Fink Prize Paper Award; 2010, 2016 and 2017 IEEE Transactions on Evolutionary Computation Outstanding Paper Awards; 2011 IEEE Transactions on Neural Networks Outstanding Paper Award; and many other best paper awards at conferences. He received a prestigious Royal Society Wolfson Research Merit Award in 2012, the IEEE CIS Evolutionary Computation Pioneer Award in 2013 and the 2020 IEEE Frank Rosenblatt Award.