IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 7, 2020, accepted September 19, 2020, date of publication September 24, 2020,
date of current version October 9, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3026339

Landscape-Based Similarity Check Strategy for Dynamic Optimization Problems

KANGJING LI, (Member, IEEE), SABER M. ELSAYED, (Member, IEEE), RUHUL SARKER, (Member, IEEE), AND DARYL ESSAM, (Member, IEEE)
School of Engineering and Information Technology, University of New South Wales at Canberra, Canberra, ACT 2600, Australia

Corresponding author: Kangjing Li (1414749039@qq.com)

This work was supported in part by the Australian Research Council Discovery Project under Grant DP190102637.

ABSTRACT In many real-world decision problems, there are specific objective, decision variables, conditions, data and/or parameters that may vary over time. When these problems are solved through an optimization process, they are generally known as dynamic optimization. Dynamic optimization is a challenging research topic as the optimal solution usually moves with any change in the problem environment. In locating the optimal point in any optimization problems, the effectiveness of the search process is influenced by the nature of the problem landscape. In order to improve the effectiveness of the search process, in this article, a new approach is developed by integrating a landscape-based strategy with appropriately designed evolutionary algorithms for solving dynamic problems. The proposed approach is named as Landscape Influenced Dynamic Optimization Algorithm (LIDOA). LIDOA checks the similarity before and after the changed environment which is then used as an input to guide the evolutionary search process. The dynamic benchmark functions from IEEE-CEC2009 are solved using LIDOA. LIDOA was able to enhance the performance of the evolutionary algorithms in their adaptation to dynamic changes, which in turn enhanced their ability to attain better results.

INDEX TERMS Fitness landscape, dynamic optimization, similarity check.

I. INTRODUCTION

The interest in optimization under dynamic environments have been growing over the years due to the fact that many real-world decisions and design problems belong to this category, such as trade market prediction, meteorological forecast, robotics motion control, computational protein design [1], intelligent watermarking in document images [2], and vehicle routing problems [3]. Common applications of dynamic optimization problems include: power system scheduling, production scheduling, and moving objects scheduling. In power system scheduling, the load on the electricity network and the generation from renewable resources changes with time. With the arrival of new jobs and experiencing defective materials and machine breakdowns, a scheduling problem changes with time. Supplying fuel to patrolling boats by a ship or aircraft, or delivering hazardous materials to multiple moving units by a robot are also typical real-world dynamic problems [4].

In general terms, optimization problems that change over time are called dynamic optimization problems (DOPs), which can be mathematically represented as [5]:

$$F = f(\vec{x}, \vec{\phi}, t) \tag{1}$$

where $\vec{x}$ is decision variable(s), $\vec{\phi}$ is parameter(s), and t is time. In DOPs, the objective function, constraints, data, and/or environmental conditions may change over time due to many different reasons. Therefore, the goal of DOPs is not only to find an optimal solution, but also to closely track the change of solutions over time [6].

In a dynamic environment, the key points for solving such problems are adaptability and efficiency. In fact, one of the main challenges in dealing with DOPs lies in reducing the computational cost in locating the new solution when there is a change in environment. In other words, a quick response to the changes and search for the optima in diverse environments are the basic requirements for methods toward these problems.

Over the last two decades, a good number of different evolutionary algorithms (EAs) have been proposed to solve DOPs. Compared with the conventional optimization methods, EAs are mostly simple to conduct the search operations and impose no requirement of specific mathematical properties, thus being more robust and flexible. For employment of

The associate editor coordinating the review of this manuscript and approving it for publication was Xiangtao Li.

EAs in DOPs, two key points are considered in this article: the optimization problems to be solved and the class of algorithms to be designed, where the key element of the former is landscape analysis and the latter frequently leads to type of the algorithm, such as framework imitating natural processes, infinite population models, ideas based on thermodynamics and statistical mechanics [7].

Landscapes analysis is one of the fundamental approaches to understand the geometry of the search space that can provide important information for the development of a search algorithm [1]. The simple landscape measures such as the ruggedness, peak number, height, separation, and clustering in the solution space, can be used to reflect the changes in the landscapes for dynamic problems [8]. These landscape indicators can be a useful component in designing the algorithm for DOPs. However, most existing landscape measures are based on static optimization problems, and have limited effects on improving the algorithm's performance.

In this article, the Landscape Influenced Dynamic Optimization Algorithm (LIDOA) is proposed, where a landscape-based strategy is integrated with appropriately designed EAs for solving DOPs. First, some points are selected from the search space based on the sequence-based deterministic initialization technique [9] and their fitness values are calculated. These points are then used to determine the landscape-based similarity indicators for comparing similarities in different environments. In this case, it is assumed that the search boundary will not change but the landscape will vary over time. For the convenience of comparisons, the landscape measures are preserved for different environments. When the fitness landscape changes, the features of the current landscape are compared with those of the previous ones based on the landscape measures. Depending on the level of similarities, the individuals from the past environments will be selected for the current one and this information would help to guide the search process in the current environment. Several classical EAs, including genetic algorithm (GA), self-adaptive differential evolution algorithm (jDE) [10] and covariance matrix adaptation evolution strategy (CMA-ES) [11], are employed to examine the efficiency of LIDOA.

The remainder of this article is organized as follows. An overview of EAs for DOPs and the basic theory of the methods employed in this article are presented in Section II and section III, respectively. The proposed LIDOA is detailed in Section IV. The analysis of the experimental results and the conclusion are provided in Sections V and VI, respectively.

II. BACKGROUND

This section reviews the popular evolutionary algorithm (EA) schemes that have been designed for DOPs. Compared to the stationary optimization problems, where the goal is to find the global optimum as fast as possible, DOPs require EAs to track the trajectory of changing optima in the search space. One of the biggest challenges that DOPs have in comparison to traditional EAs is the convergence issue: once converged, it's hard to escape from an old optimum [7]. Therefore, researchers have developed several approaches into EAs to enhance their performance for DOPs, including diversity, memory, prediction and multi-population schemes.

Diversity schemes refer to introducing diversity in population after a change happens, or maintaining diversity through the search process without explicitly detecting changes. A diversity scheme can be achieved by adapting search operators [3], introducing randomised individuals, distributing sentinel individuals [12], as well as explicitly keeping individuals from getting too close to one another [7].

Memory schemes preserve past solutions. In many cases, the best solutions from the previous landscapes are kept in memory for possible population initialization when a landscape changes [13]. The intermediate solutions and solutions from the latest problem landscapes can also comprise the memory [13], [14].

Prediction schemes can be used to track the moving optima [15], [16], locations that individuals should be re-initialized to [17], as well as the time when the next change will occur and which possible environments will appear in the next change [18].

Multi-population schemes concurrently maintain multiple sub-populations where each of them handles a different task. In this case, the sub-populations can use a single algorithm or multiple algorithms [19]. In order to improve the performance of multi-population schemes, additional strategies such as the quantum strategy [20] and scheduling approaches were employed [21].

Each of the above-mentioned schemes has its own advantages and disadvantages. Therefore, combinations of the above-mentioned schemes have been explored. The most popular method consists of a memory scheme and a multi-population scheme, where various strategies have been proposed [10], [22], [23]. Another commonly used method is the combination of a memory scheme and a diversity scheme, as was proposed in [24], [25]. In addition, prediction based on memory and/or multi-population approaches have been studied on multiple EAs [26], [27].

Despite that many methods have been proposed to deal with DOPs, very few of them consider using landscape information to direct the search process, and even fewer attempts have been made on what landscape information can be used for and how to use it. Among the works that employ landscape knowledge, they either learn from only the latest problem landscape, or seldom use the landscape information to direct the current search.

Vellasques *et al.* [2] proposed a series of memory-based dynamic PSO to deal with recurrent optimization problems. The basic idea is to build a comprehensive model of the stream of optimization problems by using representative problem instances [31]. In that, the optimal swarm obtained in each problem landscape is archived in two memories and re-evaluated when landscape changes. The re-evaluated solutions are then compared with the memory using a Kolmogorov-Smirnov statistical test to find a

TABLE 1. The advantages and disadvantages of commonly used EA schemes.

EA schemes	Advantages	Disadvantages
Diversity	Effective on continuous small or medium changes	Hard to know the diversity maintaining mechanisms and the optimal frequency for increasing diversity [28]
Memory	Effective on periodical changes; slow down the convergence [29]	Useless at significant changes [28]
Prediction	Effective on changes with regular patterns; effective on tracking and finding the global optima with accurate predictions; improving the convergence ability	Misleading under inaccurate predictions [16]; in need of sufficient training data; inefficient in fast finding/tracking [7]
Multi-population	Maintaining diversity at the global level [28]; managing multiple optima in parallel; effective on competing peaks and multimodal problems [30]	Slow search; generally high computational costs; hard to assign tasks effectively and divide sub-populations appropriately

historical landscape that is similar to the current one. If a similar historical landscape is found, the optimal swarm from that landscape is used directly to save computational time. Otherwise, L-best dynamic PSO is employed with the optimal swarm of the latest landscape as an initial population. In [31], they modified the algorithm by introducing a Gaussian mixture model built on all of the solutions to represent the problem landscape. To be specific, at the end of each environment, a short term memory stores the best individual in the last environment and the density estimation generated by the Gaussian mixture model. Then at the beginning of the changed environment, a population is re-sampled based on the density estimation and is re-evaluated. The distribution similarity between the re-evaluated population and that in the last environment is compared by using a Kolmogorov-Smirnov test, which will determine if the best solution from the last environment will be re-used. In addition, a long term memory stores the information of multiple historical environments is kept for the same purpose. Then in [32], they further modified the algorithm by using a surrogated-based PSO as an optimization method, with a portion of one population using exact fitness evaluations and the other one using a regression model for fitness.

It is worth noting that although this algorithm employs a similarity check based on the problem landscape, it is different from LIDOA in several aspects: 1) the information used to represent the problem landscape and the way to compare similarity are quite different, where the sequence-based deterministic initialization [9] is employed to generate the problem landscape in this article; 2) the optimal swarm from the similar historical landscape is used directly without being further optimized; 3) the algorithm mainly focuses on the reduction of computational cost instead of improving the fitness error, which is the other way around in this article; 4) only recurrent optimization problems, especially the document images and watermarking systems, are examined, while different functions and change types are considered in this article. These aspects are also significant components that make the proposed similarity check strategy efficient.

In order to quickly find or closely track the optima in a changing environment, it is important to efficiently use information from the current problem environment and previous environments. Although EAs are reasonably efficient for solving DOPs in each environment separately, it is still an open research issue for developing a framework that can fully use the environment information and generally benefit EAs in dealing with DOPs.

III. BASIC THEORY

A. FITNESS LANDSCAPE MEASURES

The concept of fitness landscapes originated from theoretical biology in 1932 [33]. In evolutionary computation, fitness landscapes are used to draw an analogy with real landscapes to gain a better understanding of how and where algorithms operate [34], [35]. Within this generic model, a location in the landscape is a solution to a given problem, the elevation captures how good that solution is, and solutions that are similar in nature are typically placed close to each other [36].

Fitness landscapes play a vital role in understanding the search space of optimization problems and in guiding the search process to reach an optimum solution. Despite the attempts that have been made in using landscape information to solve DOPs, there are still many issues to be addressed in terms of what characteristics of problem landscapes are useful and how to use them. In a continuously changing environment, the similarities between two problem landscapes indicate certain resemblances between fitness distributions. In this article, fitness landscape measures are used to check the similarities between different problem environments. The purpose of this similarity check is to identify the extent of changes between environments which can then be used to decide the search strategy in solving the problem in subsequent environments. Any similarity found will provide an opportunity to re-use some of the preserved solutions and any dissimilarity will provide a clue of how to generate the population and conduct the search process more effectively. Therefore, such an action will help the algorithm to find a solution with lower computational time, hence improving the performance of the algorithm.

As previously mentioned, the main goal of the similarity check strategy is to find the most similar historical landscape to the current one, so that historical information can be re-used to guide the search process. This strategy is conducted under the assumption that overlapping areas exist among different landscapes. To do so, in this article, several landscape measures are employed, including the Euclidean distance

(1-norm) between two landscapes, statistic kurtosis, statistic skewness, as well as the domination landscape matrix [37].

In the Euclidean-based landscape measure, the Euclidean distance between the landscape fitness of the current landscape and that of previous ones are calculated. The historical landscape with the shortest distance is taken as the most similar landscape. Kurtosis and skewness are measures to the symmetry and tailedness of distribution of a landscape fitness, respectively. There are different methods to calculate the two measures, with the following equations used in this article:

$$\text{kurtosis} = \frac{\sum_{i=1}^{N}(Y_i - \overline{Y})^4/N}{\sigma^4} \tag{2}$$

$$\text{skewness} = \frac{\sum_{i=1}^{N}(Y_i - \overline{Y})^3/N}{\sigma^3} \tag{3}$$

where Y_i indicates the fitness of each solution in the landscape population, $\overline{Y}$ is the mean value of all the Y_i, σ is the corresponding standard deviation, and N is the population size.

The domination landscape (DL) describes the domination relationship among solutions, which can be used in determining the parameters for fitness scaling, the EA's convergence robustness against fitness noise, as well as classifying the optimization problems [37]. In this article, the domination landscape matrix is established based on the performance of each solution. The domination relationship between n solutions can be set as a n-by-n matrix:

$$DL = \begin{pmatrix} 0 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \cdots & 0 \end{pmatrix} \tag{4}$$

where the element in the ith row and jth column is $DL_{i,j} = 1$ when the ith solution is better than the jth solution, i.e., the ith solution dominates the jth solution, otherwise $DL_{i,j} = 0$. The Hamming distance between the DL matrix in the current landscape and that in each of the previous landscapes are then calculated, and the most similar historical landscape is accordingly selected.

B. CLASSICAL EVOLUTIONARY ALGORITHMS

EAs have been proved to be effective in dealing with DOPs. This article uses three classical EAs, i.e., GA, jDE and CMA-ES, to examine the performance of the landscape-based strategy. The working mechanisms of the three algorithms are introduced in this section.

1) GA

GA is a randomized optimization algorithm inspired by natural selection and natural genetics [38]. It starts with initializing a population of individuals, where each individual represents a solution within the search range. Then the fitness of the initialized population is evaluated, according to which certain individuals are selected as parent individuals to generate new individuals for the next generation. In most cases, an individual with better fitness is more likely to be selected.

TABLE 2. Genetic algorithm (GA) operations.

Parent individuals	individual 1	1	2	3	4	5	6
	individual 2	11	12	13	14	15	16
Crossover	individual 1	**11**	**12**	3	4	5	6
	individual 2	**1**	**2**	13	14	15	16
Mutation	individual 1	11	12	3	**10**	5	6
	individual 2	1	2	13	14	15	**9**

After selection, crossover and mutation are conducted on the parent individuals to produce new individuals. Crossover is a procedure of swapping randomly selected information between two individuals. This is exemplified in Table 2, where the first two elements of two individuals are swapped. Mutation alters one or more elements in an individual to modify a solution. For example, in Table 2, the fourth elements of individual 1 and the last element of individual 2 are changed to be different values. The routine of selection, crossover and mutation is repeated in each generation until the termination is satisfied. Many other methods have been proposed to implement crossover and mutation, but the basic idea is to diversify the individuals.

2) jDE

DE is a powerful stochastic optimization algorithm [39], and its computational steps include initialization, mutation, crossover and selection. Based on DE, jDE employs self-adaptive control parameters to perform better in numerical optimization problems [40]. jDE was further modified to adapt to dynamic environments [10], which is how it is used in this article. The procedure of jDE, as used by this article, is as follows.

At the beginning, a population consisting of PS vectors is randomly initialized, where each vector represents a solution to the optimization problem. The mutation strategy is then used to create one donor vector v for each vector x, also known as target vector. A basic form of mutation strategy on the $k$the donor vector is as follows:

$$v_{k,G+1} = v_{r_1,G} + F \cdot (x_{r_2,G} - x_{r_3,G}) \tag{5}$$

where randomly selected integers $r_1, r_2, r_3 \in [1, PS]$, $r_1 \neq r_2 \neq r_3 \neq k$, v_k is the donor vector corresponding to the kth target vector x_k, $k = 1, 2, \ldots, PS$, and G is the number of generations. In jDE, the control parameter F is self-adapted:

$$F_{k,G+1} = \begin{cases} F_l + rand_1 F_u, & \text{if } rand_2 < \tau_1 \\ F_{k,G}, & \text{otherwise} \end{cases} \tag{6}$$

where F_l and F_u are fixed values, $rand_1$ and $rand_2$ are uniform random values in [0, 1], and τ_1, τ_2 represent probabilities to adjust F [41].

After the donor vector v_k is obtained, it's components are recombined with the corresponding target vector x_k to generate the trail vector u_k. This procedure is called crossover, which can enhance the potential diversity of the population [42]. The crossover rate CR is also a self-adapted parameter:

$$CR_{k,G+1} = \begin{cases} rand_3, & \text{if } rand_4 < \tau_2 \\ CR_{k,G}, & \text{otherwise} \end{cases} \tag{7}$$

where $rand_3$ and $rand_4$ are uniform random values in [0, 1].

Afterwards, the performance of x_k and u_k is compared and the better one is selected to survive to the next iteration. The mutation, crossover and selection strategies are repeated until the terminal criterion is satisfied.

On top of the aforementioned procedures, a multi-population strategy without information sharing, an aging strategy to deal with local optima, and an archive to keep the best individuals were introduced to adapt to dynamic environments. More details of jDE can be found in [10].

3) CMA-ES

CMA-ES is an evolutionary algorithm specifically designed for black-box continuous optimisation problems [43]. It has similar procedures as GA and DE, but with different operations.

In CMA-ES, a new population is initialized by sampling candidate solutions based on a multivariate normal distribution. Then new solutions are generated by adding a normally distributed random vector, which amounts to mutation. The generated solutions are evaluated and the ones with better fitness are selected as the parent solutions in the next generation. In this process, correlations between the variables are defined by a covariance matrix, which is updated by using covariance matrix adaptation (CMA) in CMA-ES. In addition, the mean of the distribution is updated on the basis of maximizing the likelihood of previously successful solutions. More details of CMA-ES can be found in [44].

C. SEQUENCE-BASED DETERMINISTIC INITIALIZATION

Sequence-based deterministic initialization is a heuristic space-filling approach to improve the performance of EAs [9]. This initialization technique takes into account both the function to be optimized and the search space, and shows promising performance in providing useful information about the problem's behavior as well as producing high-quality solutions.

To generate the initial population, the search domain $[\underline{\vec{x}}, \overline{\vec{x}}]$ is evenly divided into $q+1$ segment vectors. The interval $\vec{I}$ is determined accordingly:

$$\vec{I} = \frac{\overline{\vec{x}} - \underline{\vec{x}}}{q} \tag{8}$$

The sequence of segment vectors is started with the lower bound vector of all the decision variables, and ended with the upper bound vector of all the variables.

The kth segment vector $\vec{S}_k$ is generated based on the following equation:

$$\vec{S}_k = \begin{cases} \underline{\vec{x}} & k = 1 \\ \vec{S}_{k-1} + \vec{I} & 2 \leq k \geq q \\ \overline{\vec{x}} & k = q+1 \end{cases} \tag{9}$$

Therefore, the maximum number of individuals that can be generated is:

$$(q+1) \times ((D \times q) + 1) \tag{10}$$

The details in terms of parameter settings and further analysis can be found in [9].

IV. DESIGN OF ALGORITHM

In many real-world applications, an environmental change happens gradually instead of abruptly; therefore, overlapping areas generally exist among changed landscapes [22]. Accordingly, the knowledge learnt in historical landscapes can be re-used to guide the search process in the current landscape under limited computational resources and running time. This article proposes an algorithm based on a similarity check strategy to efficiently deal with DOPs with limited resources, which is called Landscape Influenced Dynamic Optimization Algorithm (LIDOA) in this article.

A. LANDSCAPE INFLUENCED DYNAMIC OPTIMIZATION ALGORITHM (LIDOA)

The framework of LIDOA has three main components. They are initialization, similarity check, and search process. In the first component, the sequence-based deterministic initialization [9] method is used to generate a population of a fixed number of individuals which are used to determine the fitness landscape measure in different environments. The similarity check strategy is then conducted to judge how similar the previous landscapes are to the current one, with the most similar one then selected to utilize the learnt knowledge from it in the optimization process. Finally, the evolutionary process is applied to generate solutions. Considering the general nature of the framework, any evolutionary algorithm can be applied as the search process, within this framework. Each of these components are discussed in more details below.

The steps of LIDOA are presented in Algorithm 1. It is noted that the similarity check strategy is not employed on the first several landscapes, as there would not be sufficient information to use it.

Firstly, EA parameters are initialized and an archive *Hist* to store the best found solutions in each problem environment is set. By using the sequence-based deterministic initialization approach, an initial population X_{land} is generated, and is recognized as the 'landscape population', which remains the same for calculation of landscape measures in all environments.

As shown in line 5 in Algorithm 1, at the beginning of each new environment, the landscape population X_{land} is evaluated and the corresponding fitness landscape measure $FL(l)$ is calculated and archived for the lth environment.

The search process in any environment starts with an initial random population but one can also use the landscape population as well. In this article, for the first few environments, the landscape similarity strategy is not applied as the comparison will not be useful until a reasonable amount of historical information is collected. The number of problem

Algorithm 1 Landscape Influenced Dynamic Optimization Algorithm (LIDOA)

```
1: Initialize EA parameters; set the archive $Hist$ for historical solutions
2: Generate landscape population $X_{land}$ using sequence-based deterministic initialization
3: Set landscape index $l = 0$
4: for $l = 1 : l_{max}$ do
5:     Evaluate $X_{land}$ to obtain the fitness landscape information $FL(l)$ of the $l$th problem
6:     Randomly initialize population $X$
7:     if $l > 6$ then
8:         Compare $FL(l)$ with $FL(i), i = 1, 2, \cdots, l-1$
9:         Select the $idx$th landscape that is most similar to the $l$th landscape
10:        Use $Hist(idx)$ to replace the same number of randomly selected solutions in $X$
11:    end if
12:    EA to search for the optimal solution
13:    Archive the optimal solution(s) in $Hist(l)$
14: end for
```

environments for this purpose can be set based on the variability of the environments, but this article employs 1/10 of the total number of environments, i.e. 6 when there are 60 environments in total.

When the proposed similarity check strategy is applied, the fitness landscape measure of the current problem $FL(l)$ is compared with each of the previous problems $FL(i)$, $i = 1, 2, \cdots, l-1$. Out of $l-1$ problem landscapes, the one with index idx that is most similar to the current problem can be selected. Afterwards, the best solutions archived from the selected problem $Hist(idx)$ are used to replace randomly selected initial individuals (usually one or a few only) in the current problem. In this way, the knowledge learnt from previous landscapes is conveyed and that benefits the current search process.

The above strategy is a general procedure that can be integrated with any population based algorithms to enhance their performance when solving dynamic optimization problems.

B. COMPLEXITY ANALYSIS

In this section, the computational complexity of LIDOA is discussed.

The major components of LIDOA are the initialization of the landscape population, the evaluation of landscape population, the similarity check between the current fitness landscape with the previous landscapes, and the re-use of the learnt knowledge in the search process.

According to Algorithm 1, the time complexity for the initialization of landscape population is $O(S)$, where S is the size of the landscape population (X_{land}). Note that this is done only once at the beginning of the process. At the beginning of each new problem environment, the landscape population is evaluated which requires $O(S)$. It should be noted that the fitness landscape is a vector with length L, so the time complexity of each similarity check is $O(L)$ for each problem environment, which applies to each of the four landscape measures presented in subsection III-A. In terms of re-using knowledge, a certain number of current initial solutions are replaced by the archived best ones in $O(n)$ time, where n is the number of replaced solutions, that can be one or more. In summary, the computational complexity of LIDOA in each generation where it is applied is $O(S)$, assuming S is bigger than L, which is fairly insignificant compared to the overall computational complexity required by the other processes of EAs.

The space required by LIDOA includes one fixed landscape population for all problem environments, one fitness landscape measure and one solution for each problem environment. Considering that the fitness landscape is a single row vector, the space taken by adding the proposed method to EAs is minor.

V. EXPERIMENTS

This section examines the performance of LIDOA. The employed benchmark problems are introduced. Then four landscape measures are evaluated, and the best one is selected to show the efficiency of LIDOA on several classical EAs. To conclude this section, the influence of two parameters on LIDOA is analysed.

The algorithms are implemented with C++. The processor is i7-8700, the processor speed is 3.20 GHz and the installed random-access memory is 64.0 GB.

A. BENCHMARK PROBLEMS

The generalized dynamic benchmark generator (GDBG) from the CEC'2009 competition on dynamic environments [45] were employed, including seven test functions. The first two functions are rotation peak functions with 10 peaks and 50 peaks, respectively, while the other five functions are composition of Sphere's function, composition of Rastrigin's function, composition of Griewank's function, composition of Ackley's function and a hybrid composition function.

GDBG tunes the system control parameters, such as peak heights, peak widths and peak positions, so that deviations are introduced on the solution distribution. It can be represented as follows [45]:

$$\phi(t+1) = \phi(t) \oplus \Delta\phi \tag{11}$$

where t is the real-world time, ϕ is the system control parameter, and $\Delta\phi$ is a deviation from the current system control parameters.

Six change types are covered for the dynamism of system control parameters. The framework of the dynamic changes are described as follows [45]:

1. small step:

$$\Delta\phi = \alpha \cdot \|\phi\| \cdot r \cdot \phi_{severity} \tag{12}$$

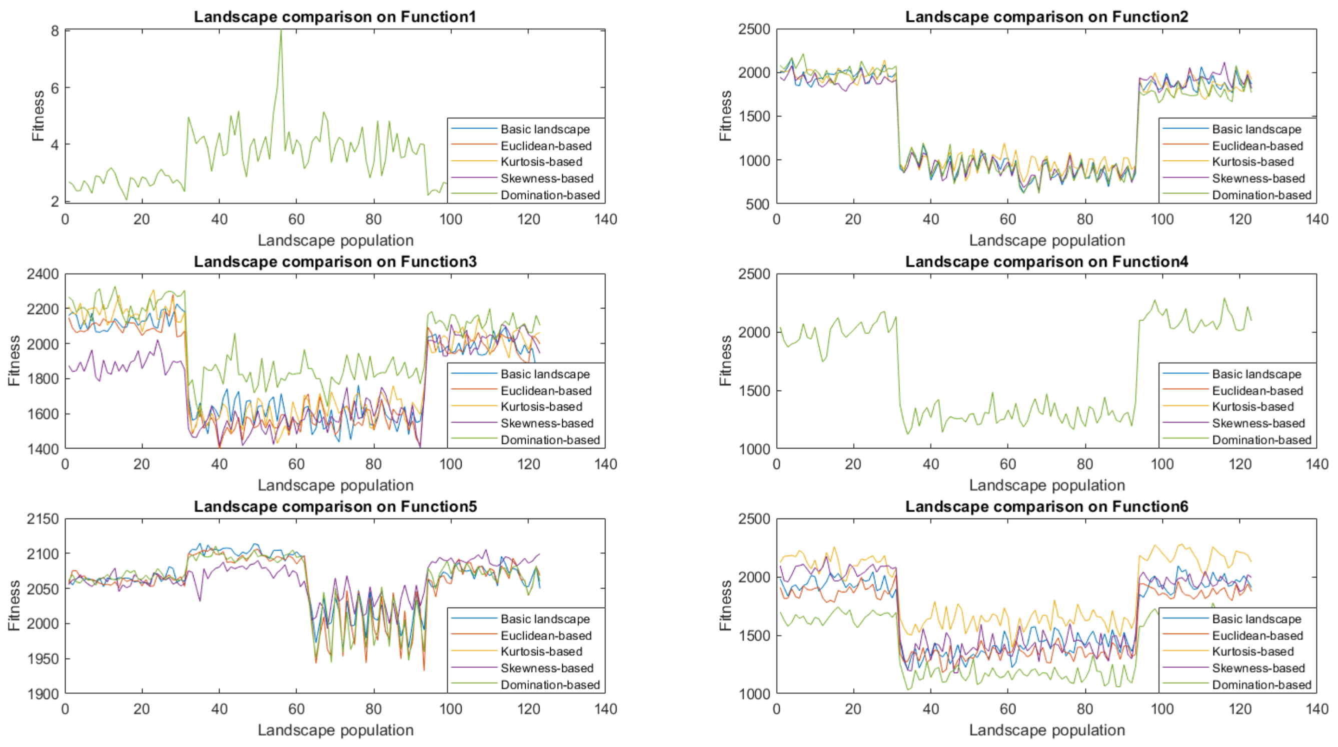

FIGURE 1. Comparison of landscape measures.

2. large step:

$$\Delta\phi = \|\phi\| \cdot (\alpha \cdot sign(r) + (\alpha_{max} - \alpha) \cdot r) \cdot \phi_{severity} \quad (13)$$

3. random:

$$\Delta\phi = N(0, 1) \cdot \phi_{severity} \quad (14)$$

4. chaotic:

$$\phi(t+1) = A \cdot (\phi(t) - \phi_{min}) \cdot (1 - (\phi(t) - \phi_{min})/\|\phi\|) \quad (15)$$

5. recurrent:

$$\phi(t+1) = \phi_{min} + \|\phi\|(sin(\frac{2\pi}{P}t + \varphi) + 1)/2 \quad (16)$$

6. recurrent with noise:

$$\phi(t+1) = \phi_{min} + \|\phi\|(sin(\frac{2\pi}{P}t + \varphi) + 1)/2 + N(0, 1) \cdot noisy_{severity} \quad (17)$$

where $\|\phi\|$, $\phi_{severity}$ and ϕ_{min} are the change range, the change severity and the minimum value of ϕ, respectively. In addition, α and α_{max} are constant values, r is a random number between $(-1, 1)$, and $noisy_{severity}$ is the noisy severity. $sign(x)$ is a sign function of x, and $N(0, 1)$ denotes a random number which obeys to the standard normal distribution. For chaotic change, A is a positive constant value between (1.0, 4.0). For the recurrent change and recurrent with noise change, P is the period, φ is the initial phase.

TABLE 3. Parameter settings.

Parameter	Value
Number of dimensions	10
Maximal fitness evaluations change	$10,000 * 10$
Number of changes	60
Search range	$[-5, 5]^n$
Period p	12
Noisy severity $noisy_{severity}$	0.8
Chaotic constant A	3.67
Step severity α	0.04
Maximum of α (α_{max})	0.1

The parameter settings are presented in Table 3. Each experiment was run for 20 times. The mean value of the best fitness error obtained over 60 landscapes were calculated, then the average of the mean value over 20 runs were employed as fitness error for further analyses.

B. LANDSCAPE MEASURES ANALYSIS

Before analysing the performance of LIDOA, the four landscape measures (Euclidean distance, domination landscape matrix, skewness and kurtosis) are examined. In order to check whether the landscape measures select the similar historical landscape, the landscape fitness values of the current landscape and selected historical landscapes are given. Six benchmark instances are exemplified in Figure 1, including function 1 (number of peaks = 50) with small changes, function 2 with large changes, function 3 with random changes, function 4 with chaotic changes, function 5 with recurrent changes, and function 6 with recurrent with noise changes.

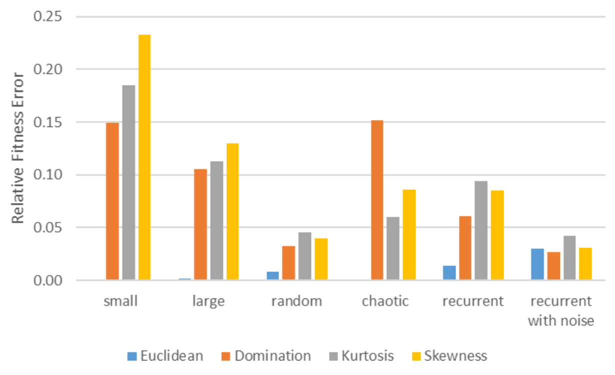

FIGURE 2. Relative fitness errors regarding change types by GA with landscape measures.

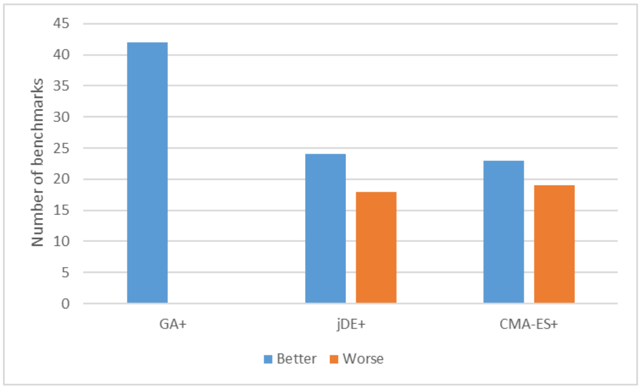

FIGURE 3. Effect of the euclidean-based measure on EAs.

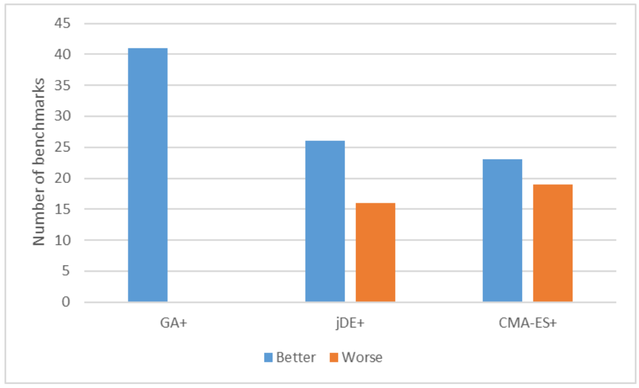

FIGURE 4. Effect of the domination-based measure on EAs.

TABLE 4. The Wilcoxon test between EAs, with and without similarity check strategy.

Significance	Euclidean	Domination matrix
GA versus GA+	0.00	0.00
jDE versus jDE+	0.52	0.32
CMA-ES versus CMA-ES+	0.43	0.69

TABLE 5. Fitness improvement rates of EAs with Euclidean distance and domination matrix landscape measures on different change types.

	Change Type	GA	jDE	CMA-ES
Euclidean distance	small	**0.37**	-0.01	0.33
	large	**0.20**	-0.02	0.09
	random	0.09	**0.39**	0.06
	chaotic	**0.42**	0.02	-0.07
	recurrent	**0.23**	-0.08	0.03
	recurrent with noise	**0.05**	-0.19	-0.20
Domination matrix	small	0.23	0.11	**0.27**
	large	0.10	**0.43**	-0.28
	random	**0.07**	-0.16	0.01
	chaotic	**0.27**	-0.30	0.20
	recurrent	**0.19**	0.17	0.08
	recurrent with noise	0.05	**0.26**	0.01

TABLE 6. The advantages of the Euclidean-based measure over the domination-based measure.

Change Type	GA	jDE	CMA-ES
small	+	-	+
large	+	-	+
random	+	+	+
chaotic	+	+	-
recurrent	+	-	-
recurrent with noise	-	-	-

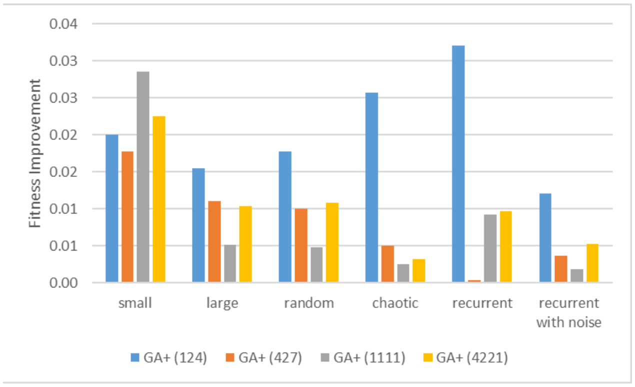

FIGURE 5. Relative fitness errors regarding change types with similarity check strategy based on Landscape population sizes of 124, 427, 1111 and 4221.

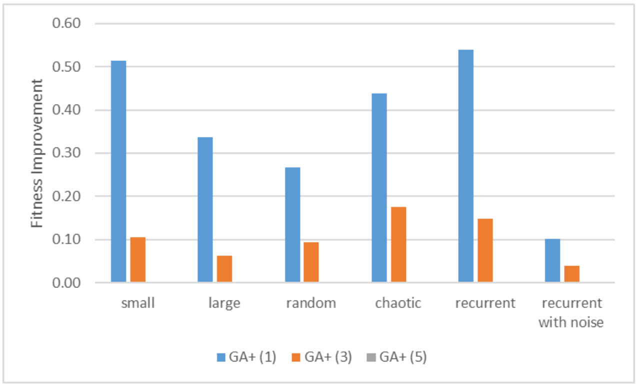

FIGURE 6. Relative fitness errors regarding change types with similarity check strategy based on 1, 3 and 5 historical best solutions.

Due to limited space, only the results from the 40th landscape as the current landscape, are presented in the paper.

For functions 1 and 4, historical landscapes selected by the four measures are all very similar to the current landscape. In terms of the other four functions, the landscapes selected by the Euclidean-based measure are generally the ones most similar to the current landscape, while the kurtosis-based, skewness-based and domination-based landscape measures

TABLE 7. Relative fitness errors achieved by GA with each landscape measure.

Function	Change Type	GA+Euclidean	GA+domination	GA+kurtosis	GA+skewness
F1	small	0.00	0.00	0.01	0.01
(10 peaks)	large	0.00	0.00	0.00	0.00
	random	0.00	0.00	0.00	0.00
	chaotic	0.00	0.01	0.01	0.01
	recurrent	0.00	0.00	0.00	0.00
	recurrent with noise	0.00	0.00	0.00	0.00
F1	small	0.00	0.00	0.00	0.00
(50 peaks)	large	0.00	0.01	0.00	0.00
	random	0.00	0.00	0.00	0.00
	chaotic	0.00	0.00	0.01	0.01
	recurrent	0.00	0.01	0.01	0.01
	recurrent with noise	0.00	0.00	0.00	0.00
F2	small	0.00	0.06	0.07	0.10
	large	0.00	0.05	0.05	0.07
	random	0.00	0.01	0.02	0.03
	chaotic	0.00	0.04	0.01	0.01
	recurrent	0.00	0.02	0.03	0.05
	recurrent with noise	0.00	0.00	0.01	0.02
F3	small	0.00	0.00	0.00	0.00
	large	0.00	0.00	0.00	0.00
	random	0.00	0.00	0.00	0.00
	chaotic	0.00	0.02	0.00	0.01
	recurrent	0.01	0.01	0.02	0.00
	recurrent with noise	0.02	0.02	0.02	0.00
F4	small	0.00	0.05	0.06	0.08
	large	0.00	0.03	0.04	0.04
	random	0.00	0.01	0.01	0.01
	chaotic	0.00	0.03	0.01	0.02
	recurrent	0.00	0.01	0.01	0.01
	recurrent with noise	0.01	0.01	0.01	0.00
F5	small	0.00	0.00	0.01	0.00
	large	0.00	0.00	0.00	0.00
	random	0.00	0.00	0.00	0.00
	chaotic	0.00	0.03	0.01	0.01
	recurrent	0.00	0.00	0.00	0.00
	recurrent with noise	0.00	0.00	0.00	0.00
F6	small	0.00	0.03	0.03	0.03
	large	0.00	0.01	0.01	0.01
	random	0.00	0.01	0.01	0.00
	chaotic	0.00	0.03	0.01	0.01
	recurrent	0.00	0.01	0.02	0.01
	recurrent with noise	0.00	0.00	0.00	0.01
Sum of relative fitness error		0.05	0.53	0.54	0.60

show unstable performances. With regard to change types, the Euclidean-based measure is able to find similar historical landscapes for all the six changes. On the other hand, the domination-based measure is less competent on both random changes and recurrent with noise changes. The kurtosis-based and skewness-based measures have difficulty in handling recurrent with noisy changes and random changes, respectively. It can be concluded that the Euclidean-based measure characterizes problem landscapes in a more comprehensive way in comparison with the other three landscape measures.

In addition, the overall performance of the four landscape measures were examined and compared on all benchmark instances based on GA. The relative fitness error of each landscape measure on the benchmark instances was calculated by the following formula:

$$F_{relative} = \frac{f_i - f_{min}}{f_{min}} \tag{18}$$

where f_i is the fitness error obtained by the ith method, $\forall i = 1, 2, 3, 4$, and f_{min} the best fitness error obtained by f_i. The smaller the sum of relative fitness error, the better the corresponding algorithm performs.

GA with Euclidean distance as its landscape measure obtained the best fitness on 30 out of 42 instances with the sum of relative fitness error 0.05. GA with domination matrix outperforms other methods on 7 out of 42 instances with the sum of relative fitness error being 0.53, which is slightly better than GA with kurtosis and GA with skewness as their landscape measure, respectively. The relative fitness error obtained by GA with each landscape measure on different instances can be found in Appendix.

The relative fitness error regarding change types is presented in Figure 2. The Euclidean-based measure was the best on all change types except for recurrent with noise changes, on which the domination-based measure outperforms the other three landscape measures. In addition, the domination-based measure ranked second on small changes, large changes, random changes and recurrent changes. Therefore, both Euclidean distance and domination landscape matrix will be employed to review the efficiency of the proposed LIDOA.

TABLE 8. Fitness errors and fitness improvement rates of EAs, with and without Euclidean distance landscape measure.

Function	Change Type	GA	GA+	FI	jDE	jDE+	FI	CMA-ES	CMA-ES+	FI
F1	small	80.29	**78.88**	0.02	0.24	0.30	-0.25	82.62	**81.16**	0.02
(10 peaks)	large	77.40	**76.86**	0.01	3.50	**3.39**	0.03	78.91	79.81	-0.01
	random	70.61	**70.40**	0.00	3.66	**3.38**	0.07	72.04	**70.77**	0.02
	chaotic	92.59	**91.37**	0.01	0.98	1.00	-0.02	93.34	93.39	0.00
	recurrent	81.49	**81.08**	0.01	1.94	**1.87**	0.04	82.95	**82.88**	0.00
	recurrent with noise	91.97	**91.83**	0.00	2.21	2.62	-0.18	93.02	**92.85**	0.00
F1	small	85.02	**83.55**	0.02	0.92	1.00	-0.09	85.52	86.17	-0.01
(50 peaks)	large	78.40	**77.80**	0.01	4.08	5.03	-0.23	80.68	**80.34**	0.00
	random	72.67	**72.36**	0.00	4.83	**4.77**	0.01	74.02	74.68	-0.01
	chaotic	92.20	**90.44**	0.02	1.03	**0.94**	0.09	92.13	92.25	0.00
	recurrent	80.45	**80.10**	0.00	1.09	1.17	-0.07	82.43	**82.37**	0.00
	recurrent with noise	91.25	**91.15**	0.00	2.44	**2.32**	0.05	91.78	**91.53**	0.00
F2	small	646.62	**569.06**	0.12	3.19	3.84	-0.20	35.35	40.70	-0.15
	large	800.03	**737.13**	0.08	47.97	**46.24**	0.04	212.30	**197.12**	0.07
	random	746.22	**717.28**	0.04	39.87	41.05	-0.03	163.10	168.74	-0.03
	chaotic	779.24	**707.86**	0.09	3.94	**3.83**	0.03	41.21	44.32	-0.08
	recurrent	739.85	**672.14**	0.09	80.42	**80.39**	0.00	198.97	205.82	-0.03
	recurrent with noise	855.29	**840.26**	0.02	4.01	**3.34**	0.17	48.65	53.54	-0.10
F3	small	1124.00	**1106.09**	0.02	15.89	**15.42**	0.03	999.23	**992.46**	0.01
	large	1378.54	**1371.23**	0.01	589.99	**554.16**	0.06	1116.78	1122.72	-0.01
	random	1316.04	**1315.96**	0.00	565.04	**554.39**	0.02	1078.92	**1076.61**	0.00
	chaotic	1326.61	**1239.29**	0.07	89.67	**83.29**	0.07	1165.44	**1159.48**	0.01
	recurrent	1350.10	**1329.62**	0.02	475.52	481.76	-0.01	1115.55	**1112.73**	0.00
	recurrent with noise	1419.88	**1419.73**	0.00	227.72	236.56	-0.04	1226.04	1235.22	-0.01
F4	small	707.57	**635.82**	0.10	4.54	**4.36**	0.04	65.02	**42.66**	0.34
	large	885.26	**823.87**	0.07	55.45	68.52	-0.24	268.94	282.30	-0.05
	random	821.63	**794.77**	0.03	69.91	**54.93**	0.21	232.36	233.72	-0.01
	chaotic	845.49	**769.56**	0.09	4.24	4.93	-0.16	57.47	69.63	-0.21
	recurrent	815.49	**761.55**	0.07	102.20	105.70	-0.03	300.70	**286.24**	0.05
	recurrent with noise	931.39	**917.81**	0.01	4.85	5.36	-0.11	67.44	76.95	-0.14
F5	small	1967.45	**1919.90**	0.02	1.88	**1.54**	0.18	54.27	**40.35**	0.26
	large	1975.94	**1960.33**	0.01	2.36	**2.03**	0.14	36.45	**34.56**	0.05
	random	1968.17	**1962.78**	0.00	1.92	**1.89**	0.01	32.77	**30.24**	0.08
	chaotic	1977.81	**1871.87**	0.05	1.63	**1.44**	0.12	51.45	**46.10**	0.10
	recurrent	1971.81	**1951.16**	0.01	1.47	**1.42**	0.03	28.44	29.50	-0.04
	recurrent with noise	1994.54	**1989.01**	0.00	0.94	1.00	-0.06	48.45	**45.92**	0.05
F6	small	1023.95	**948.55**	0.07	12.45	**8.98**	0.28	283.49	322.37	-0.14
	large	1193.92	**1167.30**	0.02	17.86	**14.69**	0.18	559.46	**541.99**	0.03
	random	1144.13	**1131.17**	0.01	13.77	**12.59**	0.09	543.45	**534.58**	0.02
	chaotic	1131.44	**1037.51**	0.08	9.56	10.54	-0.10	431.40	**385.86**	0.11
	recurrent	1160.34	**1116.18**	0.04	14.19	14.68	-0.03	528.28	**503.24**	0.05
	recurrent with noise	1215.34	**1202.38**	0.01	9.14	9.23	-0.01	458.87	460.36	0.00
Sum of fitness improvement				**1.36**			0.10			0.24

C. EFFECT OF LIDOA ON DIFFERENT EAs

To assess the performance of LIDOA, the similarity check strategy is adopted with three different algorithms (GA, jDE and CMA-ES) and compared with those without the proposed approach. The versions with LIDOA are named with '+' at the end, i.e., GA+, jDE+ and CMA-ES+. Note that, due to its good results obtained above, only the landscape measures based on Euclidean distance and domination matrix are used.

In order to analyse the overall performance, the fitness improvement achieved in each problem is calculated using Eq (19), that is, the normalised difference in both fitness errors, where a positive value of FI means the EA with LIDOA is better.

$$FI = \frac{X - X_+}{X} \tag{19}$$

where X is the fitness error achieved by the original algorithm, and X_+ the corresponding fitness error obtained by EA with LIDOA. FI is calculated on each of the 42 benchmark instances. The greater the fitness improvement FI, the better the corresponding method performs.

The comparison between EAs and EAs with a landscape-based strategy is presented in Figures 3 and 4. LIDOA with the Euclidean-based measure improves the performance of GA, jDE and CMA-ES on 42, 24 and 23 instances out of 42 instances, respectively. In other words, LIDOA enhances all of the three examined EAs on more than half the cases. In addition, the sum of FI indicates that the similarity check strategy has overall advantages in dealing with dynamic optimization problems. The detailed fitness improvement on each benchmark instance can be found in Appendix.

The performance of LIDOA is further analysed by using a Wilcoxon test. The Wilcoxon test was proposed by Frank Wilcoxon [46] to evaluate the difference between two sets of data. For two data sets X and Y with size n, the procedures to implement the Wilcoxon signed-ranks test are as follows:

TABLE 9. Fitness errors and fitness improvement rates of EAs, with and without domination matrix landscape measure.

Function	Change Type	GA	GA+	FI	jDE	jDE+	FI	CMA-ES	CMA-ES+	FI
F1	small	80.29	**79.21**	0.01	0.24	0.25	-0.03	82.62	**81.61**	0.01
(10 peaks)	large	77.40	**76.87**	0.01	3.50	**3.37**	0.04	78.91	**78.61**	0.00
	random	70.61	**70.33**	0.00	3.66	**3.55**	0.03	72.04	**71.20**	0.01
	chaotic	92.59	**91.92**	0.01	0.98	1.10	-0.13	93.34	93.53	0.00
	recurrent	81.49	**81.02**	0.01	1.94	**1.93**	0.01	82.95	83.18	0.00
	recurrent with noise	91.97	**91.84**	0.00	2.21	2.30	-0.04	93.02	**92.90**	0.00
F1	small	85.02	**84.27**	0.01	0.92	**0.88**	0.05	85.52	86.16	-0.01
(50 peaks)	large	78.40	**77.87**	0.01	4.08	4.65	-0.14	80.68	**80.23**	0.01
	random	72.67	**72.38**	0.00	4.83	5.26	-0.09	74.02	**73.55**	0.01
	chaotic	92.20	**91.03**	0.01	1.03	**0.93**	0.09	92.13	92.18	0.00
	recurrent	80.45	**80.19**	0.00	1.09	1.18	-0.08	82.43	**82.13**	0.00
	recurrent with noise	91.25	**91.19**	0.00	2.44	**2.24**	0.08	91.78	**91.49**	0.00
F2	small	646.62	**605.05**	0.06	3.19	4.64	-0.46	35.35	42.90	-0.21
	large	800.03	**775.11**	0.03	47.97	50.29	-0.05	212.30	214.42	-0.01
	random	746.22	**727.35**	0.03	39.87	46.05	-0.16	163.10	176.82	-0.08
	chaotic	779.24	**732.69**	0.06	3.94	5.41	-0.37	41.21	43.27	-0.05
	recurrent	739.85	**684.16**	0.08	80.42	**66.77**	0.17	198.97	205.12	-0.03
	recurrent with noise	855.29	**838.51**	0.02	4.01	**3.58**	0.11	48.65	**46.42**	0.05
F3	small	1124.00	**1107.51**	0.01	15.89	**13.02**	0.18	999.23	**994.60**	0.00
	large	1378.54	**1371.12**	0.01	589.99	**582.39**	0.01	1116.78	1117.66	0.00
	random	1316.04	**1310.51**	0.00	565.04	611.72	-0.08	1078.92	1090.96	-0.01
	chaotic	1326.61	**1268.00**	0.04	89.67	**89.45**	0.00	1165.44	**1147.50**	0.02
	recurrent	1350.10	**1330.15**	0.01	475.52	**435.84**	0.08	1115.55	**1105.35**	0.01
	recurrent with noise	1419.88	**1415.78**	0.00	227.72	241.36	-0.06	1226.04	1226.89	0.00
F4	small	707.57	**668.37**	0.06	4.54	**4.47**	0.02	65.02	**56.68**	0.13
	large	885.26	**852.26**	0.04	55.45	**50.14**	0.10	268.94	292.32	-0.09
	random	821.63	**805.22**	0.02	69.91	**67.14**	0.04	232.36	**231.90**	0.00
	chaotic	845.49	**791.08**	0.06	4.24	4.61	-0.09	57.47	**52.55**	0.09
	recurrent	815.49	**769.03**	0.06	102.20	124.08	-0.21	300.70	**289.98**	0.04
	recurrent with noise	931.39	**918.78**	0.01	4.85	**4.79**	0.01	67.44	69.84	-0.04
F5	small	1967.45	**1926.96**	0.02	1.88	**1.67**	0.11	54.27	**32.74**	0.40
	large	1975.94	**1967.02**	0.00	2.36	**1.80**	0.24	36.45	41.56	-0.14
	random	1968.17	**1962.10**	0.00	1.92	**1.77**	0.08	32.77	**29.22**	0.11
	chaotic	1977.81	**1919.45**	0.03	1.63	**1.29**	0.21	51.45	**48.54**	0.06
	recurrent	1971.81	**1951.14**	0.01	1.47	**1.22**	0.17	28.44	**28.12**	0.01
	recurrent with noise	1994.54	**1986.44**	0.00	0.94	**0.76**	0.19	48.45	51.55	-0.06
F6	small	1023.95	**973.12**	0.05	12.45	**9.34**	0.25	283.49	297.44	-0.05
	large	1193.92	**1178.45**	0.01	17.86	**13.63**	0.24	559.46	590.63	-0.06
	random	1144.13	**1133.68**	0.01	13.77	**13.46**	0.02	543.45	555.52	-0.02
	chaotic	1131.44	**1069.56**	0.05	9.56	9.73	-0.02	431.40	**390.81**	0.09
	recurrent	1160.34	**1129.59**	0.03	14.19	**13.64**	0.04	528.28	**498.15**	0.06
	recurrent with noise	1215.34	**1203.49**	0.01	9.14	9.40	-0.03	458.87	**433.22**	0.06
Sum of fitness improvement				**0.92**			0.51			0.29

1) Calculate the difference score D between X and Y where the size of D should also be n.
2) Omit the element in D where the value is zero, and the reduced difference score is D' with size n', $n' \leq n$.
3) Rank D' according to its absolute value in ascending order.
4) Assign the corresponding symbol '$+$' or '$-$' to the rank of D'.
5) The Wilcoxon test statistic is subsequently obtained by the sum of the ranks.

A Wilcoxon test was conducted to examine if there is a significant difference between EAs (GA, jDE and CMA-ES) and EAs that use Euclidean as the similarity check measure. The results are presented in Table 4. At the 5% significant level, a significant difference can be observed between GA and GA with Euclidean-based similarity check. However, there is no significant difference on jDE and CMA-ES. One of the reasons is that the Euclidean-based strategy improves only a limited number of instances on the two EAs.

The comparison of the original EAs and those with domination landscape matrix were conducted to further analyse LIDOA. Figure 4 shows that the modified EA with domination landscape matrix obtains better results for 42, 26 and 23 out of 42 instances, respectively. As mentioned earlier, positive FI values also show the advantage of LIDOA with domination matrix measure. Similar to the Euclidean-based measure, the domination matrix based similarity check shows a significant difference when employed on GA, but not on jDE and CMA-ES, as presented in Table 4. Though the Wilcoxon test on GA shows the capability of the similarity check strategy, the results on jDE and CMA-ES indicate that more effort should be made to make the method more efficient.

The performance of the similarity strategy regarding the change types is presented in Table 5. The fitness improvement rates of different functions from the same change type are added, so as to compare the effect of Euclidean distance and domination matrix landscape measures on various change types. As can be seen from the table, GA with

TABLE 10. Fitness errors and relative fitness errors of GA with similarity check strategy based on Landscape population sizes of 124, 427, 1111 and 4221.

Function	Change type	GA+ (124)	$F_{relative}$	GA+ (427)	$F_{relative}$	GA+ (1111)	$F_{relative}$	GA+ (4221)	$F_{relative}$
F1	small step	78.88	0.00	78.89	0.00	78.90	0.00	78.94	0.00
(10 peaks)	large step	76.86	0.00	76.85	0.00	76.84	0.00	76.84	0.00
	random	70.40	0.00	70.40	0.00	70.39	0.00	70.39	0.00
	chaotic	91.37	0.00	91.37	0.00	91.35	0.00	91.35	0.00
	recurrent	81.08	0.00	81.06	0.00	81.07	0.00	81.08	0.00
	recurrent with noise	91.83	0.00	91.83	0.00	91.82	0.00	91.84	0.00
F2	small step	83.55	0.00	83.75	0.00	83.74	0.00	83.75	0.00
(50 peaks)	large step	77.80	0.00	77.82	0.00	77.83	0.00	77.80	0.00
	random	72.36	0.00	72.44	0.00	72.45	0.00	72.45	0.00
	chaotic	90.44	0.00	90.44	0.00	90.44	0.00	90.44	0.00
	recurrent	80.10	0.00	80.11	0.00	80.13	0.00	80.14	0.00
	recurrent with noise	91.15	0.00	91.14	0.00	91.17	0.00	91.16	0.00
F2	small step	569.06	0.00	569.12	0.00	570.60	0.00	572.51	0.01
	large step	737.13	0.00	736.96	0.00	735.75	0.00	736.95	0.00
	random	717.28	0.00	715.67	0.00	717.70	0.00	717.54	0.00
	chaotic	707.86	0.01	705.08	0.00	702.75	0.00	704.17	0.00
	recurrent	672.14	0.00	671.10	0.00	671.91	0.00	673.49	0.00
	recurrent with noise	840.26	0.00	839.31	0.00	839.62	0.00	840.40	0.00
F3	small step	1106.09	0.00	1105.55	0.00	1105.67	0.00	1106.19	0.00
	large step	1371.23	0.00	1370.85	0.00	1367.34	0.00	1372.30	0.00
	random	1315.96	0.00	1314.75	0.00	1309.63	0.00	1311.15	0.00
	chaotic	1239.29	0.00	1240.36	0.00	1241.60	0.00	1240.72	0.00
	recurrent	1329.62	0.01	1321.66	0.00	1328.80	0.01	1326.10	0.00
	recurrent with noise	1419.73	0.00	1419.62	0.00	1416.32	0.00	1417.83	0.00
F4	small step	635.82	0.00	643.85	0.01	641.81	0.01	641.15	0.01
	large step	823.87	0.00	827.02	0.00	827.75	0.00	826.58	0.00
	random	794.77	0.01	794.25	0.00	790.42	0.00	794.39	0.01
	chaotic	769.56	0.00	766.33	0.00	766.19	0.00	766.19	0.00
	recurrent	761.55	0.02	746.80	0.00	747.58	0.00	746.65	0.00
	recurrent with noise	917.81	0.00	915.37	0.00	916.12	0.00	915.85	0.00
F5	small step	1919.90	0.01	1914.28	0.00	1913.68	0.00	1909.71	0.00
	large step	1960.33	0.00	1958.67	0.00	1954.45	0.00	1954.72	0.00
	random	1962.78	0.00	1959.34	0.00	1960.27	0.00	1959.80	0.00
	chaotic	1871.87	0.00	1863.99	0.00	1864.19	0.00	1863.95	0.00
	recurrent	1951.16	0.00	1942.78	0.00	1943.13	0.00	1942.80	0.00
	recurrent with noise	1989.01	0.00	1988.11	0.00	1988.61	0.00	1987.91	0.00
F6	small step	948.55	0.01	935.25	0.00	946.13	0.01	939.29	0.00
	large step	1167.30	0.01	1159.33	0.00	1158.84	0.00	1160.80	0.00
	random	1131.17	0.00	1127.54	0.00	1127.72	0.00	1128.10	0.00
	chaotic	1037.51	0.01	1028.07	0.00	1028.15	0.00	1027.63	0.00
	recurrent	1116.18	0.00	1116.14	0.00	1116.92	0.00	1118.34	0.00
	recurrent with noise	1202.38	0.01	1197.61	0.00	1196.22	0.00	1198.52	0.00
Number of advantaged instances		9		**14**		13		10	
Sum of relative fitness error			0.12		**0.05**		0.05		0.06

Euclidean distance shows overall advantage on all change types. The same conclusion can be made on GA with domination matrix. jDE with Euclidean distance outperforms jDE on random and chaotic changes, while jDE with domination matrix obtains better results on the other four change types. In terms of CMA-ES, the Euclidean distance landscape measure improves the performance on most change types except for chaotic and recurrent with noise changes, while the domination matrix landscape measure is overall superior except for on the large changes.

In Table 6, the '$+$' sign indicates that the Euclidean-based measure performs better than the domination-based measure on the corresponding instance, while the '$-$' sign indicates the opposite result. Although the Euclidean-based measure outperforms the domination-based measure on most change types, the latter is more reliable on recurrent changes and recurrent with noise changes. The comparison between the Euclidean-based measure and the domination-based measure on change types further indicates the further improvement that may be found from the integration of different landscape measures.

D. EFFECT OF THE LANDSCAPE POPULATION SIZE

The landscape population size is essential in LIDOA as it is the basis of the similarity check strategy. A larger landscape population can better reflect the characteristics of the problem, but can be time consuming as well as it can mislead the search with too many details, while a smaller population may be efficient but suffer from insufficient information. In this section, four landscape population sizes, i.e. population with 124 ($q = 3$), 427 ($q = 6$), 1111 ($q = 10$) and 4221($q = 20$) individuals, respectively, were examined to investigate the effect of this parameter. The experiments were conducted on GA because the landscape-based strategy showed greater impact on it compared to jDE and CMA-ES, as can be seen in Tables 3 and 4. Therefore, the effect imposed by different parameter settings can be more easily observed on GA.

TABLE 11. Fitness errors and relative fitness errors of GA with similarity check strategy based on 1, 3 and 5 historical best solutions.

Function	Change type	GA+ (1)	$F_{relative}$	GA+ (3)	$F_{relative}$	GA+ (5)	$F_{relative}$
F1	small step	78.88	0.04	77.18	0.02	75.84	0.00
(10 peaks)	large step	76.86	0.01	76.23	0.00	76.20	0.00
	random	70.40	0.01	70.08	0.00	69.84	0.00
	chaotic	91.37	0.03	90.00	0.02	88.30	0.00
	recurrent	81.08	0.02	80.09	0.00	79.73	0.00
	recurrent with noise	91.83	0.00	91.72	0.00	91.63	0.00
F1	small step	83.55	0.05	80.69	0.01	79.92	0.00
(50 peaks)	large step	77.80	0.01	77.23	0.00	77.02	0.00
	random	72.36	0.01	72.06	0.00	71.91	0.00
	chaotic	90.44	0.04	89.13	0.02	87.11	0.00
	recurrent	80.10	0.01	79.55	0.01	78.99	0.00
	recurrent with noise	91.15	0.00	91.04	0.00	90.95	0.00
F2	small step	569.06	0.11	515.83	0.01	512.84	0.00
	large step	737.13	0.14	665.17	0.03	648.76	0.00
	random	717.28	0.11	668.71	0.03	647.13	0.00
	chaotic	707.86	0.12	654.58	0.03	632.96	0.00
	recurrent	672.14	0.20	582.83	0.04	558.35	0.00
	recurrent with noise	840.26	0.04	823.45	0.02	810.48	0.00
F3	small step	1106.09	0.05	1074.43	0.02	1049.59	0.00
	large step	1371.23	0.03	1349.36	0.02	1325.97	0.00
	random	1315.96	0.03	1296.80	0.01	1283.51	0.00
	chaotic	1239.29	0.04	1218.86	0.02	1194.03	0.00
	recurrent	1329.62	0.06	1294.16	0.03	1258.61	0.00
	recurrent with noise	1419.73	0.01	1407.74	0.00	1402.42	0.00
F4	small step	635.82	0.10	591.86	0.02	579.98	0.00
	large step	823.87	0.10	752.44	0.01	745.79	0.00
	random	794.77	0.09	751.94	0.03	728.69	0.00
	chaotic	769.56	0.09	726.24	0.03	703.87	0.00
	recurrent	761.55	0.17	678.69	0.05	649.12	0.00
	recurrent with noise	917.81	0.03	899.89	0.01	890.73	0.00
F5	small step	1919.90	0.03	1867.34	0.00	1862.53	0.00
	large step	1960.33	0.00	1952.52	0.00	1954.03	0.00
	random	1962.78	0.00	1959.61	0.00	1960.17	0.00
	chaotic	1871.87	0.02	1829.81	0.00	1831.57	0.00
	recurrent	1951.16	0.01	1937.14	0.00	1939.34	0.00
	recurrent with noise	1989.01	0.00	1987.92	0.00	1988.08	0.00
F6	small step	948.55	0.14	855.41	0.03	834.07	0.00
	large step	1167.30	0.04	1133.67	0.01	1124.68	0.00
	random	1131.17	0.03	1115.03	0.01	1101.64	0.00
	chaotic	1037.51	0.09	993.54	0.05	949.34	0.00
	recurrent	1116.18	0.07	1065.62	0.02	1045.51	0.00
	recurrent with noise	1202.38	0.02	1190.29	0.01	1182.61	0.00
Number of advantaged instances		0		5		**37**	
Sum of relative fitness error			1.97		0.53		**0.00**

The experiments were conducted on GA with Euclidean distance as the landscape measure, which is referred to as GA+ in Figure 5. The relative fitness errors regarding change types show that the landscape population size of 427 performed best on small changes and recurrent changes, while the landscape population size of 1111 was the best on the other four change types. Similarly, the fitness error obtained on each benchmark instance indicates that the landscape population sizes of 427 and 1111 show advantages over the other two population sizes on most of the benchmark instances. It can be concluded that the selection of population size should be considered based on the optimization objectives. Additional experimental results on jDE can be found in Appendix.

E. EFFECT OF THE NUMBER OF HISTORICAL LANDSCAPES

In the real-world dynamic problems, the current problem landscape can be similar to multiple historical landscapes, as the current one can share different characteristics with each of them. Therefore, the knowledge from various historical landscapes can provide more information as well as diversify the population. However, the re-use of too many historical landscapes will not only slow down the optimization but may also distract the search. In this section, the effect of the number of historical landscapes is examined. The best solution from the 1, 3 and 5 most similar landscapes are re-used in the new problem, and the corresponding experimental results are shown in Figure 6. The experiments were conducted on GA due to the same reasons as stated in section V-D and Euclidean distance was employed as the landscape measure, referring to as GA+ in the figure.

It can be clearly seen that GA with 5 historical landscapes outperformed other settings on all change types, and GA with 3 historical landscapes is better than re-using only one landscape. This indicates that more information can generally produce better results when a landscape-based strategy is used with GA. Additional experiments on jDE that generate different results can be seen in Appendix.

TABLE 12. Fitness errors and relative fitness errors of jDE with similarity check strategy based on Landscape population sizes of 124, 427, 1111 and 4221.

Function	Change type	jDE+ (124)	$F_{relative}$	jDE+ (427)	$F_{relative}$	jDE+ (1111)	$F_{relative}$	jDE+ (4221)	$F_{relative}$
(10 peaks)	small step	0.30	0.33	**0.22**	0.00	0.24	0.09	0.29	0.29
	large step	3.39	3.53	0.75	0.00	**0.75**	0.00	0.90	0.20
	random	**3.38**	0.00	3.84	0.13	3.72	0.10	4.21	0.24
	chaotic	**1.00**	0.00	4.29	3.29	4.78	3.79	4.95	3.96
	recurrent	**1.87**	0.00	3.61	0.93	3.49	0.87	4.04	1.16
	recurrent with noise	**2.62**	0.00	4.71	0.80	4.55	0.74	4.76	0.82
(50 peaks)	small step	1.00	0.03	1.05	0.08	**0.97**	0.00	1.02	0.05
	large step	5.03	4.70	1.09	0.24	**0.88**	0.00	0.96	0.09
	random	4.77	1.60	1.85	0.01	2.05	0.12	**1.83**	0.00
	chaotic	**0.94**	0.00	1.23	0.31	1.23	0.31	1.13	0.21
	recurrent	**1.17**	0.00	1.97	0.68	2.50	1.13	1.92	0.64
	recurrent with noise	**2.32**	0.00	2.36	0.02	2.67	0.15	2.42	0.04
2	small step	3.84	0.19	**3.23**	0.00	4.73	0.46	3.90	0.21
	large step	46.24	0.01	56.26	0.22	55.88	0.21	**46.00**	0.00
	random	41.05	0.25	48.41	0.48	**32.78**	0.00	44.51	0.36
	chaotic	3.83	0.04	4.47	0.21	**3.69**	0.00	4.16	0.13
	recurrent	80.39	0.11	89.37	0.24	**72.18**	0.00	88.15	0.22
	recurrent with noise	**3.34**	0.00	5.80	0.74	4.46	0.33	4.73	0.42
3	small step	**15.42**	0.00	21.11	0.37	20.95	0.36	23.03	0.49
	large step	**554.16**	0.00	561.52	0.01	583.74	0.05	563.58	0.02
	random	**554.39**	0.00	573.00	0.03	607.41	0.10	594.48	0.07
	chaotic	83.29	0.12	76.69	0.04	82.56	0.11	**74.09**	0.00
	recurrent	481.76	0.09	452.77	0.03	455.17	0.03	**440.93**	0.00
	recurrent with noise	**236.56**	0.00	270.49	0.14	256.41	0.08	260.73	0.10
4	small step	4.36	0.12	**3.88**	0.00	4.19	0.08	4.41	0.14
	large step	68.52	0.08	80.34	0.27	65.35	0.03	**63.36**	0.00
	random	54.93	0.09	53.91	0.07	55.13	0.09	**50.45**	0.00
	chaotic	4.93	0.05	5.00	0.07	**4.69**	0.00	5.21	0.11
	recurrent	**105.70**	0.00	106.74	0.01	108.62	0.03	147.38	0.39
	recurrent with noise	5.36	0.19	4.95	0.10	**4.52**	0.00	5.06	0.12
5	small step	**1.54**	0.00	1.70	0.10	1.92	0.25	1.73	0.12
	large step	2.03	0.15	2.54	0.44	1.98	0.12	**1.76**	0.00
	random	1.89	0.09	2.00	0.15	**1.74**	0.00	2.08	0.20
	chaotic	1.44	0.14	1.52	0.20	1.39	0.10	**1.27**	0.00
	recurrent	1.42	0.01	**1.40**	0.00	1.41	0.01	1.50	0.07
	recurrent with noise	1.00	0.11	0.93	0.02	**0.90**	0.00	0.93	0.03
6	small step	**8.98**	0.00	10.14	0.13	9.68	0.08	11.03	0.23
	large step	14.69	0.04	14.21	0.01	14.43	0.02	**14.13**	0.00
	random	**12.59**	0.00	15.02	0.19	14.40	0.14	15.33	0.22
	chaotic	10.54	0.15	10.43	0.13	10.01	0.09	**9.19**	0.00
	recurrent	14.68	0.16	**12.61**	0.00	13.23	0.05	13.41	0.06
	recurrent with noise	**9.23**	0.00	9.79	0.06	10.24	0.11	9.94	0.08
Number of advantaged instances		**17**		5		10		10	
Sum of relative fitness error			12.40		10.96		**10.24**		11.48

VI. CONCLUSION

In this research, Landscape Influenced Dynamic Optimization Algorithm (LIDOA) was proposed to improve the efficiency of EAs in solving DOPs. The knowledge learned in each landscape was archived and re-utilized in the new environment. On the basis of the sequence-based deterministic initialization technique, landscape measures were employed to select the historical landscape that was most similar to the current one. The optimal solution from the chosen landscape was then re-used to enhance the optimization ability of EAs.

The commonly used benchmark from CEC2009 was employed, which consists of seven functions and six change types. Four landscape measures were examined, and it can be concluded that the Euclidean distance and domination matrix were better measures, compared to skewness and kurtosis, yet none of them could clearly dominate the other. Subsequently, for further analysis, those two measures were then adopted with GA, jDE and CMA-ES. Experimental results show the overall advantage of LIDOA.

The analysis regarding change types indicate that the Euclidean distance and domination matrix landscape measures have advantages over different instances. Accordingly, better performance can be expected with appropriate integration of multiple landscape measures. The effect of different parameters, including landscape population size and re-used landscapes, on the optimization was also analysed.

The general advantages that LIDOA shows come from multiple resources. First of all, the landscape-based similarity check strategy is independent of the problem types. Therefore, it is applicable to a large range of problems. Similarly, LIDOA can improve the performance of various EAs. In addition, the re-used global optimum helps to escape from the local optima. However, there are also shortcomings of the proposed approach. The computational complexity of EAs is not reduced after using LIDOA, as it is added to EAs without changing them. Apart from this, only limited information has been re-used in this article. The performance can be expected to be further enhanced if more knowledge is re-utilized.

TABLE 13. Fitness errors and relative fitness errors of jDE with similarity check strategy based on 1, 3 and 5 historical best solutions.

Function	Change type	jDE+ (1)	$F_{relative}$	jDE+ (3)	$F_{relative}$	jDE+ (5)	$F_{relative}$
(10 peaks)0	small step	0.30	0.14	**0.26**	0.00	0.41	0.58
	large step	3.39	3.02	0.92	0.10	**0.84**	0.00
	random	3.38	0.05	**3.22**	0.00	3.83	0.19
	chaotic	**1.00**	0.00	4.64	3.65	4.56	3.57
	recurrent	**1.87**	0.00	4.21	1.25	3.67	0.96
	recurrent with noise	**2.62**	0.00	5.42	1.07	5.13	0.96
(50 peaks)	small step	**1.00**	0.00	1.12	0.12	1.12	0.12
	large step	5.03	4.72	**0.88**	0.00	0.88	0.01
	random	4.77	1.59	2.17	0.18	**1.84**	0.00
	chaotic	**0.94**	0.00	1.15	0.22	1.24	0.33
	recurrent	**1.17**	0.00	2.28	0.94	2.29	0.95
	recurrent with noise	**2.32**	0.00	2.80	0.21	2.51	0.08
2	small step	3.84	0.33	**2.90**	0.00	4.86	0.68
	large step	46.24	0.18	**39.27**	0.00	44.70	0.14
	random	**41.05**	0.00	44.30	0.08	50.27	0.22
	chaotic	3.83	0.02	**3.78**	0.00	4.56	0.21
	recurrent	80.39	0.06	77.59	0.02	**75.84**	0.00
	recurrent with noise	3.34	0.02	3.82	0.16	**3.28**	0.00
3	small step	**15.42**	0.00	16.46	0.07	17.25	0.12
	large step	**554.16**	0.00	562.53	0.02	583.37	0.05
	random	**554.39**	0.00	560.24	0.01	559.33	0.01
	chaotic	83.29	0.09	**76.63**	0.00	84.60	0.10
	recurrent	481.76	0.11	**432.60**	0.00	446.52	0.03
	recurrent with noise	**236.56**	0.00	263.75	0.11	285.50	0.21
4	small step	**4.36**	0.00	4.40	0.01	4.66	0.07
	large step	68.52	0.05	**65.23**	0.00	74.69	0.14
	random	54.93	0.10	**50.06**	0.00	75.88	0.52
	chaotic	4.93	0.17	4.53	0.07	**4.23**	0.00
	recurrent	105.70	0.02	110.71	0.07	**103.12**	0.00
	recurrent with noise	5.36	0.43	**3.76**	0.00	4.72	0.26
5	small step	1.54	0.08	1.58	0.11	**1.42**	0.00
	large step	2.03	0.10	2.08	0.12	**1.85**	0.00
	random	1.89	0.08	1.98	0.14	**1.75**	0.00
	chaotic	1.44	0.13	**1.27**	0.00	1.31	0.03
	recurrent	1.42	0.19	1.24	0.04	**1.20**	0.00
	recurrent with noise	1.00	0.20	1.01	0.21	**0.84**	0.00
6	small step	**8.98**	0.00	11.81	0.32	10.82	0.21
	large step	14.69	0.09	14.04	0.04	**13.48**	0.00
	random	**12.59**	0.00	14.80	0.18	13.42	0.07
	chaotic	10.54	0.06	**9.99**	0.00	10.60	0.06
	recurrent	14.68	0.10	14.49	0.08	**13.39**	0.00
	recurrent with noise	9.23	0.03	**8.98**	0.00	9.53	0.06
Number of advantaged instances		**15**		14		13	
Sum of relative fitness error			12.13		**9.59**		10.93

Although this article is a step-forward to integrate landscape measures with EAs to solve DOPs, further improvements are still needed. A possible direction may be the use of multi-algorithm and multi-operator frameworks. Better approaches to share information among environments are also required.

APPENDIX

See Tables 7–13.

REFERENCES

[1] D. Simoncini, S. Barbe, T. Schiex, and S. Verel, "Fitness landscape analysis around the optimum in computational protein design," in *Proc. Genetic Evol. Comput. Conf.*, Jul. 2018, pp. 355–362.

[2] E. Vellasques, R. Sabourin, and E. Granger, "A high throughput system for intelligent watermarking of bi-tonal images," *Appl. Soft Comput.*, vol. 11, no. 8, pp. 5215–5229, Dec. 2011.

[3] A. M. F. M. Abdallah, D. L. Essam, and R. A. Sarker, "On solving periodic re-optimization dynamic vehicle routing problems," *Appl. Soft Comput.*, vol. 55, pp. 1–12, Jun. 2017.

[4] Q. Jiang, R. Sarker, and H. Abbass, "Tracking moving targets and the non-stationary traveling salesman problem," *Complex. Int.*, vol. 11, no. 2005, pp. 171–179, 2005.

[5] S. Yang, "Evolutionary computation for dynamic optimization problems," in *Proc. Companion Publication Annu. Conf. Genet. Evol. Comput.*, 2015, pp. 629–649.

[6] T. Bartz-Beielstein, J. Branke, J. Mehnen, and O. Mersmann, "Evolutionary algorithms," *Wiley Interdiscipl. Rev., Data Mining Knowl. Discovery*, vol. 4, no. 3, pp. 178–195, 2014.

[7] S. Yang and X. Yao, *Evolutionary Computation for Dynamic Optimization Problems*, vol. 490. Berlin, Germany: Springer, 2013.

[8] A. Sharifi, J. K. Kordestani, M. Mahdaviani, and M. R. Meybodi, "A novel hybrid adaptive collaborative approach based on particle swarm optimization and local search for dynamic optimization problems," *Appl. Soft Comput.*, vol. 32, pp. 432–448, Jul. 2015.

[9] S. Elsayed, R. Sarker, and C. A. Coello Coello, "Sequence-based deterministic initialization for evolutionary algorithms," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2911–2923, Sep. 2017.

[10] J. Brest, A. Zamuda, B. Boskovic, M. S. Maucec, and V. Zumer, "Dynamic optimization using self-adaptive differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, May 2009, pp. 415–422.

[11] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evol. Comput.*, vol. 9, no. 2, pp. 159–195, Jun. 2001.

[12] Z.-J. Wang, Z.-H. Zhan, K.-J. Du, Z.-W. Yu, and J. Zhang, ''Orthogonal learning particle swarm optimization with variable relocation for dynamic optimization,'' in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2016, pp. 594–600.
[13] H. R. Topcuoglu, A. Ucar, and L. Altin, ''A hyper-heuristic based framework for dynamic optimization problems,'' *Appl. Soft Comput.*, vol. 19, pp. 236–251, 2014.
[14] L. Cao, L. Xu, and E. D. Goodman, ''A neighbor-based learning particle swarm optimizer with short-term and long-term memory for dynamic optimization problems,'' *Inf. Sci.*, vol. 453, pp. 463–485, Jul. 2018.
[15] J. Zou, Q. Li, S. Yang, H. Bai, and J. Zheng, ''A prediction strategy based on center points and knee points for evolutionary dynamic multi-objective optimization,'' *Appl. Soft Comput.*, vol. 61, pp. 806–818, Dec. 2017.
[16] G. Ruan, G. Yu, J. Zheng, J. Zou, and S. Yang, ''The effect of diversity maintenance on prediction in dynamic multi-objective optimization,'' *Appl. Soft Comput.*, vol. 58, pp. 631–647, Sep. 2017.
[17] A. Zhou, Y. Jin, and Q. Zhang, ''A population prediction strategy for evolutionary dynamic multiobjective optimization,'' *IEEE Trans. Cybern.*, vol. 44, no. 1, pp. 40–53, Jan. 2014.
[18] L. Altin and H. R. Topcuoglu, ''Impact of sensor-based change detection schemes on the performance of evolutionary dynamic optimization techniques,'' *Soft Comput.*, vol. 22, no. 14, pp. 4741–4762, Jul. 2018.
[19] F. Zaman, S. M. Elsayed, T. Ray, and R. A. Sarker, ''Configuring two-algorithm-based evolutionary approach for solving dynamic economic dispatch problems,'' *Eng. Appl. Artif. Intell.*, vol. 53, pp. 105–125, Aug. 2016.
[20] F. B. Ozsoydan and A. Baykasoğlu, ''Quantum firefly swarms for multimodal dynamic optimization problems,'' *Expert Syst. Appl.*, vol. 115, pp. 189–199, Jan. 2019.
[21] J. K. Kordestani, A. E. Ranginkaman, M. R. Meybodi, and P. Novoa-Hernández, ''A novel framework for improving multi-population algorithms for dynamic optimization problems: A scheduling approach,'' *Swarm Evol. Comput.*, vol. 44, pp. 788–805, Feb. 2019.
[22] Z. Zhu, L. Chen, C. Yuan, and C. Xia, ''Global replacement-based differential evolution with neighbor-based memory for dynamic optimization,'' *Appl. Intell.*, vol. 48, no. 10, pp. 3280–3294, Oct. 2018.
[23] B. Nasiri, M. Meybodi, and M. Ebadzadeh, ''History-driven particle swarm optimization in dynamic and uncertain environments,'' *Neurocomputing*, vol. 172, pp. 356–370, Jan. 2016.
[24] R. Mukherjee, S. Debchoudhury, and S. Das, ''Modified differential evolution with locality induced genetic operators for dynamic optimization,'' *Eur. J. Oper. Res.*, vol. 253, no. 2, pp. 337–355, Sep. 2016.
[25] W. Luo, X. Lin, T. Zhu, and P. Xu, ''A Clonal selection algorithm for dynamic multimodal function optimization,'' *Swarm Evol. Comput.*, vol. 50, Nov. 2019, Art. no. 100459.
[26] C. Li, T. T. Nguyen, M. Yang, M. Mavrovouniotis, and S. Yang, ''An adaptive multipopulation framework for locating and tracking multiple optima,'' *IEEE Trans. Evol. Comput.*, vol. 20, no. 4, pp. 590–605, Aug. 2016.
[27] W. Zhang, W. Zhang, G. G. Yen, and H. Jing, ''A cluster-based clonal selection algorithm for optimization in dynamic environment,'' *Swarm Evol. Comput.*, vol. 50, Nov. 2019, Art. no. 100454.
[28] M. Mavrovouniotis, C. Li, and S. Yang, ''A survey of swarm intelligence for dynamic optimization: Algorithms and applications,'' *Swarm Evol. Comput.*, vol. 33, pp. 1–17, Apr. 2017.
[29] J. Branke, ''Evolutionary approaches to dynamic optimization problems-updated survey,'' in *Proc. GECCO Workshop Evol. Algorithms Dyn. Optim. Problems*, 2001, pp. 27–30.
[30] C. I. Moser, ''All currently known publications on approaches which solve the moving peaks problem,'' Swinburne Univ. Technol., Melbourne, VIC, Australia, Tech. Rep., 2007.
[31] E. Vellasques, R. Sabourin, and E. Granger, ''Gaussian mixture modeling for dynamic particle swarm optimization of recurrent problems,'' in *Proc. 14th Annu. Conf. Genet. Evol. Comput. Conf.*, 2012, pp. 73–80.
[32] E. Vellasques, R. Sabourin, and E. Granger, ''A dual-purpose memory approach for dynamic particle swarm optimization of recurrent problems,'' in *Recent Advances in Computational Intelligence in Defense and Security*. Cham, Switzerland: Springer, 2016, pp. 367–389.
[33] S. Wright, ''The roles of mutation, inbreeding, crossbreeding, and selection in evolution,'' in *Proc. 6th Int. Congr. Genet.*, vol. 1, no. 8. 1932, pp. 356–366.
[34] E. Pitzer and M. Affenzeller, ''A comprehensive survey on fitness landscape analysis,'' in *Recent Advances in Intelligent Engineering Systems*. Berlin, Germany: Springer, 2012, pp. 161–191.
[35] M. Wang, B. Li, G. Zhang, and X. Yao, ''Population evolvability: Dynamic fitness landscape analysis for population-based Metaheuristic algorithms,'' *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 550–563, Aug. 2018.
[36] Complexity Labs. (2016). *Fitness Landscape*. Accessed: Nov. 7, 2018. [Online]. Available: http://complexitylabs.io/fitness-landscapes/
[37] G.-S. Hao, M.-H. Lim, Y.-S. Ong, H. Huang, and G.-G. Wang, ''Domination landscape in evolutionary algorithms and its applications,'' *Soft Comput.*, vol. 23, no. 11, pp. 3563–3570, Jun. 2019.
[38] W. Roetzel, X. Luo, D. Chen, ''Optimal design of heat exchanger networks,'' in *Design and Operation of Heat Exchangers and Their Networks*. Amsterdam, The Netherlands: Elsevier, 2019, pp. 231–318.
[39] S. Das, S. S. Mullick, and P. N. Suganthan, ''Recent advances in differential evolution—An updated survey,'' *Swarm Evol. Comput.*, vol. 27, pp. 1–30, Apr. 2016.
[40] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, ''Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems,'' *IEEE Trans. Evol. Comput.*, vol. 10, no. 6, pp. 646–657, Dec. 2006.
[41] M. Georgioudakis and V. Plevris, ''A comparative study of differential evolution variants in constrained structural optimization,'' *Frontiers Built Environ.*, vol. 6, p. 102, Jul. 2020.
[42] S. Dey, S. Bhattacharyya, and U. Maulik, ''New quantum inspired meta-heuristic techniques for multi-level colour image thresholding,'' *Appl. Soft Comput.*, vol. 46, pp. 677–702, Sep. 2016.
[43] N. Hansen, ''The CMA evolution strategy: A comparing review,'' in *Towards a New Evolutionary Computation*. Berlin, Germany: Springer, 2006, pp. 75–102.
[44] N. Hansen, ''The CMA evolution strategy: A tutorial,'' 2016, *arXiv:1604.00772*. [Online]. Available: http://arxiv.org/abs/1604.00772
[45] C. Li, S. Yang, T. Nguyen, E. L. Yu, X. Yao, Y. Jin, H. Beyer, and P. Suganthan, ''Benchmark generator for CEC 2009 competition on dynamic optimization,'' Dept. Comput. Sci., Univ. Leicester, Leicester, U.K., Tech. Rep., 2008, doi: 10.13140/RG.2.1.3445.6401.
[46] F. Wilcoxon, ''Individual comparisons by ranking methods,'' *Biometrics*, vol. 1, no. 6, pp. 80–83, 1945.

KANGJING LI (Member, IEEE) received the B.E. degree in automation from Henan Polytechnic University, China, in 2014, and the M.E. degree in control theory and control engineering from the South China University of Technology, China, in 2017. She is currently pursuing the Ph.D. degree in computer science with the UNSW Canberra at the Australian Defence Force Academy, Canberra, Australia. Her research interests include evolutionary algorithm and optimization problems under dynamic and/or uncertain environments.

SABER M. ELSAYED (Member, IEEE) received the Ph.D. degree in computer science from the University of New South Wales (UNSW) at Canberra, Canberra, Australia, in 2012.

He is currently a Senior Lecturer with the School of Engineering and Information Technology (SEIT), UNSW Canberra. His research interests include the areas of evolutionary algorithms, constrained and unconstrained optimization, scheduling, multiobjective optimization, big data, and cyber-security using computational intelligence. He is also an Editorial Board Member of the *International Journal of Business Intelligence and Data Mining*. He serves as a member of the program committee of several international conferences. He was the winner of different IEEE-CEC competitions. Since 2019, he has been serving as the Chair of the IEEE Computational Intelligence Society (ACT Chapter). He is also the Local Organizing Committee Co-Chair of the IEEE Symposium Series on Computational Intelligence, in 2020. He was one of the two Program Chairs of the 20th Asia-Pacific Symposium on Intelligent and Evolutionary Systems, Canberra. He also serves as a reviewer for several international journals.

RUHUL SARKER (Member, IEEE) received the Ph.D. degree from Dalhousie University, Halifax, Canada, in 1992. He is currently a Professor with the School of Engineering and Information Technology and the Former Director of the Faculty Postgraduate Research, University of New South Wales at Canberra, ACT, Australia. He is the Lead Author of the book *Optimization Modelling: A Practical Approach* (CRC Press, 2007) and has published over 300 refereed papers in international journals, edited books, and conference proceedings. His current research interests include evolutionary optimization and applied operations research. He is also an Associate Editor of *Memetic Computing* journal, the *Journal of Industrial and Management Optimization*, and *Flexible Services and Manufacturing Journal*.

DARYL ESSAM (Member, IEEE) received the B.Sc. degree in computer science from the University of New England, Australia, in 1990, and the Ph.D. degree from the University of New South Wales (UNSW) at Canberra, Australia, in 2000. Since 1994, he has been with UNSW Canberra, where he is currently a Senior Lecturer. His research interest includes genetic algorithms, with a focus on both evolutionary optimization and large scale problems.