This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2744324, IEEE Transactions on Evolutionary Computation

Population Evolvability: Dynamic Fitness Landscape Analysis for Population-based Metaheuristic Algorithms

Mang Wang, Bin Li, *Member, IEEE,* Guofu Zhang, *Member, IEEE,* and Xin Yao, *Fellow, IEEE*

***Abstract*—Fitness Landscape Analysis (FLA) is an important approach for studying how hard problems are for metaheuristic algorithms to solve. Static FLA focuses on extracting the properties of a problem and does not consider any information about the optimization algorithms; thus, it is not adequate for indicating whether a particular algorithm is suitable for solving a problem. By contrast, dynamic FLA considers the behaviour of algorithms in combination with the properties of an optimization problem to determine the effectiveness of a given algorithm for solving that problem. However, previous dynamic FLA approaches are all individually based and lack statistical significance. In this paper, the concept of population evolvability is presented, as an extension of dynamic FLA, to quantify the effectiveness of population-based metaheuristic algorithms for solving a given problem. Specifically, two measures of population evolvability are defined that describe the probability that a population will obtain improved solutions to a problem and its ability to do so. Then, a combined measure is derived from these two measures to represent the overall population evolvability. Subsequently, the significance and validity of the proposed measures are investigated through analytical and experimental studies. Finally, the utility of the proposed measures is illustrated in an application of algorithm selection for black-box optimization problems. High accuracy in selecting the best algorithm is observed in a statistical analysis, with a low computational cost in terms of fitness evaluations.**

***Index Terms*—Population-based Metaheuristic Algorithms, Population Evolvability, Dynamic Fitness Landscape Analysis, Algorithm Selection Task**

I. Introduction

THE concept of fitness landscapes originated from theoretical biology more than eighty years ago [1]. In the evolutionary computation community, the notion of fitness landscapes is an attempt to draw an analogy with real landscapes to gain a better understanding of how and where algorithms operate [2]. Fitness Landscape Analysis (FLA) refers to a collection of data-driven techniques for extracting descriptive or numerical measures related to the properties of fitness landscapes. Hence, the purpose of FLA is to obtain knowledge about a given optimization problem. The most common application of FLA is to gain insights into the characteristics of optimization problems that are critical for algorithm performance [3].

Several descriptive characteristics have been proposed in earlier studies of FLA. For example, the modality, i.e., the distribution of local optima and plateaus, is an intuitive indicator of the hardness of a problem, although the intuitive notion that a multimodal landscape is more difficult than a unimodal landscape is not always true [4], [5]. Basins of attraction are defined with respect to local optima, whose shapes and distributions have been found to be correlated with the computational cost [6]. Fitness barriers are derived from the two concepts discussed above and are represented as a tree-like structure. Two parameters, i.e., the depth and difficulty determined from the barrier tree, play an important role in analysing the convergence speed of simulated annealing [7].

Smoothness, ruggedness and neutrality are the most frequently used measures in studies of FLA. The information provided by these three measures is useful, as in a smooth landscape, an optimization algorithm can easily generate new, fitter observations from its offspring, whereas this is not the case in neutral landscapes, where the directional information contained in the fitness information is zero, or in rugged landscapes, where the fitness values fluctuate rapidly [8]. Several numerical measures have been proposed to describe the degree of ruggedness, such as the autocorrelation function and the correlation length [9], [10]. Neutral walks, which can proceed only within a neutral area, are used to explore neutral areas and record the accumulated distance from the starting point; thus, the maximum accumulated distance can be utilised as an indicator of neutrality [11]. Other measures, such as the number of neighbouring candidate solutions with nearly equal fitness and the number of distinct fitness values surrounding a neutral area, have also been proposed to represent the degree of neutrality [12].

Other methods related to FLA have also been widely adopted. The fitness distance correlation (FDC), i.e., the correlation between fitness values and their distances from the global optimum, is designed to provide a global view of the fitness landscape [13]. In addition, exploratory landscape analysis (ELA) approaches, proposed by Mersmann *et al.* [14], [15] and extended by Muñoz *et al.* [16], attempt to extract high-level and low-level features of fitness landscapes to enable the classification of optimization problems into several categories.

Manuscript received October 14, 2016; revised January 9, 2017 and May 24, 2017; accepted August 14, 2017.

M. Wang and B. Li (Corresponding Author) are with the School of Information Science and Technology, University of Science and Technology of China, Hefei, Anhui, 230027 China (e-mail: mangwang@mail.ustc.edu.cn; binli@ustc.edu.cn). G. Zhang is with the School of Computer and Information, Hefei University of Technology, Hefei, Anhui, 230009 China (e-mail: zgf@hfut.edu.cn). X. Yao is with the Shenzhen Key Lab of Computational Intelligence, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China, and also with CERCIA, School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K. (e-mail: xiny@sustc.edu.cn).

The ultimate goal of FLA is to discover informative relations between properties of fitness landscapes and algorithm performance. Some studies have made considerable progress in this direction. For instance, empirical hardness models were built by Leyton-Brown *et al.* [17], in the context of combinatorial optimization, to predict the run time of an algorithm for a given problem instance. For numerical optimization, a machine learning model was employed by Bischl *et al.* [18] based on the low-level features extracted via the ELA method to predict well-performing algorithms within a given portfolio. Similarly, a meta-learning regression model was used by Muñoz *et al.* [19] based on five existing landscape features to select the proper parameter configurations for CMA-ES [20] on their test problem instances.

The aforementioned techniques all belong to the realm of static FLA, in which measures are extracted from optimization problems without considering any information about the optimization algorithms to be applied. However, the relation between these static measures and algorithm performance is not reliable [21]. Some problems that are judged as plausibly complicated based on their static FLA measures may be easy to solve for certain algorithms, whereas some seemingly easy problems may turn out to actually be hard. Therefore, some researchers have borrowed the notion of evolvability from evolutionary biology to define measures that consider the behaviour of algorithms in combination with the properties of optimization problems, in an approach known as dynamic FLA.

In the evolutionary computation community, Altenberg [22] defines evolvability as "the ability of a population to produce variants fitter than any yet existing". It has been demonstrated by Turney [23] that increasing evolvability is regarded as a desirable large-scale trend for evolutionary heuristics; thus, the measures of dynamic FLA are expected to be correlated with algorithm performance. Fitness evolvability portraits were defined by Smith *et al.* [24] to describe how these measures are related to both ruggedness and neutrality. The negative slope coefficient [25] and the accumulated escape probability [26] are two measures that indicate the hardness of a problem by measuring the correlation between consecutive solutions produced by the mutation operation of an algorithm. Notably, the existing measures of evolvability are all based on randomly sampled single candidate solutions in the feasible solution space; thus, they are not suitable for analysing the collective behaviour of population-based metaheuristic algorithms on a given problem. Additionally, randomness is an inherent part of the random sampling process, whereas each of these measures provides only a single value as an indicator and thus lacks statistical significance for quantifying this randomness. Therefore, the current dynamic FLA approaches are not applicable in many real optimization scenarios.

Given the above concerns, the question arises of whether it is possible to define measures that are directly correlated with the performance of population-based metaheuristic algorithms on one problem and also have statistical significance. In this paper, we attempt to address this question by presenting the concept of population evolvability, as an extension of dynamic FLA, to measure the effectiveness of population-based metaheuristic algorithms for solving a given optimization problem. Compared with previous studies, the main contributions of this paper are as follows:

1) Two measures of population evolvability are formulated, for the first time, to describe the probability that an algorithm will obtain improved solutions to a problem and its ability to do so. Then, a combined measure is derived to represent the overall population evolvability.
2) The significance and validity of the proposed measures of population evolvability are investigated through analytical and experimental studies.
3) The algorithm selection task for numerical black-box optimization problems is considered as a typical application of the proposed measures, and high accuracy in selecting the best algorithm is observed in a statistical analysis, with a low computational cost in terms of fitness evaluations (FEs).

The remainder of this paper is organized as follows. Section II describes the related work and motivation. Section III is devoted to the explanation of the proposed measures derived from the concept of population evolvability. Section IV presents experimental studies conducted to illustrate the relation between the generational measures and the convergence curve. Section V discusses the algorithm selection task for numerical black-box optimization problems as a typical application of the proposed measures. Further discussions regarding the robustness and computational cost of the proposed approach for the algorithm selection task are presented in Section VI. Finally, Section VII offers conclusions.

II. Related Work and Motivation

Over the past few decades, FLA has been demonstrated to be a useful tool for analysing the hardness of an optimizaton problem by extracting its features. Among the available FLA methods, dynamic FLA approaches are most closely related to the measures proposed in this article, as they are all based on the extraction of features from algorithm-problem pairs. In this section, a brief review of the state of the art in dynamic FLA is first presented. Then, the limitations of current methods are summarized. Finally, the motivation for this work is introduced.

A. Fitness Evolvability Portraits

Fitness evolvability portraits were presented by Smith *et al.* [24] as a means of quantifying the evolvability of a candidate solution $\mathbf{x}$. The authors also described how these portraits relate to both ruggedness and neutrality. Specifically, the fundamental component of a fitness evolvability portrait, E_a, is defined as follows:

$$E_a = \frac{|N^*(\mathbf{x})|}{|N(\mathbf{x})|} \tag{1}$$

where $N(\mathbf{x})$ denotes a set of neighbours generated by $\mathbf{x}$; $N^*(\mathbf{x}) = \{\mathbf{c} \mid \mathbf{c} \in N(\mathbf{x}), f(\mathbf{c}) \geq f(\mathbf{x})\}$ denotes a set of non-deleterious neighbours, in which $f(\cdot)$ denotes a fitness function. E_a represents the number of non-deleterious neighbours divided by the total number of neighbours; thus, it indicates the probability of non-deleterious mutations.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2744324, IEEE Transactions on Evolutionary Computation

B. Fitness Clouds and the Negative Slope Coefficient

A fitness cloud, as proposed by Philippe *et al.* [27], is a scatter plot that visualizes parent fitness vs. offspring fitness. In this approach, the authors assumed that every solution in the search space had only one unique neighbour, which was determined using a search heuristic. The fitness of the neighbour of a solution $\mathbf{x}$ was called the "bordering fitness", $b(\mathbf{x})$. Then, a fitness cloud could be obtained by constructing a scatter plot from pairs of fitness ($f(\mathbf{x})$) and bordering fitness ($b(\mathbf{x})$) values. The sets of the minimum, mean and maximum bordering fitness for every solution were defined as FC_{min}, FC_{mean} and FC_{max} respectively, and the intersections of these sets with the diagonal line (i.e., $b = f$) formed the boundaries between regions of solutions that were strictly advantageous, averagely advantageous, averagely deleterious and strictly deleterious.

Based on the fitness cloud concept, the authors extracted a singular measure, called the negative slope coefficient (nsc), as an indicator of problem hardness [25]. To calculate nsc, a set of fitness (f) and bordering fitness (b) pairs was divided into consecutive groups according to their fitness values. Then, the mean fitness and the mean bordering fitness within each group, i.e., $(\overline{f_i}, \overline{b_i})$, were calculated. Finally, nsc was calculated as the sum of all negative slopes, as follows:

$$nsc = \sum_i min(K_i, 0) \tag{2}$$

where $K_i = (\overline{b_{i+1}} - \overline{b_i})/(\overline{f_{i+1}} - \overline{f_i})$ is the slope between two adjacent groups. This measure, nsc, can be used as an indicator of problem hardness: a problem is easy if $nsc = 0$, whereas the problem is difficult if $nsc < 0$, and the magnitude of nsc quantifies the difficulty. As mentioned by the authors, nsc lacks a normalization term, so the results are not comparable across different problems.

C. Fitness-Probability Clouds and the Accumulated Escape Probability

The concept of a fitness-probability cloud (fpc) was proposed by Lu *et al.* [26] for studying evolvability based on the correlation between the fitnesses of individuals and their escape probabilities. Regarding the escape probability (or escape rate), it was first introduced by Merz [28] to quantify the number of steps required to escape from a local optimum. Thus, the escape probability, $p(f_i)$, is formulated as follows:

$$p(f_i) = \frac{1}{S_i} \tag{3}$$

where S_i represents the average number of steps required to find an improving move starting from a solution whose fitness value is f_i. The definition of the escape probability can be extended to a set of fitness values to formulate the average escape probability, p_i:

$$p_i = \frac{\sum_{f_j \in C_i} p(f_j)}{|C_i|} \tag{4}$$

where $C_i = \{f_j \mid f_j \geq f_i\}$. Then the fitness-probability cloud fpc is defined as $fpc = \{(f_0, p_0), (f_1, p_1), \ldots, (f_L, p_L)\}$.

A numerical measure called the accumulated escape probability (aep) was derived from the fitness-probability cloud concept and formulated as follows:

$$aep = \frac{\sum_{f_i \in F} p_i}{|F|} \tag{5}$$

where $F = \{f_0, f_1, \ldots, f_L \mid f_0 < f_1 < \cdots < f_L\}$. aep indicates the hardness of a problem as follows: the larger the value of aep is, the higher the evolvability is, and the easier the problem should be.

D. Limitations of Previous Works and Our Motivation

The previous proposed measures of evolvability are all based on randomly sampled single candidate solutions; thus, they are not suitable for directly analysing the collective behaviour of population-based metaheuristic algorithms on a given problem. Moreover, each of them provides only a single value as an indicator and thus lacks statistical significance for quantifying the randomness which is an inherent part of the random sampling process. Therefore, in this article, we wish to develop measures that quantify the evolvability related to sampled populations while also considering the stochastic nature of the sampling process.

The motivation for this article is as follows: in nature, evolution is correlated with the collective behaviour of species, which is controlled by natural selection [29]. Furthermore, the evolutionary rate of a species may change under different circumstances. For instance, as described by Ayala [30], in small or isolated environments such as small islands or remote valleys, evolution can be effectively accelerated. Therefore, we argue that the population evolvability is closely related to two aspects, i.e., the probability of evolution and the ability to evolve. The former represents whether the population can evolve to a fitter state. The latter expresses the degree of that evolution if it occurs.

III. Measures of Population Evolvability

A. Preliminaries

1) Success Probability: For the convergence analysis of Evolution Strategies (ESs), the success probability (denoted by p_s hereafter) was introduced by Rechenberg [31] as part of the famous 1/5th rule for investigating the performance of the (1+1) ES [32]. p_s is defined as the probability of an offspring replacing its parent. Since there are only one parent and one offspring in the (1+1) ES, the offspring will replace its parent if its fitness value is better than its parent's. Notably, the idea of p_s is similar to that of the metric E_a and that of the escape probability $p(f_i)$ defined in (1) and (3), respectively, as they are all based on an estimation of the probability of successful mutations for single individuals. Specifically, p_s is measured over a sliding window consisting of a fixed number of the most recent iterations, whereas E_a and $p(f_i)$ are estimated based on a fixed number of neighbours generated by an individual.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2744324, IEEE Transactions on Evolutionary Computation

2) Convergence Velocity and Convergence Reliability: Convergence velocity and convergence reliability analyses have achieved considerable progress in theoretical research on evolutionary algorithms. According to [33], the convergence velocity is a local measure of improvement from one iteration to the next, where the improvement can be measured in terms of either the expected quality gain, φ, or the expected change in the distance to the global optimum, d. Concretely, φ and d are formulated as follows:

$$\begin{aligned} \varphi &= E[f(\mathbf{x}_{t+1}) - f(\mathbf{x}_t)], \\ d &= E[\|\mathbf{x}^* - \mathbf{x}_t\| - \|\mathbf{x}^* - \mathbf{x}_{t+1}\|] \end{aligned} \tag{6}$$

where $\mathbf{x}^*$ denotes the global optimum and $\mathbf{x}_t$ represents the best or a representative individual of the population at generation t. A convergence reliability analysis focuses on the global convergence of an algorithm within an unlimited run time, i.e.,

$$\lim_{t\to\infty} p(\mathbf{x}^* \in P(t)) = 1 \tag{7}$$

where $P(t)$ denotes the population at generation t and $p(\cdot)$ is the probability of an event.

Regarding the above two preliminaries, the success probability estimates the probability of successful mutations, whereas the convergence velocity and convergence reliability measure the convergence ability of evolutionary algorithms. These concepts are closely related to the two aspects of evolution that we wish to assess, i.e., the probability of and ability for evolution, respectively, and thus provide inspiration for formulating our measures of population evolvability.

B. Evolutionary Probability of a Population

The concept of the probability of successful mutations of an individual is expected to be extended to that of the probability of the successful evolution of a population. As we know, metaheuristic algorithms typically include various operators and strategies, such as recombination, mutation, selection and self-adaption; consequently, the neighbours (one-step child populations) are the products of comprehensive effects of multiple operations on their common parent, not merely the results of a mutation operation, as considered in previous studies. Therefore, we need a new measure to reflect the evolutionary probability of a population.

Definition 1. *Given a sampled population P_i and a set of its generated neighbours (one-step child populations) $N(P_i) = \{ P_{i1}, \ldots, P_{iM} \mid M \geq 1 \}$, define the Evolutionary Probability of a Population (epp) to be:*

$$epp(P_i) = \frac{|N^+(P_i)|}{|N(P_i)|} \tag{8}$$

where $N^+(P_i)$ represents the set of evolved neighbours, i.e., $N^+(P_i) = \{ P_{ij} \mid P_{ij} \in N(P_i),\ f^b(P_{ij}) < f^b(P_i) \}$ for a minimization problem, in which $j = 1, \ldots, |N^+(P_i)|$ represents the index of evolved neighbours, and $f^b(P)$ denotes the best fitness value found in a population P.

Although we focus only on the best individual in the population and its generated neighbours, Definition 1 truly concerns the collective behaviour of the entire population. Moreover, through this definition, we can investigate the population evolvability for a large variety of population-based metaheuristic algorithms based on a unified concept, i.e., the probability that a population will evolve from its *current state*. Notably, the meaning of the current state may differ for different algorithms. To be more concrete, for algorithms that have no self-adaptive parameters, such as the real-coded GA [34], the current state of a population comprises the genotype and phenotype of the population and the global and fixed parameters of the algorithm, whereas for algorithms that do have self-adaptive parameters, such as CMA-ES [20], the current values of these adaptive parameters should also be included in the current state. Note that the range of epp is $[0, 1]$, as it is an estimation of a probability.

C. Evolutionary Ability of a Population

Evolutionary abilities of populations in different states are diverse. For metaheuristic algorithms, it is far from sufficient to use the convergence velocity φ defined in (6) as a measure of evolutionary ability. The reasons are as follows: first, the convergence velocity is a local metric and consequently cannot reflect the global behaviour of the algorithm, yet this kind of global behaviour is vital to metaheuristic algorithms for exploring the solution space, and second, the convergence velocity mainly focuses on the absolute changes between fitness values, which will become increasingly smaller while approaching convergence, making it unable to represent the evolution ability near convergence. Therefore, we need a new measure to reflect the evolutionary ability of an algorithm throughout the entire optimization process.

Definition 2. *Given P_i and $N^+(P_i)$ that are the same as in Definition 1, define the Evolutionary Ability of a Population (eap) to be:*

$$eap(P_i) = \begin{cases} \dfrac{\sum_{P_{ij} \in N^+(P_i)} \frac{|f^b(P_i) - f^b(P_{ij})|/NP}{\sigma(\boldsymbol{f}(P_i))}}{|N^+(P_i)|} & ,\ |N^+(P_i)| \geq 1 \\ 0 & ,\ |N^+(P_i)| = 0 \end{cases} \tag{9}$$

where NP represents the size of the population P_i, and $\sigma(\boldsymbol{f}(P_i))$ represents the standard deviation of the fitness values of the population P_i.

The meaning of Definition 2 can be interpreted as the average evolutionary ability of a population calculated over its evolved neighbours, where the evolutionary ability is expressed as follows:

$$\frac{|f^b(P_i) - f^b(P_{ij})|/NP}{\sigma(\boldsymbol{f}(P_i))} \tag{10}$$

In (10), the numerator is the absolute improvement in the best fitness value between the population P_i and its evolved neighbour P_{ij}. The reason for dividing by *NP* is to obtain equitable improvements for populations of different sizes. As for the denominator, it is the standard deviation of the fitness values of P_i, which is a distance-based diversity measure with respect to the population phenotype, as proposed in [35]. For this equation, on the one hand, if the denominator remains

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2744324, IEEE Transactions on Evolutionary Computation

constant, the larger the numerator is, the greater the evolutionary ability will be; that is to say, if the diversity measures of different populations are equal, then the population that can achieve the highest absolute fitness improvement is the one that has the greatest evolutionary ability. On the other hand, if the numerator remains constant, the smaller the denominator is, the greater the evolutionary ability will be; that is to say, if the absolute fitness improvements of different populations are the same, then the population with the lowest diversity is considered to have the greatest evolutionary ability. In both situations, this equation is able to provide a reasonable explanation for the evolutionary ability. Notably, if $\sigma(\boldsymbol{f}(P_i))$ is equal or infinitely approaching to zero and the numerator is nonzero, then eap could be infinitely large. To avoid this situation, we set $\sigma(\boldsymbol{f}(P_i)) = \varepsilon$ when $\sigma(\boldsymbol{f}(P_i)) < \varepsilon$ (where ε is a small constant). In this paper, ε is set to $1e{-}8$.

Considering the optimization process of metaheuristic algorithms, large fitness improvements and high diversity are the features of earlier stages, whereas relatively smaller fitness improvements and lower diversity are the ordinary states of later stages. We expect that eap can serve as a measure of evolutionary ability throughout the entire process of optimization, although we do not exclude the possibility that measures of other forms might achieve the same purpose. Finally, it should be noted that the range of eap is $[0, +\infty)$.

D. Evolvability of a Population

From an algorithmic perspective, a problem is hard either if it is hard for evolution to occur (i.e., the evolutionary probability is low) or if the degree of evolution is small (i.e., the evolutionary ability is low). Thus, we wish to consider both phenomena simultaneously to formulate a combined measure of problem hardness for population-based metaheuristic algorithms. Certainly, there are various possible ways to combine these two aspects; in particular, we adopt a concise and meaningful one, namely, multiplying them together, to reflect the overall population evolvability.

Definition 3. *Given epp and eap, define the Evolvability of a Population (evp) to be:*

$$evp(P_i) = epp(P_i) \times eap(P_i) = \begin{cases} \dfrac{\sum_{P_{ij} \in N^+(P_i)} \frac{|f^b(P_i) - f^b(P_{ij})|/NP}{\sigma(\boldsymbol{f}(P_i))}}{|N(P_i)|} & , \ |N^+(P_i)| \geq 1 \\ 0 & , \ |N^+(P_i)| = 0 \end{cases} \tag{11}$$

Definition 3 represents the average evolutionary ability over the entire set of generated neighbours, where the evolution abilities of unevolved neighbours are considered to be 0. In addition, because epp is defined as the probability of evolution and eap is a measure of evolutionary ability, evp actually represents the expectation of the evolutionary ability for a population P_i. In particular, when P_i exists in a given state, $evp(P_i)$ reflects the expected evolutionary ability from that state, which is correlated with the hardness of the problem for the algorithm in that state. Clearly, the range of evp is $[0, +\infty)$.

IV. Experimental Studies

This section presents experimental studies conducted to demonstrate the validity of the proposed measures of population evolvability. First, thirteen benchmark functions and a test problem adapted from a real application are introduced. Then, five representative population-based algorithms are described. Finally, the results are presented and analysed in detail.

A. Problem Set

Thirteen traditional numerical problems among the benchmark functions of CEC 2013 [36] are considered first. All of these functions are minimization problems, whose search space lies in $[-100, 100]^D$ and whose global optimum is randomly generated within $[-80, 80]^D$. Note that the fitness value of the global optimum is separately shifted to positive, zero and negative values to evaluate the proposed measures under various circumstances. In addition, we treat all selected test functions as black-box optimization problems, each with 30 dimensions, i.e., $D = 30$. Table I lists the details of the selected test functions, where the first five functions are unimodal and the last eight functions are multimodal.

A real-world optimization problem for the design of spread-spectrum radar polyphase codes [37] (denoted by f_{radar} hereafter) is also used as a test problem. It can be modelled as a min-max nonlinear and nonconvex optimization problem with continuous variables and numerous local optima, which has been proven to be NP-hard [38]. This problem can be formulated as follows:

$$\begin{aligned} &\text{global}\min f(\mathbf{x}) = \max\{\, \varphi_1(\mathbf{x}), \ldots, \varphi_{2m}(\mathbf{x}) \,\} \\ &\mathbf{x} = \{\, (x_1, \ldots, x_D) \in \mathbb{R}^D \mid 0 \leq x_j \leq 2\pi,\ j = 1, \ldots, D \,\} \\ &\varphi_{2i-1}(\mathbf{x}) = \sum_{j=1}^{D} \cos\left(\sum_{k=|2i-j-1|+1}^{j} x_k \right), i = 1, \ldots, D \\ &\varphi_{2i}(\mathbf{x}) = 0.5 + \sum_{j=i+1}^{D} \cos\left(\sum_{k=|2i-j|+1}^{j} x_k \right), i = 1, \ldots, D-1 \\ &\varphi_{m+i}(\mathbf{x}) = -\varphi_i(\mathbf{x}), i = 1, \ldots, m;\ m = 2D-1 \end{aligned} \tag{12}$$

The objective of this optimal design problem is to minimize the largest module among the samples of the auto-correlation function, which is correlated with the complex envelope of the compressed radar pulse at the output of an optimal receiver. The surface plot for this problem is shown in Fig. 1, which illustrates its complexity in $D = 2$. In this article, the dimensionality of this problem is set to 20 (i.e., $D = 20$), which makes it even more complicated to solve.

B. Algorithm Settings

Five representative and state-of-the-art population-based metaheuristic algorithms [39], including evolutionary and swarm-based optimization techniques for numerical function optimization, i.e., the real-coded GA [34], CMA-ES [20], CoDE [40], SPSO2011 [41] and ABC [42], are studied. Among them, the real-coded GA was implemented by modifying the "ga" function of the MATLAB Global Optimization

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2744324, IEEE Transactions on Evolutionary Computation

TABLE I
THIRTEEN REVISED TEST FUNCTIONS FROM CEC 2013

Function	Definition	Transformation	f_i^*
Sphere Function	$f_1(\mathbf{x}) = \sum_{i=1}^{D} z_i^2 + f_1^*$	$\mathbf{z} = \mathbf{x} - \mathbf{o}$	-600
High Conditioned Elliptic Function	$f_2(\mathbf{x}) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} z_i^2 + f_2^*$	$\mathbf{z} = T_{osz}(\mathbf{x} - \mathbf{o})$	-500
Bent Cigar Function	$f_3(\mathbf{x}) = z_1^2 + 10^6 \sum_{i=2}^{D} z_i^2 + f_3^*$	$\mathbf{z} = T_{asy}^{0.5}(\mathbf{x} - \mathbf{o})$	-400
Discus Function	$f_4(\mathbf{x}) = 10^6 z_1^2 + \sum_{i=2}^{D} z_i^2 + f_4^*$	$\mathbf{z} = T_{osz}(\mathbf{x} - \mathbf{o})$	-300
Different Powers Function	$f_5(\mathbf{x}) = \sqrt{\sum_{i=1}^{D} \lvert z_i \rvert^{2+4\frac{i-1}{D-1}}} + f_5^*$	$\mathbf{z} = \mathbf{x} - \mathbf{o}$	-200
Rosenbrock's Function	$f_6(\mathbf{x}) = \sum_{i=1}^{D-1} (100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2) + f_6^*$	$\mathbf{z} = \frac{2.048(\mathbf{x}-\mathbf{o})}{100} + 1$	-100
Schaffers $F7$ Function	$f_7(\mathbf{x}) = \left(\frac{1}{D-1} \sum_{i=1}^{D-1} (\sqrt{z_i} + \sqrt{z_i}\sin^2(50 z_i^{0.2})) \right)^2 + f_7^*$	$z_i = \sqrt{y_i^2 + y_{i+1}^2}$ $\mathbf{y} = \Lambda^{10} T_{asy}^{0.5}(\mathbf{x} - \mathbf{o})$	0
Ackley's Function	$f_8(\mathbf{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\right) + 20 + e + f_8^*$	$\mathbf{z} = \Lambda^{10} T_{asy}^{0.5}(\mathbf{x} - \mathbf{o})$	100
Weierstrass Function	$f_9(\mathbf{x}) = \sum_{i=1}^{D} \left(\sum_{k=0}^{k_{max}} [a^k \cos(2\pi b^k (z_i + 0.5))] \right) - D \sum_{k=0}^{k_{max}} [a^k \cos(2\pi b^k \cdot 0.5))] + f_9^*$	$a = 0.5, b = 3, k_{max} = 20,$ $\mathbf{z} = \Lambda^{10} T_{asy}^{0.5}\left(\frac{0.5(\mathbf{x}-\mathbf{o})}{100}\right)$	200
Griewank's Function	$f_{10}(\mathbf{x}) = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos(\frac{z_i}{\sqrt{i}}) + 1 + f_{10}^*$	$\mathbf{z} = \Lambda^{100}\left(\frac{600(\mathbf{x}-\mathbf{o})}{100}\right)$	300
Rastrigin's Function	$f_{11}(\mathbf{x}) = \sum_{i=1}^{D} (z_i^2 - 10\cos(2\pi z_i) + 10) + f_{11}^*$	$\mathbf{z} = \Lambda^{10} T_{asy}^{0.2}\left(T_{osz}\left(\frac{5.12(\mathbf{x}-\mathbf{o})}{100}\right)\right)$	400
Schwefel's Function	$f_{12}(\mathbf{x}) = 418.9829 \times D - \sum_{i=1}^{D} g(z_i) + f_{12}^*$	$\mathbf{z} = \Lambda^{10}\left(\frac{1000(\mathbf{x}-\mathbf{o})}{100}\right) + 420.9687462275036$	500
Katsuura Function	$f_{13}(\mathbf{x}) = \frac{10}{D^2}\prod_{i=1}^{D}\left(1 + i\sum_{j=1}^{32}\frac{\lvert 2^j z_i - round(2^j z_j)\rvert}{2^j}\right)^{\frac{10}{D^{1.2}}} - \frac{10}{D^2} + f_{13}^*$	$\mathbf{z} = \Lambda^{100}\left(\frac{5(\mathbf{x}-\mathbf{o})}{100}\right)$	600

$\mathbf{o}$ represents a random shift added to the global optimum; the definitions of Λ^{α}, T_{asy}^{β}, T_{osz} and $g(z_i)$ in f_{12} can be found in [36].

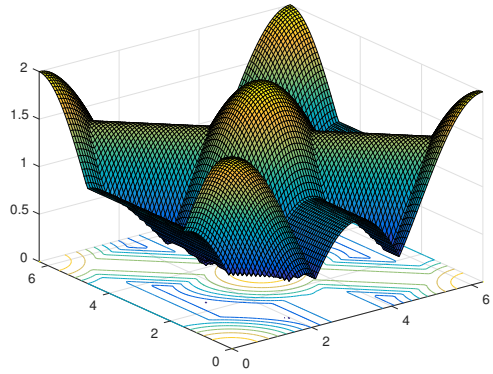


Fig. 1. Surface plot of the spread spectrum radar polyphase code design problem.

Toolbox [43], where the strategy for parent selection is "rank fitness scaling" followed by "stochastic universal sampling (SUS)"; the crossover and mutation operators are "whole arithmetic" and "uniform", respectively; and survival selection is performed using "elitism". The other four algorithms were all implemented from the source codes provided online by their authors. In addition, the bounded constraint strategy "absorb" (i.e., $\min(\max(\mathbf{x}, lb), ub)$) [44] was added to all algorithms except for CMA-ES and SPSO2011, as these algorithms possess specific strategies provided by their authors for handling the search space boundaries. We set the parameters of the real-coded GA, i.e., the crossover rate (p_c), mutation rate (p_m) and elite ratio (r_e), to $p_c = 0.8$, $p_m = 0.05$ and $r_e = 0.05$, respectively. The population sizes for all algorithms were set following the recommendations of their authors; i.e., 100 for the real-coded GA, $4 + \lfloor 3 + \ln D \rfloor$ for CMA-ES, 30 for CoDE, 40 for SPSO2011 and 50 for ABC[1].

C. Results of Visualization

The relationship between the proposed measures of population evolvability and the convergence curve is visualized for all test problems. In this process, we unified the budget of FEs to $10^3 \times D$ and assumed the fitness value of the global optimum to be known a priori for better visualization of the convergence curve. Each of the studied algorithms was

[1]The source code required to run all of the experiments described in this article is available at https://github.com/mangwang/TEVC17.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2744324, IEEE Transactions on Evolutionary Computation

regularly run on every test problem to generate a sequence of P_i; then, the proposed measures of population evolvability were estimated for 10 neighbours separately following equations (8), (9) and (11) with $f^b(P)$ replaced with $f_t^b(P)$, which represents the best fitness value found up through generation t. Each convergence curve plot showed the generational mean and median of 50 convergence curves that were generated by independently running the studied algorithm on the test problem. We initialized the random number generator with a time-dependent seed to ensure the randomness of each run and independence among all runs. Additionally, in accordance with the IEEE Standard for Floating-Point Arithmetic (IEEE 754), the total precision of the double-precision floating-point format is 53 bits, which corresponds to 15-17 significant decimal digits of precision. Therefore, to strike a balance between precision and practicability, we set the threshold to $1e{-}8$ and revised the fitness value based on the following strategy: if $f - \lfloor f \rfloor < 1e{-}8, \text{then } f = \lfloor f \rfloor$. Because of the page limit, we show the results for only three typical test problems, i.e., the Sphere Function, Schwefel's Function and the spread-spectrum radar polyphase code design problem (called the Radar Problem hereafter), as these problems represent simple unimodal, complicated multimodal and NP-hard numerical optimization problems, respectively. The results of the rest of the test problems are included in the supplementary material.

Fig. 2, Fig. 3 and Fig. 4 present the visualization results for the application of the studied algorithms to the Sphere Function, Schwefel's Function and the Radar Problem. From these figures, several interesting findings can be extracted. First, for an easy optimization problem such as the Sphere Function, the values of epp for all studied algorithms remain relatively high throughout the entire process until the global optimum is reached (with the floating-point precision correction), whereas the magnitudes of eap for the different algorithms are diverse. Further inspection of the convergence curves reveals that larger magnitudes of eap correspond to faster convergence towards the global optimum. In this respect, CMA-ES has the largest eap magnitudes; thus, it is the fastest algorithm that finds the global optimum. Second, for hard problems such as Schwefel's Function and the Radar Problem, either epp is relatively low (e.g., for CMA-ES on Schwefel's Function and for CoDE on the Radar Problem, as shown in Figs. 3b and 4c, respectively), or eap is relatively low (e.g., for SPSO2011 on Schwefel's Function and the Radar Problem, as shown in Figs. 3d and 4d, respectively). Such observations illustrate that these algorithms have difficulty evolving towards a fitter region because of either a low evolutionary probability or a low evolutionary ability. Moreover, the performances of these algorithms are also not good, as shown in Figs. 3b, 4c, 3d and 4d, which indicates that these two aspects of problem hardness for a population-based metaheuristic algorithm are effectively reflected in the values of epp and eap. As a result, the evp measure is expected to be correlated with the overall problem hardness. In fact, all of the figures illustrate that the values of evp are strongly related to the performances of the studied algorithms, regardless of the test problem. Concretely, if evp remains at a relatively high value, as shown in Fig. 2 for example, the best fitness values obtained by the algorithm in generations will continuously improve. If, however, the evp measure exhibits continuous regions of zero values, as for CMA-ES and SPSO2011 on Schwefel's Function (shown in Figs. 3b and 3d), then the best fitness values achieved by the algorithm will remain nearly unchanged, and improvement will be difficult to achieve.

V. The Algorithm Selection Task for Black-box Optimization Problems – An Application

From the visualization results presented above, the values of evp are expected to be closely related to the problem hardness for metaheuristic algorithms. Therefore, in this section, we utilize the sequence of evp values as an indicator for performing the algorithm selection task for the test problems described in Section IV. First, the algorithm selection task is introduced. Then, the overall framework for algorithm selection using the evp sequence is explained in detail. Finally, the results achieved using the proposed algorithm selection framework are evaluated through a statistical comparison with the traditional performance evaluation approach, and high accuracy is found to be achieved at a low computational cost.

A. The Algorithm Selection Task

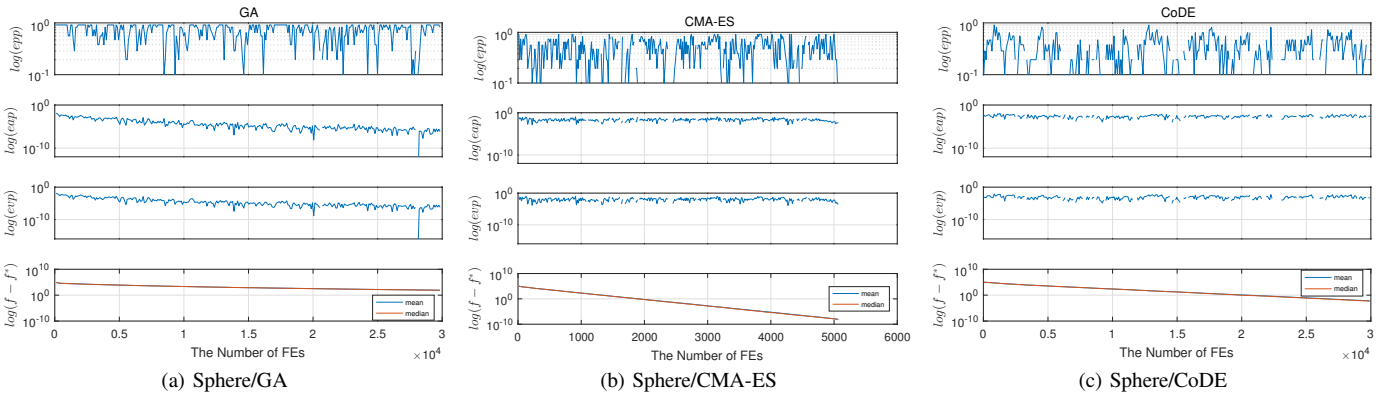
The algorithm selection task, which was first proposed by Rice [45], consists of four components: the problem space, F; the algorithm space, A; the performance space, Q; and the characteristic space, C. For a given problem instance $f \in F$, the algorithm $\alpha \in A$ that achieves the best performance in Q can be selected based on the extracted characteristics from C. According to this definition, in our case, the values of the evp measure can be viewed as the characteristics obtained via dynamic FLA for population-based metaheuristic algorithms; thus, they are expected to be feasible for the algorithm selection task.

B. The Overall Framework

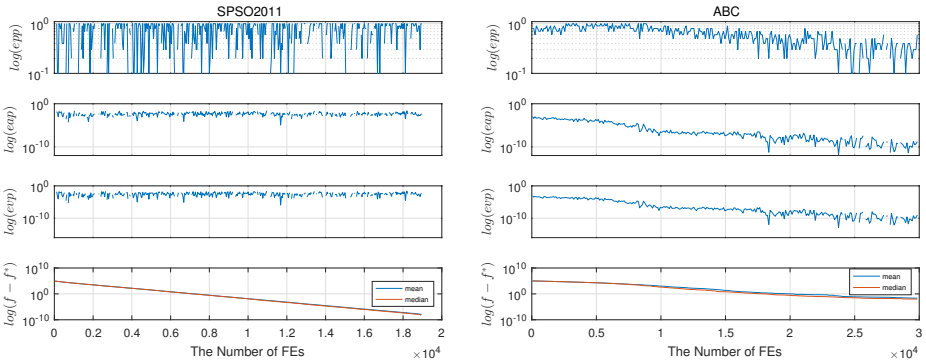
Fig. 5 depicts the overall framework for employing the values of $evp(P_i)$ for the algorithm selection task, where the input to this framework is a set of candidate algorithms A, and the best algorithm α is selected as its output. This framework consists of four relevant components: the sampling of the feasible solution space to obtain sampled populations, the generation of a fixed number of neighbours (one-step child populations), the calculation of the $evp(P_i)$ values, and a statistical analysis for selecting the appropriate algorithm. These components are described in detail below.

1) Sampling the Solution Space: In general, sampling should be used to generate populations (i.e., P_1, P_2, ..., P_T shown in Fig. 5) that can represent various states, as the size of the feasible solution space does not permit the consideration of all possible regions. Sampling can be achieved either offline or online. With regard to offline sampling, such as Latin Hypercube Sampling (LHS) and random sampling [46], the same weights are assigned to all valid search areas, whereas the Metropolis method [47] assigns higher weights to "important" areas. Because not all areas are equally important for

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2744324, IEEE Transactions on Evolutionary Computation

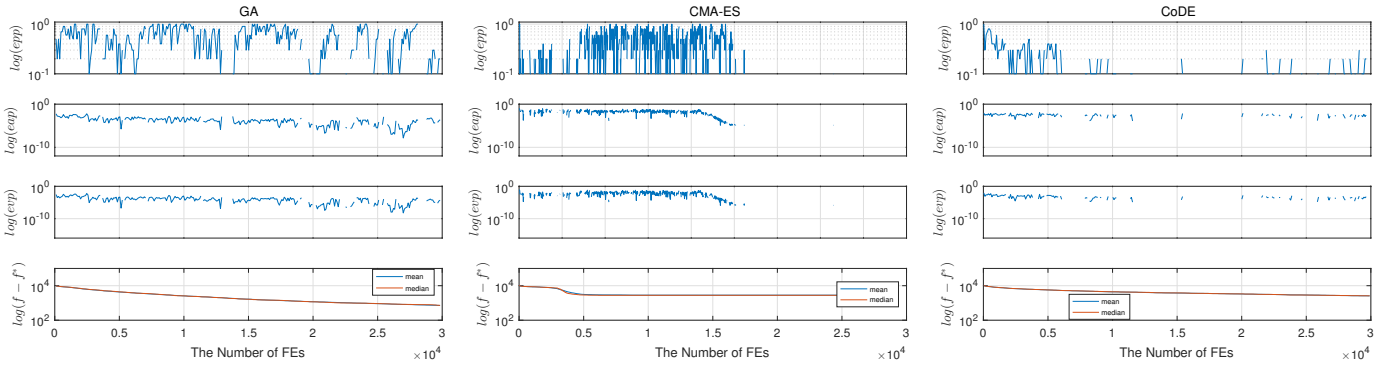


(a) Sphere/GA (b) Sphere/CMA-ES (c) Sphere/CoDE

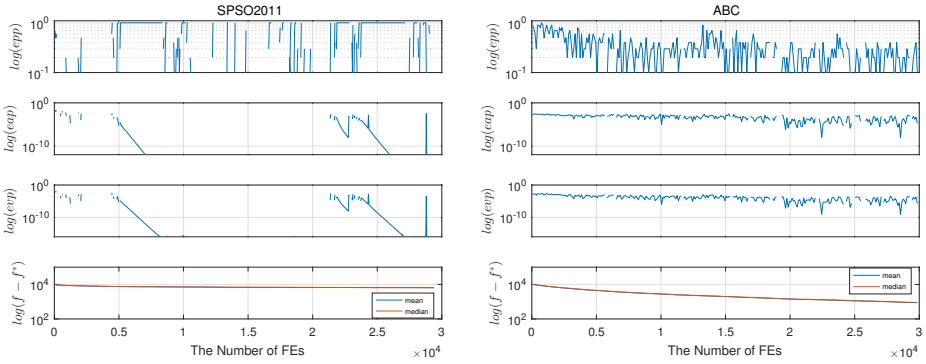


(d) Sphere/SPSO2011 (e) Sphere/ABC

Fig. 2. The relationship between the generational measures and the convergence curve for the Sphere Function. In each group of four, the top graph shows $log(epp)$, the second graph shows $log(eap)$, the third graph shows $log(evp)$ and the bottom graph shows $log(f - f^*)$. As the y axes of these graphs are on a logarithmic scale, gaps appear in continuous regions where the measures take a value of zero.



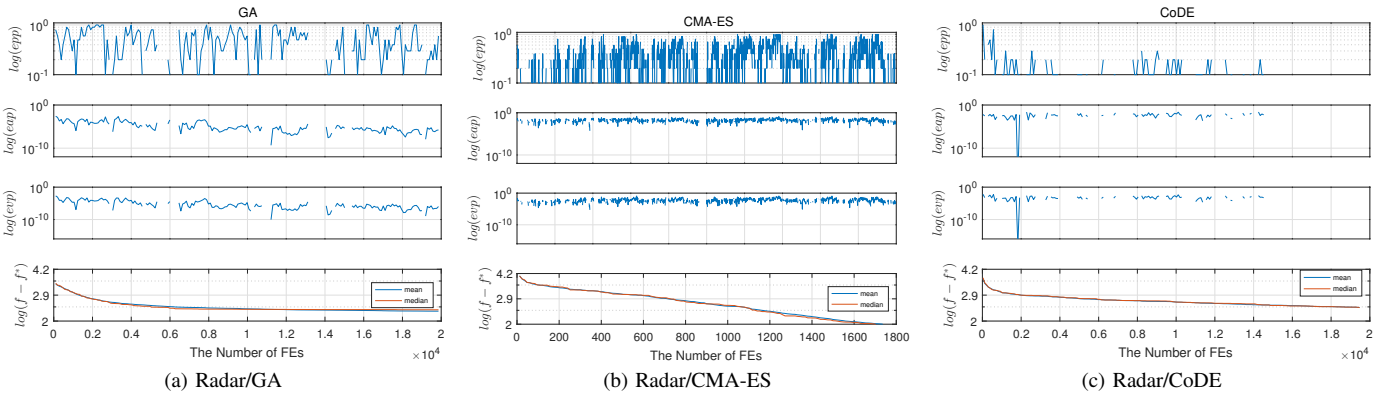
(a) Schwefel/GA (b) Schwefel/CMA-ES (c) Schwefel/CoDE



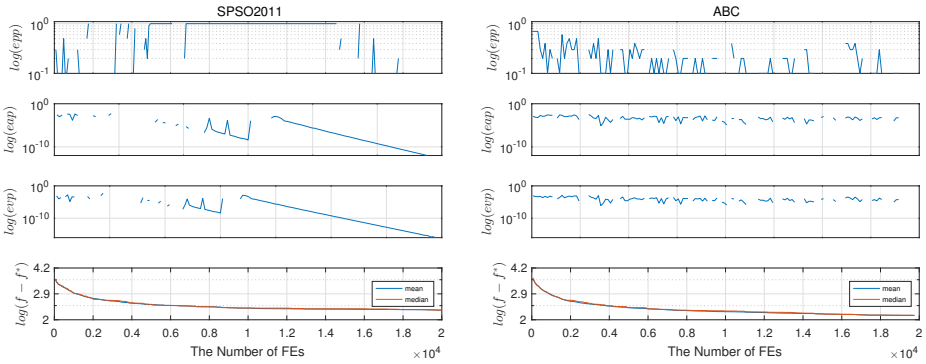
(d) Schwefel/SPSO2011 (e) Schwefel/ABC

Fig. 3. The relationship between the generational measures and the convergence curve for Schwefel's Function. In each group of four, the top graph shows $log(epp)$, the second graph shows $log(eap)$, the third graph shows $log(evp)$ and the bottom graph shows $log(f - f^*)$. As the y axes of these graphs are on a logarithmic scale, gaps appear in continuous regions where the measures take a value of zero.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2744324, IEEE Transactions on Evolutionary Computation



(a) Radar/GA (b) Radar/CMA-ES (c) Radar/CoDE



(d) Radar/SPSO2011 (e) Radar/ABC

Fig. 4. The relationship between the generational measures and the convergence curve for the Radar Problem. In each group of four, the top graph shows $log(epp)$, the second graph shows $log(eap)$, the third graph shows $log(evp)$ and the bottom graph shows $log(f - f^*)$. As the y axes of these graphs are on a logarithmic scale, gaps appear in continuous regions where the measures take a value of zero.

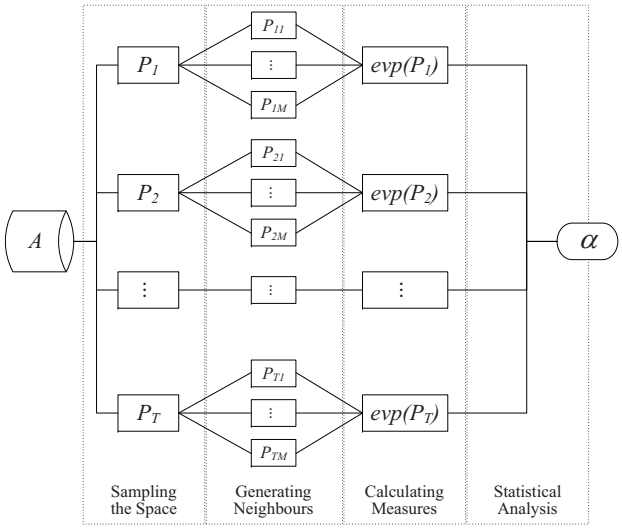


Fig. 5. The overall framework for employing evp for the algorithm selection task.

metaheuristic algorithms, the Metropolis method is preferred. However, the size and location of the search region sampled by an algorithm may constantly change during optimization. Moreover, some self-adaptive algorithms contain adaptive parameters, and no offline sampling method can obtain information about the search region or the current state of adaptive parameters, which means that such methods are not suitable for our purposes. By contrast, with an online sampling method, the sampling procedure is performed during the execution of the algorithm itself. Specifically, the first population P_1 is uniformly randomly sampled from the entire feasible search space, and subsequent populations are generated throughout the execution of the algorithm until a termination criterion is met. Online sampling methods are able to discover important areas and update adaptive parameters automatically by virtue of the behaviour of metaheuristic algorithms, and hence, the online sampling approach is employed in this paper.

When online sampling is adopted, a termination criterion should be established. As different algorithms may have different population sizes, we use FEs as our measure of computational cost [48], and a budget of FEs is used as the termination criterion. Concretely, the FEs budget for online sampling (denoted by *samFEs* hereafter) are initially limited to a number much smaller than the *maxFEs* budget[2], such as $samFEs = 20\% maxFEs$. Then, for a given optimization problem, a candidate algorithm performs online sampling in the feasible space until the *samFEs* budget is reached.

2) Generating Neighbours and Calculating Measure Values: Online sampling generates parental populations, and the best fitness value obtained by each algorithm is recorded during the sampling process. As we know, on simple optimization problems such as the Sphere Function, certain algorithms (e.g., CMA-ES) can rapidly reach the global optimum (with the

[2]The *maxFEs* budget depends on the optimization problem and is usually set based on either experience or typical recommendations from others.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2744324, IEEE Transactions on Evolutionary Computation

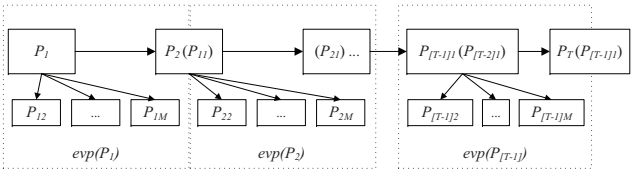


Fig. 6. The procedure for calculating $evp(P_i)$, in which P_{ij} denotes the neighbours of P_i. Because of the online sampling procedure, P_{i+1} is also a neighbour (shown in parentheses) of P_i.

floating-point precision correction), such that the number of FEs used may be even fewer than *samFEs*. In this situation, if we were still to calculate the evp values after the convergence of the algorithm to the global optimum, they would be consecutive zeros, which is unreasonable. Therefore, to ensure fair comparisons among the candidate algorithms and in consideration of the fact that the fitness value of the global optimum is usually unknown in real applications, the best fitness value achieved by the best-performing algorithm during online sampling is considered as an approximation of the fitness value of the global optimum. Thus, the number of FEs corresponding to the first hitting time of the algorithm that is the fastest to reach the fitness value of approximate global optimum is considered as the unified budget for subsequent procedures (denoted by *uniFEs* hereafter), which means that sampled populations that use more FEs than *uniFEs* are discarded in later procedures. In brief, *uniFEs* can be less than or equal to *samFEs*, depending on whether the algorithm can hit the fitness value of the approximate global optimum before reaching the *samFEs* budget.

For each candidate algorithm, the sampled populations within the *uniFEs* budget are treated as parents to separately generate M neighbours from their current states. All neighbours are randomly and independently generated throughout the algorithm to ensure the generalizability of the proposed framework. Then, each $evp(P_i)$ value can be calculated following the procedure shown in Fig. 6, in which, for two consecutive parental populations, i.e., P_i and P_{i+1} (where $i = 1, 2, \ldots, T-1$), P_{i+1} is also a neighbour of P_i; thus, we need to generate only $(M-1)$ new neighbours to estimate the value of $evp(P_i)$. Fig. 6 also indicates that the minimum value of M is 1, in which case, each $evp(P_i)$ value is estimated directly through the online sampling process.

3) Statistical Analysis: For a given optimization problem, each candidate algorithm generates a sequence of $evp(P_i)$ values. The sequence sizes for different algorithms may be diverse because of different population sizes. Further investigation of the relations indicates that because of the intrinsic behaviour of metaheuristic algorithms, the $evp(P_i)$ values in each sequence are dependent and not normally distributed, and all sequences are independent of each other. Moreover, there are usually at least two candidate algorithms, which means that the number of sequences is usually greater than two. These properties of the evp sequences satisfy all of the requirements for a non-parametric statistical test, i.e., the Kruskal-Wallis test [49]. In this test, the null hypothesis H_0 is that the distributions of all k sequences are equal, where k is the number of candidate algorithms. Then, the total evp values from all sequences are globally ranked. Let N_s denotes the total number of evp values across all sequences, r_{ij} denotes the global rank of evp value j from sequence i, n_i denotes the number of evp values in sequence i, and $R_i = \sum_{j=1}^{n_i} r_{ij}$ denotes the sum of the ranks from the ith sequence, then $N_s = \sum_{i=1}^{k} n_i$, and the test statistic is calculated in the following form:

$$H = \frac{12}{N_s(N_s+1)} \sum_{i=1}^{k} \frac{R_i^2}{n_i} - 3(N_s+1) \tag{13}$$

Any tied values are assigned the average of their original ranks, and a correction for ties can be applied by dividing the H value calculated in (13) by $1 - \sum_{g=1}^{G}(t_g^3 - t_g)/(N_s^3 - N_s)$, where G represents the number of groupings of different tied ranks and t_g is the number of tied ranks within group g. The p-value of the Kruskal-Wallis test can be approximated from the χ^2 distribution with $(k-1)$ degrees of freedom. If the p-value is less than the pre-set significance level (such as 0.05), a post hoc procedure can be used for multiple comparisons between pairs of sequences to obtain stochastic dominance [50]. In this study, we adopt the Conover-Iman test [51] as the post hoc procedure because it can directly use the ranks generated from the Kruskal-Wallis test. Its test statistic is formulated as follows:

$$\begin{aligned} S^2 &= \frac{1}{N_s - 1}\left[\sum_{i=1}^{k}\sum_{j=1}^{n_i} r_{ij}^2 - N_s\frac{(N_s+1)^2}{4}\right] \\ \left|\overline{R_i} - \overline{R_j}\right| &> t_{1-\alpha}\sqrt{S^2\frac{N_s - 1 - H}{N_s - k}}\sqrt{\frac{1}{n_i} + \frac{1}{n_j}} \end{aligned} \tag{14}$$

where $\overline{R_i} = R_i/n_i$ represents the average rank in sequence i and t represents Student's t-distribution. Then, the p-value can be calculated from the upper α quantile of the t-distribution with $(N_s - k)$ degrees of freedom and adjusted by means of the Holm procedure [52] to revise the family-wise error rate (FWER), which represents the probability that one or more false conclusions will be drawn among all hypotheses when performing multiple pair-wise tests. Eventually, the best algorithm or a set of equally best algorithms can be selected according to the results of the statistical tests described above.

C. Performing the Algorithm Selection Task via the Performance Evaluation Approach

The algorithm selection task can also be performed by evaluating the performances of all candidate algorithms on a given optimization problem. In this study, 50 independent runs of each candidate algorithm on each test problem were executed with a pre-set FEs budget of $10^4 \times D$. Then, statistical tests similar to those described in Section V-B3 were applied to compare the performances of the candidate algorithms on that problem. Concretely, the Kruskal-Wallis test was first used to check for differences among the entire set of candidate algorithms. If the null hypothesis of the Kruskal-Wallis test was rejected based on the pre-set significance level, the Conover-Iman test was further employed as a post hoc procedure for pair-wise comparisons of stochastic dominance. Then, the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2744324, IEEE Transactions on Evolutionary Computation

Holm procedure was applied to adjust the p-value obtained from the Conover-Iman test. Finally, the best algorithm or a set of equally best algorithms was selected based on the final results of these statistical tests.

D. Results for the Algorithm Selection Task

For each test problem, the best algorithms were selected based on the proposed algorithm selection framework and the performance evaluation approach. As suggested by Naudts *et al.* [53], the results obtained through performance evaluation can be regarded as the gold standard for the algorithm selection task, against which we can evaluate the results obtained by the proposed framework. As both approaches are based on statistical analysis, the evaluation criterion is as follows: for a test problem, if one of the best algorithms selected by the proposed framework is contained in the algorithm set chosen based on the performance evaluation approach, we consider the former to be correct; otherwise, it is considered a mistake.

Table II shows the selected algorithms for each test problem based on the two previously described approaches. For the proposed framework, the average rank in each sequence (i.e., $\overline{R_i}$) is shown as the statistical indicator. For the performance evaluation approach, we show the mean and standard deviation of the best fitness values obtained in 50 independent runs. As the statistical analysis procedures used for both approaches are the same, the results are comparable in terms of statistical significance. According to the evaluation criterion described above, the proposed approach achieves remarkable performance (with an accuracy of 100%) in the algorithm selection task. Moreover, in this study, the FEs budget for online sampling and the number of neighbours for calculating the evp values were set to $samFEs = 20\%maxFEs$ and $M = 10$, respectively, and consequently, the proposed algorithm selection framework incurred a much lower computational cost than the performance evaluation approach. We will discuss the details of further reducing the computational cost in the next section.

The detailed differences between the algorithms selected using the two approaches can also be explored based on Table II. Concretely, for the unimodal test problems, i.e., f_1 to f_5, the proposed framework selects only one algorithm, whereas more than two algorithms are selected by the performance evaluation approach. After further checking the number of FEs required for the convergence of algorithms, we discover that the proposed framework prioritizes the algorithm that has the best performance and also converges fastest. This phenomenon can be easily understood, as the best-performing and fastest-converging algorithm has the highest evp values. For the multimodal test problems, i.e., f_6 to f_{13}, the proposed framework always selects the best-performing algorithm in terms of the best mean and the lowest standard deviation. Although other algorithms, such as CoDE, are also selected by the performance evaluation approach for several problems, the proposed framework tends to select the best-performing (i.e., with the best mean) and the most robust (i.e., with the lowest standard deviation) algorithm. Notably, the proposed framework can make the correct choice even on the NP-hard numerical optimization problem (i.e., f_{radar}), which indicates its general applicability for the algorithm selection task.

VI. Further Discussions

A. Robustness of the Proposed Framework

Here, the robustness of the proposed framework for the algorithm selection task is discussed. As mentioned earlier, the proposed framework for the algorithm selection task is based on online sampling and estimates of the evp measure. Both the online sampling and measure estimation procedures contain randomness, and the seed of the random number generator is reset based on the current time before either procedure is performed. Therefore, if the results are robust to this intrinsic randomness, the proposed algorithm selection framework can be considered reasonable and trustworthy.

An experiment was conducted to test the robustness of the proposed framework. In this experiment, we first fixed the FEs budget for online sampling and the number of neighbours to be generated for estimating the evp values to $samFEs = 20\%maxFEs$ and $M = 10$, respectively, and then ran the proposed framework 10 times. The results obtained from each run were statistically evaluated with respect to those achieved using the performance evaluation approach (shown in Table II). We found that the results obtained using the proposed framework are stable. Specifically, the proposed framework achieved an accuracy of 100% three times and made one mistake four times. For the remainder of the runs, it made more than one but fewer than four mistakes, most of which occurred for the functions f_8, f_{10} and f_{13}. Notably, these functions that confuse the proposed framework are all complicated multimodal functions with many local optima, which may affect the information obtained via online sampling. However, in all of the cases in which the proposed framework failed to choose the correct algorithm, it ranked that algorithm second, i.e., was not far off.

B. Accuracy of the Proposed Framework Versus Computational Cost

In this subsection, two tunable parameters of the proposed framework are discussed. The first one is the number of neighbours (i.e., M) used for measure estimation. This parameter can be interpreted as specifying the trade-off between the accuracy of estimation and the computational cost: the larger the number of neighbours is, the more accurate the measure estimates should become, at the price of a higher computational cost. To further investigate the influence of this parameter on the accuracy of the framework for the algorithm selection task, we first fixed the FEs budget for online sampling to $samFEs = 20\%maxFEs$ and observed the relation between the results of the algorithm selection task and the number of neighbours.

As shown in Table III, if the number of neighbours M decreases below 5, the number of test functions for which the proposed framework makes mistakes will start to increase. Therefore, the neighbour number threshold that ensures the accuracy of the proposed framework is 5. More importantly, when $samFEs$ and M are separately set to $20\%maxFEs$ and 5,

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2744324, IEEE Transactions on Evolutionary Computation

TABLE II
COMPARISON OF THE RESULTS ACHIEVED USING THE PROPOSED APPROACH AND THE PERFORMANCE EVALUATION APPROACH FOR THE ALGORITHM SELECTION TASK

Function	Measure	GA	CMA-ES	CoDE	SPSO2011	ABC
f_1	avgEvpRank	201	**407**†	182	228	88
	performEval	-6.0000E+02(3.54E-04)	**-6.0000E+02(0.00E+00)**†	**-6.0000E+02(0.00E+00)**†	**-6.0000E+02(0.00E+00)**†	**-6.0000E+02(0.00E+00)**†
f_2	avgEvpRank	916	**3479**†	1870	1951	817
	performEval	-4.9988E+02(1.12E-01)	**-5.0000E+02(0.00E+00)**†	**-5.0000E+02(0.00E+00)**†	2.1386E+05(6.98E+04)	**-5.0000E+02(0.00E+00)**†
f_3	avgEvpRank	320	**1202**†	698	749	235
	performEval	-9.9465E+01(2.86E+02)	**-4.0000E+02(0.00E+00)**†	**-4.0000E+02(0.00E+00)**†	1.8478E+03(2.63E+03)	**-4.0000E+02(0.00E+00)**†
f_4	avgEvpRank	1839	2118	**2648**†	1174	1594
	performEval	-3.0000E+02(4.52E-04)	**-3.0000E+02(0.00E+00)**†	**-3.0000E+02(0.00E+00)**†	1.7126E+03(6.77E+02)	**-3.0000E+02(0.00E+00)**†
f_5	avgEvpRank	1332	**4797**†	2905	3659	1103
	performEval	-2.0000E+02(2.04E-03)	**-2.0000E+02(0.00E+00)**†	**-2.0000E+02(0.00E+00)**†	-2.0000E+02(3.98E-05)	**-2.0000E+02(0.00E+00)**†
f_6	avgEvpRank	898	**3017**†	2086	1455	960
	performEval	-9.1746E+01(2.17E+01)	**-9.8563E+01(1.01E+01)**†	**-9.7931E+01(1.09E+01)**†	-3.0970E+01(3.40E+01)	-9.9995E+01(5.33E-03)
f_7	avgEvpRank	4181	3173	**5397**†	4582	3889
	performEval	1.4193E-01(1.05E-01)	1.7880E+01(1.38E+01)	**0.0000E+00(0.00E+00)**†	4.6581E+01(1.52E+01)	**0.0000E+00(0.00E+00)**†
f_8	avgEvpRank	**5436**†	2991	4657	4220	**5334**†
	performEval	**1.2000E+02(2.33E-04)**†	1.2024E+02(1.23E-01)	1.2000E+02(3.00E-03)	1.2075E+02(8.62E-02)	1.2000E+02(8.26E-04)
f_9	avgEvpRank	5475	3073	4234	4321	**5913**†
	performEval	2.0006E+02(2.30E-02)	2.0445E+02(2.54E+00)	2.0050E+02(4.25E-01)	2.1400E+02(3.52E+00)	**2.0000E+02(2.44E-05)**†
f_{10}	avgEvpRank	4565	2710	**5583**†	**5639**†	4467
	performEval	3.0008E+02(4.94E-02)	3.0002E+02(1.16E-02)	**3.0000E+02(0.00E+00)**†	3.0021E+02(1.15E-01)	**3.0000E+02(0.00E+00)**†
f_{11}	avgEvpRank	5441	2758	4823	4937	**5832**†
	performEval	4.0000E+02(2.76E-03)	4.5015E+02(1.22E+01)	**4.0002E+02(1.39E-01)**†	4.7118E+02(1.69E+01)	**4.0000E+02(0.00E+00)**†
f_{12}	avgEvpRank	5616	3098	4250	4152	**5987**†
	performEval	**5.0530E+02(3.35E+00)**†	3.3316E+03(5.94E+02)	5.6013E+02(6.08E+01)	5.0315E+03(5.90E+02)	**5.0481E+02(2.04E+00)**†
f_{13}	avgEvpRank	**4924**†	3782	4400	2876	4381
	performEval	**6.0000E+02(1.27E-04)**†	6.0003E+02(1.41E-02)	6.0000E+02(3.94E-04)	6.0161E+02(4.00E-01)	6.0002E+02(3.10E-03)
f_{radar}	avgEvpRank	2394	**3601**†	1528	1316	1580
	performEval	2.0584E+00(2.24E-01)	**1.2365E+00(1.61E-01)**†	**1.2614E+00(1.64E-01)**†	1.9877E+00(1.55E-01)	1.9024E+00(8.60E-02)

The selected algorithms are highlighted in boldface.

"avgEvpRank" and "performEval" denote the average evp rank in each sequence (i.e., $\overline{R_i}$) and the performance evaluation approach, respectively.

In all "performEval" rows, every measure is presented as the mean and standard deviation (in parentheses) for 50 independent runs.

"†" indicates that the selected algorithms are statistically significantly superior to the unselected algorithms at the 0.05 significance level and the 0.05 FWER level.

TABLE III
RELATION BETWEEN THE RESULTS OF THE ALGORITHM SELECTION TASK AND THE NUMBER OF NEIGHBOURS

M	20	15	10	5	4	3	2	1
MisFunc	/	/	/	/	f_7, f_{10}	f_7, f_{10}, f_{11}	$f_4, f_7, f_9, f_{10}, f_{11}$	$f_4, f_5, f_7, f_8, f_9, f_{10}, f_{11}, f_{radar}$
Cnt	0	0	0	0	2	3	5	8

"M" denotes the number of neighbours.

"MisFunc" denotes the test functions on which the proposed framework made mistakes.

"Cnt" denotes the number of functions in MisFunc.

TABLE IV
RELATION BETWEEN THE RESULTS OF THE ALGORITHM SELECTION TASK AND THE ONLINE SAMPLING BUDGET

samFEs	10%*maxFEs*	20%*maxFEs*	30%*maxFEs*	40%*maxFEs*	50%*maxFEs*
MisFunc	f_8, f_{10}, f_{13}	/	/	/	/
Cnt	3	0	0	0	0

"*samFEs*" denotes the budget of FEs for online sampling.

"MisFunc" denotes the test functions on which the proposed framework made mistakes.

"Cnt" denotes the number of functions in MisFunc.

the total number of FEs required to execute the entirety of the proposed framework for the algorithm selection task is equal to or fewer than *maxFEs*, which is at most sufficient to perform one run in the performance evaluation approach. Generally, performing only one run for each metaheuristic algorithm in a black-box optimization problem is not sufficient for a

statistically sound determination of which algorithm is the best choice for that problem, whereas the proposed framework can produce statistically significant results with high accuracy at a similar or lower computational cost in terms of FEs.

For the other parameter, i.e., the FEs budget for online sampling (*samFEs*), a similar study was performed with the number of neighbours set to $M = 10$. As illustrated in Table IV, when *samFEs* is set to 10%*maxFEs*, the proposed approach makes three mistakes, for functions f_8, f_{10} and f_{13}. When *samFEs* is increased from 20%*maxFEs* up to 50%*maxFEs*, the proposed framework remains an accuracy of 100%. It should be noted that additional experiments with continuously increasing values of this parameter were not performed because such values would markedly increase the computational cost of the proposed framework. In this article, we recommend *samFEs* be set to 20%*maxFEs*.

VII. Conclusions

In this paper, the concept of population evolvability is presented as an extension of dynamic FLA. Two measures, i.e., epp and eap, are proposed to describe the probability that a population will obtain improved solutions to an optimization problem and its ability to do so. Then, a combined measure, i.e., evp, is derived to represent the overall population evolvability. Experimental studies illustrate the validity of the proposed measures. The algorithm selection task for black-box optimization problems is considered as an application to demonstrate that the proposed framework is a generally applicable and low-cost approach for performing the algorithm selection task with high accuracy and statistical significance.

It is noteworthy that this work is merely a preliminary step towards the goal of developing more practical FLA techniques. Additional studies based on the proposed concept will be conducted in the future. From the perspective of algorithm designers, further work can be focused on the design and application of the proposed measures for the hyper-parameter tuning of population-based metaheuristic algorithms, including parameter control, algorithm portfolios and hybridization.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grants 61473271, 61573125 and 61329302, the Ministry of Science and Technology of China (Grant No. 2017YFC0804003), the Science and Technology Innovation Committee Foundation of Shenzhen (Grant No. ZDSYS201703031748284), and EPSRC (Grant No. EP/K001523/1). Xin Yao was also supported by a Royal Society Wolfson Research Merit Award. The authors would like to thank Prof. Thomas Weise, Dr. Bo Yuan, Dr. Fang Tao and Dr. Laijuan Xu for proofreading the entire manuscript and providing valuable comments.

References

[1] S. Wright, "The roles of mutation, inbreeding, crossbreeding and selection in evolution," in *Proc. 6th Int. Congr. Genet.*, 1932, vol. 1, pp. 356–366.

[2] E. Pitzer and M. Affenzeller, "A comprehensive survey on fitness landscape analysis," in *Recent Adv. Intell. Eng. Syst.*, 2012, pp. 161–191.

[3] H. Richter and A. Engelbrecht, *Recent advances in the theory and application of fitness landscapes*. Springer, 2014.

[4] T. Weise, M. Zapf, R. Chiong, and A. J. Nebro, "Why is optimization difficult?" in *Nature-Inspired Algorithms for Optimisation*. Springer, 2009, pp. 1–50.

[5] T. Weise, R. Chiong, and K. Tang, "Evolutionary optimization: Pitfalls and booby traps," *J. Comput. Sci. Technol.*, vol. 27, pp. 907–936, 2012.

[6] Y. Liu, X. Yao, Q. Zhao, and T. Higuchi, "Scaling up fast evolutionary programming with cooperative coevolution," in *Proc. 2001 IEEE Congr. Evol. Comput.*, vol. 2, 2001, pp. 1101–1108.

[7] P. F. Stadler, "Fitness landscapes," in *Biol. Evol. Stat. Phys.*, 2002, pp. 183–204.

[8] V. K. Vassilev, T. C. Fogarty, and J. F. Miller, "Smoothness, ruggedness and neutrality of fitness landscapes: from theory to application," in *Adv. Evol. Comput. Theory Appl.*, 2003, pp. 3–44.

[9] E. Weinberger, "Correlated and uncorrelated fitness landscapes and how to tell the difference," *Biol. Cybern.*, vol. 63, no. 5, pp. 325–336, 1990.

[10] T. Jones, "Evolutionary algorithms, fitness landscapes and search," Ph.D. dissertation, Univ. of New Mexico, May 1995.

[11] C. M. Reidys and P. F. Stadler, "Neutrality in fitness landscapes," *Appl. Math. Comput.*, vol. 117, no. 2, pp. 321–350, Jan. 2001.

[12] L. Barnett, "Ruggedness and neutrality: The nkp family of fitness landscapes," in *Proc. 6th Int. Conf. Artif. life*, 1998, pp. 18–27.

[13] T. Jones and S. Forrest, "Fitness distance correlation as a measure of problem difficulty for genetic algorithms," in *Proc. 6th Int. Conf. Genet. Algorithms*, vol. 95, 1995, pp. 184–192.

[14] O. Mersmann, M. Preuss, and H. Trautmann, "Benchmarking evolutionary algorithms: Towards exploratory landscape analysis," in *Proc. 11th Int. Conf. Parallel Probl. Solving from Nat.*, 2010, pp. 73–82.

[15] O. Mersmann, B. Bischl, H. Trautmann, M. Preuss, C. Weihs, and G. Rudolph, "Exploratory landscape analysis," in *Proc. 13th Genet. Evol. Comput. Conf.*, 2011, pp. 829–836.

[16] M. A. Muñoz, M. Kirley, and S. K. Halgamuge, "Exploratory landscape analysis of continuous space optimization problems using information content," *IEEE Trans. Evol. Comput.*, vol. 19, no. 1, pp. 74–87, 2015.

[17] K. Leyton-Brown, E. Nudelman, and Y. Shoham, "Empirical hardness models: Methodology and a case study on combinatorial auctions," *J. ACM*, vol. 56, pp. 22:1–22:52, Jul. 2009.

[18] B. Bischl, O. Mersmann, H. Trautmann, and M. Preuß, "Algorithm selection based on exploratory landscape analysis and cost-sensitive learning," in *Proc. 14th Genet. Evol. Comput. Conf.*, 2012, pp. 313–320.

[19] M. A. Muñoz, M. Kirley, and S. K. Halgamuge, "A meta-learning prediction model of algorithm performance for continuous optimization problems," in *Proc. 12th Int. Conf. Parallel Probl. Solving from Nat.*, 2012, pp. 226–235.

[20] N. Hansen, "The cma evolution strategy: A tutorial," Apr. 2016, arXiv preprint arXiv:1604.00772. [Online]. Available: https://arxiv.org/abs/1604.00772

[21] K. A. Smith-Miles, "Cross-disciplinary perspectives on meta-learning for algorithm selection," *ACM Comput. Surv.*, vol. 41, no. 1, pp. 6:1–6:25, Jan. 2009.

[22] L. Altenberg, "The evolution of evolvability in genetic programming," *Adv. Genet. Program.*, pp. 47–74, 1994.

[23] P. D. Turney, "Increasing evolvability considered as a large-scale trend in evolution," in *Proc. 1999 Genet. Evol. Comput. Conf.*, 1999, pp. 43–46.

[24] T. Smith, P. Husbands, P. Layzell, and M. O'Shea, "Fitness landscapes and evolvability," *Evol. Comput.*, vol. 10, no. 1, pp. 1–34, 2002.

[25] L. Vanneschi, M. Clergue, P. Collard, M. Tomassini, and S. Vérel, "Fitness landscapes and problem hardness in genetic programming," in *Proc. 6th Genet. Evol. Comput. Conf.*, 2004, pp. 690–701.

[26] G. Lu, J. Li, and X. Yao, "Fitness-probability cloud and a measure of problem hardness for evolutionary algorithms," in *Eur. Conf. Evol. Comput. Comb. Optim.*, 2011, pp. 108–117.

[27] C. Philippe, S. Vérel, and C. Manuel, "Local search heuristics: Fitness cloud versus fitness landscape," in *Proc. 16th Eureopean Conf. Artif. Intell.*, 2004, pp. 973–974.

[28] P. Merz, "Advanced fitness landscape analysis and the performance of memetic algorithms," *Evol. Comput.*, vol. 12, no. 3, pp. 303–325, 2004.

[29] C. Darwin, *The origin of species by means of natural selection*. London, UK: John Murray, 1859.

[30] F. J. Ayala, *Population and evolutionary genetics: a primer.* Menlo Park, California, USA: The Benjamin/Cummings Publishing Company, Inc., 1982.

[31] I. Rechenberg, *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Stuttgart: Frommann-Holzboog Verlag, 1973.

[32] H.-G. Beyer and H.-P. Schwefel, "Evolution strategies–a comprehensive introduction," *Nat. Comput.*, vol. 1, no. 1, pp. 3–52, 2002.

[33] S. Nijssen and T. Bäck, "An analysis of the behavior of simplified evolutionary algorithms on trap functions," *IEEE Trans. Evol. Comput.*, vol. 7, no. 1, pp. 11–22, 2003.

[34] D. E. Goldberg, *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley, 1989.

[35] M. Črepinšek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: A survey," *ACM Comput. Surv.*, vol. 45, no. 3, pp. 1–33, 2013.

[36] J. Liang, B. Qu, P. N. Suganthan, and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the cec 2013 special session on real-parameter optimization," *Zhengzhou University and Nanyang Technological University, Tech. Rep.*, 2013.

[37] N. Mladenović, J. Petrović, V. Kovačević-Vujčić, and M. Čangalović, "Solving spread spectrum radar polyphase code design problem by tabu search and variable neighbourhood search," *Eur. J. Oper. Res.*, vol. 151, no. 2, pp. 389–399, 2003.

[38] V. V. Kovačević-Vujčić, M. M. Čangalović, M. D. Ašić, L. Ivanović, and M. Dražić, "Tabu search methodology in global optimization," *Comput. Math. with Appl.*, vol. 37, no. 4-5, pp. 125–133, 1999.

[39] T. Weise, *Global Optimization Algorithms – Theory and Application*. Germany: it-weise.de, 2009.

[40] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 55–66, 2011.

[41] M. Clerc, "Standard particle swarm optimisation – from 2006 to 2011," Dec. 2012, hal id: hal-00764996. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00764996/document

[42] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm," *J. Glob. Optim.*, vol. 39, no. 3, pp. 459–471, 2007.

[43] *MATLAB version 9.1.0.441655 (R2016b)*, The Mathworks, Inc., Natick, Massachusetts, 2016.

[44] A. H. Gandomi and X. S. Yang, "Evolutionary boundary constraint handling scheme," *Neural Comput. Appl.*, vol. 21, no. 6, pp. 1449–1462, 2012.

[45] J. R. Rice, "The algorithm selection problem," *Adv. Comput.*, vol. 15, pp. 65–118, 1976.

[46] M. D. McKay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[47] N. N. Madras, *Lectures on monte carlo methods*. American Mathematical Society, 2002, vol. 16.

[48] T. Weise, R. Chiong, K. Tang, J. Lässig, S. Tsutsui, W. Chen, Z. Michalewicz, and X. Yao, "Benchmarking optimization algorithms: an open source framework for the traveling salesman problem," *IEEE Comput. Intell. Mag.*, vol. 9, pp. 40–52, 2014.

[49] W. H. Kruskal and W. A. Wallis, "Use of ranks in one-criterion variance analysis," *J. Am. Stat. Assoc.*, vol. 47, no. 260, pp. 583–621, 1952.

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

[51] W. J. Conover and R. L. Iman, "On multiple-comparisons procedures," *Los Alamos Sci. Lab. Tech. Rep.*, 1979.

[52] S. Holm, "A simple sequentially rejective multiple test procedure," *Scand. J. Stat.*, pp. 65–70, 1979.

[53] B. Naudts and L. Kallel, "A comparison of predictive measures of problem difficulty in evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 4, no. 1, pp. 1–15, 2000.

Mang Wang received his Ph.D. degree in Information and Communication Engineering from the School of Information Science and Technology, University of Science and Technology of China (USTC), Hefei, China, in 2017.

He is currently a Senior Engineer with Huawei Software Technologies Co., Ltd. His major research interests include statistical learning, deep learning, evolutionary computation, computer vision and their applications in industry.

Bin Li (M'07) received his B.Sc. degree from the Hefei University of Technology, Hefei, China, in 1992, M.Sc. degree from the Institute of Plasma Physics, Chinese Academy of Sciences, Hefei, in 1995, and Ph.D. degree from the University of Science and Technology of China (USTC), Hefei, in 2001.

He is currently a Professor with the School of Information Science and Technology, USTC. He has authored or co-authored over 40 refereed publications. His current research interests include evolutionary computation, pattern recognition, human-computer interaction and real-world applications. He is the Founding Chair of the IEEE Computational Intelligence Society Hefei Chapter, a Counselor of the IEEE USTC Student Branch, a Senior Member of the Chinese Institute of Electronics (CIE), and a member of the Technical Committee of the Electronic Circuits and Systems Section of CIE.

Guofu Zhang (M'11) received his B.Sc. and Ph.D. degrees in computer science from the Hefei University of Technology, Hefei, China, in 2002 and 2008, respectively.

He is currently an Associate Professor with the School of Computer and Information, Hefei University of Technology. His current research interests include multi-agent systems, evolutionary computation, and search-based software engineering.

Xin Yao (M'91-SM'96-F'03) received his B.Sc. degree from the University of Science and Technology of China (USTC), Hefei, China, in 1982, M.Sc. degree from the North China Institute of Computing Technology, Beijing, China, in 1985, and Ph.D. degree from USTC in 1990.

He is currently 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 a Distinguished Lecturer of IEEE Computational Intelligence Society (CIS). His major research interests include evolutionary computation, ensemble learning, and their applications in software engineering. His research won the 2001 IEEE Donald G. Fink Prize Paper Award, 2010, 2016, and 2017 IEEE Transactions on Evolutionary Computation Outstanding Paper Awards, 2010 BT Gordon Radley Award for Best Author of Innovation (Finalist), 2011 IEEE Transactions on Neural Networks Outstanding Paper Award, and many other best paper awards. He received the prestigious Royal Society Wolfson Research Merit Award in 2012 and the IEEE CIS Evolutionary Computation Pioneer Award in 2013. He was the the President (2014-15) of IEEE CIS, and the Editor-in-Chief (2003-08) of IEEE Transactions on Evolutionary Computation.