Adaptive Guidance-based Differential Evolution with Iterative Feedback Archive Strategy for Multimodal Optimization Problems

Hong Zhao, *Student Member*, *IEEE*, Zhi-Hui Zhan (Corresponding Author), *Senior Member*, *IEEE*, and Jun Zhang, *Fellow*, *IEEE*

***Abstract*—Multimodal optimization problems (MMOPs) target to locate multiple global optima simultaneously, which require the algorithms not only can maintain the population diversity to locate the global optima as many as possible, but also can ensure the convergence on each found optimal region to refine the solutions as high accuracy as possible. Aiming to these two goals and to efficiently deal with MMOPs, an adaptive guidance-based differential evolution (AGDE) with archive strategy is proposed in this paper, including three novel components. Firstly, an adaptive mutation strategy (AMS) is introduced, which guides the current individual to move towards the peak that is closest to itself. Secondly, an iterative feedback archive (IFA) strategy is used to store the global optima of the population in every iteration. Thirdly, a Gaussian disturbance-based elite learning (GDEL) strategy is performed on the archive to refine the accuracy of the solutions. The AMS strategy helps to locate more peaks, while the IFA and GDEL strategies help to maintain the found solutions and refine their accuracy. The performance of AGDE is tested on 20 widely used multimodal benchmark functions of CEC'2013. The experimental results of AGDE are competitive with the results obtained by the state-of-the-art multimodal algorithms.**

***Keywords*—*Differential evolution*; *Multimodal optimization problems*; *Adaptive mutation strategy*.**

I. INTRODUCTION

Many real-word problems require to find multiple optimal solutions simultaneously, such as data mining [1], power system [2], and protein structure prediction [3]. These problems have many global optima that are difficult to be simultaneously obtained by the traditional methods, which can be called as multimodal optimization problems (MMOPs). Due to the evolutionary computation (EC) algorithms have achieved a great success in many complex problems [4]-[8], many researchers try to use EC algorithms to solve MMOPs in recent years [9][10]. The common EC algorithms include genetic algorithm [11], ant colony optimization [12], particle swarm optimization (PSO) [13], and differential evolution (DE) [14][15]. Generally speaking, there are three methods to deal with MMOPs with EC algorithms.

The first method is the well-known niching technique. The crowding DE (CDE) [16] and speciation DE (SDE) [17][18] are two widely used niching techniques, which can locate the multiple peaks by dividing population into several independent or overlapped sub-populations. However, these methods introduce additional parameters (i.e., crowding size and species radius), and the performance of these algorithms are sensitive to the values of these parameters. To reduce the effect of the niching parameters, a parameter-free ring topology method was proposed by Li [19], which used the individual index to form a niching, named r2pso and r3pso. In addition, a self-adaptive parameter control method was proposed by Gao *et al*. [20], which used a novel adaptive DE based on cluster method (Self-CCDE) to deal with MMOPs. Recently, Wang *et al*. [21] used a random cluster number to reduce the sensitivity of the clustering parameters and designed a dual-strategy DE. Moreover, they further proposed to use the affinity propagation clustering that did not use sensitive clustering parameters for automatic niching DE [22]. Zhao *et al*. [23] borrowed the local binary operator idea in image processing to help efficiently form the niches. Chen *et al*. [24] treated each individual as a distributed niche to design a distributed individual DE that can locate the global optima as many as possible.

The second method adopts the novel evolutionary operators to enhance the algorithm ability to deal with MMOPs. Qu *et al*. [25] designed a neighborhood mutation strategy by integrating with various DE and a distance-based locally informed particle swarm algorithm, forming NCDE, NSDE and LIPS, respectively. Subsequently, an improved parent-centric normalized neighborhood mutation operator with DE (PNPCDE) and a parent selection scheme based on the locally informative CDE (LoICDE) [26] were proposed by Biswas *et al*. [27]. Although these methods perform better in solving MMOPs, most of the methods needed to set extra parameters, or they were designed too complicatedly and exhausted many fitness evolutions (FEs).

This work was supported in part by the National Key Research and Development Program of China with 2019YFB2102102, the Outstanding Youth Science Foundation with 61822602, the National Natural Science Foundations of China with 61772207 and 61873097, the Key-Area Research and Development of Guangdong Province with 2020B010166002 the Guangdong Natural Science Foundation Research Team under 2018B030312003, and Ministry of Science and ICT through the National Research Foundation of Korea (NRF-2019H1D3A2A01101977). (*Corresponding author: Zhi-Hui Zhan*).

H. Zhao and Z.-H. Zhan are with the School of Computer Science and Engineering, South China University of Technology, 510006 Guangzhou, China and also with the Guangdong Provincial Key Laboratory of Computational Intelligence and Cyberspace Information and the State Key Laboratory of Subtropical Building Science, South China University of Technology, 510006 Guangzhou, China (e-mail: zhanapollo@163.com).

J. Zhang is with the Hanyang University, Ansan 15588, South Korea.

978-1-7281-6929-3/20/$31.00 ©2020 IEEE

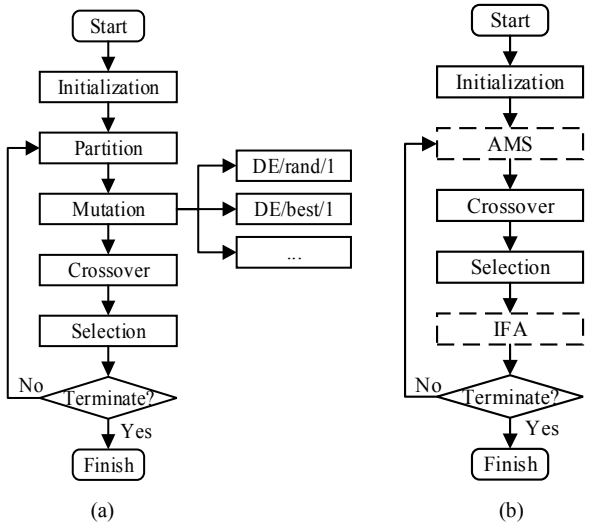


Fig. 1. The difference between a common method and AGDE in solving MMOPs. (a) The framework of the common method to solve MMOPs. (b) The framework of the AGDE to solve MMOPs.

The third method adopts the multiobjective technique by transforming MMOPs into the multiobjective optimization problems [28]-[30]. The optimization objectives usually are the fitness value and the diversity indicator. In particular, Wang *et al*. [30] designed two conflict objectives according to the definition of multiobjective optimization to transform MMOPs into the multiobjective optimization problems, resulting in MOMMOP. Although this method performs well when dealing with low-dimensional problems (LDPs), it is difficult to locate all the global optima due to the large search space in high-dimensional problems (HDPs).

In summary, although the above-mentioned methods have enhanced the DE performance in solving MMOPs, there are still some limitations for DEs. First, most of them use only a single greedy/rand strategy (e.g., DE/best/1, DE/rand/1), which will cause the individuals difficult to suitably search information according to their evolutionary status. That is, they cannot adaptively guide the evolution of individuals based on the most suitable guiding individual. Second, some existing methods try to divide the population into several niches by introducing extra parameters, but the performance of the algorithm is sensitive to the values of these parameters. Third, some existing methods do not effectively maintain the found optima and refine their accuracy.

Motivated by the above problems, this paper proposes an adaptive guidance-based differential evolution (AGDE) with archive strategy to solve MMOPs. Fig. 1 shows the difference between a common method and AGDE in solving MMOPs. Fig. 1(a) and Fig. 1(b) introduce the frameworks of the common method and the AGDE when dealing with MMOPs, respectively. The content of the dashed box in Fig. 1(b) are the improvement over the common method to solve MMOPs. As shown in Fig. 1, there are three advantages of AGDE when dealing with MMOPs. Firstly, AGDE borrows the information of the nearest individual distanced to the current individual to form an adaptive mutation strategy (AMS), which makes each individual move towards the nearest peak to itself. AMS also includes the global perturbation to improve the global search ability of the algorithm. Secondly, an iterative feedback archive (IFA) strategy uses an archive to store the global optimum of the population in every iteration to preserve and utilize the good search information to feedback the population during the evolutionary process. Moreover, the archive is cleared every T iterations to ensure the stored solutions in archive have a positive feedback to the evolution. Thirdly, to refine the accuracy of the solutions, a Gaussian disturbance-based elite learning (GDEL) strategy is performed in the archive.

The AMS can help the AGDE locate as many peaks as possible. The IFA strategy together with the GDEL strategy can maintain the found solutions via the archive and refine their accuracy as high as possible. The performance of AGDE is tested on the CEC'2013. The results of AGDE are better than or at least comparable to the results obtained by the state-of-the-art multimodal algorithms.

The rest parts of AGDE are as follows. Section II presents the details of the proposed AGDE for solving MMOPs. In Section III, experiments are designed to verify the effectiveness of our AGDE. In the end, Section IV makes a conclusion and high lights the future works.

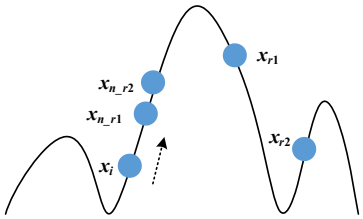
II. AGDE

This section introduced the detailed process of AGDE algorithm. Firstly, the AMS is proposed to make each individual learn from the best and nearest individual distanced to itself, which can ensure each individual gradually approach the nearest peak distanced to the current individual. Moreover, the global disturbance is integrated with AMS to improve the global search ability of the algorithm. Secondly, the IFA strategy is introduced to collect the global optimum of the population in every iteration. Besides, the GDEL strategy is performed on all these stored solutions to obtain new promising solutions. Lastly, the completed AGDE algorithm is listed.

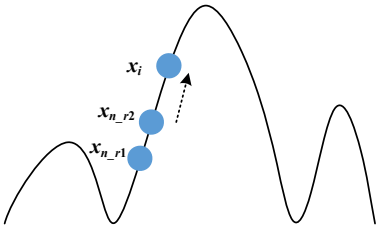
A. AMS

The mutation operation is a key step that significantly affects the performance of DE. Many mutation methods such as DE/rand/1, DE/best/1, make DE perform well in many optimization problems. However, these methods do not consider the specific distribution information of the individuals that around the current individual, thus they cannot guide the evolution of individuals adaptively. In fact, a better mutation strategy requires well balance the diversity and convergence of the population. Herein, the AMS is proposed, which is divided into two situations according to the problem dimensions.

One situation is when the problem dimension (D) is less than or equal to 3, which can be called the LDPs. The LDPs are relatively simple, and the current individual $\boldsymbol{x}_i$ can be guided by itself. Moreover, to avoid the individual $\boldsymbol{x}_i$ being trapped into local optima and to improve the global search ability of the algorithm, a global disturbance is added to the individual evolution as:



(a) x_{n_r1} or x_{n_r2} is the best individual



(b) x_i is the best individual.

Fig. 2. The two cases of AMS in HDPs. (a) x_{n_r1} or x_{n_r2} is the best individual. (b) x_i is the best individual.

$$v_i = x_i + F \times (x_{r_1} - x_{r_2}), \quad D \le 3 \tag{1}$$

where x_i is the current individual, $x_{r1}, x_{r2} \in \{1, 2, \ldots, NP\}$, $i \neq r_1 \neq r_2$, NP is the population size, and F is the scaling factor.

The other situation is when D is more than 3, which can be called the HDPs. Due to the HDPs are relatively complicated and it can use the two nearest individuals distanced to the current individual x_i to help the individual evolution. The process of AMS in HDPs can be divided into three steps:

i) Find the two nearest individuals (x_{n_r1}, x_{n_r2}) distanced to x_i;

ii) Compare the fitness of x_i, x_{n_r1}, x_{n_r2} and select the best individual x_{nbest} by

$$x_{nbest} = \text{best}\{x_i, x_{n\_r_1}, x_{n\_r_2}\} \tag{2}$$

iii) Use x_{nbest} to guide x_i, and use x_{n_r1} and x_{n_r2} to accelerate the population convergence. The detailed process is shown as

$$\begin{aligned} v_i = x_{nbest} + F \times (x_{n\_r_1} - x_{n\_r_2}) + \\ \alpha \times F \times (x_{r_1} - x_{r_2}), \ D > 3 \end{aligned} \tag{3}$$

where x_{n_r1} and x_{n_r2} are the two nearest individuals distanced to x_i. x_{nbest} is the best individual among x_i, x_{n_r1} and x_{n_r2}. α is a control parameter, and α=(sign($f(x_{nbest})$-$f(x_i)$)+1)/2, $f(x_i)$ is the fitness of the individual x_i. The other parameters have the same meanings as Eq. (1).

There are two cases when AMS deals with HDPs as shown in Fig. 2. The first case is that x_{n_r1} or x_{n_r2} is the best individual x_{nbest} as shown in Fig. 2(a). It means that $f(x_{nbest})$ is greater than $f(x_i)$ and therefore the value of sign($f(x_{nbest})$ − $f(x_i)$) is 1, namely α=1. In this case, the x_{n_r1} or x_{n_r2} is used to guide the current individual. Moreover, the global individuals (x_{r1} and x_{r2}) are added into the individual evolution to improve the global search ability of the algorithm. The second case is that x_i is the best individual (x_{nbest}) as shown in Fig. 2(b). It means that $f(x_{nbest})$ is equal to $f(x_i)$ and the value of sign($f(x_{nbest})$ − $f(x_i)$) is 0, namely α=0. That is to say, x_i is closer to the peak than x_{n_r1} and x_{n_r2}. In this case, it only requires a slight disturbance to avoid the individual trapped into local optima due to the greedy guidance. The global disturbance from x_{r1} and x_{r2} is not need.

B. IFA Strategy

In order to preserve and utilize the good search information during the evolutionary process, the IFA strategy uses an archive to store the global optimum in every iteration. Moreover, the DGEL strategy is performed on all these stored solutions to obtain new promising solutions. These new solutions will feedback to the population to enhance the diversity. The IFA strategy is carried in every iteration cooperative with the DGEL to feedback the population, with its process as the following three steps.

Firstly, after the evolution of every iteration, the global optimum is added into the archive A.

Secondly, the GDEL strategy is performed on A to make a local refine of each solution in the archive as

$$A_i = \text{Gaussian}(A_i, \sigma) \tag{4}$$

where A_i is the i^{th} solution in A and σ is the variance that is set as σ=1.0 in this paper.

Thirdly, to increase the diversity of the population, all the solutions after Gaussian disturbance in the archive are added into the population. There are two points to note here: i) After adding the $|A|$ new generated solutions by GDEL to the population, $|A|$ individuals are randomly removed to ensure the population size does not change. ii) To ensure the stored solutions in A have a positive feedback to the evolution, the A is cleared (i.e., all the solutions are removed) every T iterations. In addition, it should be noticed that T is set to 5 according to investigation in Section III-C.

C. Complete AGDE Algorithm

The AGDE introduces two components including AMS and IFA. Firstly, AMS can locate as many peaks as possible by borrowing the nearest individual distanced to the current individual. Secondly, IFA strategy collects the elite information of the population and feedback to the population in every iteration. Besides, GDEL strategy is used to improve the accuracy of solutions.

In DE, the crossover and selection operations should be performed after AMS. The detailed of crossover and selection operations are as (5) and (6), respectively.

$$u_{ij} = \begin{cases} v_{ij}, & \text{if } \text{rand}(0,1) \le CR \text{ or } j = jrand \\ x_{ij}, & \text{otherwise} \end{cases} \tag{5}$$

where u_i comes from v_i or x_i determined by a crossover rate CR, j = 1, 2, …, D, and $jrand$ is a random index in {1,2,…,D} to ensure that at least one dimension of u_i comes from v_i.

$$x_i = \begin{cases} u_i, \text{if } f(u_i) \ge f(x_p) \\ x_p, \text{otherwise} \end{cases} \tag{6}$$

where x_p is the nearest individual in the parent population to the current individual x_i. For the maximization MMOPs, if the fitness of u_i is better than the fitness of x_p, the u_i is selected to enter the next iteration; otherwise, x_p is select to enter the next iteration. The complete AGDE is shown in **Algorithm 1**.

Algorithm 1 AGDE

Begin

Randomly initialize the $\boldsymbol{pop}_t$ (t=0) and set archive $\boldsymbol{A}$ as empty;
While FEs<$MaxFEs$
  If (t%T==0) /*Clear the set $\boldsymbol{A}$ every T iterations*/
    Clear the set $\boldsymbol{A}$;
  End If
  For each individual $\boldsymbol{x}_i$
    Find the two nearest individuals of $\boldsymbol{x}_i$;
    Produce $\boldsymbol{v}_i$ using AMS by (1) or (3);
    Produce $\boldsymbol{u}_i$ using crossover operator by (5);
  End For
  For each trial vector $\boldsymbol{u}_i$
    Evaluate the fitness value of $\boldsymbol{u}_i$;
    Select $\boldsymbol{x}_i$ to enter the next iteration $\boldsymbol{pop}_{t+1}$ by (6);
  End For
  Find the global optimum in $\boldsymbol{pop}_{t+1}$ and store it in the archive $\boldsymbol{A}$;
  Perform GDEL on the solutions in $\boldsymbol{A}$ by (4);
  Merge the new solutions generated by GDEL into $\boldsymbol{pop}_{t+1}$;
  Randomly remove $|\boldsymbol{A}|$ individuals from $\boldsymbol{pop}_{t+1}$;
  t=t+1;
End While

End

III. Experimental Studies

This section introduces the experimental study, which is divided into three parts. Firstly, the test functions and experimental settings are introduced. Secondly, we introduce the compared algorithms and the related parameter settings. Thirdly, the performance of AGDE and the compared algorithms are analyzed.

A. Test Functions and Experimental Settings

The performance of AGDE is tested on the widely used CEC'2013 benchmark set [31] in dealing with MMOPs, which includes 20 multimodal test functions. Among them, F1, F2, and F3 are the one-dimensional problems, which have 2, 5, and 1 global optima/optimum, respectively. F4-F7 and F10-F13 are the two-dimensional problems, which are complex problems and have many local optima. F8, F9, F14, and F15 are the three-dimensional problems. The test functions F1-F15 can be named the LDPs. F16-F20 are the more complex problems, which are called the HDPs.

The peak ratio (*PR*), success rate (*SR*), and average fitness evolutions (*AveFEs*) are used to evaluate the performance of AGDE and each compared algorithms. The definition of *PR*, *SR*, and *AveFEs* are same as [23]. The AGDE algorithm is compared with 11 state-of-the-art multimodal optimization algorithms as in [23]. The different *NP* and *MaxFEs* for different test functions are same as [23], which are adopted in each algorithm.

In AGDE, the scaling factor F and the crossover rate CR are all set to 0.5. For fair comparison, each algorithm is conducted in 51 independent runs and the average results are reported. We mainly discuss the experimental results with the accuracy level ε=1.0E-04. Besides, to further test the performance of each algorithm, we test the performance of some algorithms with the different ε. The Wilcoxon rank-sum test [32] is used to statistically evaluate the *PR* results of AGDE and the compared algorithms in the 51 runs, and the significance level is set as 0.05. The symbols "+", "-", and "≈" represent the AGDE is significantly better, worse, and similar than the compared algorithms, respectively.

B. Comparison with State-of-the-Art Algorithms

To better investigate the performance of AGDE and the compared algorithms, we designed three different experiments. The first experiment is calculated the *PR* and *SR* results in each algorithm. The second experiment is calculated the *AveFEs* of AGDE, Self_CCDE, PNPCDE, LoICDE, NCDE, LIPS, and CDE with the accuracy ε are 1.0E-02, 1.0E-03, and 1.0E-04. The third experiment shows the landscape of AGDE in different iterations.

1) The PR and SR Results of Different Algorithms

For the test functions F1-F20, each algorithm is terminated when achieves the *MaxFEs*. The results of *PR* and *SR* on F1-F20 with the accuracy ε=1.0E-04 are shown in Table I. The best *PR* values are **bolded** in all the algorithms.

From Table I, we can find that the performance of AGDE is the best on 14 functions out of all the 20 functions. For F1-F6 and F10, the *SR* results of AGDE are 1.000 in each run, which means that the AGDE can find all the global peaks. Among all the compared algorithms, only MOMMOP can achieve the same performance as AGDE. The other algorithms fail on at least one function, such as the CDE fails on F4 and F6, the Self_CCDE and PNPCDE fail on F6. MOMMOP performs better than AGDE in F7-F9, but our AGDE still find most of the global peaks. Especially for F8, AGDE find all the global peaks in most of the runs (i.e., *SR*=0.727).

The functions F11-F15 are the complex composition functions with the LDPs. Our AGDE performs best on F14 and F15, and performs slightly worse than Self_CCDE, NCDE, LIPS, and MOMMOP on F11. For F12 and F13, AGDE performs competitive with NCDE, LIPS and MOMMOP. However, the performance of AGDE is still promising and significantly better than most of the other compared algorithms.

For F16-F20, the advantage of AGDE is obvious, which means that our AGDE has a better performance in dealing with the complex composition functions. Compared with the other algorithms, AGDE can find the most peaks of the function F16-F20 in each run. Especially for F19 and F20, they are 10-dimensional and 20-dimensional functions, the compared algorithms are difficult to locate the global peaks. But our AGDE still obtains a best *PR* results for F19 and F20, which are 0.450 and 0.282, respectively. It indicates that the AMS and IFA strategies of AGDE have a significance effect on these HDPs functions.

In the bottom row of Table I, we list the number of functions that AGDE performs significantly better, worse, and similar than the compared algorithms, which are counted by the results of Wilcoxon rank-sum test. From the results, we can find that the number of "+" is more than the number of "-", which means that the AGDE has the overall better performance in solving MMOPs.

TABLE I. THE PR AND SR OF DIFFERENT ALGORITHMS

Func	AGDE		CDE		Self_CCDE		PNPCDE		LoICDE		NCDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000
F2	**1.000**	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000
F3	**1.000**	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000
F4	**1.000**	1.000	0.985(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	0.975(+)	0.902	**1.000**(≈)	1.000
F5	**1.000**	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000
F6	**1.000**	1.000	0.640(+)	1.000	0.942(+)	0.490	0.537(+)	0.000	**1.000**(≈)	1.000	0.305(+)	0.000
F7	0.850	0.000	0.861(–)	0.000	0.884(–)	0.020	0.874(–)	0.000	0.705(+)	0.020	0.873(–)	0.000
F8	0.952	0.727	0.000(+)	0.000	0.994(–)	0.882	0.000(+)	0.000	0.000(+)	0.000	0.002(+)	0.000
F9	0.366	0.000	0.474(–)	0.000	0.459(–)	0.000	0.474(–)	0.000	0.187(+)	0.000	0.461(–)	0.000
F10	**1.000**	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	0.988(+)	0.863
F11	0.667	0.000	0.330(+)	0.000	0.778(–)	0.137	0.667(≈)	0.000	0.660(+)	0.000	0.727(–)	0.059
F12	0.750	0.000	0.002(+)	0.000	0.422(+)	0.000	0.002(+)	0.000	0.495(+)	0.000	0.253(+)	0.000
F13	0.667	0.000	0.140(+)	0.000	0.653(+)	0.000	0.461(+)	0.000	0.510(+)	0.000	0.667(≈)	0.000
F14	**0.667**	0.000	0.024(+)	0.000	0.520(+)	0.000	0.258(+)	0.000	0.657(+)	0.000	**0.667**(≈)	0.000
F15	**0.702**	0.000	0.005(+)	0.000	0.343(+)	0.000	0.015(+)	0.000	0.299(+)	0.000	0.319(+)	0.000
F16	**0.667**	0.000	0.000(+)	0.000	0.655(+)	0.000	0.000(+)	0.000	0.556(+)	0.000	**0.667(≈)**	0.000
F17	**0.479**	0.000	0.000(+)	0.000	0.246(+)	0.000	0.000(+)	0.000	0.222(+)	0.000	0.250(+)	0.000
F18	**0.579**	0.000	0.167(+)	0.000	0.337(+)	0.000	0.150(+)	0.000	0.219(+)	0.000	0.500(+)	0.000
F19	**0.450**	0.000	0.000(+)	0.000	0.113(+)	0.000	0.000(+)	0.000	0.032(+)	0.000	0.148(+)	0.000
F20	**0.282**	0.000	0.000(+)	0.000	0.024(+)	0.000	0.000(+)	0.000	0.126(+)	0.000	0.150(+)	0.000
+ (AGDE is better)			12		10		11		14		9	
– (AGDE is worse)			2		4		2		0		3	
≈			6		6		7		6		8	

Func	LIPS		MOMMOP		SDE		NSDE		r2pso		r3pso	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F1	0.833(+)	0.686	**1.000**(≈)	1.000	0.657(+)	0.373	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000
F2	**1.000**(≈)	1.000	**1.000**(≈)	1.000	0.737(+)	0.529	0.776(+)	0.667	**1.000**(≈)	1.000	**1.000**(≈)	1.000
F3	0.961(+)	0.961	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000	**1.000**(≈)	1.000
F4	0.990(+)	0.961	**1.000**(≈)	1.000	0.284(+)	0.000	0.240(+)	0.000	0.670(+)	0.784	0.966(+)	0.863
F5	**1.000**(≈)	1.000	**1.000**(≈)	1.000	0.922(+)	0.843	0.745(+)	0.490	**1.000**(≈)	1.000	**1.000**(≈)	1.000
F6	0.246(+)	0.000	**1.000**(≈)	1.000	0.056(+)	0.000	0.056(+)	0.000	0.388(+)	0.000	0.687(+)	0.000
F7	0.400(+)	0.000	**1.000**(–)	1.000	0.053(+)	0.000	0.053(+)	0.000	0.509(+)	0.000	0.434(+)	0.000
F8	0.086(+)	0.000	**1.000**(–)	1.000	0.013(+)	0.000	0.013(+)	0.000	0.000(+)	0.000	0.421(+)	0.000
F9	0.108(+)	0.000	**1.000**(–)	1.000	0.013(+)	0.000	0.006(+)	0.000	0.091(+)	0.353	0.127(+)	0.000
F10	0.748(+)	0.000	**1.000**(≈)	1.000	0.147(+)	0.000	0.098(+)	0.000	0.788(+)	0.000	0.850(+)	0.157
F11	**0.974**(–)	0.843	0.710(–)	0.040	0.314(+)	0.000	0.248(+)	0.000	0.667(+)	0.000	0.650(+)	0.000
F12	0.574(+)	0.000	**0.955**(–)	0.600	0.208(+)	0.000	0.135(+)	0.000	0.448(+)	0.000	0.537(+)	0.000
F13	**0.794**(–)	0.176	0.667(≈)	0.000	0.297(+)	0.000	0.225(+)	0.000	0.660(+)	0.000	0.647(+)	0.000
F14	0.644(+)	0.000	0.667(≈)	0.000	0.216(+)	0.000	0.190(+)	0.000	0.403(+)	0.000	0.637(+)	0.000
F15	0.336(+)	0.000	0.618(+)	0.000	0.108(+)	0.000	0.125(+)	0.000	0.103(+)	0.000	0.213(+)	0.000
F16	0.307(+)	0.000	0.630(+)	0.000	0.108(+)	0.000	0.170(+)	0.000	0.000(+)	0.000	0.431(+)	0.000
F17	0.168(+)	0.000	0.505(+)	0.000	0.076(+)	0.000	0.108(+)	0.000	0.000(+)	0.000	0.096(+)	0.000
F18	0.098(+)	0.000	0.497(+)	0.000	0.026(+)	0.000	0.163(+)	0.000	0.000(+)	0.000	0.100(+)	0.000
F19	0.000(+)	0.000	0.230(+)	0.000	0.105(+)	0.000	0.098(+)	0.000	0.000(+)	0.000	0.032(+)	0.000
F20	0.000(+)	0.000	0.125(+)	0.000	0.000(+)	0.000	0.123(+)	0.000	0.000(+)	0.000	0.078(+)	0.000
+	16		6		19		18		16		16	
–	2		5		0		0		0		0	
≈	2		7		1		2		4		4	

'+', '–', and '≈' indicate that the results of the algorithm are significantly better than, worse than, and similar to the ones of AGDE by Wilcoxon's rank sum test with α=0.05.

2) The AveFEs Results with Different Accuracies

From Table II, we can find that our AGDE and all the compared algorithms can find all the global peaks in F1-F5. To distinguish the performance of these algorithms in F1-F5, the *AveFEs* is used to measure these algorithms with different accuracy. Table II shows the *AveFEs* results of AGDE, Self_CCDE, PNPCDE, LoICDE, NCDE, LIPS, and CDE in function F1-F5.

From Table II, the *AveFEs* results of AGDE are significantly better than Self_CCDE, LoICDE, NCDE, and LIPS on F1 with different accuracy. In ε=1.0E-02, AGDE performs not significantly difference with PNPCDE and CDE for F1 (i.e., the *AveFEs* of AGDE, PNPCDE, and CDE are 1.59E+02, 1.62E+02, and 1.60E+02, respectively). For F2 and F4, AGDE performs significantly better than Self_CCDE, PNPCDE, LoICDE, NCDE, LIPS, and CDE when ε=1.0E-02.

In ε=1.0E-03, the *AveFEs* results of AGDE are similar with PNPCDE and CDE on the function F1. Moreover, AGDE performs better than Self_CCDE, PNPCDE, LoICDE, NCDE, and CDE on F2. For F3-F4, AGDE still is a winner in Self_CCDE, PNPCDE, LoICDE, NCDE, and CDE. For F5, the results of AGDE is promising and it performs better than Self_CCDE, PNPCDE, LoICDE, and CDE.

In ε=1.0E-04, AGDE performs better than Self_CCDE, PNPCDE, LoICDE, NCDE, and CDE in F1, F4, and F5. For F2 and F3, the *AveFEs* results of AGDE are worse than LIPS and NCDE, but AGDE still performs better than Self_CCDE, PNPCDE, LoICDE, and CDE. In general, AGDE performs better than Self_CCDE, PNPCDE, LoICDE, NCDE, LIPS, and CDE in different accuracy.

TABLE II. THE AVEFES OF DIFFERENT ALGORITHMS ON F1-F5

				ε=1.0E-02			
Func	AGDE	Self_CCDE	PNPCDE	LoICDE	NCDE	LIPS	CDE
F1	**1.59E+02**	3.55E+02 (+)	**1.62E+02 (≈)**	1.73E+02 (+)	6.85E+02 (+)	1.64E+04 (+)	**1.60E+02 (≈)**
F2	**1.63E+02**	4.02E+02 (+)	3.97E+02 (+)	4.41E+02 (+)	4.58E+02 (+)	3.04E+02 (+)	4.36E+02 (+)
F3	**1.42E+02**	2.23E+02 (+)	3.37E+02 (+)	2.45E+02 (+)	2.21E+02 (+)	2.60E+02 (+)	3.64E+02 (+)
F4	**2.07E+03**	4.15E+03 (+)	1.21E+04 (+)	5.07E+03 (+)	2.60E+03 (+)	**2.09E+03 (≈)**	1.61E+04 (+)
F5	**3.71E+02**	9.68E+03 (+)	1.33E+03 (+)	6.51E+02 (+)	7.84E+02 (+)	5.17E+02 (+)	1.52E+03 (+)
+ (AGDE is better)		5	4	5	5	4	4
- (AGDE is worse)		0	0	0	0	0	0
≈		0	1	0	0	1	1
				ε=1.0E-03			
Func	AGDE	Self_CCDE	PNPCDE	LoICDE	NCDE	LIPS	CDE
F1	**1.60E+02**	3.55E+02 (+)	**1.62E+02 (≈)**	1.73E+02 (+)	6.85E+02 (+)	1.64E+04 (+)	**1.60E+02 (≈)**
F2	**4.45E+02**	7.61E+02 (+)	1.08E+03 (+)	1.24E+03 (+)	9.27E+02 (+)	**4.93E+02 (≈)**	1.39E+03 (+)
F3	**5.55E+02**	**5.55E+02 (≈)**	8.33E+02 (+)	5.84E+02 (+)	**4.94E+02 (≈)**	1.40E+03 (+)	1.11E+03 (+)
F4	**3.26E+03**	7.07E+03 (+)	2.28E+04 (+)	1.48E+04 (+)	3.73E+03 (+)	4.28E+03 (+)	2.71E+04 (+)
F5	2.65E+03	**2.67E+03 (≈)**	3.69E+04 (+)	1.88E+03 (-)	**1.57E+02 (-)**	9.35E+02 (-)	5.42E+03 (+)
+ (AGDE is better)		3	4	4	3	3	4
- (AGDE is worse)		0	0	1	1	1	0
≈		2	1	0	1	1	1
				ε=1.0E-04			
Func	AGDE	Self_CCDE	PNPCDE	LoICDE	NCDE	LIPS	CDE
F1	**1.56E+02**	7.18E+02 (+)	1.65E+02 (+)	1.75E+02 (+)	6.88E+02 (+)	1.66E+04 (+)	1.76E+02 (+)
F2	1.63E+03	2.36E+04 (+)	2.66E+03 (+)	3.00E+03 (+)	1.66E+03 (≈)	**7.62E+02 (-)**	3.23E+03 (+)
F3	1.42E+03	1.25E+03 (≈)	2.43E+03 (+)	1.54E+03 (≈)	**9.84E+02 (-)**	2.68E+03 (+)	3.36E+03 (+)
F4	**1.61E+03**	4.42E+04 (+)	3.38E+04 (+)	2.73E+04 (+)	4.96E+03 (+)	5.00E+03 (+)	3.93E+04 (+)
F5	**1.22E+03**	1.28E+04 (+)	8.99E+03 (+)	3.50E+03 (+)	2.48E+03 (+)	**1.50E+03 (≈)**	1.12E+04 (+)
+ (AGDE is better)		4	5	4	3	3	5
- (AGDE is worse)		0	0	0	1	1	0
≈		1	0	1	1	1	0

3) *The Landscape of AGDE in Different Generation*

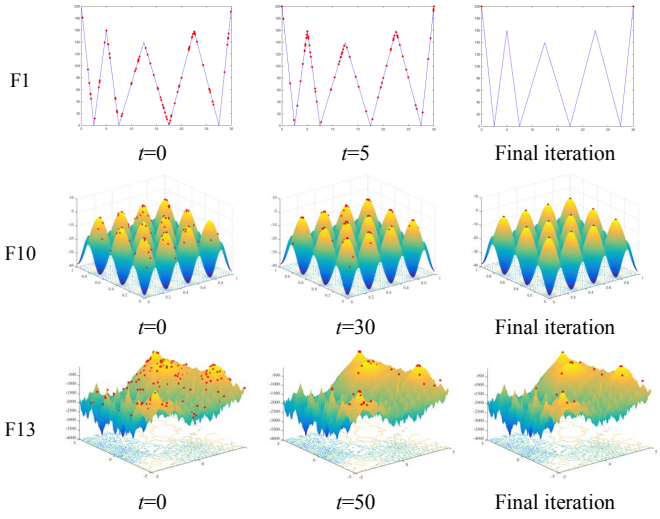


Fig. 3. Distribution of solutions in different iterations on F1, F10, and F13.

To observe the evolution process of the population under AGDE algorithm, we recorded the distribution of individuals in different iterations. Gen=0 represents the initial distribution of individuals. According to the features of each problem, we describe three typical problems. The first type is the one-dimensional problems (i.e., F1) when iteration is 0, 5, and the final iteration. The second type is F10, which have many global optima (i.e., 12). We list the landscape of F10 when iteration is 0, 30, and the final iteration. The third type is the complex composition problems (i.e., F13), and we list the landscape of F13 when iteration is 0, 50, and the final iteration. From Fig. 3, we have the following three observations.

(1) F1 is the simple LDPs, which have two global optima and three local optima, respectively. From Fig. 3, it can find that our AGDE almost can locate all the global optima in the 5^{th} iterations. In the later iterations, most of the individuals converge to the global/local optima. Namely, the individuals that in the local peaks can jump out of the local optima and move towards the global optima. It indicates that the AGDE has the better ability of balance the diversity and convergence in solving MMOPs.

(2) When dealing with MMOPs, the ability to locate multiple peaks simultaneously is a key indicator to measure an algorithm. From Fig. 3, we can find that our AGDE can quickly locate many global peaks of F10. It indicates that AGDE performs better when solving the problems with many global optima. With the increase of the iteration, many individuals gradually move towards the peaks until locate the global optima.

(3) According to the distribution of individuals in F13, it can be seen that with the increase of iterations, each individual is moving towards the different peaks. This shows that AGDE has a better search ability in complex problems.

TABLE III. THE PR AND SR OF AGDE ON F1-F20 WITH DIFFERENT *T* VALUES

T / Func	T=1		T=3		T=5		T=7		T=9	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F2	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F3	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F4	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F5	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F6	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F7	0.790	0.000	0.794	0.000	**0.850**	0.000	0.833	0.000	0.813	0.000
F8	0.942	0.857	0.894	0.633	**0.952**	0.727	0.943	0.571	0.940	0.714
F9	0.349	0.000	0.348	0.000	0.366	0.000	**0.373**	0.000	0.349	0.000
F10	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F11	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000
F12	**0.750**	0.000	0.725	0.000	**0.750**	0.000	0.732	0.000	0.732	0.000
F13	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000
F14	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000
F15	0.696	0.000	0.675	0.000	**0.702**	0.000	0.607	0.000	0.625	0.000
F16	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000
F17	0.464	0.000	**0.500**	0.000	0.479	0.000	0.446	0.000	0.393	0.000
F18	0.548	0.000	0.567	0.000	**0.579**	0.000	0.548	0.000	0.571	0.000
F19	0.334	0.000	0.350	0.000	**0.450**	0.000	0.357	0.000	0.411	0.000
F20	0.175	0.000	0.250	0.000	**0.282**	0.000	0.232	0.000	0.230	0.000
NBPR	12		12		**18**		12		11	

C. *Impacts of Parameter Settings*

In this section, we investigate the effect of parameter *T* on the AGDE. Table III lists the *PR* and *SR* of AGDE on F1-F20 when *T* is set as 1, 3, 5, 7, and 9, respectively, and the best results are marked as **bold**. The last row shows the number of best *PR* (NBPR) values obtained by setting different *T* values. From Table III, we can find that the NBPR is the largest (i.e., 18) when *T* is 5. Moreover, compared with other *T* values (i.e., 1, 3, 7, and 9), the AGDE performs best in HDPs when *T* is 5. That is, our AGDE algorithm obtains the best results on F18, F19, and F20. It indicates that the performance of AGDE is best when *T* is 5. Therefore, *T* is set to 5 in this paper.

IV. CONCLUSION

This paper introduces a novel AGDE algorithm for solving MMOPs. The AMS and IFA strategy are the two key techniques in AGDE. AMS is proposed to control the evolution direction of individuals by using the nearest information of the current individual. By this way, the AGDE algorithm can balance the diversity and convergence of the population. The IFA strategy collects the global optima by introducing an archive. Besides, the GDEL strategy can refine the accuracy of the solutions. By comparing with the state-of-the-art multimodal optimization algorithms, the AGDE shows the better performance in the CEC'2013 benchmark.

In the future, we will further improve the performance of the AGDE and extend it to solve the real-word MMOPs, such as electricity marks[33], resource constrained project scheduling [34], and cloud computing [35][36].

REFERENCES

[1] W. Sheng, S. Swift, L. Zhang, and X. Liu, "A weighted sum validity function for clustering with a hybrid niching genetic algorithm," *IEEE Trans. Cybern*., vol. 35, no. 6, pp. 1156-1167, Dec. 2005.

[2] A. Y. Goharrizi, R. Singh; A. M. Gole, S. Filizadeh, J. C. Muller, and R. P. Jayasinghe, "A parallel multimodal optimization algorithm for simulation-based design of power systems," *IEEE Trans. Power Del*., vol. 30, no. 5, pp. 2128-2137, Oct. 2015.

[3] K. C. Wong, K. S. Leung, and M. H. Wong, "Protein structure prediction on a lattice model via multimodal optimization techniques," in *Proc. IEEE Congr. Evol. Comput*., 2010, pp. 155-162.

[4] Z. J. Wang, Z. H. Zhan, S. Kwong, H. Jin, and J. Zhang, "Adaptive granularity learning distributed particle swarm optimization for large-scale optimization," *IEEE Trans. Cybern*., DOI: 10.1109/TCYB.2020. 2977956, Feb. 2020.

[5] X. Zhang, K. J. Du, Z. H. Zhan, S. Kwong, T. L. Gu, and J. Zhang, "Cooperative co-evolutionary bare-bones particle swarm optimization with function independent decomposition for large-scale supply chain network design with uncertainties," *IEEE Trans. Cybern*., DOI: 10.1109/TCYB. 2019.2937565, Aug. 2019.

[6] X. F. Liu, *et al.*, "Neural network-based information transfer for dynamic optimization," *IEEE Trans. Neural Netw. and Learn. Syst.*, DOI: 10.1109/TNNLS.2019.2920887, Jun. 2019.

[7] Z. J. Wang, Z. H. Zhan, W. J. Yu, Y. Lin, J. Zhang, T. L. Gu, and J. Zhang, "Dynamic group learning distributed particle swarm optimization for large-scale optimization and its application in cloud workflow scheduling," *IEEE Trans. Cybern*., DOI: 10.1109/TCYB. 2019.2933499, Aug. 2019.

[8] D. Liang, Z. H. Zhan, Y. Zhang, and J. Zhang, "An efficient ant colony system approach for new energy vehicle dispatch problem," *IEEE Trans. Intell. Transp. Syst*., DOI:10.1109/TITS.2019.2946711, Oct. 2019.

[9] O. J. Mengshoel and D. E. Goldberg, "The crowding approach to niching in genetic algorithms," *Evol. Comput*., vol. 16, no. 3, pp. 315-354, Sep. 2008.

[10] J. Y. Li, Z. H. Zhan, C. Wang, H. Jin, and J. Zhang, "Boosting data-driven evolutionary algorithm with localized data generation," *IEEE Trans. Evol. Comput*., DOI:10.1109/TCYB.2020.2977956, Feb. 2020.

[11] S. P. Hoseini Alinodehi, S. Moshfe, M. Saber Zaeimian, A. Khoei, and K. Hadidi, "High-speed general purpose genetic algorithm processor," *IEEE Trans. Cybern*., vol. 46, no. 7, pp. 1551-1565, Jul. 2016.

[12] M. Birattari, P. Pellegrini, and M. Dorigo, "On the invariance of ant colony optimization," *IEEE Trans. Evol. Comput*., vol. 11, no. 6, pp. 732-742, Dec. 2007.

[13] X. F. Liu, Z. H. Zhan, Y. Gao, J. Zhang, S. Kwong, and J. Zhang, "Coevolutionary particle swarm optimization with bottleneck objective learning strategy for many-objective optimization," *IEEE Trans. Evol. Comput*., vol. 23, no. 4, pp. 587-602, Aug. 2019.

[14] Z. H. Zhan, Z. J. Wang, H. Jin, and J. Zhang, "Adaptive distributed differential evolution," *IEEE Trans. Cybern*., DOI: 10.1109/TCYB. 2019.2944873, Oct. 2019.

[15] Z. H. Zhan, *et al.*, "Cloudde: A heterogeneous differential evolution algorithm and its distributed cloud version," *IEEE Trans*. *Parallel and Dist*. *Syst*., vol. 28, no. 3, pp. 704-716, March. 2017.

[16] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," in *Proc*. *IEEE Congr*. *Evol*. *Comput*., 2004, pp. 1382-1389.

[17] X. Li, "Efficient differential evolution using speciation for multimodal function optimization," in *Proc*. *Genetic Evol*. *Comput*. *Conf*., 2005, pp. 873-880.

[18] J. P. Li, M. E. Balazs, G. T. Parks, and P. J. Clarkson, "A species conserving genetic algorithm for multimodal function optimization," *IEEE Trans. Evol. Comput*., vol. 10, no. 3, pp. 207-234, 2002.

[19] X. Li, "Niching without niching parameters: particle swarm optimization using a ring topology," *IEEE Trans. Evol. Comput*., vol. 14, no. 1, pp. 150-169, Feb. 2010.

[20] W. Gao, G. G. Yen, and S. Liu, "A cluster-based differential evolution with self-adaptive strategy for multimodal optimization," *IEEE Trans. Cybern*., vol. 44, no. 8, pp. 1314-1327, Aug. 2014.

[21] Z. J. Wang, *et al.*, "Dual-strategy differential evolution with affinity propagation clustering for multimodal optimization problems," *IEEE Trans*. *Evol*. *Comput*., vol. 22, no. 6, pp. 894-908, Dec. 2018.

[22] Z. J. Wang, *et al.*, "Automatic niching differential evolution with contour prediction approach for multimodal optimization problems," *IEEE Trans*. *Evol*. *Comput*., vol. 24, no. 1, pp. 114-128, Feb. 2020.

[23] H. Zhao, *et al.*, "Local binary pattern based adaptive differential evolution for multimodal optimization problems", *IEEE Trans. Cybern*., DOI: 10.1109/TCYB.2019.2927780, 2019.

[24] Z. G. Chen, Z. H. Zhan, H. Wang, and J. Zhang, "Distributed individuals for multiple peaks: A novel differential evolution for multimodal optimization problems," *IEEE Trans. Evol. Comput*., DOI: 10.1109/TEVC. 2019.2944180, Oct. 2019.

[25] B. Y. Qu, P. N. Suganthan, and J. J. Liang, "Differential evolution with neighborhood mutation for multimodal optimization," *IEEE Trans. Evol. Comput*., vol. 16, no. 5, pp. 601-614, Oct. 2012.

[26] S. Biswas, S. Kundu, and S. Das, "Inducing niching behavior in differential evolution through local information sharing," *IEEE Trans. Evol. Comput*., vol. 19, no. 2, pp. 246-263, Apr. 2015.

[27] S. Biswas, S. Kundu, and S. Das, "An improved parent-centric mutation with normalized neighborhoods for inducing niching behavior in differential evolution," *IEEE Trans. Cybern*., vol. 44, no. 10, pp. 1726-1737, Oct. 2014.

[28] R. Cheng, M. Q. Li, K. Li, and X. Yao, "Evolutionary multiobjective optimization-based multimodal optimization: titness landscape approximation and peak detection," *IEEE Trans. Evol. Comput*., vol. 22, no. 5, pp. 692-706, Oct. 2018.

[29] A. Basak, S. Das, and K. C. Tan, "Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection," *IEEE Trans. Evol. Comput*., vol. 17, no. 5, pp. 666-685, Oct. 2013.

[30] Y. Wang, H. X. Li, G. G. Yen, and W. Song, "MOMMOP: Multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems," *IEEE Trans. Cybern*., vol. 45, no. 4, pp. 830-843, Apr. 2015.

[31] X. Li, A. Engelbrecht, and M. G. Epitropakis, "Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization," Evol. Comput. Mach. Learn. Group, RMIT Univ., Melbourne, VIC, Australia, Tech. Rep., 2013. [Online].Available:http://titan.csit.rmit.edu.au/~e46507/cec13-niching/ competition/cec2013-niching-benchmark-tech-report.pdf.

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

[33] F. Zaman, S. M. Elsayed, T. Ray, and R. A. Sarkerr, "Evolutionary algorithms for finding Nash equilibria in electricity markets," *IEEE Trans. Evol. Comput*., vol. 22, no. 4, pp. 536-549, Aug. 2018.

[34] S. Elsayed, R. Sarker, T. Ray, and C. Coello Coello, "Consolidated optimization algorithm for resource-constrained project scheduling problems," *Inf. Sci.*, vols. 418-419, pp. 346-362, Dec. 2017.

[35] X. F. Liu, Z. H. Zhan, J. D. Deng, Y. Li, T. L. Gu, and J. Zhang, "An energy efficient ant colony system for virtual machine placement in cloud computing," *IEEE Trans. Evol. Comput*., vol. 22, no. 1, pp. 113-128, Feb. 2018.

[36] Z. H. Zhan, X. F. Liu, Y. J. Gong, J. Zhang, H. Chung, and Y. Li, "Cloud computing resource scheduling and a survey of its evolutionary approaches," *ACM Computing Surveys*, vol. 47, no. 4, pp. 1-33, Jul. 2015.