Swarm and Evolutionary Computation 75 (2022) 101148

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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

An offline learning co-evolutionary algorithm with problem-specific knowledge

Fuqing Zhao [a,*], Bo Zhu [a], Ling Wang [b], Tianpeng Xu [a], Ningning Zhu [a], Jonrinaldi Jonrinaldi [c]

[a] School of Computer and Communication, Lanzhou University of Technology, Lanzhou 730050, China
[b] Department of Automation, Tsinghua University, Beijing 10084, China
[c] Department of Industrial Engineering, Universitas Andalas, Padang 25163, Indonesia

ARTICLE INFO

Keywords:
Fitness landscape
Random forest
Offline-learning
Estimation of distribution
Differential evolution

ABSTRACT

The meta-heuristics is an effective way to solve the complex optimization problems. However, the applicability of meta-heuristic is restricted in real applications due to the various characteristics of the corresponding problems. An offline learning co-evolutionary algorithm (OLCA) based on the fitness landscape analysis that introduces the Gaussian estimation of distribution algorithm (EDA) and a variant of differential evolution (DE) for enhancing the search ability, is proposed for complex continuous real-valued problems. The relationship between strategies and fitness landscapes is established by using offline learning of a random forest. The suitable strategy is determined based on the properties of the fitness landscape trained by a random forest before the beginning of the evolutionary process. The proposed OLCA is tested by using the CEC 2017 benchmark test suite and is compared with several state-of-the-art algorithms. The results show that the proposed OLCA is efficient and competitive for solving complex continuous optimization problems. In addition, the effectiveness of the proposed OLCA is also verified by using 19 IEEE CEC 2011 benchmark problems for tackling real-world problems.

1. Introduction

Optimization problems are an important domain in computer science, management sciences, and engineering applications since various real problems are essentially aimed at achieving optimal configurations [1]. The complex continuous optimization problems are usually expressed as minimizing the objective function $Min\ f(\boldsymbol{x})$ and the optimal solution vector $\boldsymbol{x} = (x_1, x_2, \ldots x_i, \ldots x_D)$, where D denotes the dimension of the problem [2]. It is notable that the difficulty of computation increases exponentially with an increase in the number of dimensions. The complex continuous optimization problems are characterized by non-linear, non-convex, multi-modal, non-differentiable, and non-separable variables [3,4]. The traditional mathematical methods have a strong theoretical foundation and work well on linear, integer programming problems. However, the mathematical methods are unable to efficiently solve the complex optimization problems with the aforementioned properties. The gradient information is utilized frequently in the mathematical approach for finding a satisfactory solution. The black-box optimization problems without gradient information are hard to solve based on the mathematical methods [5]. The meta-heuristics are an effective way to overcome the shortcomings of mathematical methods.

Various meta-heuristics have been presented in literature to address the complex optimization problems [6,7] including genetic algorithm (GA) [8,9], differential evolution (DE) [10–13], particle swarm optimization (PSO) [14,15], and estimation of distribution algorithm (EDA) [16,17]. As compared to mathematical methods, meta-heuristics are more robust, less demanding for rigorous mathematical formulations, and require no gradient information. Besides, meta-heuristics usually operate on a population (implicit parallelism), thus being less prone to the local optima [18]. Although the proposed meta-heuristics have achieved satisfactory results, different meta-heuristics have different biases. It is noteworthy that no algorithm performs best in all the cases as stated by the no free lunch theorem [19].

The selection of appropriate algorithm based on the different characteristics of the problem is a feasible way to solve the optimization problems [20,21]. The selection of an algorithm based on an intelligent policy to achieve the co-evolution between various algorithms is an effective way for the no free lunch theorem dilemma [22]. Note that meta-heuristics only focus on the algorithm itself and ignore the characteristics of the optimization problems. The fitness landscape analysis techniques provide an effective way to understand the features of problems. Although the fitness landscape is often adopted to determine

* Corresponding author.
E-mail address: Fzhao2000@hotmail.com (F. Zhao).

https://doi.org/10.1016/j.swevo.2022.101148
Received 22 December 2021; Received in revised form 21 July 2022; Accepted 2 August 2022
Available online 3 August 2022

the complexity of an optimization problem, there are very few works that utilize the fitness landscape for designing algorithms [23].

Machine learning has become a powerful tool for addressing problems in various fields. Although meta-heuristics and machine learning techniques are developed for different purposes, they are adopted to perform common tasks such as feature selection or solving optimization problems [24]. Meta-heuristics and machine learning techniques frequently interact to improve their search or learning capabilities [25]. The machine learning community uses meta-learning for the automatic selection of algorithms. Meta-learning aims to understand the relationship between the problem features and the performance of an algorithm. Considering the learning style of meta-learning, the process of algorithm selection is classified into two types, including online and offline learning [26]. The offline learning aims to construct the problem-algorithm mapping based on a set of training samples, and predicts the problem-algorithm mapping for unseen data samples. The advantage of offline learning is the low computational cost once the training process is completed [27]. There are few works presented in literature that apply meta-learning to complex continuous optimization problems.

In order to effectively address the aforementioned problems in the complex real-valued optimization problems, a co-evolutionary algorithm that integrates EDA and DE as the primary search strategies, based on the fitness landscape analysis is proposed in this work. The EDA and DE are embedded in an offline learning co-evolutionary algorithm (OLCA) to adapt various optimization problems with different properties. The main contributions of this work are presented below.

(1) An OLCA that introduces EDA and DE based on the fitness landscape is proposed to solve complex continuous optimization problems.
(2) The fitness landscape analysis technique is embedded in the proposed OLCA to extract the structural information of the problems and to guide the prediction of the random forest.
(3) A random forest is introduced to establish the problem-algorithm mapping based on offline learning. The search behavior of OLCA is analyzed with a visualized methodology to intuitively observe the execution process of the algorithm.

The rest of this paper is organized as follows. Section 2 presents the literature review, including the improvement history of EDA and DE, fitness landscape analysis techniques, and offline learning for optimization problems. Section 3 illustrates the framework of proposed OLCA and the training of a random forest based on the fitness landscape. Section 4 reports the experimental results and analysis. Finally, the paper is summarized in Section 5 and the future work is discussed.

2. Related works

The estimation of distribution algorithm (EDA) as an evolutionary algorithm was first proposed by H.Mühlenbein and G.Paaβ in 1996 [28]. EDA is characterized by the way of generating offspring, i.e., sampling solutions based on a probability distribution instead of through crossover and mutation operators as generated in other kinds of evolutionary algorithms. During the past few decades, EDAs have been applied widely in both combinatorial and continuous domains [28]. There are various models that have been introduced in EDAs, such as the Gaussian model, Cauchy model, and histogram model, to describe the distribution of high-quality solutions. The Gaussian EDAs (GEDA) are generally classified into three categories based on the variable dependencies, i.e., (1) univariate EDA that assumes that all the variables are independent [30], (2) bivariate EDA that only considers some pairwise variable interactions [31], and (3) multivariate EDA that considers the interactions among multiple variables [32].

Although GEDAs possess a huge potential for solving complex optimization problems, they often suffer from premature convergence. This defect is attributed to the rapid shrinking in the variances. Various improvements have been performed to address this defect. Ocenasek et al. [33] proposed a new evolution strategy that combines the mixed Bayesian optimization algorithm and variance adaption. Grahl et al. [34] proposed an adaptive variance scaling (AVS) strategy, which tunes the variances when it identifies that the algorithm is traversing a slope. The authors developed two identification strategies, i.e., the strategies based on correlation triggering and standard deviation ratio (SDR). Few researchers achieved variance tuning by modifying the eigenvalues of the estimated covariance matrix instead of performing variance scaling directly [35,36].

Note that the performance of GEDA not only depends on its search scope but also on the direction of search. There are few works that show that the main search direction of GEDA tends to become orthogonal to the improvement direction of the fitness functions [37]. In [38], a powerful variant of EDA variant known as AMaLGaM is developed based on the anticipated mean shift (AMS) technique. It estimates the covariance matrix by shifting part of the selected solutions along the anticipated gradient direction, such that the main search direction is improved to a certain extent. Liang et al. [39] proposed a variant of EDA called EDA^2. In this work, a historical memory mechanism, which conserves a certain number of populations generated in the former generations, is developed to estimate the covariance matrix of the Gaussian model. This mechanism contributes to the EDA for discovering a more promising search space. The covariance matrix adaption evolution strategy (CMA-ES) [40] is a special EDA that employs the rank-μ-update operator to increase the variance along the direction of gradient.

Additionally, extensive efforts have been made to improve the efficiency of EDA. Dong et al. [41] presented a novel framework based on EDA and model complexity control (EDA-MCC). This framework for the first time designed an EDA based on a multivariate model that can be applied to 500D problems. In [40], a weakly dependent variable identification and subspace modeling technique are employed to solve high dimensional problems. In addition, this method also provides the structural information of the problems and saves the computational resources. Zhao et al. [42] developed a new hybrid optimization algorithm that combines the EDA with a differential evolution based on chaos theory. In cDE/EDA, the chaos strategy was integrated in the differential evolution to strengthen the search ability of the DE. In cDE/EDA, the value of F, CR, and γ (the decisive factor) was updated during the search process based on the chaotic operation. The parameter γ determines the proportion of individuals that perform the mutation operation. Ren et al. [43] proposed a new EDA variant named AAVS-EDA that adopts a novel anisotropic adaptive variance scaling (AAVS) technique to strengthen the performance of the traditional EDA. The AAVS-EDA not only adjusts the search scope of the EDA but also considers the improvement of main search directions. Furthermore, an auxiliary global monitor was introduced to assist the AAVS-EDA to converge to a promising area based on a shrinking variable variance if no improvement is achieved in a generation. Tang et al. [44] integrated Kalman filtering to revise the data generated by the Gaussian model to construct an accurate model. Furthermore, a learning strategy is proposed to improve the sampling operation in accordance with the sampling outcomes.

The canonical DE algorithm consists of four basic steps, including initialization, mutation, crossover, and selection. Recently, various variants of DE are developed to resolve continuous optimization and combinatorial optimization problems. DE has emerged as one of the most powerful and versatile evolutionary optimizers [45]. Although, it is difficult to classify the DE methods under a well-defined taxonomy since some of the improvements combine the multiple mechanisms together, these approaches can be roughly classified into two categories, including DE methods with strategies and control parameters adaptions and hybrid DE algorithms.

The self-adaptive differential evolution algorithm (SaDE) is proposed to address the issue of basic DE being intensely dependent on the pa-

rameters [46]. The authors present a learning strategy to choose a suitable mutation strategy among "rand/1/bin" and "current to best/2/bin" based on a probability formulation. Moreover, the crossover rate *CR* and scale factor *F* of the control parameters are adapted based on the success histories of the generated trial vectors. Wu et al. [47] proposed a novel variant of DE with multi-population-based ensemble of mutation strategies (MPEDE). In MPEDE, three mutation strategies, including "current-to-pbest/1″, "current-to-rand/1″, and "rand/1″, are executed in each generation. The amount of population is regarded as a resource for rewarding the winner mutation strategy. Zhang and Sanderson [48] presented an adaptive DE with an optional external archive called JADE. In JADE, a new mutation strategy denoted "DE/current--to-pbest" with an external archive is implemented to guide the direction of the search. Furthermore, the update of the *F* and *CR* is performed based on a self-adaptive approach. Tanabe and Fukunaga [49] presented an improved JADE named success-history-based adaptive DE (SHADE), which updates the parameters based on the historical memory of successful parameters settings. The SHADE not only enhances the robustness of JADE but also leads to the future parameter selection. The LSHADE that adopted a linear population size reduction (LPSR) method is proposed to improve the SHADE [50]. The LPSR reduces the population size in each generation according to a linear function and alleviates the wastage of computation resources.

The blending of DE and other algorithms is an effective way to enhance the performance of an algorithm [45]. Elsayed et al. [51] hybridized differential operators of DE with real code GA to solve the problems presented in CEC 2011 competition. The hybrid technique performed the best on the competition problems. Boussaïd et al. [52] proposed a synergy between DE and biogeography-based optimization (BBO) for implementing optimal power allocation in wireless sensor networks. Li et al. [53] proposed a new hybrid algorithm based on the DE framework that integrates the key features of CMA-ES. In this algorithm, a trial vector is generated by using a DE/rand/1/bin strategy followed by an evolution path mutation of CMA-ES. Rakshit et al. [54] proposed an adaptive memetic search algorithm that uses a hybrid DE and local reinforcement learning based refiner to solve the multi-robot path planning problem. Poikolainen and Neri [55] proposed a DE variant, which combined the regular DE and Hooke-Jeeves method based local search. In this algorithm, the number of solutions *Np* is selected uniformly from the search space to refine and form the initial population. Then, the offspring for the next iteration is generated by using a DE mutation strategy.

The concept of the fitness landscape was proposed by Wright in 1932 to provide an intuitive picture of the evolution process of an algorithm [56]. Recently, the research on the fitness landscapes has developed rapidly not only in theory but also in the applications of optimization and machine learning. In [57], the authors reviewed the study of fitness landscape analysis techniques based on the previous works and summarized 33 methods. In addition, the random walk algorithms are often applied in fitness landscape analysis techniques to capture the structural features of space [58]. In [59], Malan et al. reviewed the random walks in general and random walks in fitness landscapes. They proposed a progressive random walk algorithm that used the multiple walks to sample neighborhood structure in continuous multi-dimensional spaces.

The fitness landscape analysis techniques are roughly classified into four categories. First, the techniques related to the modality and global structure. The local optima network (LON) is proposed by Ochoa et al. to represent the global structure of the search space [60]. The relationship between the characters of attraction basin and LON is investigated in [59]. The search trajectory network (STN) is presented by Malan et al. to visualize the behavior of different meta-heuristics [61]. The STN is established based on the data obtained from the process of the algorithm running without extra sampling. Second, the techniques that consider the ruggedness and neutrality. Malan and Engelbrecht proposed the ruggedness of information entropy (RIE) to quantify the ruggedness of continuous landscapes [62]. Vanneschi et al. proposed the measure on neutral networks to analyze the characteristics of genetic programming Boolean landscape. Third, techniques that consider the level of searchability. The exploratory landscape analysis (ELA) of continuous search space is proposed by Mersmann et al. [63]. The six low-level features, including convexity, *y*-distribution, levelset, meta-model, local search, and curvature are defined by a sample of random solutions. The test results obtained on the BBOB'09/10 contest illustrate the successful prediction of predefined function groups. The local multiobjective landscape feature is proposed by Liefooghe et al. to solve multi-objective combinatorial optimization problems [64]. This method provides an effective way to evaluate the influence of problem features on the performance of the algorithms. In addition, the authors also study the importance of ruggedness and multimodality to describe the landscape of multi-objective combinatorial optimization problems. Lastly, the techniques that consider quantifying epistasis and deception. In [65], the maximum entropic epistasis (MEE) is proposed to quantify the interactions between the variables. The experimental results obtained by using 24 multi-dimensional continuous optimization functions show the robustness of this method. The bit-wise epistasis is proposed by Fonlup et al. to show the dependence between variables [66]. In [67], the epistasis variance is proposed as an estimate of the amount of non-linearity in the function. The epistasis variance is calculated based on a linear composition of a string solution from its bits. Despite a large number of techniques that are developed to analyze the characteristics of the fitness landscape, there are few research works that have applied the fitness landscape analysis technique for enhancing the performance of the algorithm.

Note that the offline learning is often embedded in meta-heuristics to solve the combinatorial optimization problems, such as flow-shop scheduling problem (FSP) [68–71], traveling salesman problem (TSP) [72], and job-shop scheduling problem (JSP) [73]. In [74], the fitness landscape features, and random forest are utilized for meta-learning and selecting a corresponding algorithm for different quadratic assignment problem (QAP) instances. In [75], an algorithm selection model, based on linear regression is proposed to address the timetabling problem (TTP). In [76], a multilayer perceptron classifier is combined with a wrapper meta-feature selection method to predict a suitable meta-heuristic for given vehicle routing problem (VRP) instances. A data-mining approach combined with GA and PSO is proposed in [77] to address the JSP. In [73], the apriori technique is employed to extract the rules behind the optimal schedules of JSP. In [78], supervised learning is adopted to solve the two-stage stochastic integer programming problems. In [74], artificial neural network and linear regression are utilized to minimize the mean square error between the true and predicted scenarios. In [79], the data-mining-based approach is used to solve the VRP. The authors extract the common characteristics of good solutions and used them for generating the neighbors. The offline learning combined with meta-heuristics can be performed effectively on combinatorial optimization problems. However, there are few research works that have applied meta-heuristics on complex continuous optimization problems.

The aforementioned works show that the EDA and DE have been investigated deeply in the past few years. However, the difference and relationship between EDA and DE are not considered by these works. Instead, most algorithms only focus on the algorithm itself and ignore the characteristics of the problem, thus limiting the performance of these algorithms. The combination of machine learning and evolutionary algorithms provides a suitable way of improving the performance of an algorithm. In addition, the application of the fitness landscape analysis is also helpful in designing the algorithms and for improving the performance of these algorithms. In order to enhance the adaptation of the algorithm to different problems, this work proposes a framework embedded with the fitness landscape analysis and random forest.

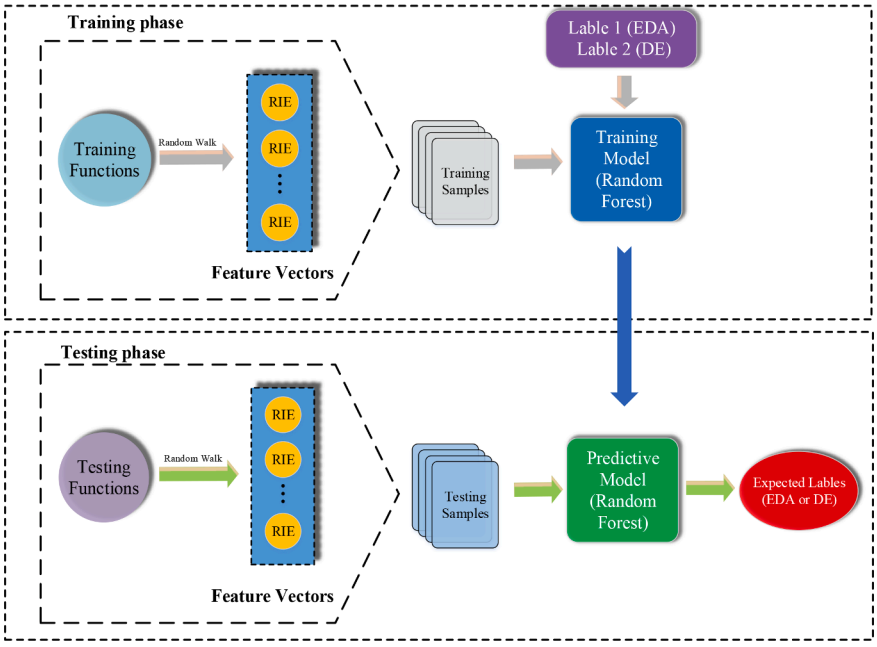


Fig. 1. The framework of OLCA.

3. Offline learning co-evolutionary algorithm

In this section, the proposed OLCA is presented in detail. First, the algorithm portfolio containing improved GEDA and LSHADE is presented and explained in Section 3.1. Then, the implementation process of the proposed OLCA is described step by step. The general framework of the proposed OLCA is provided in Fig. 1. The pseudo-code of the proposed method is presented in Algorithm 3.

3.1. Algorithm portfolio

3.1.1. Improved GEDA

Inspired by meta-learning and no free lunch theorem, the proposed OLCA consists of two algorithms, i.e., the GEDA and the LSHADE, instead of a single algorithm for solving complex optimization problems.

Generally, EDA consists of four phases including selection, modeling, sampling, and combination. First of all, the truncation selection method is utilized in EDA to obtain a set of good individuals from the total population. The solution space is described comprehensively by the selected individuals. In the second phase, a probabilistic model is constructed based on the information extracted from the selected individuals. After modeling is completed, a set of individuals is sampled from the constructed model. In every generation, the modelled individuals and the sampled individuals are combined to obtain the offspring population.

Generally, the Gaussian model is adopted to describe the solution space in continuous EDAs [29,34,38,40,42]. The probability density function of the Gaussian model is expressed as follows:

$$G_{(\mu,\ C)}(x) = \frac{1}{(2\pi)^{\frac{n}{2}}(detC)^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T(C)^{-1}(x-\mu)} \tag{1}$$

where, x is an n-dimensional random vector, μ is the mean of x, and C is the covariance matrix of x. The new μ and C for the next generation are typically estimated by using the following maximum-likelihood (ML) method based on the selected solutions in the current generation:

$$\mu^t = \frac{1}{|S^t|}\sum_{i=1}^{|S^t|} S_i^t \tag{2}$$

$$C^t = \frac{1}{|S^t|}\sum_{i=1}^{|S^t|}\left(S_i^t - \mu^t\right)\left(S_i^t - \mu^t\right)^T \tag{3}$$

where, $|S^t|$ represents the number of selected individuals, S_i^t denotes the ith solution in the selected solution set S^t, and the mean value of S^t is represented by μ^t.

There are few works presented in literature that show the employment of ML methods for estimating the Gaussian model makes the main search direction of the EDA orthogonal to the fitness improvement direction on slope-like regions [37]. This issue restricts EDA to discover a promising search area. Thus, a weighted average of candidate solutions is developed in our work to estimate the mean based on the fitness of the candidate solutions. The mean is mathematically expressed as follows:

Algorithm 1
Linear Search Strategy.

Algorithm 1 Linear Search Strategy

Input: μ^t, δ_1, δ_2 and η_{max}
Output: $\hat{\mu}^t$
$\hat{\mu}^t = \mu^t, \eta = 0$
While $f\left(\mu^t + \eta(\delta_1 + \delta_2)\right) < f(\mu^t)$ && $\eta < \eta_{max}$
    $\hat{\mu}^t = \mu^t + \eta(\delta_1 + \delta_2)$
    $\eta = \eta + 1$
end

$$\mu^t = \frac{\sum_{i=1}^{|S^t|}(FITNESS - fitness(i))S_i^t}{\sum_{i=1}^{|S^t|}(FITNESS - fitness(i))} \tag{4}$$

$$FITNESS = \sum_{i=1}^{|S^t|} fitness(i) \tag{5}$$

where, $|S^t|$ is the number of selected individuals and S_i^t denotes the ith solution in the candidate solutions.

Line search is a simple algorithm that is introduced in various research to enhance or improve the performance of the algorithm [28, 80–82]. In this paper, a linear search method presented in Algorithm 1 is designed to accelerate the search process as follows:

$$\delta_1 = \mu^t - \mu_0 \tag{6}$$

$$\delta_2 = \mu^t - \mu_1 \tag{7}$$

$$\widehat{\mu}^t = \begin{cases} \mu^t + \eta(\delta_1 + \delta_2) \ if \ f(\mu^t + \eta(\delta_1 + \delta_2)) < f(\mu^t) \\ \mu^t \ otherwise \end{cases} \tag{8}$$

where, $\widehat{\mu}^t$ denotes the shifted mean in the tth generation. The $\widehat{\mu}^t$ is used to generate the offspring when the estimation process is finished. δ_1 represents the difference between μ^t and μ_0 where μ^t is the mean of the candidate solutions and μ_0 is the mean of the total population. δ_2 represents the difference between μ^t and μ_1, where μ_1 represents the mean of the individuals that are not selected from the population. η is a shifting factor that is greater than or equal to 0. The purpose of this operation is to balance the exploration and exploitation of the algorithm and to avoid premature convergence. η_{max} is set to 10 in this work.

3.1.2. DE with the self-adaptive mechanisms

The differential evolution is a simple but effective algorithm for addressing various optimization problems and is widely used in various works [45]. The main steps of differential evolution comprise mutation, combination, and selection. DE is an evolutionary algorithm based on population. In this work, a variant of DE namely LSHADE [50], which is coevolution with GEDA, is adopted in the proposed OLCA to deal with complex optimization problems. The population in DE is expressed as follows:

$$X_i^g = \left(x_{i,1}^g, x_{i,2}^g, x_{i,3}^g \ldots x_{i,d}^g \ldots x_{i,D}^g\right) \tag{9}$$

where, g is the generation and $x_{i,d}^g$ is the dth variable of ith individuals in generation g. Note that each variable is in the range of domain space.

$$x_{min} \le x_{i,d}^g \le x_{max} \tag{10}$$

The type of distribution used in initialization of population is UNIFORM. The initial population of DE is expressed as:

$$x_{i,d}^g = x_{min} + r * (x_{max} - x_{min}) \tag{11}$$

where, r denotes a uniformly selected random number ranging from $[0, 1]$. After initialization, the mutation operator is applied on each target vector x_i^g to obtain the mutant vectors. The mutant population is denoted as $V_i^g = \{v_i^g | i = 1 \ldots NP\}$. In LSHADE, the mutation strategy $DE/current - to - pbest/1$ is used to generate the mutant vectors, and is expressed as follows:

$$DE/current - to - pbest/1 : v_i^g = x_i^g + F * \left(x_{pbest}^g - x_i^g\right) + F * (x_{r1}^g - x_{r2}^g) \tag{12}$$

Where, $r1$ and $r2$ are random integers in the range of $[1, NP]$, NP is the population size and F is the scale factor preset by the users. The crossover operator is exerted on the mutant vector consequently to produce the trial population. This process is mathematically expressed as follows:

$$u_{i,j}^g = \begin{cases} v_{i,j}^g \ if \ j = j_r \ or \ cr \le CR \\ x_{i,j}^g \ otherwise \end{cases} j = 1, \ldots, D \tag{13}$$

where, j_r is a random integer ranging from the $[1, D]$, cr is a uniformly distributed random number ranging from $[0, 1]$, and $CR \in [0, 1]$ is the crossover rate. A selection operator is required to determine the individuals that survive in this generation and pass to the one. The offspring population is generated by using the greedy selection from the trial population and current population based on the fitness values. Because of the selection operator, the generated offspring does not become inferior. The selection operator is expressed as follows:

$$X_i^g = \begin{cases} u_i^g \ if \ f(u_i^g) \le f(x_i^g) \\ x_i^g \ otherwise \end{cases} i = 1, \ldots, NP \tag{14}$$

The differential evolution is significantly influenced by the parameters, such as scale factor F and crossover rate CR. Therefore, the mechanism for linear population size reduction and parameter adaption based on success history are also introduced to avoid the tunning parameters by users in LSHADE. Due to the loss of diversity in the population during later iterations, the linear population reduction is an effective way to save the computation resources. The mathematical expression of this method is presented as follows:

$$N^{g+1} = round\left(\left(\frac{N_{min} - N_{init}}{Max\_Nfes}\right) * nfes + N_{init}\right) \tag{15}$$

where, N_{init} and N_{min} represent the size of the initial population and minimum size of the population respectively. $nfes$ and Max_Nfes are the numbers of fitness evaluations used so far and evaluations of maximum available, respectively. In addition, the parameters are selected adaptively by using the following expressions:

$$CR_i^g = randn\left(M_{CR,r_i}, 0.1\right) \tag{16}$$

$$F_i^g = randc\left(M_{F,r_i}, 0.1\right) \tag{17}$$

where, $randn$ and $randc$ are the random numbers that obey the Gaussian distribution and Cauchy distribution, respectively, and the standard deviations of both distributions are all set to 0.1. M_{CR} and M_F represent the parameters of length H stored in the historical memory archive that performed efficiently in the past, and the initial values of M_{CR} and $M_F$are 0.5. For each individual, the parameters F and CR are selected from the memory archive and r_i is the index that ranges from $[1, H]$. In each generation, if a trial individual u_i^g survives to the next generation, the corresponding F and CR are regarded as successful and denoted as S_F and S_{CR}, separately. The memory update at the end of generation is expressed as follows:

$$M_{CR,k}^{g+1} = \begin{cases} mean_{WA}(S_{CR}) \ if \ S_{CR} \ne \varnothing \\ M_{CR,k}^g, \ otherwise \end{cases} \tag{18}$$

where, $mean_{WA}(S_{CR})$ is the weighted arithmetic mean generated by using the following expression:

$$mean_{WA}(S_{CR}) = \sum_{k=1}^{|S_{CR}|} \omega_k * S_{CR,k} \tag{19}$$

ω_k is computed as:

$$\omega_k = \frac{\Delta f_k}{\sum_{k=1}^{|S_{CR}|} \Delta f_k} \tag{20}$$

$$\Delta f_k = |f(u_k^g) - f(x_k^g)| \tag{21}$$

The update of F memory is performed as follows:

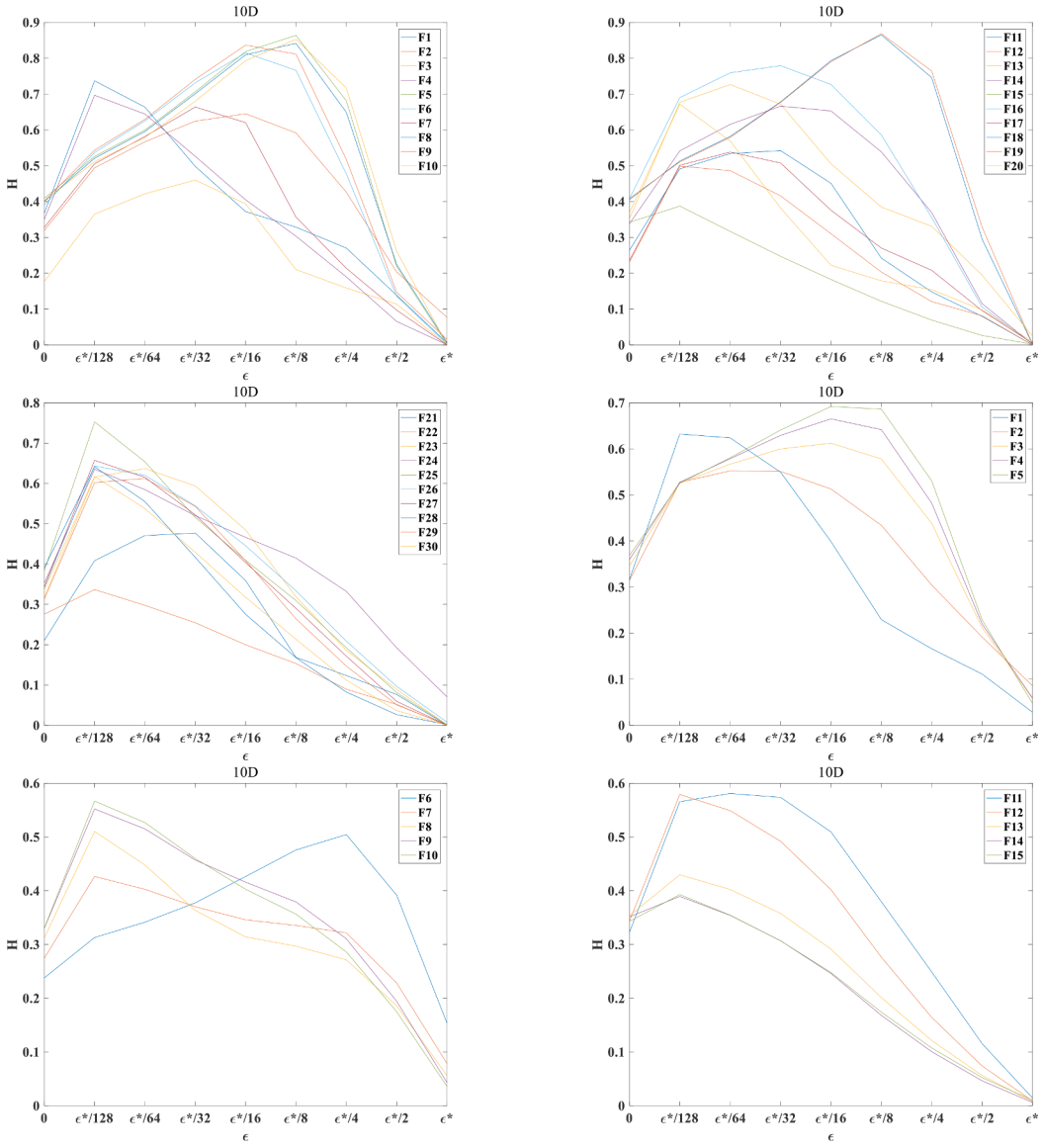


Fig. 2. RIE values of different functions on 10D.

$$M_{F,k}^{g+1} = \begin{cases} mean_{WL}(S_F) \ if \ S_F \neq \varnothing \\ M_{F,k}^{g} \ otherwise \end{cases} \tag{22}$$

where, $mean_{WL}(S_F)$ denotes the Lehmer mean and is computed as follows:

$$mean_{WL}(S_F) = \frac{\sum_{k=1}^{|S_F|} \omega_k * S_{F,k}^2}{\sum_{k=1}^{|S_F|} \omega_k * S_{F,k}} \tag{23}$$

where, k is the index that determines the position of the parameters in the historical memory.

3.2. The implementation of the proposed OLCA

3.2.1. The framework of OLCA

The framework of OLCA comprises two stages, i.e., the training phase and the testing phase. A random forest [83] is used for training and testing. It is a critical component of OLCA, which has learning and predicting functions. The random forest, which is introduced by Breiman in 2001 for the first time [84], is a collection of classification and regression trees [85]. It is a simple model that using binary splits on predictor variables to determine outcome predictions. In a random forest, many classification and regression trees are built by utilizing randomly selected training datasets and random subsets of predictor variables to model outcomes. The final results of random forest are determined by aggregating the results of each decision tree and a prediction for each observation is given. The version of random forest used in this work is C4.5, which is an extended form of ID3 [86]. The details of C4.5 is described in Section 3.2.4.

During the training phase, a random walk is used to obtain various landscape points in the domain space. Subsequently, the ruggedness of information entropy is calculated by using these landscape points. GEDA and LSHADE are used as the training functions to determine the training label, i.e., 1 or 2, for a certain function according to the final results of the two algorithms. Then, the landscape features and training labels are used to establish the problem-algorithm mapping and to form the complete training set. The training process of training is continued until all the decision trees in the random forest are formed.

During the testing phase, a random walk is first applied to the testing functions to obtain new landscape points in the domain space. Then, the ruggedness of information entropy is calculated to form the testing set. The prediction labels are obtained by using the trained random forest

Algorithm 2
Random Increasing Walk.

Algorithm 2 Random Increasing Walk

Input: $Dimensions$, $Domain$ ($[-100,100]$), $Walk\ Steps = 3000$, $StepSize = 10$
Output: A walk sequence
Initialize an empty array for storing the walk
Produce a random position for the walk ($walk[0]$)
$count = 0$
while $count < Walk\ Steps$
  for $i = 1 : Dimensions$
    Generate a random number $step$ in the range $[0, Step\ Size]$
    Set $walk[count]_i = walk[count]_{i-1} + step$
    if $walk[count]_i$ beyond the maximum bound
      Set $walk[count]_i = walk[count]_i - 200$
      $count = count + 1$
    end
  end
end

model.

3.2.2. Training samples and testing instances

In order to avoid overfitting, 45 benchmark functions containing 30 CEC 2014 [87] functions and 15 CEC 2015 functions [88] are used to train the random forest, and 30 CEC 2017 [89] functions are used to verify the generalization performance of the random forest model. The CEC benchmark functions are composed of unimodal functions, simple multimodal functions, hybrid functions, and composition functions. These functions have different characteristics and are used to calculate the fitness landscape. They are used for training the random forest.

3.2.3. Fitness landscape evaluation

It is difficult to compute a global fitness landscape. Usually, the local fitness landscape is calculated by sampling or random walk in the domain space. The ruggedness of information entropy (RIE) adopted in this work is related to the number and distribution of the local optima [62].

The random walk method is used to obtain the fitness features RIE. In this work, the population size of each training function is set to $10 \times D$, and the walk steps and step size are set to 3000 and 10, respectively. Furthermore, the search domain space is $[-100, 100]^D$ and the number of evaluations is set to $10^4 D$, where D is the dimension of the problem. The method is trained for 30 times independently and the mean value of 30 results are reserved. The ruggedness of information entropy values for 45 training functions are presented in Fig. 2. Although each function has a different curve, similarities exist in the RIE curve of functions that have the same characters. Similarities contribute to the classification of the training set. Therefore, a feature vector consisting of entropy values that comprise nine different values of ε values form a training sample. The ε is an important parameter that determines the sensitivity of $H(\varepsilon)$ for different landscapes, where H is the entropy. The random walk strategy used in this work is shown in Algorithm 2. The RIE values for each 10D training function are shown in Fig. 2. The landscape feature vector described in (24) is produced by the ruggedness of information entropy.

$$Feature\ Vector = \left\{ H(0), H\left(\frac{\varepsilon^*}{128}\right), H\left(\frac{\varepsilon^*}{64}\right), H\left(\frac{\varepsilon^*}{32}\right), H\left(\frac{\varepsilon^*}{16}\right), H\left(\frac{\varepsilon^*}{8}\right), H\left(\frac{\varepsilon^*}{4}\right), H\left(\frac{\varepsilon^*}{2}\right), H(\varepsilon^*) \right\} \tag{24}$$

3.2.4. Training and testing on benchmark

In the proposed OLCA, a random forest is adopted to construct the problem-algorithm mapping on training functions and predict the problem-algorithm mapping for new problem instances. In this work, a C4.5, decision tree that is appropriate for performing classification of continuous features is adopted to construct the random forest. As aforementioned, C4.5 is an improvement version of ID3 algorithm [90]. In C4.5, the information Gain ratio criterion is adopted for decision tree

Table 1
The optimal strategy for training functions.

CEC 2014	10D	30D	50D	100D	CEC 2014	10D	30D	50D	100D	CEC 2015	10D	30D	50D	100D
f_1	1,2	1,2	1	1	f_{16}	2	2	2	2	f_1	1,2	1,2	1	1
f_2	1,2	1,2	1,2	2	f_{17}	2	1	1	1	f_2	1,2	1,2	1,2	2
f_3	1,2	1,2	1,2	1,2	f_{18}	1	1	1	1	f_3	2	2	2	2
f_4	2	2	2	1,2	f_{19}	2	1	2	2	f_4	1	1	1	1
f_5	1	2	2	2	f_{20}	2	1	1	1	f_5	1	1	1	1
f_6	1,2	1,2	1	1	f_{21}	2	1	1	1	f_6	1	1	1	1
f_7	1	1,2	1,2	1,2	f_{22}	2	2	1	1	f_7	2	1	2	1
f_8	2	2	2	2	f_{23}	1,2	1,2	1,2	1,2	f_8	2	1	1	1
f_9	1	1	1	1	f_{24}	1	1	1	1	f_9	1,2	1	2	1,2
f_{10}	2	2	2	2	f_{25}	2	1,2	2	2	f_{10}	1,2	1	1	1
f_{11}	1	1	1	1	f_{26}	1,2	1,2	1,2	1,2	f_{11}	2	1	2	1
f_{12}	2	2	2	2	f_{27}	2	1,2	2	1	f_{12}	1	1	2	1,2
f_{13}	1	1	1	1	f_{28}	2	1	2	1	f_{13}	2	2	2	2
f_{14}	2	2	2	2	f_{29}	1,2	1	1	1	f_{14}	2	1	1	2
f_{15}	2	2	1	1	f_{30}	2	1	2	1	f_{15}	1,2	1,2	1,2	1,2

Algorithm 3
OLCA.

Algorithm 3 OLCA

Set $NP_{EDA} = 4000$, $\tau = 0.3$; $M_{CR} = M_F = 0.5$, $Archive = \varphi$, $g = 1$, $NP_{DE}^{max} = 180 \times D$; $steps = 3000$; $size = 10$
Generate the initial population $\boldsymbol{X} = [\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3} \ldots \boldsymbol{x_i} \ldots \boldsymbol{x_D}]$
$Gbest$ is used to store the best results
Perform a random walk to obtain points in the domain space according to **Algorithm 2**
Calculate the fitness landscape feature vector according to obtained points
Predict strategy by trained random forest and produce the prediction labels $\{\mathbf{1}, \mathbf{2}\}$
$nfes = 0$, max_$nfes = 10^4 * D$
While $nfes \leq max\_nfes$
  if $label = 1$
    Produce the best $\lfloor \tau \times NP \rfloor$ solutions by truncation selection
    Estimate the mean value by Eqs. (4) and (5)
    Perform **Algorithm 1**
    Estimate the covariance matrix by Eq. (3)
    Generate the new population by the Gaussian model
  else $(label = 2)$
    $S_{CR} = \varphi$; $S_F = \varphi$
    $r_i = Select\ from\ [1, H]\ randomly$
    $CR_i^g = randn(M_{CR,r_i}, 0.1)$; $F_i^g = randc(M_{F,r_i}, 0.1)$
    Generate the mutant vector by $DE/current\text{-}to\text{-}pbest/1$
    Generate the trial vector
    Perform the selection operator
    Update M_{CR} and M_F based on S_{CR}, S_F;
    Perform the LPSR [50]
  end
end
$nfes = nfes + NP$
return $Gbest$

classification instead of Gain split criterion [86]. Unlike random forest that builds a few thousand classification trees, C4.5 takes the training data and generates a single tree, and modifies the internal structure leading to a substantial reduction in computational resource and energy usage [91].

In this work, two algorithms are considered as strategies including GEDA and LSHADE to solve the optimization problems. These algorithms are trained on 45 benchmark functions for 51 times independently to identify a suitable algorithm for a certain benchmark problem. The mean value and standard variation are calculated on the training functions with dimensions of 10, 30, 50, and 100. The existence of a significant difference between the two algorithms on a function is identified by using the Friedman-test. As shown in Table 1, both algorithms might perform similarly, which is represented as "1,2″, on certain functions. Note that only one label (i.e. "1″ or "2″) is selected randomly for training. In Table 1, "1″ denotes that GEDA obtained better results as compared to LSHADE, and vice versa.

During the testing phase, the landscape features of testing functions are obtained by using the random walk before the execution of the algorithm. Subsequently, an algorithm is predicted for each testing function by classifying the landscape features using trained random forest.

4. Experimental results and analysis

4.1. Experimental environment and parameter settings

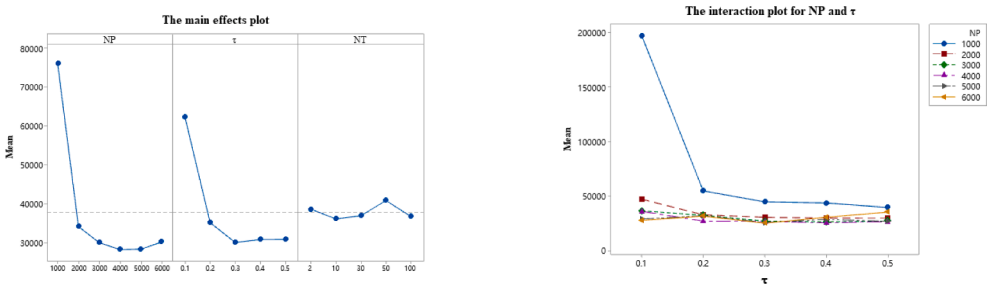
In order to verify the performance of the proposed OLCA, experiments are performed by using the CEC 2017 benchmark functions with different dimensions. The adopted benchmark test suite includes four types of functions including unimodal functions ($f_1 \sim f_3$), simple multimodal functions ($f_4 \sim f_{10}$), hybrid functions ($f_{11} \sim f_{20}$), and composition functions ($f_{21} \sim f_{30}$). It is difficult to find the global optima for the composition functions, which are characterized by multi-modal, non-separable, asymmetrical, processing different properties around different local optima, and different properties for different variable subcomponents. For the sake of fairness, all of the algorithms are executed for the same number of fitness evaluations (MaxFEs is set to 10000 $\times$ D, where D is the dimension) In Algorithm 3, note that the number of evaluations consumed in the calculation of RIE is subtracted from the total number of evaluations. The calculation of RIE can refer to [62]. In addition, we run the process 51 times independently on each function. The experiments are conducted on Microsoft Windows Server 2019 Standard 64-bit Opening System, 2.30 GHz CPU, and PC 64GB of RAM. The programming language is MATLAB 2016a. The source code of OLCA can be downloaded at https://github.com/OmertaZB/Algorithms.git.

Table 2
The ANOVA results of parameters.

Source	Sum of squares	Degrees of freedom	Mean Squares	F-ratio	$p-value$
NP	4.453e+10	5	8.907e+09	49.96	**0**
τ	2.291e+10	4	5.728e+09	32.13	**0**
NT	4.396e+08	4	1.099e+08	0.62	0.652
Interactions					
$NP * \tau$	7.181e+10	20	3.591e+09	20.14	**0**
$NP * NT$	3.134e+09	20	1.567e+08	0.88	0.613
$\tau * NT$	4.314e+09	16	2.697e+08	1.51	0.116
Error	1.426e+10	80	1.783e+08		
Total	1.614e+11	149			

We use few state-of-the-art algorithms for performing comparisons. These algorithms include CMA-ES (which is well known in the research community) [92], jSO (which performed well on CEC 2017) [93], LSHADE (which performed well on CEC 2014) [54], iL-SHADE [94], AAVS-EDA [57], PBILc [95], and cDE/EDA [96]. In addition, note that to verify the effectiveness of the random forest on OLCA, the random selection method is utilized to replace the random forest of OLCA namely RSCA is also included in the comparison algorithms.

In this work, the mean value (Mean), and the standard deviation (Std) metrics are adopted as the evaluation criterion. The search capability of the algorithms is reflected by the Mean and the stability of algorithms is represented by Std. The smaller mean or standard deviation represents a better performance. The Std is compared when the mean values of two compared algorithms are the same. The best results are represented in bold. Note that the error value is taken as zero if it is smaller than 10^{-8}.



(a) The Main effects plot. (b) The Interaction plot.

Fig. 3. Parameter analysis for the OLCA. (a) The Main effects plot. (b) The Interaction plot.

4.2. Parameters analysis

In this section, the design of the experiment (DOE) [97] is used to analyze the control parameters. The parameters of DE are adopted from the previous works because the parameters of this type of DE are determined by self-adaption and are executed without manual tuning. Therefore, the critical parameters of the proposed OLCA include population size (*NP*) and truncation rate τ of EDA, and the number of decision trees in the random forest (*NT*). The levels of the chosen parameters are listed as follows: $NP = \{1000, 2000, 3000, 4000, 5000, 6000\}$, $\tau = \{0.1, 0.2, 0.3, 0.4, 0.5\}$, and $NT = \{2, 10, 30, 50, 100\}$. Therefore, there are $6 \times 5 \times 5 = 150$ combinations of parameters. Each combination contains 30 functions and is executed for runs 30 times. The iterations continue till the termination criterion (exhaustion of maximum functional evaluations) is satisfied. The multifactor analysis of variance (ANOVA), which tests the significance of the difference between two or more samples, is adopted to investigate the experimental results.

As illustrated in Table 2, the two $p-values$ of the population size (*NP*) and τ are smaller than 0.05, which means that the adjustment of the two parameters is leads to a significant change in the algorithm with a 95% confidence level. Moreover, there is an interaction effect between *NP* and τ. The analysis of parameters for the OLCA, such as the figure of main effects and the interaction figure is shown in Fig. 3. As shown in Fig. 3(a), the best results are obtained when the population size (*NP*) is set to 4000, whereas the worst results are obtained when the *NP* is 1000, and neutral results are obtained for other settings. The small population size leads to inferior results due to poor diversity of the population. On the other hand, the wastage of computation resources in each iteration if the population size is too large. For the truncation rate τ, 0.3 is the best selection as shown in Fig. 3(a) and the results are worse, τ has a different value. The truncation rate is critical to the EDA and determines the quality and quantity of the selected solutions. The quality of the selected solutions is ensured if τ is small. However, the population diversity is lost, which may lead to the algorithm being prone to the local optima. Conversely, the search efficiency is lower if τ is too large. Although a large τ enables the algorithm to avoid the local optima, the accuracy of the model is reduced due to more inferior solutions contained in the selected solutions. The algorithm performs effectively if the parameters setting of *NP* and τ are 4000 and 0.3, respectively, as illustrated in the interaction figure. These results match with the results of the main effects plot. In Fig. 3(a), the optimal results are obtained when the number of decision trees is set to 10, which is not a key parameter according to the ANOVA. Therefore, the optimal combination of the control parameters is $NP = 4000$, $\tau = 0.3$, and $NT = 10$.

4.3. Performance comparison

In order to verify the performance of the proposed OLCA, the simulation are performed by using the CEC2017 benchmark functions. The comparison algorithms include CMA-ES, jSO, LSHADE, iL-SHADE, AAVS-EDA, PBILc, cDE/EDA, and RSCA. The experimental results on 100D are shown in Table 3 and the results for 10D, 30D, and 50D are present in Tables S4-S6 in the supplementary material. W/L/T, i.e. win/loss/tie, denotes that the other competitor performed better than, worse than, or equal compared with the proposed OLCA on the benchmark test suite.

The comparison results of 9 algorithms on 10D are shown in Table S4. For 3 unimodal functions, the OLCA converges to the global optimal similar to CMA-ES, jSO, LSHADE, iL-SAHDE, and RSCA. AAVS-EDA converges to the global value on f_2 and f_3. cDE/EDA converges to the global value on f_3. The performance of the proposed OLCA on all the unimodal functions is always better than or equal to the performance of the compared algorithms. The results show that the proposed OLCA has a strong local search capability. On multimodal functions $f_4 \sim f_{10}$, the proposed OLCA achieves the best performance. CMA-ES, jSO, LSHADE, iL-SHADE, and RSCA also converge to the global optimum on f_4. The performance of jSO, LSHADE, iL-SHADE, and PBILc on f_6 is as good as the performance of OLCA. PBILc also performs well on f_7. AAVS-EDA converges to the global optimum on f_8. On f_9, LSHADE, iL-SAHDE, cDE/EDA, RSCA, and OLCA performed well. The optimization results illustrate that the proposed OLCA performed better than the compared algorithms in terms of global search capability. The OLCA performed best on most of the hybrid functions except f_{12}, f_{13}, and f_{19}.

The iL-SHADE, jSO and RSCA also obtain the optimal solution on f_{11}. The performance of jSO on f_{12} is the best. AAVS-EDA performs better than the other compared algorithms on f_{13}. On f_{19}, the LSHADE has the best performance. On the composition functions $f_{21} \sim f_{30}$, the performance of the OLCA is still competitive. jSO performs best on f_{21}. All the algorithms obtain the same results on f_{22}. The proposed OLCA performs best on f_{23}.

On f_{24}, the CMA-ES performs much better as compared to the other algorithms. The performance of AAVS-EDA on f_{25} is better as compared to the others algorithms. On f_{26}, the CMA-ES also performs well. The cDE/EDA performs best for f_{27}. The AAVS-EDA obtains a better solution as compared to the other algorithms on f_{28}. The proposed OLCA performs well on f_{29}. The CMA-ES has the best performance on f_{30}. In short, the OLCA performs efficiently well as a whole and has a superior performance to other algorithms. Moreover, the proposed OLCA is competitive when compared with algorithms such as jSO and CMA-ES.

In order to further verify the performance of the proposed OLCA, it is compared with CMA-ES, jSO, LSHADE, iL-SAHDE, AAVS-EDA, PBILc, cDE/EDA, and RSCA on 30D, 50D, and 100D respectively. As illustrated in Tables S5 and S6, CMA-ES, jSO, LSHADE, RSCA, and OLCA perform well with 30D and 50D on the unimodal functions. Considering the problems with 100D, only CMA-ES is able to obtain the global optimum for all the unimodal functions. jSO and LSHADE perform well on f_1. RSCA and OLCA converge to the global optimum on f_1 and f_3. On the problems with 30D, the proposed OLCA performs well on the multimodal and hybrid functions as compared to the other methods. Although jSO, LSHADE, AAVS-EDA, and RSCA perform well on several functions among these two types of problems, their performance as a whole is worse than the proposed OLCA. On the problems with 50D and 100D,

Table 3
Comparison results on 100D benchmark functions.

Function		CMA-ES [81]	jSO [82]	LSHADE [49]	iL-SHADE [83]	AAVS_EDA [42]	PBILc [84]	cDE/EDA [85]	RSCA	OLCA
F1	Mean	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	9.37E+09	2.41E+07	6.05E+03	**0.00E+00**	**0.00E+00**
	Std.	0.00E+00	0.00E+00	0.00E+00	0.00E+00	4.39E+09	3.54E+07	7.99E+03	0.00E+00	0.00E+00
F2	Mean	**0.00E+00**	8.94E+00	4.23E+01	1.03E+02	1.00E+30	1.00E+30	8.62E+29	1.83E+03	2.04E+00
	Std.	1.03E-08	2.42E+01	1.43E+02	3.77E+02	2.84E+14	2.84E+14	3.21E+29	7.03E+00	6.19E+03
F3	Mean	**0.00E+00**	2.39E-06	1.23E-06	9.62E-07	5.65E+04	4.14E+05	6.44E+04	**0.00E+00**	**0.00E+00**
	Std.	0.00E+00	2.72E-06	3.17E-06	7.73E-07	1.01E+04	4.18E+04	1.61E+04	0.00E+00	0.00E+00
F4	Mean	**1.72E+01**	1.90E+02	1.98E+02	1.94E+02	7.20E+03	4.56E+02	1.55E+02	2.77E+02	1.96E+02
	Std.	5.09E-01	2.89E+01	7.98E+00	1.54E+01	1.42E+03	3.96E+01	4.59E+01	3.23E+01	9.09E+00
F5	Mean	3.43E+01	4.39E+01	3.86E+01	**1.70E+01**	2.20E+02	5.51E+01	1.94E+02	3.70E+01	3.62E+01
	Std.	4.52E+00	5.61E+00	4.80E+00	3.40E+00	3.64E+01	7.76E+00	3.24E+01	4.22E+00	5.30E+00
F6	Mean	2.14E+01	2.02E-04	7.05E-03	1.07E-03	2.08E+01	4.55E-05	5.07E-01	6.05E-03	**2.01E-06**
	Std.	2.81E+01	6.20E-04	5.08E-03	1.13E-03	3.52E+00	2.96E-04	2.22E-01	2.43E-07	3.18E-07
F7	Mean	1.32E+02	1.45E+02	1.40E+02	4.59E+02	4.49E+02	1.36E+02	3.23E+02	1.38E+02	**1.13E+02**
	Std.	3.77E+00	6.70E+00	4.65E+00	2.96E+01	5.25E+01	4.41E+00	3.74E+01	4.48E+00	2.23E+00
F8	Mean	3.46E+01	4.22E+01	3.77E+01	**1.69E+01**	2.45E+02	5.13E+01	1.96E+02	2.18E+01	3.77E+01
	Std.	5.48E+00	5.52E+00	4.52E+00	3.42E+00	4.06E+01	7.72E+00	3.36E+01	4.64E+00	4.69E+00
F9	Mean	2.21E+04	4.59E-02	4.93E-01	1.23E-02	4.10E+03	6.08E+01	6.01E+02	6.33E-01	**0.00E+00**
	Std.	7.98E+01	1.15E-01	4.12E-01	3.11E-02	9.65E+02	2.54E+01	3.88E+02	0.00E+00	3.78E-01
F10	Mean	9.50E+03	9.70E+03	1.04E+04	2.23E+04	7.78E+03	4.96E+03	1.40E+04	**1.33E+03**	1.41E+03
	Std.	1.31E+03	6.82E+02	4.69E+02	6.70E+02	1.13E+03	9.84E+02	1.36E+03	4.72E+02	5.44E+02
F11	Mean	1.13E+03	1.13E+02	4.69E+02	1.22E+02	1.03E+04	5.16E+04	7.04E+02	4.49E+02	**3.72E+01**
	Std.	2.82E+02	4.32E+01	1.19E+02	3.48E+01	4.21E+03	1.06E+04	2.59E+02	2.88E+01	2.36E+01
F12	Mean	5.33E+03	1.84E+04	2.29E+04	1.77E+04	2.39E+10	6.99E+07	4.11E+06	**3.16E+03**	**3.16E+03**
	Std.	8.81E+02	8.35E+03	7.37E+03	6.48E+03	4.46E+09	2.26E+07	2.32E+06	4.57E+02	7.58E+02
F13	Mean	3.90E+03	1.45E+02	4.15E+02	1.51E+02	1.16E+09	4.36E+03	4.50E+03	4.65E+02	**8.32E+01**
	Std.	9.32E+02	3.80E+01	1.87E+02	4.19E+01	7.10E+08	1.77E+03	3.82E+03	3.69E+01	3.59E+01
F14	Mean	3.26E+02	6.43E+01	2.49E+02	9.59E+01	2.34E+02	7.40E+06	2.13E+05	**2.64E+01**	2.67E+01
	Std.	6.83E+01	1.09E+01	3.28E+01	2.32E+01	5.45E+01	2.18E+06	1.85E+05	3.40E+00	2.41E+00
F15	Mean	4.44E+02	1.62E+02	2.47E+02	2.33E+02	1.12E+07	8.33E+02	2.27E+03	**5.79E+01**	2.31E+02
	Std.	1.19E+02	3.81E+01	4.47E+01	4.67E+01	3.79E+07	3.70E+02	1.91E+03	5.82E+01	4.05E+01
F16	Mean	**4.61E+02**	1.86E+03	1.63E+03	2.54E+03	1.47E+03	1.50E+03	3.10E+03	1.60E+03	1.75E+03
	Std.	2.31E+02	3.49E+02	2.93E+02	3.29E+02	4.16E+02	4.39E+02	4.92E+02	3.75E+02	3.57E+02
F17	Mean	1.31E+03	1.28E+03	1.16E+03	1.66E+03	1.15E+03	8.81E+02	2.64E+03	1.11E+02	**1.10E+02**
	Std.	4.27E+02	2.38E+02	1.86E+02	2.45E+02	3.47E+02	3.09E+02	4.66E+02	6.70E+01	6.46E+01
F18	Mean	2.16E+02	1.67E+02	2.32E+02	2.12E+02	6.87E+03	4.65E+06	5.12E+05	2.23E+02	**2.82E+01**
	Std.	5.05E+01	3.65E+01	6.27E+01	4.35E+01	6.73E+03	1.69E+06	2.54E+05	6.52E+01	2.48E+00
F19	Mean	3.73E+02	**1.05E+02**	1.80E+02	1.62E+02	7.59E+06	1.10E+03	2.98E+03	1.69E+02	1.68E+02
	Std.	9.89E+01	2.01E+01	2.64E+01	2.33E+01	4.56E+07	9.25E+02	3.96E+03	5.55E+00	2.21E+01
F20	Mean	3.89E+03	1.38E+03	1.53E+03	2.07E+03	7.58E+02	**7.41E+02**	2.33E+03	1.56E+03	1.55E+03
	Std.	1.84E+02	2.43E+02	2.21E+02	2.13E+02	1.97E+02	2.65E+02	4.10E+02	9.35E+00	2.18E+02
F21	Mean	2.58E+02	2.64E+02	2.60E+02	2.46E+02	4.44E+02	2.72E+02	4.35E+02	2.58E+02	**2.42E+02**
	Std.	5.82E+00	6.43E+00	4.30E+00	4.03E+00	4.33E+01	9.06E+00	3.05E+01	5.95E+00	6.25E+00
F22	Mean	**1.84E+03**	1.02E+04	1.13E+04	2.25E+04	3.80E+03	5.85E+03	1.45E+04	1.13E+04	1.14E+04
	Std.	8.83E+02	2.18E+03	5.27E+02	8.66E+02	3.25E+03	8.80E+02	1.33E+03	6.89E+02	5.29E+02
F23	Mean	5.66E+02	5.71E+02	5.70E+02	5.71E+02	7.48E+02	6.15E+02	7.69E+02	**5.58E+02**	5.69E+02
	Std.	1.11E+01	1.07E+01	8.75E+00	1.03E+01	6.50E+01	1.33E+01	3.73E+01	1.21E+01	1.00E+01
F24	Mean	8.95E+02	9.02E+02	9.09E+02	9.12E+02	1.49E+03	9.36E+02	1.12E+03	9.08E+02	**7.86E+02**
	Std.	7.15E+00	7.89E+00	8.46E+00	7.92E+00	4.05E+02	1.44E+01	4.83E+01	8.03E+00	2.22E+01
F25	Mean	7.33E+02	7.36E+02	7.35E+02	7.43E+02	2.85E+03	1.08E+03	7.69E+02	6.71E+02	**6.60E+02**
	Std.	3.47E+01	3.53E+01	3.91E+01	3.59E+01	4.72E+02	5.19E+01	7.30E+01	3.43E+01	3.29E+01
F26	Mean	3.23E+03	3.27E+03	3.30E+03	3.31E+03	2.52E+03	3.72E+03	6.09E+03	3.29E+03	**2.45E+03**
	Std.	1.49E+02	8.02E+01	7.37E+01	8.64E+01	1.08E+03	1.87E+02	4.75E+02	6.01E+01	2.18E+02
F27	Mean	**5.00E+02**	5.85E+02	6.29E+02	6.12E+02	6.34E+02	7.96E+02	**5.00E+02**	6.02E+02	5.49E+02
	Std.	3.87E-04	2.17E+01	1.85E+01	1.75E+01	3.25E+01	3.04E+01	2.67E-04	2.42E+01	7.75E+00
F28	Mean	**5.00E+02**	5.27E+02	5.24E+02	5.28E+02	4.19E+03	1.52E+03	5.02E+02	6.29E+02	5.36E+02
	Std.	6.22E-04	2.73E+01	2.64E+01	2.28E+01	6.59E+02	1.79E+02	1.03E+01	2.65E+01	3.39E+01
F29	Mean	1.61E+03	1.26E+03	1.22E+03	2.12E+03	2.22E+03	1.92E+03	2.67E+03	8.13E+02	**7.97E+02**
	Std.	3.14E+02	1.91E+02	1.58E+02	2.26E+02	3.28E+02	4.22E+02	6.79E+02	5.69E+01	8.79E+01
F30	Mean	**1.38E+03**	2.33E+03	2.44E+03	2.34E+03	1.10E+07	2.63E+05	4.09E+03	2.39E+03	2.41E+03
	Std.	3.71E+02	1.19E+02	1.53E+02	1.28E+02	2.41E+07	6.46E+05	4.57E+03	7.54E+01	1.73E+02
W/L/T	10/18/2	7/22/1	4/24/2	6/24/0	3/27/0	3/27/0	3/27/0	8/19/3	–	

the proposed OLCA is the most competitive on multimodal functions, hybrid functions, and composition functions. As illustrated by the mean and std values presented in Tables 3 and S4–S6, the statistical results show that the proposed OLCA obtains better performance on benchmark functions. No algorithm performs the best on all problems according to the no free lunch theorem. The proposed OLCA shows good performance on several problems and is competitive as compared with the other algorithms.

In summary, the proposed OLCA is competitive compared with its competitors which contain the state-of-the-art. This kind of performance mainly profits from the combination of the random forest and the fitness landscape analysis. The knowledge that is implied in the fitness landscape is utilized in the algorithm procedure. Therefore, the proposed OLCA can select the appropriate method to adapt to distinct problems.

4.4. Behavior analysis of the proposed OLCA

In this work, an experiment is conducted to analyze the search behavior and visualize the execution process of the proposed OLCA for two-dimensional benchmark functions. The execution details of the al-

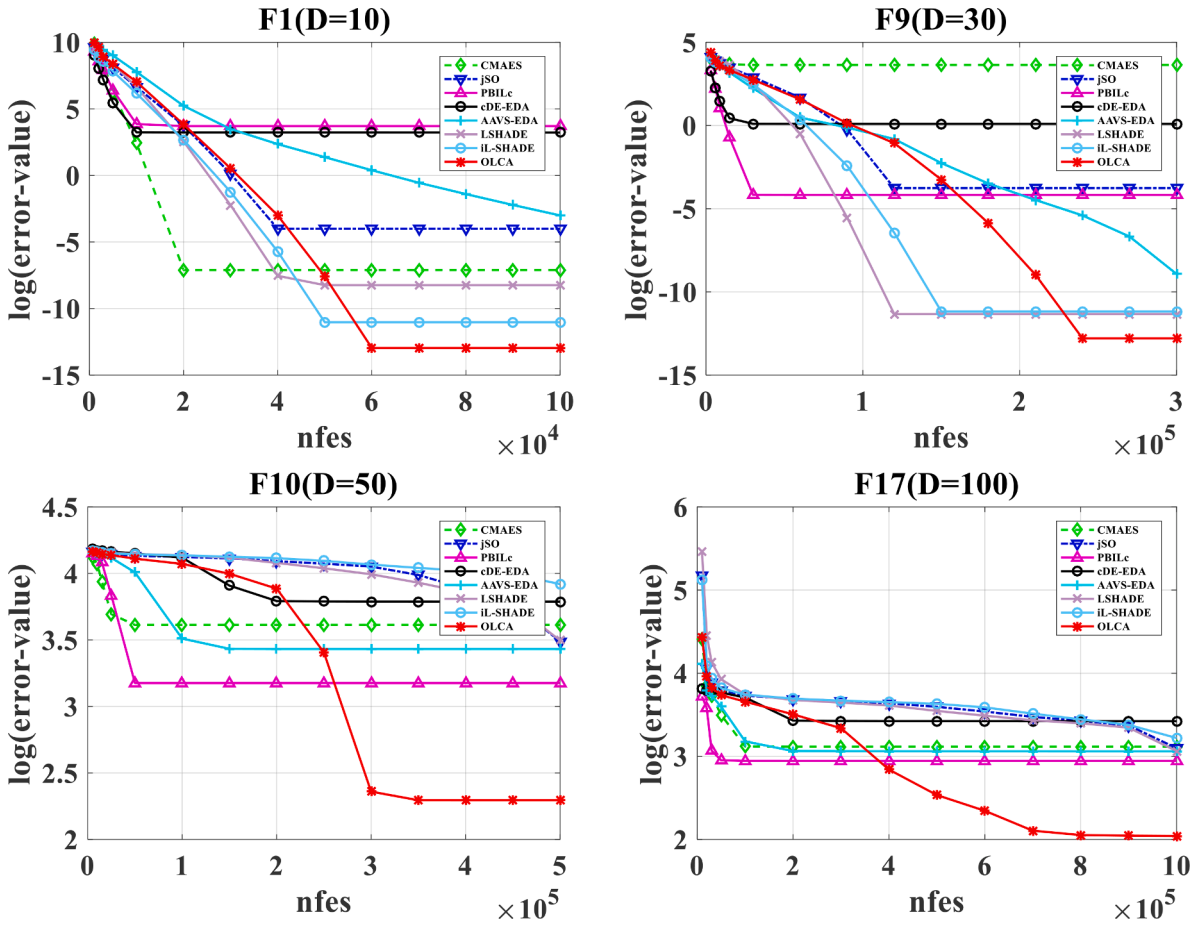


Fig. 4. The convergence curves plot on 10D, 30D, 50D, 100D.

gorithm such as convergence speed and diversity of the population are presented in Figs. 6–8. The fitness landscape figures for $f_1, f_2, f_4, f_{10}, f_{22}$, and f_{28} are also presented to illustrate the search behavior of OLCA. Moreover, the stereogram and planform of each function are also displayed to understand the running process intuitively. The population of OLCA is marked by the red dots. f_1 and f_2 are unimodal functions that easily find the global optimal generally. f_4 and f_{10} are multimodal functions that are unable to find the global optima easily due to the plenty of peaks and valleys. f_{22} and f_{28} denote the composition functions that are unable to obtain the global optima easily because of various attraction basins.

As shown in Fig. 6, f_1 is a unimodal function with a non-separable, smooth, but narrow ridge. It is difficult to optimize f_1 for algorithms, which have a poor local search ability, such as PBILc. On the contrary, the proposed OLCA performs well on f_1 due to good local search ability of EDA and DE. Note that the proposed OLCA has good performance on f_1 regardless of which component (EDA or DE) is executed. The only difference is that the EDA rapidly converges to the optima while the convergence speed of DE is slow. f_2 is a unimodal function that is non-separable and symmetric. The fitness landscape with symmetric rotation of the coordinate system leads to severe algorithmic performance losses [98]. The proposed OLCA has good performance on f_2 as EDA and DE are insusceptible to coordinate system rotation. Similarly, the convergence speed of the algorithm is fast if the EDA component is executed. It is noteworthy that the proposed OLCA performs much better than some algorithms, such as AAVS_EDA, PBILc, and cDE/EDA, on f_2 for higher dimensions. f_4 is a multi-modal function that is a non-separable and has a large number of local optima. The proposed OLCA performs well on f_4 as the population diversity of the DE component is maintained during the evolution process. f_{10} is a multi-modal function that is a non-separable and possesses a large number of local optima. In addition, the second better local optimum is far from the global optimum. Note that a high population diversity of an algorithm is required for this type of function. As shown in Fig. 7, the population diversity of EDA is lost due to the rapid reduction in the variance, which makes it far from the global optimum. In contrast, the defect in the DEA is complemented by the DE component. The DE explores multiple attraction basins with the help of mutation and crossover operators, thus enabling it to find the global optimum. In Fig. 8, f_{22} is a composition function, which is multi-modal, non-separable, asymmetrical, and possesses different properties around different local optima.

As shown in the stereogram, there are two attraction basins in the fitness landscape and the global optima lie in the small basin. It is difficult for the EDA to cross the area between the two basins since most individuals in the population in EDA are trapped in the large basin, thus making it difficult to shift the center of the population to a promising area. The location of the mean determines if it is possible to find the global optimum since the population of EDA is generated around the mean. On the contrary, the DE is less affected by the attraction basin as compared to the EDA. Similarly, f_{28} has the same properties as f_{22}. The global optimum is hard to discover by the EDA due to the existence of various attraction basins. As compared to EDA, the DE has the advantage of dividing the population automatically. This allows DE to explore different areas of the search space and discover the global optimum. Note that the properties of the fitness landscape are changed with an increase in the number of dimensions. The number of local optima increases exponentially and the difficulty of finding the global optimum increases significantly due to the curse of dimensionality.

In short, the EDA and DE are two algorithms that possess different characteristics. In the proposed framework, the diversity of the

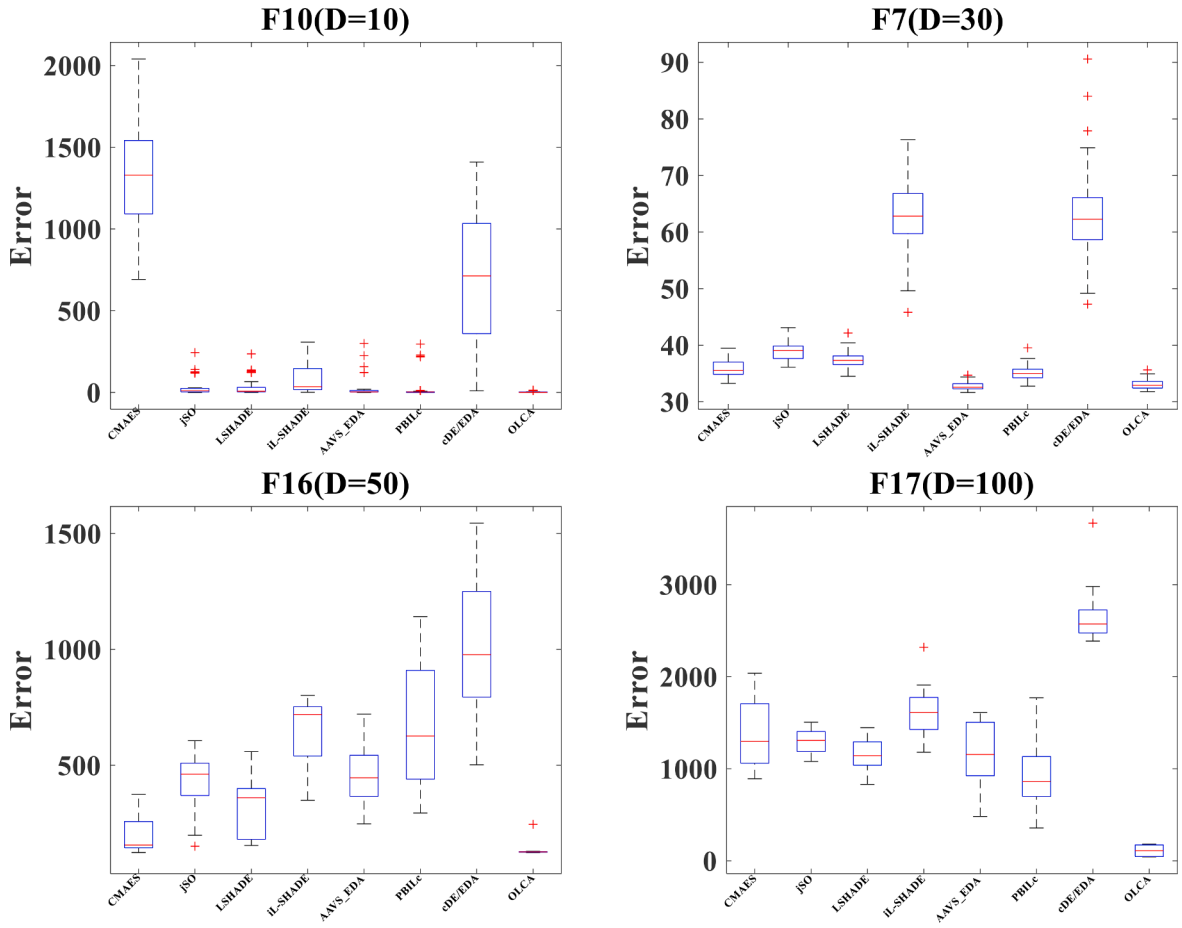


Fig. 5. The box plot on 10D, 30D, 50D, 100D.

algorithms is helpful to solve the problems that have various features. The different characteristics of EDA and DE are exploited to discover the global optimum in a large-scale search space.

4.5. *Effect of random forest on OLCA*

In this section, the contribution of random forest in the performance of the proposed OLCA is analyzed. The RSCA as a variant of OLCA adopts random selection for selecting EDA and DE as compared with other algorithms discussed in Section 4.3. As presented in Tables 3 and S4–S6, the performance of the proposed OLCA is better as compared to the RSCA in all dimensions as the selection mechanism of random forest based on fitness landscape analysis has made a correct decision with a high probability to select a suitable strategy to solve the problem. Note that a single selection method is unable to make the right decision every time. As the algorithm adopts a mechanism that possesses the guidance information, such as fitness landscape rather than a random operation, the performance of the algorithm is overall good. In Fig. 9, the proposed OLCA denoted by a red dotted line is below the RSCA denoted by a blue solid line, especially for f_6, f_9, and f_{20}. As shown in Fig. 9, the two lines are close to each other because the two algorithms, including EDA and DE, have a good performance in solving several problems, i.e., the performance of RSCA based on random selection is good as well. As described in section 3.5, the behaviors of the two algorithms are complementary, which means that the two algorithms play their respective roles in different problems.

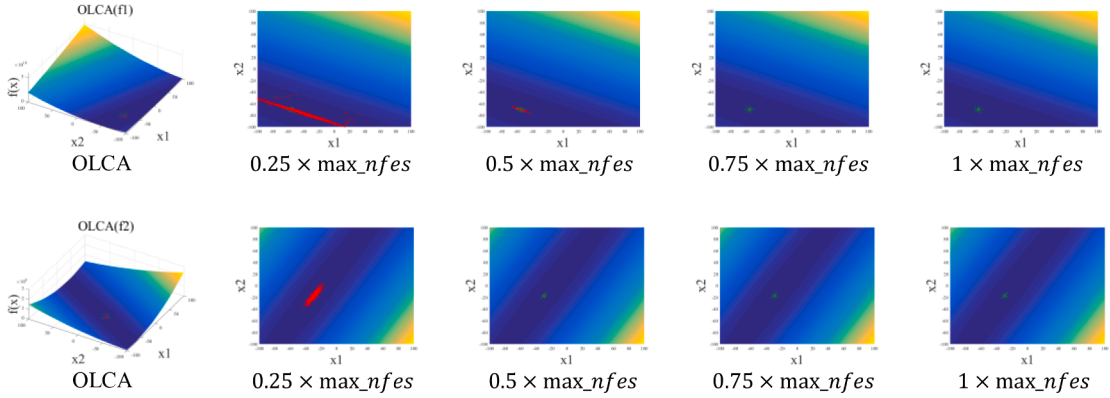


Fig. 6. Stereogram and planform of unimodal functions.

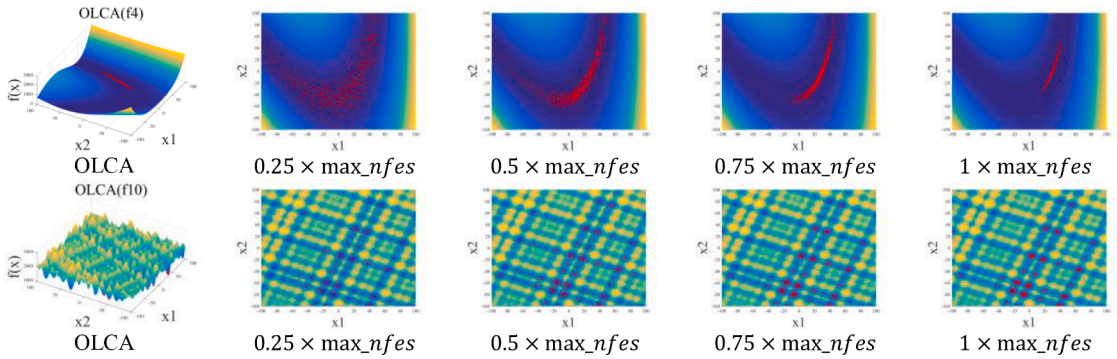


Fig. 7. Stereogram and planform of DE running on multimodal functions.

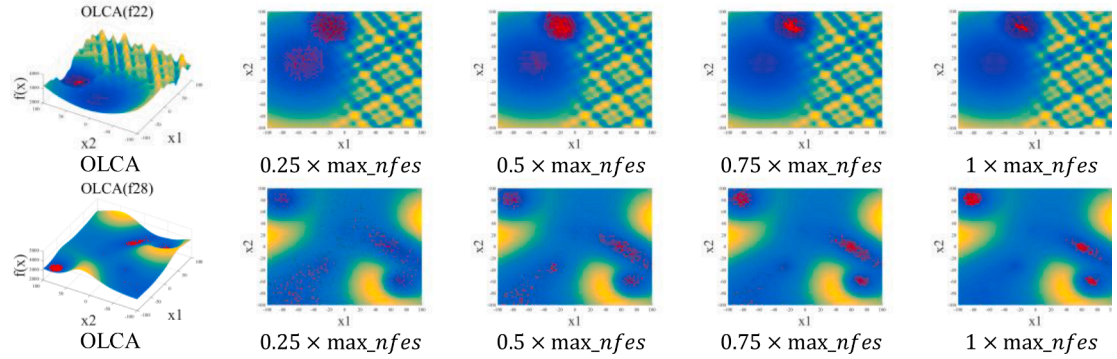


Fig. 8. Stereogram and planform of DE running on compositions functions.

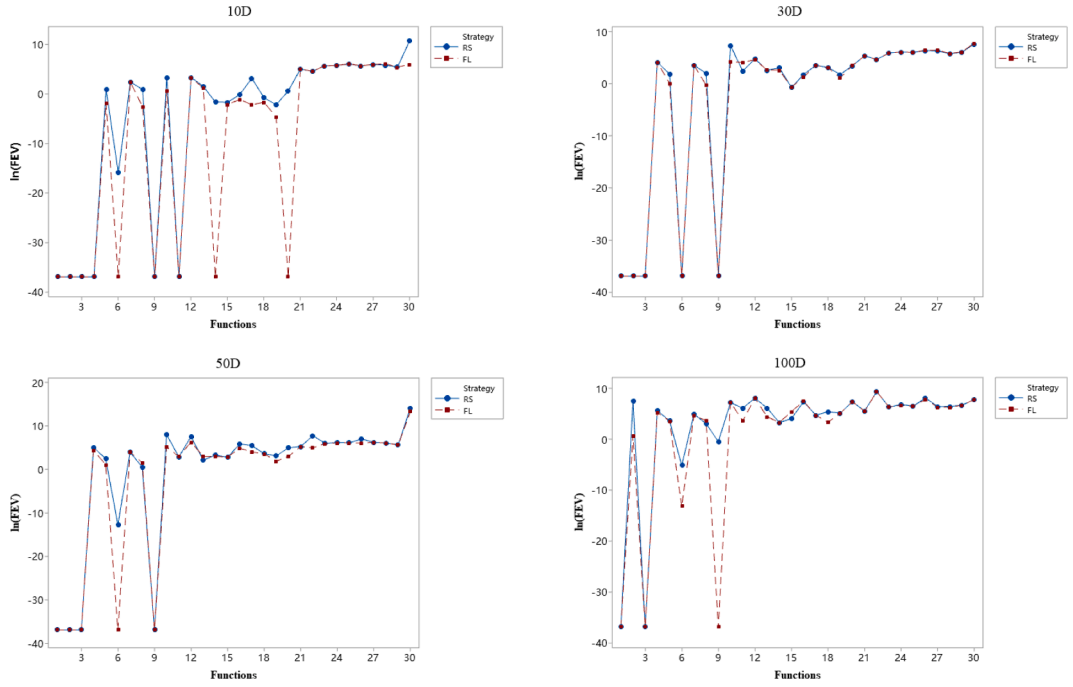


Fig. 9. Comparison between OLCA and RSCA on 30 functions.

4.6. Convergence and stability comparison

In order to further elucidate the convergence speed and stability of the proposed OLCA, one function is selected from each type of benchmark function to plot the convergence speed curves and box plot. According to the definition of CEC 2017, fourteen data samples are selected from the original experimental results to plot the convergence speed curves. After obtaining the logarithm of the error values, i.e., along longitudinal coordinate-axis y, some curves vanish from some point if the data is zero. In order to avoid this situation, the last non-zero data is used to replace those data points that reach zero. Note that some algorithms obtain the predefined accuracy instead of being stagnate. As

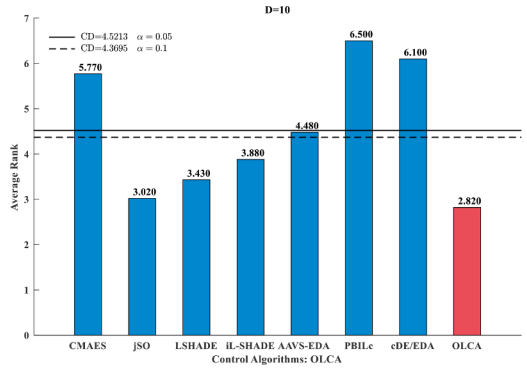


Fig. 10. The Friedman test results of 10D.

shown in Fig. 4, the proposed OLCA denoted as a red line, has a good performance in terms of convergence. Although, there are some algorithms that converge faster than the proposed OLCA, the quality of convergence is worse than the proposed OLCA. A significant distinction is that the proposed OLCA still tries to find the global optimum while the others algorithms converge to the local optima. This occurs during the later stages of these algorithms. The results show that some algorithms are prone to premature convergence and fall into stagnation due to the monotonous mechanism. Generally, an algorithm performs well on some functions while it performs unsatisfactory on others. The proposed OLCA predicts a suitable strategy based on the fitness landscape information to discover a better solution. Fig. 5 presents the box plots that show the stability of the algorithms. One function is selected from each type of benchmark function to perform comparisons. The stability of the proposed OLCA is effective for the four types of benchmark functions. The good stability indicates that a good solution is frequently obtained by the algorithm rather than obtaining it occasionally in each iteration. In addition, the other convergence speed curves and box plots are presented in Figs S1-S8 in the supplementary material.

4.7. Non-parametric test

In this section, the non-parametric test, i.e., Friedman and Wilcoxon tests are performed to further assess the proposed OLCA. The Friedman test is a nonparametric statistical method for determining the significant differences in multiple related samples [99]. As shown in Fig. 10 and the Fig. S9 of supplementary information, the mean rank of the OLCA is the smallest in all dimensions, which means that the performance of this algorithm is the best. Note that there is a significant difference between the proposed OLCA and CMA-ES, AAVS-EDA, PBILc, and cDE/EDA with a 90% confidence level. In addition, there is a significant difference between the proposed OLCA and CMA-ES, PBILc, and cDE/EDA under the 95% confidence level. Considering the 30-dimensional problems, the mean rank of the OLCA is minimum. Under the 90% confidence level, the difference between the proposed OLCA and CMA-ES, AAVS-EDA, PBILc, and cDE/EDA is significant. Under the 95% confidence level, the test results are the same as under the confidence level is 90%. The difference between OLCA and the other algorithms is significant under 90% confidence level in the 50-dimensional problems. The results show that only the mean rank of the proposed OLCA is below the critical differences represented by the solid and the dotted lines. Moreover, the results also show that the OLCA performs well on the 50-dimensional problems. As presented in Fig. S9, there exists a significant difference between the proposed OLCA and LSHADE, AAVS-EDA, PBILc, and cDE/EDA in case of 100-dimensional problems, no matter what the confidence level is. On the contrary, there is no significant difference between the OLCA and CMA-ES under the two confidence levels. Note that the mean rank of the proposed OLCA is still smaller as compared to the other algorithms. In conclusion, the OLCA is competitive as compared with its competitors.

The results of the Wilcoxon sign rank test are shown in Tables 4 and S1–S3 of the supplementary material. In Table 4, the number of positive ranks is represented by $R+$ which means the number of compared algorithms that are worse than the proposed OLCA is $R+$. Meanwhile, the number of the negative ranks is represented by R- which means there are R- comparison algorithms that are superior to the OLCA. As shown in Tables 4 and S1–S3, there exists a significant difference in a pairwise algorithm if the p-value is smaller than α, which is represented by Yes. In case of the 10 and 30-dimensional problems, the proposed OLCA is significantly better as compared to some algorithms but is not significantly better than some other algorithms such as jSO, LSHADE, iL-SHADE, and AAVS-EDA. The performance of the proposed OLCA is significantly better than the compared algorithms for the 50- and 100-dimensional problems.

In summary, the OLCA is feasible and better than the compared algorithms in solving the benchmark test suite. As compared with other algorithms, this framework of algorithms solves the problem at a higher level, i.e., algorithms are strategies, rather than tuning the parameters of an algorithm or several operators of the algorithm. In addition, the fitness landscape information without prior knowledge about the global optimum is utilized to guide the selection of strategies in the framework. Two algorithms that possess different characteristics are adopted to discover the global optimum. A complementation exists in the two algorithms contributing to the algorithm for solving the problems that have different features.

4.8. IEEE CEC2011 real-world benchmark problems test

In this subsection, the capability of the proposed OLCA is further assessed based on the nineteen real-world benchmark problems selected from IEEE CEC 2011. These benchmarks are adopted widely by the researchers for measuring the performance of algorithms proposed for solving the practical problems. The details of nineteen selected benchmark problems from IEEE CEC 2011 are presented in Table S7 of the supplementary materials. The definition of these functions and other details are presented in [100].

The proposed OLCA is compared with state-of-the-art algorithms by using the IEEE CEC 2011 real-world benchmark problems including GA with a new multi-parent crossover (GA-MPC), adaptive differential evolution with optional external archive (JADE), jSO, AAVS-EDA, EDcDE, LSHADE, iL-SHADE, and DE. The parameters of the compared algorithms are obtained from the original works. In order to ensure a fair comparison, all the compared algorithms are simulated by using the same hardware and software configurations. Moreover, all the compared algorithms are executed for 25 epochs independently, and the

Table 4
p-value of Wilcoxon's rank-sum test for $D = 10$.

OLCA *VS*	$R+$	R-	+	≈	–	Z	p-value	α=0.05	α=0.1
CMAES	274.0	51.0	22	5	3	−3.000	2.70E-03	**Yes**	**Yes**
jSO	102.0	129.0	13	9	8	−0.469	6.39E-01	No	No
LSHADE	148.0	105.0	15	8	7	−0.698	4.85E-01	No	No
iL-SHADE	144.5	86.5	14	9	7	−1.008	3.13E-01	No	No
AAVS_EDA	228.5	96.5	20	5	5	−1.776	7.57E-02	No	Yes
PBILc	378.0	0.0	27	3	0	−4.541	6.00E-06	**Yes**	**Yes**
cDE/EDA	291.0	60.0	23	4	3	−2.933	3.35E-03	**Yes**	**Yes**

Table 5
The statistical results of Wilcoxon's rank-sum test.

OLCA *vs*	R+	R-	+	≈	–	Z	*p*-value	α=0.05	α=0.1
JADE	141.0	12.0	16	2	1	−3.054	2.26E-03	**Yes**	**Yes**
jSO	54.0	12.0	9	8	2	−1.868	6.18E-02	No	No
LSHADE	27.0	9.0	6	11	2	−1.262	2.07E-01	No	No
iL-SHADE	85.0	20.0	12	5	2	−2.042	4.11E-02	Yes	Yes
GA-MPC	141.0	30.0	12	1	6	−2.417	1.56E-02	**Yes**	**Yes**
AAVS_EDA	171.0	0.0	18	1	0	−3.724	1.96E-04	**Yes**	**Yes**
DE	190.0	0.0	19	0	0	−3.823	1.32E-04	**Yes**	**Yes**
cDE/EDA	178.0	12.0	18	0	1	−3.341	8.33E-04	**Yes**	**Yes**

Table 6
Comparison results on IEEE CEC 2011 problems.

Function		JADE [47]	jSO [82]	LSHADE [49]	iL-SHADE [83]	AAVS_EDA [42]	GA-MPC [50]	cDE/EDA [85]	DE [12]	OLCA
F1	Mean	3.64E+00	4.68E-01	2.61E-04	3.37E-01	2.45E+01	**0.00E+00**	2.98E+00	1.16E+01	3.14E-04
	Std.	3.92E+00	2.34E+00	1.01E-03	1.68E+00	1.71E+00	0.00E+00	5.48E+00	3.37E+00	1.56E-03
F2	Mean	−1.99E+01	−2.17E+01	−2.63E+01	−2.23E+01	−1.02E+01	**−2.73E+01**	−2.60E+01	−5.28E+00	−2.64E+01
	Std.	1.16E+00	8.67E-01	5.28E-01	8.82E-01	7.51E-01	6.10E-01	1.10E+00	4.80E-01	7.12E-01
F5	Mean	−3.38E+01	−3.12E+01	−3.63E+01	−3.18E+01	−2.04E+01	−3.48E+01	−3.18E+01	−1.99E+01	**−3.66E+01**
	Std.	3.87E-01	1.13E+00	5.09E-01	1.09E+00	8.37E-01	1.58E+00	2.53E+00	1.27E+00	3.12E-01
F6	Mean	−2.77E+01	−2.49E+01	**−2.92E+01**	−2.61E+01	−1.60E+01	−2.65E+01	−2.17E+01	−1.34E+01	**−2.92E+01**
	Std.	8.29E-01	1.83E+00	7.30E-03	1.28E+00	1.16E+00	2.45E+00	4.21E+00	9.71E-01	6.97E-03
F7	Mean	1.33E+00	1.30E+00	1.19E+00	1.26E+00	2.17E+00	9.51E-01	9.46E-01	1.44E+00	**6.17E-01**
	Std.	1.15E-01	1.04E-01	9.48E-02	8.60E-02	1.19E-01	2.72E-01	2.15E-01	7.58E-02	1.28E-01
F9	Mean	**1.43E+03**	2.52E+03	9.12E+03	1.23E+04	2.47E+06	5.07E+03	8.56E+03	1.98E+05	9.80E+03
	Std.	4.21E+02	8.22E+02	4.76E+03	5.67E+03	2.93E+04	8.95E+03	7.08E+03	1.61E+04	5.15E+03
F10	Mean	−2.15E+01	**−2.17E+01**	−2.16E+01	−2.16E+01	−2.14E+01	**−2.17E+01**	−2.14E+01	−1.65E+01	−2.16E+01
	Std.	1.31E-01	5.49E-02	1.01E-01	6.49E-02	6.25E-02	1.13E-01	1.72E-01	1.17E+00	9.07E-02
F11	Mean	5.23E+04	**5.19E+04**	5.20E+04	5.20E+04	1.51E+06	5.27E+04	9.92E+04	1.65E+07	**5.19E+04**
	Std.	5.92E+02	5.55E+02	5.47E+02	5.95E+02	1.72E+05	6.89E+02	8.36E+04	3.93E+06	3.63E+02
F12	Mean	**1.07E+06**	**1.07E+06**	**1.07E+06**	**1.07E+06**	4.12E+06	1.08E+06	1.08E+06	2.88E+06	**1.07E+06**
	Std.	1.73E+03	1.76E+03	1.48E+03	1.55E+03	2.17E+05	9.33E+03	2.09E+04	1.20E+05	1.33E+03
F13	Mean	**1.54E+04**	**1.54E+04**	**1.54E+04**	**1.54E+04**	1.57E+04	**1.54E+04**	1.55E+04	1.55E+04	**1.54E+04**
	Std.	2.41E+00	7.43E-12	2.46E-06	1.22E+00	5.53E+02	7.43E-12	1.50E+01	5.03E+00	7.43E-12
F14	Mean	1.83E+04	**1.81E+04**	**1.81E+04**	**1.81E+04**	**1.81E+04**	1.83E+04	1.83E+04	1.94E+04	**1.81E+04**
	Std.	3.65E+02	2.69E+01	2.85E+01	1.33E+01	5.02E+01	7.89E+01	6.06E+01	2.01E+02	1.66E+01
F15	Mean	3.30E+04	**3.27E+04**	**3.27E+04**	**3.27E+04**	3.29E+04	3.28E+04	3.29E+04	3.30E+04	**3.27E+04**
	Std.	5.97E+01	4.96E-01	3.10E-01	2.72E-01	6.24E+01	3.43E+01	6.47E+01	5.01E+01	1.95E-01
F16	Mean	1.33E+05	1.24E+05	1.24E+05	**1.23E+05**	1.25E+05	1.36E+05	1.32E+05	1.39E+05	1.24E+05
	Std.	4.76E+03	5.72E+02	4.36E+02	5.47E+02	9.78E+02	2.79E+03	2.43E+03	3.28E+03	5.55E+02
F17	Mean	1.93E+06	**1.85E+06**	**1.85E+06**	1.87E+06	1.91E+06	2.60E+07	1.92E+06	2.80E+06	**1.85E+06**
	Std.	4.78E+04	1.03E+04	1.16E+04	1.24E+04	6.93E+03	1.20E+08	1.71E+04	4.79E+05	1.36E+04
F18	Mean	9.44E+05	9.34E+05	**9.33E+05**	9.34E+05	6.64E+07	1.05E+06	9.43E+05	2.37E+06	**9.33E+05**
	Std.	1.30E+04	1.64E+03	1.57E+03	1.59E+03	3.99E+06	1.67E+05	1.14E+04	3.66E+05	1.64E+03
F19	Mean	1.07E+06	9.41E+05	**9.40E+05**	9.42E+05	4.24E+07	1.21E+06	1.18E+06	3.01E+06	**9.40E+05**
	Std.	1.91E+05	1.38E+03	1.28E+03	1.18E+03	1.97E+06	1.81E+05	1.11E+05	4.18E+05	1.11E+03
F20	Mean	9.41E+05	9.32E+05	9.32E+05	**9.31E+05**	4.47E+07	1.02E+06	9.40E+05	1.87E+06	9.32E+05
	Std.	1.13E+04	1.38E+03	1.38E+03	1.06E+03	3.78E+06	4.95E+04	2.51E+03	2.45E+05	1.06E+03
F21	Mean	1.90E+01	1.54E+01	1.53E+01	1.54E+01	1.55E+01	**1.38E+01**	1.58E+01	2.01E+01	1.47E+01
	Std.	1.69E+00	1.56E+00	1.32E+00	9.52E-01	1.77E+00	4.04E+00	1.99E+00	2.81E+00	1.85E+00
F22	Mean	1.75E+01	1.49E+01	1.22E+01	1.31E+01	1.74E+01	**1.04E+01**	1.96E+01	1.79E+01	1.21E+01
	Std.	2.55E+00	3.56E+00	2.78E+00	2.99E+00	2.02E+00	3.10E+00	2.56E+00	2.33E+00	2.58E+00
W/L/T	1/16/2	2/9/8	2/6/11	2/12/5	0/18/1	6/12/1	1/18/0	0/19/0	–	

maximum number of fitness evaluations is set as 150,000.

A detailed comparison between the proposed OLCA and other counterparts on nineteen real-world problems is shown in Table 6, including Mean and Std values for each function. The statistical results show that the proposed OLCA is superior to GA-MPC which is the winner of IEEE CEC 2011 real-world benchmark problems on 12 functions. The proposed OLCA wins JADE on 16 functions but loses on 1 function. Note that there is a significant difference between the proposed OLCA and JADE in statistical terms. The proposed OLCA outperforms jSO on 9 functions, but it is worse than jSO on 8 functions and shows a tie on 2 functions. Although there is no significant difference between OLCA, LSHADE, and jSO, the proposed OLCA is competitive as compared with LHSADE and jSO, which is the winner of IEEE CEC 2017 benchmark problems. The proposed OLCA performs better on 12 functions as compared with the iL-SHADE. The OLCA is better than AAVS-EDA on 18 functions but loses on 1 function. As compared with cDE/EDA, the

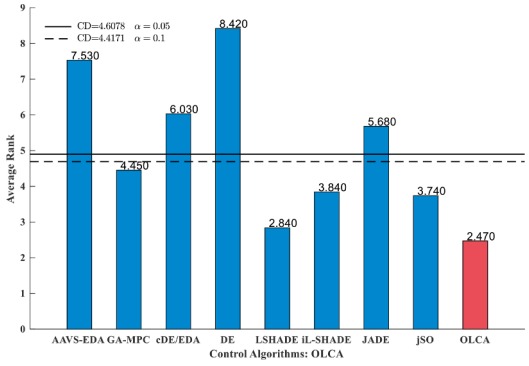


Fig. 11. The Friedman test results of compared algorithms on IEEE CEC 2011.

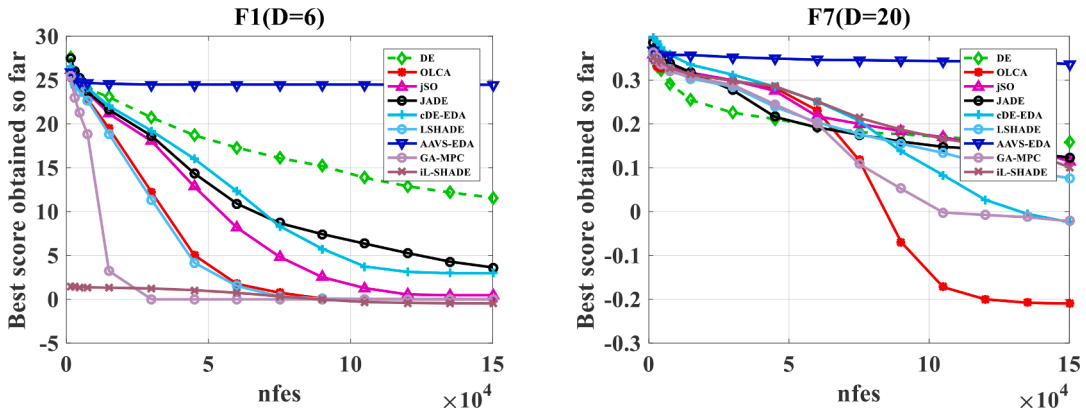


Fig. 12. The convergence curves plot of compared algorithms on IEEE CEC 2011.

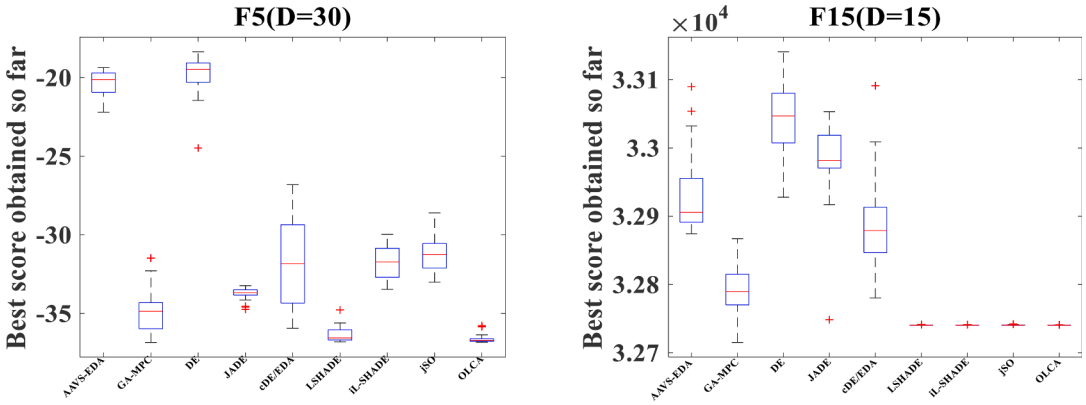


Fig. 13. The box plot of compared algorithms on IEEE CEC 2011.

proposed OLCA shows a superior performance on 18 functions but it has a poor performance on one function. It is noteworthy that the proposed OLCA does not fail against DE on any function.

Table 5 shows the comparison results between the proposed OLCA and other advanced methods based on the Wilcoxon's rank-sum test with a significance level of 0.05 and 0.1. Although there is no significant difference between the proposed OLCA and some of its competitors (GA-MPC, LSHADE, iL-SHADE, and jSO), the OLCA performs well on certain problems. The Friedman test results of all the competitors are demonstrated in Fig. 11. The results show that the proposed OLCA attains the lowest ranking score with 1.84 and has the best performance. All these statistical results illustrate the effectiveness of the proposed OLCA in solving the real-world optimization problems. Based on the convergence curves reported in Fig. 12, it is evident that found that the proposed OLCA performs well on certain problems. The OLCA focuses on the selection of a suitable algorithm for certain problems instead of improving the part of an algorithm. Consequently, the convergence speed of the OLCA is not the best. Similarly, the stability of the proposed OLCA is competitive as compared with other algorithms rather than being the best as illustrated by the boxplots presented in Fig. 13. The other convergence speed curves and box plots are presented in Figs. S10 and S11 in the supplementary material.

Note that the real-world benchmark problems are more challenging to solve as compared to the benchmark functions, such as IEEE CEC 2017. However, the proposed OLCA performs well on these problems as well. There are two main reasons for this performance. First, the two algorithms with different advantages are integrated in the proposed framework of OLCA to boost the adaptability of OLCA for different problems. Secondly, the fitness landscape analysis technique is employed to extract the problem-specific knowledge for the selection of algorithms for different problems. Furthermore, the machine learning techniques are applied for training the OLCA to establish the problem-algorithm mapping and predict a suitable method for a new problem.

5. Conclusions and future work

In this work, an offline learning co-evolutionary algorithm (OLCA) based on the fitness landscape is proposed to address the complex continuous optimization problems. The visualization of executing the algorithms is displayed and utilized to analyze the search behavior of the proposed OLCA. The OLCA is tested on CEC 2017 benchmark test suite to verify the performance and compared with several state-of-the-art algorithms. The mean value and standard variance illustrate that the OLCA is competitive. The non-parametric test is performed to analyze the statistical differences between the pairing algorithms. The test results show that there is a significant difference between the proposed OLCA and compared algorithms. Moreover, the proposed OLCA is tested on CEC 2011 real-world benchmark problems. The results demonstrate that OLCA is competitive as compared with some advanced algorithms. This is because each algorithm is selected as a strategy in the framework and, the performance of the proposed OLCA is dependent on each algorithm. The performance of the OLCA is limited if the selected algorithms selected are similar in terms of searching behavior. In summary, the proposed OLCA is an effective algorithm.

In future work, the framework proposed in this work can be further extended on other algorithms and machine learning models. The integration of other algorithms, mechanisms, and strategies will enhance the performance of the proposed algorithm. The selection of an efficient machine learning model is important for improving the search capability of intelligent algorithms. Moreover, it is a challenging to apply the

fitness landscape analysis technique in practice. The introduction of the fitness landscape contributes in solving the complex problems. In addition, the visualization analysis of algorithms is an interesting direction and helps to design a more efficient algorithm. Furthermore, the performance of the proposed OLCA needs further verification on practical problems.

CRediT authorship contribution statement

Fuqing Zhao: Funding acquisition, Investigation, Supervision. **Bo Zhu:** Investigation, Software, Writing – original draft. **Ling Wang:** Methodology, Resources. **Tianpeng Xu:** Project administration, Writing – review & editing. **Ningning Zhu:** Conceptualization, Formal analysis. **Jonrinaldi Jonrinaldi:** Visualization.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was financially supported by the National Natural Science Foundation of China under grant 62063021. It was also supported by the Key talent project of Gansu Province (ZZ2021G50700016), the Key Research Programs of Science and Technology Commission Foundation of Gansu Province (21YF5WA086), Lanzhou Science Bureau project (2018-rc-98), and Project of Gansu Natural Science Foundation (21JR7RA204) , respectively.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.swevo.2022.101148.

References

[1] Z.H. Zhan, L. Shi, K.C. Tan, J. Zhang, A Survey On Evolutionary Computation For Complex Continuous Optimization, Springer, Netherlands, 2021, https://doi.org/10.1007/s10462-021-10042-y, 0123456789.
[2] Z. Tan, K. Li, Y. Wang, Differential evolution with adaptive mutation strategy based on fitness landscape analysis, Inf. Sci. 549 (2021) 142–163, https://doi.org/10.1016/j.ins.2020.11.023.
[3] M.N. Omidvar, X. Li, X. Yao, A review of population-based metaheuristics for large-scale black-box global optimization: part A, IEEE Trans. Evol. Comput. (2021), https://doi.org/10.1109/tevc.2021.3130838, 1–1Nov.
[4] M.N. Omidvar, X. Li, X. Yao, A review of population-based metaheuristics for large-scale black-box global optimization: part B, IEEE Trans. Evol. Comput. (2021), https://doi.org/10.1109/tevc.2021.3130835, 1–1Nov.
[5] H. Dong, Z. Dong, Surrogate-assisted grey wolf optimization for high-dimensional, computationally expensive black-box problems, Swarm Evol. Comput. 57 (2020), https://doi.org/10.1016/j.swevo.2020.100713. Sep.
[6] K.Z. Gao, Z.M. He, Y. Huang, P.Y. Duan, P.N. Suganthan, A survey on meta-heuristics for solving disassembly line balancing, planning and scheduling problems in remanufacturing, Swarm Evol. Comput. 57 (2020), https://doi.org/10.1016/j.swevo.2020.100719. Sep.
[7] C. Witt, Theory of estimation-of-distribution algorithms, *GECCO 2020 Companion*, in: Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, 2020, pp. 1254–1282, https://doi.org/10.1145/3377929.3389888. Jul.
[8] B.B. Oliveira, M.A. Carravilla, J.F. Oliveira, A diversity-based genetic algorithm for scenario generation, Eur. J. Oper. Res. (2021), https://doi.org/10.1016/j.ejor.2021.09.047 xxxx.
[9] L. Manzoni, L. Mariot, E. Tuba, Balanced crossover operators in genetic algorithms, Swarm Evol. Comput. 54 (2020), https://doi.org/10.1016/j.swevo.2020.100646. May.
[10] A. Kumar, P.P. Biswas, P.N. Suganthan, Differential evolution with orthogonal array-based initialization and a novel selection strategy, Swarm Evol. Comput. 68 (2022), https://doi.org/10.1016/j.swevo.2021.101010. Feb.
[11] J. Cheng, Z. Pan, H. Liang, Z. Gao, J. Gao, Differential evolution algorithm with fitness and diversity ranking-based mutation operator, Swarm Evol. Comput. 61 (2021), https://doi.org/10.1016/j.swevo.2020.100816. Mar.
[12] S. Das, S.S. Mullick, P.N. Suganthan, Recent advances in differential evolution-an updated survey, Swarm Evol. Comput. 27 (2016) 1–30, https://doi.org/10.1016/j.swevo.2016.01.004. Apr.
[13] F. Zhao, L. Zhao, L. Wang, H. Song, A collaborative LSHADE algorithm with comprehensive learning mechanism, Appl. Soft Comput. J. 96 (2020), https://doi.org/10.1016/j.asoc.2020.106609. Nov.
[14] A.P. Piotrowski, J.J. Napiorkowski, A.E. Piotrowska, Population size in particle swarm optimization, Swarm Evol. Comput. 58 (2020), https://doi.org/10.1016/j.swevo.2020.100718. Nov.
[15] E.H. Houssein, A.G. Gad, K. Hussain, P.N. Suganthan, Major advances in particle swarm optimization: theory, analysis, and application, Swarm Evol. Comput. 63 (2021), https://doi.org/10.1016/j.swevo.2021.100868. Jun.
[16] M. Hauschild, M. Pelikan, An introduction and survey of estimation of distribution algorithms, Swarm Evol. Comput. 1 (3) (2011) 111–128, https://doi.org/10.1016/j.swevo.2011.08.003. Sep.
[17] J.P. Martins, A.C.B. Delbem, Pairwise independence and its impact on estimation of distribution algorithms, Swarm Evol. Comput. 27 (2016) 80–96, https://doi.org/10.1016/j.swevo.2015.10.001. Apr.
[18] M. Gendreau, J.Y. Potvin, Eds., Handbook of Metaheuristics, 272, Springer, 2019.
[19] D.H. Wolpert and W.G. Macready, "No free lunch theorems for optimization," 1997.
[20] J.R. Rice, The algorithm selection problem, Adv. Comput. 15 (1976) 65–118, https://doi.org/10.1016/S0065-2458(08)60520-3. CJan.
[21] F. Zhao, S. Di, J. Cao, J. Tang, A novel cooperative multi-stage hyper-heuristic for combination optimization problems, Complex Syst. Model. Simul. 1 (2) (2021), https://doi.org/10.23919/csms.2021.0010.
[22] L. Wang, J.N. Shen, S.Y. Wang, J. Deng, Advances in co-evolutionary algorithms, in: Kongzhi Yu Juece/Control and Decision, 30, Northeast University, 2015, pp. 193–202, https://doi.org/10.13195/j.kzyjc.2014.0624. Feb. 01.
[23] K.M. Malan, A.P. Engelbrecht, A survey of techniques for characterising fitness landscapes and some possible ways forward, Inf. Sci. 241 (2013) 148–163, https://doi.org/10.1016/j.ins.2013.04.015. Aug.
[24] M. Karimi-Mamaghan, M. Mohammadi, P. Meyer, A.M. Karimi-Mamaghan, E. G. Talbi, Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: a state-of-the-art, Eur. J. Oper. Res. 296 (2) (2022) 393–422, https://doi.org/10.1016/j.ejor.2021.04.032.
[25] S. Wagner, M. Affenzeller, HeuristicLab: a generic and extensible optimization environment. Adaptive and Natural Computing Algorithms, Springer, 2005, pp. 538–541. Dec.
[26] P. Kerschke, H.H. Hoos, F. Neumann, H. Trautmann, Automated algorithm selection: survey and perspectives, Evol. Comput. 27 (1) (2018) 3–45, https://doi.org/10.1162/evco_a_00242.
[27] J. Wawrzyniak, M. Drozdowski, É. Sanlaville, Selecting algorithms for large berth allocation problems, Eur. J. Oper. Res. 283 (3) (2020) 844–862, https://doi.org/10.1016/J.EJOR.2019.11.055. Jun.
[28] H. Mühlenbein, J. Bendisch, H.M. Voigt, From recombination of genes to the estimation of distributions II. Continuous parameters, in: Lecture Notes in Computer Science, 1141, Springer, 1996 (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).
[29] Y. Liang, et al., An efficient estimation of distribution algorithm with rank-one modification and population reduction, Biosystems 181 (2019), https://doi.org/10.1016/j.biosystems.2019.04.001.
[30] S. Baluja, "Population-based incremental learning: a method for integrating genetic search based function optimization and competitive learning," Tech. Rep. CMU-CS94-163, School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, 1994.
[31] J.S. de Bonet, C.L. Isbell, P. Viola, MIMIC: Finding Optima By Estimating Probability Densities, MIT Press, 1997.
[32] M. Pelikan, D.E. Goldberg, and E. Cantt U-Paz, "BOA: the Bayesian optimization algorithm," 1999.
[33] J. Ocenasek, S. Kern, N. Hansen, P. Koumoutsakos, A mixed bayesian optimization algorithm with variance adaptation, in: Lecture Notes in Computer Science, 3242, Springer, 2004 (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).
[34] J. Grahl, P.A.N. Bosman, F. Rothlauf, The correlation-triggered adaptive variance scaling IDEA, in: Proceedings of the *GECCO Genetic and Evolutionary Computation Conference* 1, 2006, https://doi.org/10.1145/1143997.1144071.
[35] W. Dong, X. Yao, Unified eigen analysis on multivariate Gaussian based estimation of distribution algorithms, Inf. Sci. 178 (15) (2008), https://doi.org/10.1016/j.ins.2008.01.021.
[36] M. Wagner, A. Auger, and M. Schoenauer, "EEDA : A new robust estimation of distribution algorithms," 2006. [Online]. Available: https://hal.inria.fr/inria-00070802.
[37] Y. Cai, X. Sun, H. Xu, and P. Jia, "Cross entropy and adaptive variance scaling in continuous EDA," 2007. doi:10.1145/1276958.1277081.
[38] P.A.N. Bosman, J. Grahl, D. Thierens, Enhancing the performance of maximum-likelihood gaussian EDAs using anticipated mean shift, in: Lecture Notes in Computer Science, 5199, Springer, 2008 (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)LNCS.
[39] Y. Liang, Z. Ren, X. Yao, Z. Feng, A. Chen, W. Guo, Enhancing gaussian estimation of distribution algorithm by exploiting evolution direction with archive, IEEE Trans. Cybern. 50 (1) (2020), https://doi.org/10.1109/TCYB.2018.2869567.
[40] N. Hansen, A. Ostermeier, Completely derandomized self-adaptation in evolution strategies, Evol. Comput. 9 (2) (2001), https://doi.org/10.1162/106365601750190398.

[41] W. Dong, T. Chen, P. Tino, X. Yao, Scaling up estimation of distribution algorithms for continuous optimization, IEEE Trans. Evol. Comput. 17 (6) (2013), https://doi.org/10.1109/TEVC.2013.2247404.

[42] F. Zhao, Z. Shao, J. Wang, C. Zhang, A hybrid differential evolution and estimation of distribution algorithm based on neighbourhood search for job shop scheduling problems, Int. J. Prod. Res. 54 (4) (2016), https://doi.org/10.1080/00207543.2015.1041575.

[43] Z. Ren, Y. Liang, L. Wang, A. Zhang, B. Pang, B. Li, Anisotropic adaptive variance scaling for Gaussian estimation of distribution algorithm, Knowl. Based Syst. 146 (2018) 142–151, https://doi.org/10.1016/j.knosys.2018.02.001. Apr.

[44] L. Tang, X. Song, J. Liu, C. Liu, An estimation of distribution algorithm with filtering and learning, IEEE Trans. Autom. Sci. Eng. 18 (3) (2021), https://doi.org/10.1109/TASE.2020.3019694.

[45] S. Das, S.S. Mullick, P.N. Suganthan, Recent advances in differential evolution-an updated survey, Swarm Evol. Comput. 27 (2016), https://doi.org/10.1016/j.swevo.2016.01.004.

[46] A.K. Qin, P.N. Suganthan, Self-adaptive differential evolution algorithm for numerical optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation, IEEE CEC 2, 2005, https://doi.org/10.1109/cec.2005.1554904.

[47] G. Wu, R. Mallipeddi, P.N. Suganthan, R. Wang, H. Chen, Differential evolution with multi-population based ensemble of mutation strategies, Inf. Sci. 329 (2016), https://doi.org/10.1016/j.ins.2015.09.009.

[48] J. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Trans. Evol. Comput. 13 (5) (2009), https://doi.org/10.1109/TEVC.2009.2014613.

[49] R. Tanabe, A. Fukunaga, Success-history based parameter adaptation for differential evolution, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2013, pp. 71–78, https://doi.org/10.1109/CEC.2013.6557555. *CEC*2013.

[50] R. Tanabe and A.S. Fukunaga, "Improving the search performance of SHADE using linear population size reduction".

[51] S.M. Elsayed, R.A. Sarker, and D.L. Essam, "GA with a new multi-parent crossover for solving IEEE-CEC2011 competition problems," 2011. doi:10.1109/CEC.2011.5949731.

[52] I. Boussaïd, A. Chatterjee, P. Siarry, M. Ahmed-Nacer, Hybridizing biogeography-based optimization with differential evolution for optimal power allocation in wireless sensor networks, IEEE Trans. Veh. Technol. 60 (5) (2011), https://doi.org/10.1109/TVT.2011.2151215.

[53] Y.L. Li, Z.H. Zhan, Y.J. Gong, W.N. Chen, J. Zhang, Y. Li, Differential evolution with an evolution path: a DEEP evolutionary algorithm, IEEE Trans. Cybern. 45 (9) (2015), https://doi.org/10.1109/TCYB.2014.2360752.

[54] P. Rakshit, et al., Realization of an adaptive memetic algorithm using differential evolution and q-learning: a case study in multirobot path planning, IEEE Trans. Syst. Man Cybern. Part A Syst. Hum. 43 (4) (2013), https://doi.org/10.1109/TSMCA.2012.2226024.

[55] I. Poikolainen and F. Neri, "Differential evolution with concurrent fitness based local search," 2013. doi:10.1109/CEC.2013.6557595.

[56] K.M. Malan, A.P. Engelbrecht, A survey of techniques for characterising fitness landscapes and some possible ways forward, Inf. Sci. 241 (2013) 148–163, https://doi.org/10.1016/j.ins.2013.04.015.

[57] K.M. Malan, A survey of advances in landscape analysis for optimisation, Algorithms 14 (2) (2021), https://doi.org/10.3390/a14020040.

[58] F. Xia, J. Liu, H. Nie, Y. Fu, L. Wan, and X. Kong, "Random walks: a review of algorithms and applications," Aug. 2020, doi:10.1109/TETCI.2019.2952908.

[59] K.M. Malan and A.P. Engelbrecht, "A progressive random walk algorithm for sampling continuous fitness landscapes," 2014. doi:10.1109/CEC.2014.6900576.

[60] G. Ochoa, M. Tomassini, C. Darabos, S. Vérel, A study of NK landscapes' basins and local optima networks, in: Proceedings of the GECCO'08: 10th Annual Conference on Genetic and Evolutionary Computation, 2008, pp. 555–562, https://doi.org/10.1145/1389095.1389204.

[61] G. Ochoa, K.M. Malan, C. Blum, Search trajectory networks: a tool for analysing and visualising the behaviour of metaheuristics, Appl. Soft Comput. 109 (2021), 107492, https://doi.org/10.1016/j.asoc.2021.107492.

[62] K.M. Malan, A.P. Engelbrecht, Quantifying ruggedness of continuous landscapes using entropy, in: Proceedings of the IEEE Congress on Evolutionary Computation, CEC, 2009, pp. 1440–1447, https://doi.org/10.1109/CEC.2009.4983112, 2009.

[63] O. Mersmann, B. Bischl, H. Trautmann, M. Preuss, C. Weihs, G. Rudolph, Exploratory landscape analysis, in: Proceedings of the Genetic and Evolutionary Computation Conference 11, GECCO, 2011, pp. 829–836, https://doi.org/10.1145/2001576.2001690. February 2014.

[64] A. Liefooghe, F. Daolio, S. Verel, B. Derbel, H. Aguirre, K. Tanaka, Landscape-aware performance prediction for evolutionary multiobjective optimization, IEEE Trans. Evol. Comput. 24 (6) (2020) 1063–1077, https://doi.org/10.1109/TEVC.2019.2940828.

[65] Y. Sun, M. Kirley, S.K. Halgamuge, Quantifying variable interactions in continuous optimization problems, IEEE Trans. Evol. Comput. 21 (2) (2017) 249–264, https://doi.org/10.1109/TEVC.2016.2599164.

[66] C. Fonlupt, D. Robilliard, P. Preux, A bit-wise epistasis measure for binary search spaces, in: Lecture Notes in Computer Science, 1498, Springer, 1998, pp. 47–56 (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)LNCS.

[67] Y. Davidor, "Epistasis variance: a viewpoint on GA-hardness," vol. 1, pp. 23–35, Jan. 1991, doi:10.1016/B978-0-08-050684-5.50005-7.

[68] F. Zhao, R. Ma, L. Wang, A self-learning discrete jaya algorithm for multiobjective energy-efficient distributed no-idle flow-shop scheduling problem in heterogeneous factory system, IEEE Trans. Cybern. (2021), https://doi.org/10.1109/TCYB.2021.3086181.

[69] F. Zhao, X. He, L. Wang, A two-stage cooperative evolutionary algorithm with problem-specific knowledge for energy-efficient scheduling of no-wait flow-shop problem, IEEE Trans. Cybern. 51 (11) (2021) 5291–5303, https://doi.org/10.1109/TCYB.2020.3025662. Nov.

[70] F. Zhao, L. Zhang, J. Cao, J. Tang, A cooperative water wave optimization algorithm with reinforcement learning for the distributed assembly no-idle flowshop scheduling problem, Comput. Ind. Eng. 153 (2021), https://doi.org/10.1016/j.cie.2020.107082. Mar.

[71] F. Zhao, L. Zhang, Y. Zhang, W. Ma, C. Zhang, H. Song, A hybrid discrete water wave optimization algorithm for the no-idle flowshop scheduling problem with total tardiness criterion, Expert Syst. Appl. 146 (2020), https://doi.org/10.1016/j.eswa.2019.113166.

[72] S. Fairee, C. Khompatraporn, S. Prom-On, B. Sirinaovakul, Combinatorial artificial bee colony optimization with reinforcement learning updating for travelling salesman problem, in: Proceedings of the 16th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, ECTI-CON, 2019, pp. 93–96, https://doi.org/10.1109/ECTI-CON47248.2019.8955176. *2019*Jul.

[73] C.Y. Cheng, P. Pourhejazy, K.C. Ying, C.F. Lin, Unsupervised learning-based artificial bee colony for minimizing non-value-adding operations, Appl. Soft Comput. 105 (2021), 107280, https://doi.org/10.1016/J.ASOC.2021.107280. Jul.

[74] A. Dantas, A. Pozo, On the use of fitness landscape features in meta-learning based algorithm selection for the quadratic assignment problem, Theor. Comput. Sci. 805 (2020) 62–75, https://doi.org/10.1016/J.TCS.2019.10.033. Jan.

[75] F. de la Rosa-Rivera, J.I. Nunez-Varela, J.C. Ortiz-Bayliss, H. Terashima-Marín, Algorithm selection for solving educational timetabling problems, Expert Syst. Appl. 174 (2021), 114694, https://doi.org/10.1016/J.ESWA.2021.114694. Jul.

[76] F. Arnold, K. Sörensen, What makes a VRP solution good? The generation of problem-specific knowledge for heuristics, Comput. Oper. Res. 106 (2019) 280–288, https://doi.org/10.1016/J.COR.2018.02.007. Jun.

[77] M.M. Nasiri, S. Salesi, A. Rahbari, N. Salmanzadeh Meydani, M. Abdollai, A data mining approach for population-based methods to solve the JSSP, Soft Comput. 23 (21) (2019) 11107–11122, https://doi.org/10.1007/S00500-018-3663-2/TABLES/11. Nov.

[78] Y. Bengio, E. Frejinger, A. Lodi, R. Patel, and S. Sankaranarayanan, "A learning-based algorithm to quickly compute good primal solutions for stochastic integer programs," Dec. 2019, [Online]. Available: http://arxiv.org/abs/1912.08112.

[79] F. Lucas, R. Billot, M. Sevaux, K. Sörensen, Reducing space search in combinatorial optimization using machine learning tools, in: Lecture Notes in Computer Science, 12096, Springer, 2020, pp. 143–150 (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) LNCSMay.

[80] N. Ghalavand, E. Khorram, V. Morovati, An adaptive nonmonotone line search for multiobjective optimization problems, Comput. Oper. Res. 136 (2021), https://doi.org/10.1016/j.cor.2021.105506.

[81] B. Ivorra, B. Mohammadi, A. Manuel Ramos, A multi-layer line search method to improve the initialization of optimization algorithms, Eur. J. Oper. Res. 247 (3) (2015), https://doi.org/10.1016/j.ejor.2015.06.044.

[82] X. Zhao, W. Lin, Q. Zhang, Enhanced particle swarm optimization based on principal component analysis and line search, Appl. Math. Comput. 229 (2014), https://doi.org/10.1016/j.amc.2013.12.068.

[83] M. Schonlau, R.Y. Zou, The random forest algorithm for statistical learning, Stata J. 20 (1) (2020), https://doi.org/10.1177/1536867X20909688.

[84] L. Breiman, Random forests, Mach. Learn. 45 (1) (2001), https://doi.org/10.1023/A:1010933404324.

[85] J.L. Speiser, M.E. Miller, J. Tooze, E. Ip, A comparison of random forest variable selection methods for classification prediction modeling, Expert Syst. Appl. 134 (2019), https://doi.org/10.1016/j.eswa.2019.05.028.

[86] S. Sundaramurthy, P. Jayavel, A hybrid grey wolf optimization and particle swarm optimization with C4.5 approach for prediction of rheumatoid arthritis, Appl. Soft Comput. J. 94 (2020), https://doi.org/10.1016/j.asoc.2020.106500.

[87] J.J. Liang, B.Y. Qu, and P.N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization," 2013. [Online]. Available: http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2014.

[88] J.J. Liang, B.Y. Qu, P.N. Suganthan, and Q. Chen, "Problem definitions and evaluation criteria for the CEC 2015 competition on learning-based real-parameter single objective optimization," 2014. [Online]. Available: http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2015/CEC2015.htm.

[89] N.H. Awad, M.Z. Ali, J. Liang, B.Y. Qu, and P.N. Suganthan, *Problem definitions and evaluation criteria for the CEC 2017 special session and competition on real-parameter optimization*, no. August. 2016.

[90] X. Meng, P. Zhang, Y. Xu, H. Xie, Construction of decision tree based on C4.5 algorithm for online voltage stability assessment, Int. J. Electr. Power Energy Syst. 118 (2020), https://doi.org/10.1016/j.ijepes.2019.105793.

[91] B.F. Tanyu, A. Abbaspour, Y. Alimohammadlou, G. Tecuci, Landslide susceptibility analyses using Random Forest, C4.5, and C5.0 with balanced and unbalanced datasets, Catena 203 (2021), https://doi.org/10.1016/j.catena.2021.105355 *(Amst)*.

[92] X. Tong, B. Yuan, B. Li, Model complex control CMA-ES, Swarm Evol. Comput. 50 (2019), https://doi.org/10.1016/j.swevo.2019.100558. Nov.

[93] J. Brest, M.S. Maučec, and B. Bošković, "Single objective real-parameter optimization: algorithm jSO," 2017. doi:10.1109/CEC.2017.7969456.

[94] J. Brest, M.S. Maučec, B. Bošković, IL-SHADE: improved L-SHADE algorithm for single objective real-parameter optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2016, pp. 1188–1195, https://doi.org/10.1109/CEC.2016.7743922. *CEC*Nov. 2016.
[95] M. Sebag and A. Ducoulombier, "Extending population-based incremental learning to continuous search spaces, 2022".
[96] F. Zhao, Z. Shao, R. Wang, C. Zhang, J. Wang, A hybrid EDA with Chaotic DE algorithm and its performance analysis, J. Comput. Inf. Syst. 11 (4) (2015) 1505–1512, https://doi.org/10.12733/jcis13564. Feb.
[97] K. Hinkelmann, *Design and analysis of experiments*, vol. 3. 2012. doi:10.1002/9781118147634.
[98] R. Salomon, *Re*-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. A survey of some theoretical and practical aspects of genetic algorithms, Biosystems 39 (3) (1996) 263–278, https://doi.org/10.1016/0303-2647(96)01621-8.
[99] A. Viktorin, R. Senkerik, M. Pluhacek, T. Kadavy, A. Zamuda, Distance based parameter adaptation for success-history based differential evolution, Swarm Evol. Comput. 50 (2019), 100462, https://doi.org/10.1016/J.SWEVO.2018.10.013. Nov.
[100] S. Das, P.N. Suganthan, and P.K. Rout, "Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems," 2010. [Online]. Available: http://www.esa.int/act.