Multimodal Estimation of Distribution Algorithms

Qiang Yang, *Student Member, IEEE*, Wei-Neng Chen, *Member, IEEE*, Yun Li, *Member, IEEE*, C. L. Philip Chen, *Fellow, IEEE*, Xiang-Min Xu, *Member, IEEE*, and Jun Zhang, *Senior Member, IEEE*

***Abstract*—Taking the advantage of estimation of distribution algorithms (EDAs) in preserving high diversity, this paper proposes a multimodal EDA. Integrated with clustering strategies for crowding and speciation, two versions of this algorithm are developed, which operate at the niche level. Then these two algorithms are equipped with three distinctive techniques: 1) a dynamic cluster sizing strategy; 2) an alternative utilization of Gaussian and Cauchy distributions to generate offspring; and 3) an adaptive local search. The dynamic cluster sizing affords a potential balance between exploration and exploitation and reduces the sensitivity to the cluster size in the niching methods. Taking advantages of Gaussian and Cauchy distributions, we generate the offspring at the niche level through alternatively using these two distributions. Such utilization can also potentially offer a balance between exploration and exploitation. Further, solution accuracy is enhanced through a new local search scheme probabilistically conducted around seeds of niches with probabilities determined self-adaptively according to fitness values of these seeds. Extensive experiments conducted on 20 benchmark multimodal problems confirm that both algorithms can achieve competitive performance compared with several state-of-the-art multimodal algorithms, which is supported by nonparametric tests. Especially, the proposed algorithms are very promising for complex problems with many local optima.**

***Index Terms*—Estimation of distribution algorithm (EDA), multimodal optimization, multiple global optima, niching.**

Manuscript received August 29, 2015; revised November 24, 2015; accepted January 21, 2016. This work was supported in part by the National Natural Science Foundation of China Project under Grant 61379061, Grant 61332002, and Grant 61511130078, in part by the Natural Science Foundation of Guangdong for Distinguished Young Scholars under Grant 2015A030306024, in part by the Guangdong Special Support Program under Grant 2014TQ01X550, and in part by the Guangzhou Pearl River New Star of Science and Technology Project under Grant 201506010002. This paper was recommended by Associate Editor P. P. Angelov. *(Corresponding author: Wei-Neng Chen.)*

Q. Yang is with the Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, Guangzhou 510006, China, also with the Engineering Research Center of Supercomputing Engineering Software, Ministry of Education, Guangzhou 510006, China, and also with the School of Data and Computer Science, Sun Yat-sen University, Guangzhou 510006, China.

W.-N. Chen is with School of Computer Science and Engineering, South China University of Technology, Guangzhou, 510006, China, also with the Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, Guangzhou, 510006, China, and with the Engineering Research Center of Supercomputing Engineering Software, Ministry of Education, Guangzhou, 510006, China (e-mail: cwnraul634@aliyun.com).

Y. Li is with the School of Engineering, University of Glasgow, Glasgow G12 8LT, U.K.

C. L. P. Chen is with the Department of Computer and Information Science, University of Macau, Macau 999078, China.

X.-M. Xu is with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou 510006, China.

J. Zhang is with the Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, Guangzhou 510006, China, and also with the Engineering Research Center of Supercomputing Engineering Software, Ministry of Education, Guangzhou 510006, China (e-mail: junzhang@ieee.org).

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors. This includes a PDF file that contains additional tables not included within the paper itself. The total size of the file is 1.17 MB.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2016.2523000

I. Introduction

MULTIMODAL optimization, which seeks multiple optima simultaneously, has received much attention recently. Many real-world problems involve multiple optima, such as protein structure prediction [1], holographic design [2], data mining [3]–[5], and electromagnetic design [6], [7]. Hence, it is desirable to find as many optima as possible in the global optimization. However, different from finding just one optimum in single optimization [8]–[13], locating multiple global or local optima simultaneously is qualitatively more challenging.

For such problems, classical evolutionary algorithms (EAs), including differential evolution (DE) [9], [14]–[19], genetic algorithm (GA) [20], [21], and particle swarm optimization (PSO) [12], [22]–[25], lose feasibility and effectiveness, because their overall learning and updating makes the population tend to converge toward one dominating candidate. Therefore, to locate multiple optima simultaneously using classical EAs, a multimodality-specific mechanism is necessary.

So far, niching [26]–[29] has been widely used to help an EA maintain a diverse population in multimodal optimization. Niching achieves this by partitioning the whole population into subpopulations using techniques such as crowding [29] and speciation [30]–[32] and each subpopulation is responsible for one area to locate one or a small number of optima. Because of the sensitivity to parameters in niching methods, for instance, crowding [29] is sensitive to the crowding size and speciation is sensitive to the niching radius [30], [31], various parameter-free or parameter-insensitive techniques are hence developed to improve niching, such as hill-valley [32]–[34], recursive middling [35], history-based topological speciation [36], and clustering [37], [38]. However, these niching techniques either cost a number of fitness evaluations or require a large memory or introduce less sensitive parameters to partition the population into niches.

Alternative to niching, learning, and updating strategies are modified to enhance the diversity for EAs, such as GA [26], [28], [35], DE [38]–[40], and PSO [41], [42].

The resultant algorithms extend learning beyond dominant individuals by offering relatively poor individuals improved survival to the next generation [38]–[40]. Together, they maintain the diversity of the population at the individual level.

Recently, a new family of EAs—the estimation of distribution algorithms (EDAs) [10], [11], [43]–[46] has emerged, which preserves diversity at the population level. Generally, an EDA generates offspring by sampling from the probability distribution estimated from promising individuals. However, current EDAs are designed for single optimization and so far, there exists no report in the literature on applying EDAs to deal with multimodal optimization.

Since an EDA maintains significant diversity at the population level, it should offer improved assistance for locating multiple optima. Motivated by this observation, we propose a multimodal EDA (MEDA), to cope with multimodal optimization. Specifically, the characteristics of MEDA are as follows.

1) Two popular niching strategies, crowding and speciation, are incorporated in MEDA, leading to MCEDA and MSEDA, respectively. Instead of operating at the population level, MEDAs (MCEDA and MSEDA) operate at the niche level. Further, different from traditional EDAs, all individuals in every niche participate in the estimation of distribution of that niche.
2) A dynamic cluster sizing strategy is proposed to cooperate with the two niching methods. This strategy not only affords a potential balance between exploration and exploitation, but also reduces the sensitivity of the used niching methods to the cluster size.
3) Gaussian distribution and Cauchy distribution are alternatively used to generate offspring for each niche, instead of just using Gaussian distribution in classical EDAs. Taking the advantages of both distributions, this alternative usage offers an extra potential balance between exploration and exploitation.
4) A new local search scheme based on Gaussian distribution is proposed to refine the obtained solutions, leading to local search-based MEDAs (LMEDAs), containing LMCEDA and LMSEDA. This local search is probabilistically performed around seeds of niches with probabilities self-determined according to fitness values of these seeds.

Extensive experiments have been conducted on 20 widely used benchmark multimodal functions and the experimental results supported by nonparametric Wilcoxon rank-sum test consistently demonstrate that LMEDAs are able to provide better and more consistent performance than the state-of-the-art multimodal algorithms on majority of the benchmark functions without incurring any serious computational burden.

Following a review of multimodal optimization techniques and EDAs in Section II, the proposed MEDAs are detailed in Section III. Extensive experiments are conducted to test MEDAs in Section IV. Finally, the conclusions and discussions are presented in Section V.

Algorithm 1 Crowding-Based DE (CDE) [29]

```
Input: population size $NP$, crowding size $K$
1: Randomly initialize the population;
2: For $i = 1$ to $NP$
3:     Produce a child $\boldsymbol{o_i}$ using standard DE;
4:     Randomly select $K$ individuals in the population to
       form a crowd;
5:     Compare the fitness of $\boldsymbol{o_i}$ with that of the nearest
       individual found by Eq. (1) and replace it if $\boldsymbol{o_i}$ is
       better;
6: End For
7: Stop if the termination criterion is met. Otherwise go to
   Step 2;
Output: the whole population
```

II. Multimodal Optimization and Estimation of Distribution Algorithms

Without loss of generality, maximization multimodal problems are considered in this paper as in [26], [29], [30], [32], [36], [37], [39]–[42], and [47]–[49]. In addition, in the literature, locating multiple global optima is the main objective of multimodal optimization, and hence is also the concern of this paper.

A. Multimodal Optimization Algorithms

Generally, to cope with multimodal optimization efficiently, two major issues need to be addressed: 1) diversification and 2) intensification. To settle these issues, so far, various niching methods have been proposed and combined with EAs to tackle multimodal problems. At present, crowding [29] and speciation [30] are the two most popular niching methods.

In crowding [29], each generated child is compared with the nearest individual from a crowd formed by randomly selecting K parents in the population. Then if the child is better, it will replace the compared parent. This process is formulated as

$$\arg\min_{\boldsymbol{x}_j \in \boldsymbol{C}_i} \operatorname{dist}(\boldsymbol{x}_j, \boldsymbol{o}_i) = \sqrt{\sum_{d=1}^{D}\left(x_j^d - o_i^d\right)} \tag{1}$$

where $\boldsymbol{o}_i$ is the generated child, $\boldsymbol{C}_i$ is the crowd of this child with K randomly selected parents, $\boldsymbol{x}_j$ is the jth individual in the crowd, $\operatorname{dist}(\boldsymbol{x}_j, \boldsymbol{o}_i)$ is the Euclidean distance between $\boldsymbol{x}_j$ and $\boldsymbol{o}_i$, and D is the dimension size. When applied to DE, the framework of crowding is outlined in Algorithm 1.

In speciation [30], the population is first divided into several species according to the following formula:

$$\boldsymbol{S}_i = \left\{ \boldsymbol{x}_j | \boldsymbol{x}_j \in \boldsymbol{P} \ \& \ \operatorname{dist}(\boldsymbol{x}_{\text{seed}}, \boldsymbol{x}_j) = \sqrt{\sum_{d=1}^{D}\left(x_{\text{seed}}^d - o_i^d\right)} \le r \right\} \tag{2}$$

where $\boldsymbol{S}_i$ is the ith species, $\boldsymbol{P}$ is the exclusive population, where individuals in the $(i-1)$ species are excluded, $\boldsymbol{x}_{\text{seed}}$ is the seed of the species, which is defined as the best individual in this species, and r is the species radius.

Algorithm 2 Speciation-Based DE (SDE) [30]

```
Input: population $\boldsymbol{P}$ with $NP$ members, species radius $r$, minimal species size $m$
1: Randomly initialize the population $\boldsymbol{P}$;
2: Sort the individuals in descending order according to fitness values;
// Determine the species of the current population
3: While $\boldsymbol{P}$ is not empty
4:     The best individual is set as a seed to create a new species;
5:     Find other individuals in this species according to Eq. (2);
6:     Delete all individuals of this species from $\boldsymbol{P}$;
7: End While
8: For each species
9:     Produce an equal number of children using standard DE;
10:    If the species size is less than $m$
11:        Randomly generate new individuals within the radius of the species seed so that the species size reaches $m$;
12:    End If
13:    If a child's fitness is the same as that of the species seed
14:        Replace this child with a new randomly generated individual within the radius of the species seed;
15:    End If
16: End For
17: Combine the children and the parents and then keep the $NP$ best ones;
18: Stop if the termination criterion is met. Otherwise go to Step 2;
Output: the whole population
```

Algorithm 3 Crowding Clustering [37]

```
Input: population $\boldsymbol{P}$, cluster size $M$
1: Randomly generate a reference point $\boldsymbol{R}$;
2: While $\boldsymbol{P}$ is not empty
3:     Select the nearest individual $\boldsymbol{P}_{near}$ to $\boldsymbol{R}$ in $\boldsymbol{P}$ ;
4:     Combine $M$-1 individuals nearest to $\boldsymbol{P}_{near}$ and $\boldsymbol{P}_{near}$ as a crowd;
5:     Eliminate these $M$ individuals from $\boldsymbol{P}$;
6: End While
Output: a set of crowds
```

Algorithm 4 Speciation Clustering [37]

```
Input: population $\boldsymbol{P}$, cluster size $M$
1: Sort $\boldsymbol{P}$ according to fitness;
2: While $\boldsymbol{P}$ is not empty
3:     Select the best individual $\boldsymbol{P}_{best}$ in $\boldsymbol{P}$ as a new seed;
4:     Combine $M$-1 individuals nearest to $\boldsymbol{P}_{best}$ and $\boldsymbol{P}_{best}$ as a species;
5:     Eliminate these $M$ individuals from $\boldsymbol{P}$;
6: End While
Output: a set of species
```

After partition, each species is evolved individually using an optimizer. When incorporated into DE, the framework of speciation is presented in Algorithm 2.

Even though these two niching strategies have shown their efficiency and effectiveness on the tested problems in [29] and [30], respectively, they both encounter two dilemma. First, their performance is seriously dependent on the setting of their parameters, namely the crowding size (K) and the species radius (r). Second, they lose feasibility substantially in larger and more complex search spaces, such as the cases where masses of local optima exist. These two limitations restrict their wide application in practice.

To reduce the sensitivity of niching algorithms to parameters, certain topology-based niching methods [32], [35], [36] have been proposed. For example, hill-valley [50], [51] was developed to examine the landscape topography along the line segment connecting two sampled individuals. If there exists a third point along the line segment whose fitness value is lower than those of both individuals, a valley is said to be detected, separating the two individuals into two different niches. Recursive middling [35], [52] is another niching technique that adopts a similar mechanism but borrows the idea of binary search, which reduces the number of sampled points to a deterministic amount. Combining these two methods with a seed preservation strategy, topological species conservation [32], [33] was put forward to reduce extinction of species that have a diminishing number of individuals.

Although these techniques can self-adaptively determine niches, they cost an extra number of fitness evaluations. To combat this, Li and Tang [36] proposed a history-based topological speciation method to detect valleys without sampling extra points. It maintains a large archive to store all historical points for valley detection. However, some valleys may be missed, and this is more so during the first few generations as the number of historical points is initially small.

To improve partition, Gao *et al.* [37] and Qu *et al.* [38] have utilized clustering techniques to determine niches. Neighborhood-based CDE (NCDE) [38] generates offspring for each individual according to its M (termed as the neighborhood size) nearest neighbors. While neighborhood-based SDE (NSDE) [38] divides the whole population into species with an equal number of individuals, as in Algorithm 4. Self-adaptive clustering-based CDE (Self_CCDE) and SDE (Self_CSDE) proposed in [37] partition the whole population into crowds and species as presented in Algorithms 3 and 4, respectively. These tactics partition a population by introducing a less sensitive parameter (the neighborhood size or the cluster size) to improve the performance of CDE and SDE.

Subsequently, variants of conventional EAs have also been developed with niching to solve multimodal problems. For example, Li [42] proposed a variant of PSO (RPSO) using ring neighborhood topology for niching. It utilizes the particles' local memories to form a stable network and retain the best positions found so far. Qu *et al.* [41] put forward a distance-based locally informed PSO (LIPS), which uses several local best positions to guide the search of each particle instead of the

Algorithm 5 EDA

```
Input: population size $NP$, the number of selected individuals $K$
1: Randomly initialize the population;
2: While the termination criteria is not satisfied
3:     Select $K$ best individuals from the population;
4:     Estimate the probability distribution of the population
         according to the selected individuals;
5:     Sample new individuals according to the estimated
         distribution;
6:     Combine the sampled individuals and the old population
         to create a new population with $NP$ individuals;
7: End While
Output: the best individual and its fitness
```

global best position. LIPS can operate as a stable niching algorithm by using the information provided by particles' neighborhoods assessed by Euclidean distance. Biswas *et al.* [40] developed locally informative niching DE (LoINDE, containing LoICDE combined with CDE and LoISDE combined with SDE), which absorbs an improved information sharing mechanism among individuals for inducing efficient niching behavior. Further, using normalized search neighborhoods, they presented a parent-centric normalized mutation with proximity-based CDE (PNPCDE) [39]. It restricts randomness of members without inhibiting the explorative power, and facilitates in tracking and maintaining optima without loss of niches in multimodal basins [39].

So far, multimodal algorithms are all based on DE [38]–[40], GA [26], [28], [35], or PSO [41], [42], which exhibit various deficiencies, such as inefficiency in dealing with complex problems, where masses of local optima exist. Few attempts have been made to the development of EDAs for multimodal application, despite EDAs feature substantial exploration with high diversity at the population level [10], [46], [53] that would be beneficial for multimodal optimization.

B. Estimation of Distribution Algorithms

EDAs [11], [54] form a new family of EAs, which generate offspring according to a probability distribution, and have been intensively studied in the context of single optimization. A general framework of EDAs is outlined in Algorithm 5.

EDAs have achieved success in both combinatorial [55], [56] and continuous domains [10], [44], [57]–[60]. However, they are all essentially designed for single optimization, and few attempts have been made to locate multiple global optima for multimodal optimization. Although Yang *et al.* [45] proposed a novel maintaining and processing multiple submodels to enhance the ability of EDAs on multimodal problems, it is still designed for single optimization. In addition, despite niching-based EDA [46] was reported, it is also for single global optimization.

In this paper, taking the advantage of EDAs in preserving high diversity at the population level, we concentrate on developing two MEDAs for multimodal optimization.

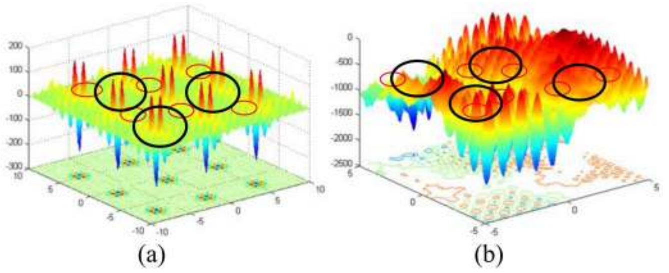

Fig. 1. Fitness landscapes of F_6 and F_{12} selected from CEC'2013 multimodal benchmark function set. (a) F_6. (b) F_{12}.

III. Multimodal EDAs

From Algorithm 5, we can see that without modification, an EDA is unsuitable for multimodal optimization. Take F_6 and F_{12} from the CEC'2013 benchmark multimodal function set [61] for example. Fig. 1 displays the fitness landscapes of these two functions, where a number of global maxima are distributed across various areas surrounded by multiple local maxima. In addition, some global optima are far away from one another, while some are very close to one another. In such an environment, an EDA is more likely to converge toward one or a small number of global optima, because it estimates the probability distribution of the whole population based on selected individuals.

Consequently, to cope with this situation, we develop an MEDA with niching for multimodal optimization. Specifically, MEDA augments EDAs with crowding outlined in Algorithm 3 and speciation in Algorithm 4, leading to two different versions of MEDA, named MCEDA and MSEDA, respectively. To afford a balance between exploration and exploitation, a dynamic cluster sizing strategy is absorbed in the niching methods for MEDAs. To further promote the diversity, both Gaussian and Cauchy distributions are alternatively utilized to generate offspring, taking the advantages of both distributions. Besides, to enhance solution accuracy, a local search algorithm outlined in Algorithm 6 is added to these MEDAs, resulting in LMCEDA and LMSEDA, respectively.

Having discussed the primary idea behind this paper, the concrete description of each algorithmic component will be elucidated in the following sections.

A. Dynamic Cluster Sizing

Without loss of generality, referring to F_6 in Fig. 1, for a given population size, a small cluster size leads to a large number of niches with narrow ranges of information (shown in red circles). This may be beneficial for exploitation, but may lead to low diversity of each niche, resulting in poor exploration and local traps due to such a narrow range of information. Conversely, if the cluster size is too large, a small number of niches covering wide areas (displayed in black circles) are obtained. This situation affords high diversity but leads to poor exploitation. The same dilemma can be seen for more complex functions, such as F_{12} shown in Fig. 1.

Thus, a balance between exploration and exploitation should be achieved. To this end, a dynamic cluster sizing strategy should be brought up. When a niche converges to one local area, the cluster size can be increased in the following generations to enhance the diversity of niches. In contrast, when more promising areas are discovered, the cluster size can be reduced to refine the search. However, without prior knowledge about the landscape of a problem and heuristic information on the number of global optima found during the evolution, it is hard to precisely determine a proper cluster size M in different evolution stages. To make a compromise, in this paper, a randomness-based dynamic sizing is utilized to realize the above idea for simplicity of dynamism.

In details, every generation, a number is chosen uniformly at random from a set $\boldsymbol{C}$ containing different integers as the cluster size M. With such a mechanism, the proposed methods are able to potentially balance the exploration and exploitation demands in spite of different features of different objective functions and evolution stages. This simple scheme is demonstrated to work well experimentally (see Section IV-B1). In this way, we can also reduce the sensitivity of choosing a suitable cluster size M in the niching methods used in our algorithms.

Through this sizing dynamism, MEDAs are less restricted by the maturing niches, hence providing diversity dynamism while balancing refinement and globalism without necessity to set sensitive parameters of the algorithms. The efficiency of this strategy will be fully tested in Section IV-B1.

B. Distribution Estimation and Offspring Generation

After partitioning the population into niches, MEDAs start to estimate the probability distribution of each niche. Suppose the population size is NP, and the selected cluster size is M. Then the total number of niches is $\lceil \mathrm{NP}/M \rceil$, denoted as s (when $\mathrm{NP}\%M \neq 0$, the last niche contains $\mathrm{NP}\%M$ individuals, where $\%$ stands for the modulo operation).

First, differing from an EDA [10], [44], [57]–[60] that estimates the probability distribution of the whole population using a number of selected individuals, MEDAs estimate the distribution at the niche level, i.e., MEDAs estimate each niche's distribution. Because a niche is considerably small, all individuals in each niche are potentially useful and thus all participate in the distribution estimation of that niche in MEDAs. Such mechanism may be helpful for diversity enhancement of each niche and thus potentially beneficial for finding more promising areas.

In the literature of EDAs, many distribution models are utilized, such as univariate Gaussian distribution [54], [62], multivariate Gaussian distribution [53], [63]–[65], and histogram model [10], [45]. Generally, these models can be applied to MEDAs, but in this paper, for brevity, we adopt univariate Gaussian distribution because it suffices with a low level of computational complexity [10], [53]. Therefore, the distribution estimation of each niche is computed as follows:

$$\mu_i^d = \frac{1}{M}\sum_{j=1}^{M} X_j^d$$
$$\delta_i^d = \sqrt{\frac{1}{M-1}\sum_{j=1}^{M}\left(X_j^d - \mu_i^d\right)^2} \tag{3}$$

where $\boldsymbol{\mu_i} = [\mu_i^1, \ldots, \mu_i^d, \ldots, \mu_i^D]$ and $\boldsymbol{\delta_i} = [\delta_i^1, \ldots, \delta_i^d, \ldots, \delta_i^D](1 = i = s)$ are, respectively, the mean and standard deviation (std) vectors of the ith niche, $\boldsymbol{X_j} = [X_j^1, \ldots, X_j^d, \ldots, X_j^D]$ is the jth individual in the ith niche, and D is the dimension size of the multimodal problem.

The next step is the generation of offspring. Most variants of EDAs only adopt Gaussian distribution to sample points [10], [44], [57]–[60]. However, Gaussian distribution generally has a narrow sample space, especially when the standard deviation δ is small, which would limit the exploration ability of an EDA. To improve this situation, we turn to Cauchy distribution [66], which is similar to Gaussian distribution, but has a long fat tail, leading to a wide sample space. Equipped with this distribution, an EDA is potentially more capable of escaping from local areas.

Overall, we find that Gaussian distribution is more suitable for the exploitation stage owing to its narrow sampling range, while Cauchy distribution is fitter for the exploration stage because of its wide sampling space [66], [67]. This motivates us to alternatively use these two distributions to generate offspring. Since MEDAs are based on niches, it is obvious that these two distributions should be operated at the niche level, namely each niche should randomly select one of these two distributions to generate offspring. For simplicity, we assign these two distributions with the same probability for each niche. In a word, the offspring generation of each niche is as follows:

$$\begin{cases} \boldsymbol{C_i} = \mathrm{Gaussian}(\boldsymbol{\mu_i}, \boldsymbol{\delta_i}) & \text{if rand()} < 0.5 \\ \boldsymbol{C_i} = \mathrm{Cauchy}(\boldsymbol{\mu_i}, \boldsymbol{\delta_i}) & \text{otherwise} \end{cases} \tag{4}$$

where $\boldsymbol{C_i}$ is the M offspring of the ith niche, and rand() is a uniformly random number generator that generates numbers within [0, 1]. Implicitly, such alternative usage can potentially offer a balance between exploration and exploitation for MEDAs based on the features of the two distributions.

After the generation of offspring, it comes to the selection of promising individuals. For MCEDA, we adopt the selection procedure in CDE [29], namely the offspring will replace its nearest parent in the current population if it is better. While for MSEDA, unlike the selection process in SDE [30], we also adopt the one in CDE [29], but it is operated within niches, namely, the offspring of each niche will replace its most similar parent in its niche if the offspring is better. Different selection procedures are adopted here on account of different niching strategies used for partitioning the population into niches.

The cooperation between the distribution selection operated at the niche level and the individual selection is likely to arm MEDAs with competitive exploration and exploitation abilities. Experiments in Section IV-B will demonstrate the superiority of employing both Cauchy and Gaussian distributions for sampling over using only one of these two distributions.

C. Local Search

Usually an EDA is prone to difficulty in improving the accuracy of solutions because of its unsubtle sampling strategy. This can be improved through local search strategies [10].

Likewise, in this paper, we propose a new local search strategy based on Gaussian distribution for MEDAs, leading to LMEDAs (containing LMCEDA and LMSEDA). Gaussian distribution is utilized here because it owns a narrow sampling space, especially when the standard deviation is small, which is beneficial for local search. Thus, this inspires us to set a small local standard deviation σ for the local search. In this paper, $\sigma = 1.0\text{E-}4$ is utilized.

In general, more promising solutions can be found around the best individuals in the current generation. For MEDAs, to avoid performing local search in the same area, we conduct local search at the niching level. Consequently, local search is executed only around the seed of each niche, which is defined as the best individual in a niche.

Suppose the seeds of all niches are stored in $\boldsymbol{S}$, containing $s = \lceil \text{NP}/M \rceil$ seeds. Owing to the sampling strategy used in the local search, it is evident that enough points should be sampled around each seed so that better solutions could be obtained. However, the number of points (denoted as N) sampled by the local search should not be too large, because it would waste fitness evaluations if there is no improvement around some seeds. Besides, it should not be too small either, because it cannot afford the improvement around the seeds if only few points are sampled. In preliminary experiments, we find $N = 5$ is enough to compromise the above concerns.

As a consequence, the local search is performed as follows:

$$\boldsymbol{LC}_{\mathbf{i}} = \text{Gaussian}(\boldsymbol{S}_{\mathbf{i}}, \sigma) \tag{5}$$

where $\boldsymbol{LC_i}$ is the N individuals of the ith niche generated by the local search and $\boldsymbol{S_i}$ is the seed of the ith niche. Since the sampled points $\boldsymbol{LC_i}$ are only produced around the seed of the ith niche, it is reasonable that we should use the sampled points to only replace the seed if they are better. That is, the whole procedure of local search is only related to the seeds of niches.

The purpose of local search is to improve the accuracy of promising solutions, so that the gap between the global optima and the obtained solutions can be narrowed. This indicates that it would be useless to do local search on the niches, which fall into local areas. Inspired by this, we consider conducting local search with probability.

Apparently, the better the seed is, the greater chance it has to do local search. This suggests that the probability of carrying out local search on a niche should be proportional to the fitness of its seed. Thus the following formula can be obtained:

$$\text{Pr}_i = \frac{F_i}{F_{\max}} \tag{6}$$

where Pr_i is the probability for the ith niche to do local search, F_i is the seed fitness of the ith niche, and $F_{\max}$ is the maximum fitness in $\boldsymbol{F}$, which contains the fitness of all seeds in $\boldsymbol{S}$.

To cover functions that have negative or zero fitness values, (6) is extended to

$$\text{Pr}_i = \frac{F_i + |F_{\min}| + \xi}{F_{\max} + |F_{\min}| + \xi} \tag{7}$$

where $F_{\min}$ is the minimal fitness value in $\boldsymbol{F}$ and ξ is a small positive value, which is used to accommodate the case where $F_{\min} = F_{\max} = 0$.

Algorithm 6 Local Search

Input: seeds set $\boldsymbol{S}$, the number of seeds s, fitness of these seeds $\boldsymbol{F}$, local std value σ, the number of sampled individuals N

```
1: $F_{min}$ = min($\boldsymbol{F}$), $F_{max}$ = max($\boldsymbol{F}$), flag = false;
2: If $F_{min} \leq 0$
3:     $F_{max} = F_{max} + |F_{min}| + \xi$;
4:     flag = true;
5: End If
//Calculate the probability for each seed to perform local search
6: For $i = 1{:}s$
7:     If flag
8:         $Pr[i] = (\boldsymbol{F}[i] + |F_{min}| + \xi) / F_{max}$;
9:     else
10:        $Pr[i] = \boldsymbol{F}[i]/F_{max}$;
11:    End If
12: End For
13: For $i = 1{:}s$
14:     If $rand(\,) \leq Pr[i]$
15:         For $j = 1{:}N$
16:             Generate a new individual $\boldsymbol{LC_j}$ using Gaussian($\boldsymbol{S}[i]$, $\sigma$);
17:             If $\boldsymbol{LC_j}$ is better than $\boldsymbol{S}[i]$
18:                 Replace $\boldsymbol{S}[i]$ with $\boldsymbol{LC_j}$;
19:             End If
20:         End For
21:     End If
22: End For
```

Output: Seeds $\boldsymbol{S}$ and their fitness $\boldsymbol{F}$

From (6) and (7), we can observe that all seeds with equal fitness values have the same probability to do local search. In addition, local search is always performed for the best individual. Besides, the worse the fitness of the seed, the lower the probability it would have to do local search and vice versa.

Overall, the procedure of the local search is summarized in Algorithm 6, and the complete procedures of LMCEDA and LMSEDA are outlined in Algorithms 7 and 8, respectively.

D. Complexity Analysis

Given that the population size is NP, the dimension size is D and the cluster size is M, LMCEDA and LMSEDA have the same complexity with $O(\text{NP} \times D)$ as previous EDAs in the procedures of distribution estimation and offspring generation, even though there are differences in these procedures. Comparing Algorithms 7 and 8 with Algorithm 5, we can see that compared with previous EDAs, the difference in complexity for LMCEDA and LMSEDA mainly lies in the procedures of niche generation (line 3), individual selection (lines 11–14), and the local search (line 15) in Algorithms 7 and 8, respectively. For niching generation, LMCEDA needs $O(D)$(line 1) + $(O(\text{NP} \times D) + O(\text{NP}^2 \times D))$ (lines 2–6) in Algorithm 3, while LMSEDA needs $O(\text{NP} \times \log(\text{NP}))$(line 1) + $O(\text{NP}^2 \times D)$(lines 2–6) in Algorithm 4. In terms of individual selection, LMCEDA costs $O(\text{NP}^2 \times D)$

Algorithm 7 Local Search-Based MCEDA (LMCEDA)

```
Input: population size $NP$, cluster size set $\boldsymbol{C}$, local search std $\sigma$
1: Randomly initialize the population;
2: Randomly select a number from $\boldsymbol{C}$ as the cluster size $M$;
3: Using Algorithm 3 to partition the population into crowds;
4:    For each crowd
5:        Estimate the probability distribution (mean $\boldsymbol{\mu}$ and
          std $\boldsymbol{\delta}$) of this crowd using (3);
6:        If $rand(\ )\leq 0.5$
7:            Using Cauchy($\boldsymbol{\mu}$,$\boldsymbol{\delta}$) to generate $M$ new individuals;
8:        else
9:            Using Gaussian($\boldsymbol{\mu}$,$\boldsymbol{\delta}$) to generate $M$ new
              individuals;
10:       End If
11: End For
12: For each new individual $\boldsymbol{c_i}$
13:       Compare the fitness of $\boldsymbol{c_i}$ with that of the most similar
          individual in the current population and replace it if
          $\boldsymbol{c_i}$ is better;
14: End For
15: Perform local search according to Algorithm 6;
16: Stop if the termination criterion is met. Otherwise go to
    Step 2;
Output: the whole population
```

Algorithm 8 Local Search-Based MSEDA (LMSEDA)

```
Input: population size $NP$, cluster size set $\boldsymbol{C}$, local search std $\sigma$
1: Randomly initialize the population;
2: Randomly select a number from $\boldsymbol{C}$ as the cluster size $M$;
3: Using Algorithm 4 to partition the population into species;
4:  For each species
5:    Estimate the probability distribution (mean $\boldsymbol{\mu}$ and std $\boldsymbol{\delta}$)
      of this species;
6:    If $rand(\ )\leq 0.5$
7:        Using Cauchy($\boldsymbol{\mu}$,$\boldsymbol{\delta}$) to generate $M$ new individuals;
8:    else
9:        Using Gaussian($\boldsymbol{\mu}$,$\boldsymbol{\delta}$) to generate $M$ new individuals;
10:   End If
11:   For each new individual $\boldsymbol{c_i}$ in the species
12:       Compare the fitness of $\boldsymbol{c_i}$ with that of the most
          similar individual in current species and replace it
          if $\boldsymbol{c_i}$ is better;
13:   End For
14: End For
15: Perform local search according to Algorithm 6;
16: Stop if the termination criterion is met. Otherwise go to
    Step 2;
Output: the whole population
```

(lines 12–14) in Algorithm 7, while LMSEDA takes $O(M \times \text{NP} \times D)$ (lines 11–14) in Algorithm 8. Additionally, the complexity of previous EDAs in individual selection usually is $O(2\text{NP} \times \log(2\text{NP}))$ [10]. As for the local search, suppose the number of points sampled in the local search is N, then the complexity is $O(\text{NP}/M \times N \times D)$ for both algorithms. Note that NP/M and N are usually much smaller than NP.

In total, the overall complexities of both LMCEDA and LMSEDA are $O(\text{NP}^2 \times D)$. Comparing to the complexity of classic EDAs with $O(\text{NP} \times D)$, we can see that the complexities of the proposed MEDAs and classic EDAs both are linearly proportional to the dimension size D. The only difference is that the increment speed for MEDAs is NP^2, which is caused by the niching generation, which is unavoidable for niching-based multimodal algorithms [37], [38], [40], [45].

In summary, with the dynamic cluster sizing strategy, the alternative usage of Gaussian and Cauchy distributions and the local search, the proposed MEDAs are capable of solving multimodal problems with little extra computational burden, which will be substantiated in the following section.

IV. Experiment Studies

In this section, a series of experiments are carried out to verify the feasibility and efficiency of MEDAs developed in this paper. First, Section IV-A describes the benchmark function set and evaluation protocols we adopt in the experiments. Then, the influence of each component in MEDAs is observed in Section IV-B, where the comparison between LMCEDA and LMSEDA is also conducted. Section IV-C will present the detailed comparison results between LMEDAs and several state-of-the-art multimodal algorithms with the results attached to the supplemental material. In addition, to better understand the abbreviations of algorithms used in the experiments, we list the names of all algorithms in details in Table SI, which is left in the supplemental material for the consideration of saving space.

A. Benchmark Functions and Evaluation Protocols

To demonstrate the effectiveness of LMEDAs, we conduct experiments on a widely used benchmark function set—the CEC'2013 multimodal function set [61] containing 20 functions, which are designed for the 2013 IEEE CEC Special Session on Niching Methods for Multimodal Optimization.[1] To save space, the main characteristics of these functions are summarized in Table SII in the supplemental material. For more details, readers can refer to [61].

To evaluate the performance of LMEDAs and to make fair comparisons with the state-of-the-art multimodal algorithms, peak ratio (PR), success rate (SR), and convergence speed (CS) are selected as the evaluation protocols, which are also adopted in the corresponding competition on niching methods for multimodal optimization at the special session.

Given a fixed maximum number of function evaluations (denoted as Max_Fes) and a specific accuracy level ε, PR is defined as the percentage of the number of the global optima found out of the total number of global optima averaged over multiple runs. SR measures the ratio of successful runs out of all runs and a successful run is defined as a run where all known global optima are found. CS is computed by counting the number of function evaluations (denoted as Fes)

[1]http://goanna.cs.rmit.edu.au/~xiaodong/cec13-niching/

TABLE I
PARAMETER SETTINGS

Function	Max_Fes	Population Size
F_1-F_5	5.0E+4	80
F_6	2.0E+5	100
F_7	2.0E+5	300
F_8-F_9	4.0E+5	300
F_{10}	2.0E+5	100
F_{11}-F_{13}	2.0E+5	200
F_{14}-F_{20}	4.0E+5	200

required to locate all known global optima at a specific accuracy level ε. If one algorithm cannot locate all global optima when the maximum number of fitness evaluations is exhausted, then Max_Fes is used when calculating CS [61].

The formula to compute these three protocols can be found in [61], which also contains the method to compute the number of global optima found in the current population.

In the experiments, five different accuracy levels, namely: $\varepsilon = 1.0\text{E-}01$, $\varepsilon = 1.0\text{ E-}02$, $\varepsilon = 1.0\text{E-}03$, $\varepsilon = 1.0\text{E-}04$, and $\varepsilon = 1.0\text{E-}05$, are adopted. However, for saving space, in this paper, unless otherwise stated, we mainly report the results at $\varepsilon = 1.0\text{E-}04$, which is common in [35], [37], [38], [42], [47], and [48].

To make fair comparisons, the maximum number of fitness evaluations (Max_Fes) and the population size (NP) are set to the same for all compared multimodal methods as shown in Table I. Further, all experiments are carried out for 51 independent runs for statistics.

Additionally, it is worth mentioning that all experiments are conducted on a PC with 4 Intel Core i5-3470 3.20 GHz CPUs, 4 GB memory and the Ubuntu 12.04 LTS 64-bit system.

B. Observations of LMEDAs

Before investigating the influence of the three techniques adopted in LMEDAs, namely the dynamic cluster sizing, the alternative usage of Gaussian and Cauchy distributions, and the local search, we should announce the setting for the cluster size set $\boldsymbol{C}$ introduced in LMEDAs. It should be noticed that the cluster size should be neither too large nor too small, since a too large cluster size would lead to too few niches, bringing in too wide area one niche covers and a too small cluster size would result in too many niches, making many niches possibly located at local areas. In this paper, for simplicity, $\boldsymbol{C}$ is set as a range of integers varying from 2 to 10, namely $\boldsymbol{C} = [2, 10]$. Actually, in preliminary experiments, we find our algorithms are not sensitive to $\boldsymbol{C}$ if we keep $\boldsymbol{C}$ in a wide range.

1) Influence of Dynamic Cluster Sizing: To testify the usefulness of the dynamic cluster sizing strategy, we conduct comparison experiments between LMEDAs with the dynamic cluster sizing and the ones with different fixed cluster sizes. For these fixed sizes, to make it simple, we set them as $\{2, 4, 5, 6, 8, 10\}$, which is mainly for exact division. Table II exhibits the comparison results between different versions of LMCEDA (the left part of the bolded line) and LMSEDA (the right part of the bolded line) with respect to PR at the accuracy level $\varepsilon = 1.0\text{E-}04$ and the best PR is highlighted in bold. The numbers under unit "LMCEDA" or "LMSEDA" denote the fixed cluster sizes, and "dynamic" means LMCEDA or LMSEDA adopts the dynamic cluster sizing strategy.

Observing Table II, we can draw the following conclusions.

1) First, on F_1–F_5 and F_{13}–F_{14}, different cluster sizes make no difference on both LMCEDA and LMSEDA. However, on F_6–F_{12}, a small cluster size is preferred for both algorithms, and on F_{15}–F_{20}, neither a small nor a large cluster size is attractive, and only an appropriate cluster size is beneficial. This observation demonstrates that the optimal cluster size of different problems is not the same and such a size for different problems needs to be fine-tuned.
2) LMCEDA with the dynamic cluster sizing can achieve comparable performance on most problems (17 functions) when compared with the one with the optimal cluster size. The PR results obtained by the dynamic version on these functions are very close to those achieved by the one with the optimal cluster size. Specially, the dynamic version even obtains the best PR result on F_{17} and only loses the competition on F_8 and F_{20}.
3) LMSEDA with the dynamic cluster sizing shows its superiority to the one with the optimal cluster size. It achieves considerably similar PR results on 16 functions and obtains the best PR results on F_6, F_{15}, F_{17}, and F_{19} in comparison with the version with the fixed optimal size.

In summary, the dynamic cluster sizing strategy is helpful for both LMCEDA and LMSEDA, and this benefit is more evident for LMSEDA. Such benefit comes from the potential balance between exploration and exploitation resulted from the changing cluster size. Additionally, the dynamic strategy on the cluster size also eliminates the sensitivity to this parameter and relieves users from the task of fine tuning.

2) Effects of Local Search and Combination of Distributions: Here, we take a close observation at the influence of the local search scheme and the combination of Gaussian and Cauchy distributions. In the experiments, we denote LMEDAs only with Gaussian distribution as LMEDA_Gs containing LMCEDA_G and LMSEDA_G. Likewise, LMEDAs only with Cauchy distribution are denoted as LMEDA_Cs, including LMCEDA_C and LMSEDA_C. The proposed LMEDAs without local search are represented as MEDAs (MCEDA and MSEDA). The version with both techniques is still be LMEDAs (LMCEDA and LMSEDA). Table III presents the comparison results among these versions with respect to PR at accuracy level $\varepsilon = 1.0\text{E-}04$ with the left part of the bolded line related to LMCEDA and the right part associated with LMSEDA. The first two columns of each part, namely columns "G" and "C," represent the results of LMCEDA_G (LMSEDA_G) and LMCEDA_C (LMSEDA_C), respectively. Additionally, the best PR results are highlighted in bold.

First, in terms of the combination of the two distributions, comparing LMCEDA with LMCEDA_G and LMCEDA_C, we can see LMCEDA performs similarly to both LMCEDA_G and LMCEDA_C on 12 functions (F_1–F_6, F_{10}–F_{14}, and F_{16}). LMCEDA defeats LMCEDA_G and LMCEDA_C on 3 (F_7, F_{18}, and F_{19}) and 8 (F_7–F_9, F_{15},

TABLE II
COMPARISON RESULTS ABOUT PR BETWEEN LMEDAS WITH DYNAMIC CLUSTER SIZE AND THOSE WITH FIXED CLUSTER SIZES AT ACCURACY LEVEL $\varepsilon = 1.0\text{E-}04$. THE BEST PR IS HIGHLIGHTED IN BOLD

	1.0E-04													
F	LMCEDA							LMSEDA						
	2	4	5	6	8	10	Dynamic	2	4	5	6	8	10	Dynamic
F_1	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
F_2	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
F_3	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
F_4	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
F_5	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
F_6	**0.995**	0.972	0.946	0.911	0.810	0.696	0.990	0.962	0.912	0.914	0.857	0.751	0.638	**0.972**
F_7	**0.755**	0.670	0.641	0.626	0.604	0.558	0.734	**0.698**	0.606	0.575	0.551	0.524	0.488	0.673
F_8	**0.633**	0.382	0.336	0.315	0.279	0.223	0.347	**0.682**	0.563	0.469	0.403	0.323	0.278	0.613
F_9	**0.321**	0.246	0.219	0.203	0.175	0.148	0.284	**0.269**	0.209	0.188	0.170	0.144	0.126	0.248
F_{10}	**1.000**	**1.000**	**1.000**	0.998	0.992	0.992	**1.000**	**0.998**	0.997	0.997	0.993	0.975	0.873	**0.998**
F_{11}	**0.683**	0.667	0.667	0.667	0.667	0.667	0.667	0.882	**0.935**	0.859	0.778	0.670	0.667	0.892
F_{12}	**0.782**	0.748	0.750	0.748	0.748	0.738	0.750	0.929	0.988	**0.993**	0.983	0.885	0.789	0.990
F_{13}	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**
F_{14}	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**
F_{15}	0.561	0.681	0.708	**0.716**	0.691	0.681	0.696	0.625	0.725	0.730	0.733	0.662	0.620	**0.738**
F_{16}	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	0.621	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**
F_{17}	0.424	0.436	0.387	0.365	0.304	0.275	**0.456**	0.400	0.591	0.556	0.544	0.463	0.390	**0.620**
F_{18}	0.480	0.663	**0.667**	0.663	0.647	0.601	0.657	0.000	0.657	0.657	**0.660**	0.650	0.641	**0.660**
F_{19}	0.304	**0.485**	0.463	0.456	0.387	0.277	0.451	0.105	0.449	0.453	0.441	0.404	0.331	**0.458**
F_{20}	0.000	**0.238**	0.169	0.083	0.064	0.042	0.059	0.000	0.248	**0.250**	**0.250**	**0.250**	**0.250**	0.248

TABLE III
COMPARISON RESULTS IN PR AMONG DIFFERENT VERSIONS OF MEDAS AT ACCURACY LEVEL $\varepsilon = 1.0\text{E-}04$ WITH BEST PR BOLDED

	1.0E-04							
F	LMCEDA		MCEDA	LMCEDA	LMSEDA		MSEDA	LMSEDA
	G	C			G	C		
F_1	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
F_2	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
F_3	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
F_4	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
F_5	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
F_6	**0.991**	0.989	0.008	0.990	0.963	0.969	0.916	**0.972**
F_7	0.725	0.717	0.690	**0.734**	0.641	**0.686**	0.633	0.673
F_8	**0.386**	0.294	0.000	0.347	**0.640**	0.509	0.267	0.613
F_9	**0.289**	0.256	0.270	0.284	0.241	0.244	0.230	**0.248**
F_{10}	**1.000**	**1.000**	**1.000**	**1.000**	0.997	**1.000**	0.995	0.998
F_{11}	0.667	0.667	**0.676**	0.667	**1.000**	0.771	0.935	0.892
F_{12}	**0.750**	**0.750**	0.319	**0.750**	0.934	0.985	0.924	**0.990**
F_{13}	**0.667**	**0.667**	**0.667**	**0.667**	**0.670**	0.667	0.667	0.667
F_{14}	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**
F_{15}	**0.706**	0.642	0.375	0.696	**0.748**	0.664	0.652	0.738
F_{16}	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**
F_{17}	**0.485**	0.316	0.382	0.456	**0.625**	0.505	0.525	0.620
F_{18}	0.621	0.000	**0.667**	0.657	0.582	0.637	0.657	**0.660**
F_{19}	0.439	0.000	**0.473**	0.451	0.424	0.319	0.446	**0.458**
F_{20}	**0.252**	0.000	0.250	0.059	0.248	0.000	**0.250**	0.248

and F_{17}–F_{20}) functions, respectively. Besides, there is no loss of competition for LMCEDA in comparison with LMCEDA_C, and LMCEDA only seriously loses its advantage on F_{20} when compared with LMCEDA_G. When it comes to LMSEDA, observing the right part, we find the combination of the two distributions is very beneficial, which helps LMSEDA defeat LMSEDA_G and LMSEDA_C on six functions (F_6, F_7, F_9, F_{12}, F_{18}, and F_{19}) and nine functions (F_6, F_8, F_9, F_{11}, F_{15}, and F_{17}–F_{20}), respectively. And there is no serious loss of advantage for LMSEDA when compared with the two versions. Comprehensively, we can see the combination of the two distributions provides potential help for both LMCEDA and LMSEDA, and the benefit is more obvious on LMSEDA. Such benefit origins from the potential balance between exploration and exploitation afforded by the alternative usage of the two distributions, which takes advantages of both distributions.

Second, from the perspective of the local search, comparing LMCEDA with MCEDA and LMSEDA with MSEDA, respectively, which corresponds to the last two columns of the two parts, we can find that the local search is very useful for LMCEDA on F_6, F_8, F_{12}, and F_{15}, while it benefits LMSEDA specially on F_6, F_8, F_{12}, F_{15}, and F_{17}. On other functions, even though such benefit is not evident, the performance of LMCEDA and LMSEDA is comparable to that of MCEDA and MSEDA, or even a little better, such as on F_7, F_9, and F_{17} for LMCEDA and on F_7, F_9, F_{10}, F_{18}, and F_{19} for LMSEDA. As a whole, we can see that the local search scheme makes significant difference on both LMCEDA and LMSEDA.

In short, we can conclude that both the combination of the two distributions and the local search are beneficial for LMEDAs in locating more global optima.

3) Overall Performance of LMCEDA and LMSEDA: After observing the influence of the three techniques on LMCEDA and LMSEDA from the above two series of experiments, we present the overall performance of these two algorithms at all accuracy levels. Table IV shows the comparison results between LMCEDA and LMSEDA with respect to PR and SR on all 20 functions at all accuracy levels and the best PR results are emphasized in bold.

From Table IV, we can observe the following.

1) Both algorithms achieve almost the same performance on nine functions (F_1–F_5, F_{10}, F_{13}, F_{14}, and F_{16}) at all five accuracy levels. Besides, on F_1–F_5, they can locate all known global optima successfully at any accuracy level, and on F_{10}, LMCEDA can successfully find all global optima at all accuracy levels, while LMSEDA can nearly successfully locate all global optima with only one run missing finding only one global optima out of 51 independent runs. On F_{13}, F_{14}, and F_{16}, both LMCEDA and LMSEDA can almost successfully find all global optima at accuracy level $\varepsilon = 1.0\text{E-}01$, while at other levels, even though they cannot find all optima in any run, they are able to find most of the global optima (4 out of 6).
2) Compared with LMSEDA, overall, LMCEDA is slightly better on F_6, and shows its great superiority to

TABLE IV
COMPARISON RESULTS IN PR AND SR BETWEEN LMCEDA AND LMSEDA ON 20 FUNCTIONS AT DIFFERENT ACCURACY LEVELS

ε	F_1				F_2				F_3				F_4				F_5			
	LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
1.0E-02	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
1.0E-03	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
1.0E-04	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
1.0E-05	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
ε	F_6				F_7				F_8				F_9				F_{10}			
	LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	**0.998**	0.961	0.975	0.608	**1.000**	1.000	**1.000**	1.000	0.359	0.000	**0.638**	0.000	**0.424**	0.000	0.344	0.000	**1.000**	1.000	0.998	0.980
1.0E-02	**0.995**	0.902	0.973	0.588	**0.848**	0.000	0.753	0.000	0.354	0.000	**0.627**	0.000	**0.401**	0.000	0.328	0.000	**1.000**	1.000	0.998	0.980
1.0E-03	**0.990**	0.843	0.973	0.588	**0.782**	0.000	0.712	0.000	0.352	0.000	**0.622**	0.000	**0.333**	0.000	0.281	0.000	**1.000**	1.000	0.998	0.980
1.0E-04	**0.990**	0.843	0.972	0.588	**0.734**	0.000	0.673	0.000	0.347	0.000	**0.613**	0.000	**0.284**	0.000	0.248	0.000	**1.000**	1.000	0.998	0.980
1.0E-05	**0.990**	0.843	0.972	0.588	**0.710**	0.000	0.658	0.000	0.293	0.000	**0.556**	0.000	**0.256**	0.000	0.228	0.000	**1.000**	1.000	0.998	0.980
ε	F_{11}				F_{12}				F_{13}				F_{14}				F_{15}			
	LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	**1.000**	1.000	**1.000**	1.000	0.919	0.471	**0.990**	0.922	**1.000**	1.000	0.980	0.941	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	0.995	0.980
1.0E-02	0.667	0.000	**0.944**	0.667	0.755	0.000	**0.990**	0.922	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	0.699	0.000	**0.738**	0.000
1.0E-03	0.667	0.000	**0.905**	0.451	0.750	0.000	**0.990**	0.922	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	0.699	0.000	**0.738**	0.000
1.0E-04	0.667	0.000	**0.892**	0.392	0.750	0.000	**0.990**	0.922	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	0.696	0.000	**0.738**	0.000
1.0E-05	0.667	0.000	**0.879**	0.333	0.745	0.000	**0.988**	0.902	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	0.686	0.000	**0.735**	0.000
ε	F_{16}				F_{17}				F_{18}				F_{19}				F_{20}			
	LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA		LMCEDA		LMSEDA	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**0.806**	0.667	0.770	0.569	**1.000**	1.000	**1.000**	1.000
1.0E-02	**0.667**	0.000	**0.667**	0.000	0.458	0.000	**0.620**	0.000	0.657	0.000	**0.660**	0.000	0.451	0.000	**0.461**	0.000	**0.250**	0.000	**0.250**	0.000
1.0E-03	**0.667**	0.000	**0.667**	0.000	0.456	0.000	**0.620**	0.000	0.657	0.000	**0.660**	0.000	0.451	0.000	**0.458**	0.000	**0.250**	0.000	**0.250**	0.000
1.0E-04	**0.667**	0.000	**0.667**	0.000	0.456	0.000	**0.620**	0.000	0.657	0.000	**0.660**	0.000	0.451	0.000	**0.458**	0.000	0.059	0.000	**0.248**	0.000
1.0E-05	**0.667**	0.000	**0.667**	0.000	0.417	0.000	**0.576**	0.000	**0.657**	0.000	**0.657**	0.000	0.439	0.000	**0.458**	0.000	0.000	0.000	**0.248**	0.000

LMSEDA on F_7 and F_9. Conversely, LMSEDA displays its little superiority on F_{18} and F_{19}, and wins the competition with great advantages on 6 functions (F_8, F_{11}, F_{12}, F_{15}, F_{17}, and F_{20}).

3) From the perspective of accuracy levels, both LMCEDA and LMSEDA can almost successfully locate all global optima for almost all functions at the accuracy level ε = 1.0E-01, except for F_8, F_9, F_{12}, and F_{19} for LMCEDA and F_6, F_8, F_9, and F_{19} for LMSEDA. As for other accuracy levels, LMCEDA cannot achieve a successful run on 13 functions (F_7–F_9 and F_{11}–F_{20}) and LMSEDA cannot obtain any successful run on 11 functions (F_7–F_9 and F_{13}–F_{20}) owing to the influence of an ocean of local optima. Such phenomena is common for the state-of-the-art multimodal algorithms, but ours can locate more global optima in the unsuccessful runs, which will be testified in the following section.

In addition, to have a better view of the comparison between LMCEDA and LMSEDA, we visualize the final fitness landscape of the final population when the maximum number of fitness evaluations is exhausted on ten visual functions, which is shown in Fig. 2.

First, we observe that both LMCEDA and LMSEDA can locate the global optima. And when there is not so many or even no local optima, as shown in Fig. 2(b)–(d), (f), and (g), LMCEDA and LMSEDA achieve very similar performance, namely the individuals are located around or at the global optima. However, when there are many or even masses of local optima, the two algorithms perform very differently. LMCEDA not only can locate global optima, but also can locate many local optima, while LMSEDA can locate more global optima but at the loss of locating local optima, which can be clearly seen in Fig. 2(e) and (h)–(j). Such phenomena may be attributed to the different individual selection procedures adopted in LMCEDA and LMSEDA as shown in Algorithms 7 and 8, respectively.

In summary, we can conclude that from the perspective of locating more global optima as concerned in this paper, LMSEDA is a little superior to LMCEDA. But, from Table IV, we can see that both algorithms are promising for solving multimodal optimization problems.

C. Comparisons With State-of-the-Art Algorithms

To further demonstrate the superiority of LMEDAs (LMCEDA and LMSEDA), we conduct comparison experiments between LMEDAs and several state-of-the-art multimodal algorithms. The compared algorithms are CDE [29], SDE [30], LIPS [41], R2PSO, R3PSO [42], Self_CCDE, Self_CSDE [37], NCDE, NSDE [38], LoICDE, LoISDE [40], and PNPCDE [39]. The brief description of these algorithms can be found in Section II-A. To make fair comparisons, the population size and the maximum number of fitness evaluations are set the same for all algorithms according to Table I. Other parameters introduced in the corresponding algorithms are set as recommended in the related papers.

For saving space, we leave all the comparison results at the five accuracy levels in the supplemental material. Tables SIII–SVII (in the supplemental material) show the comparison results with respect to PR, SR, and CS of different multimodal algorithms at the five accuracy levels, and the best PRs are highlighted in bold. The row named "bprs" counts the number of functions where one algorithm obtains the best PR results, namely the number of bolded PRs. Tables SVIII–SXII (in the supplemental material) present nonparametric Wilcoxon rank-sum test[2] results with respect to PR between LMEDAs and the state-of-the-art methods

[2]http://en.wikipedia.org/wiki/Mann–Whitney_U_test

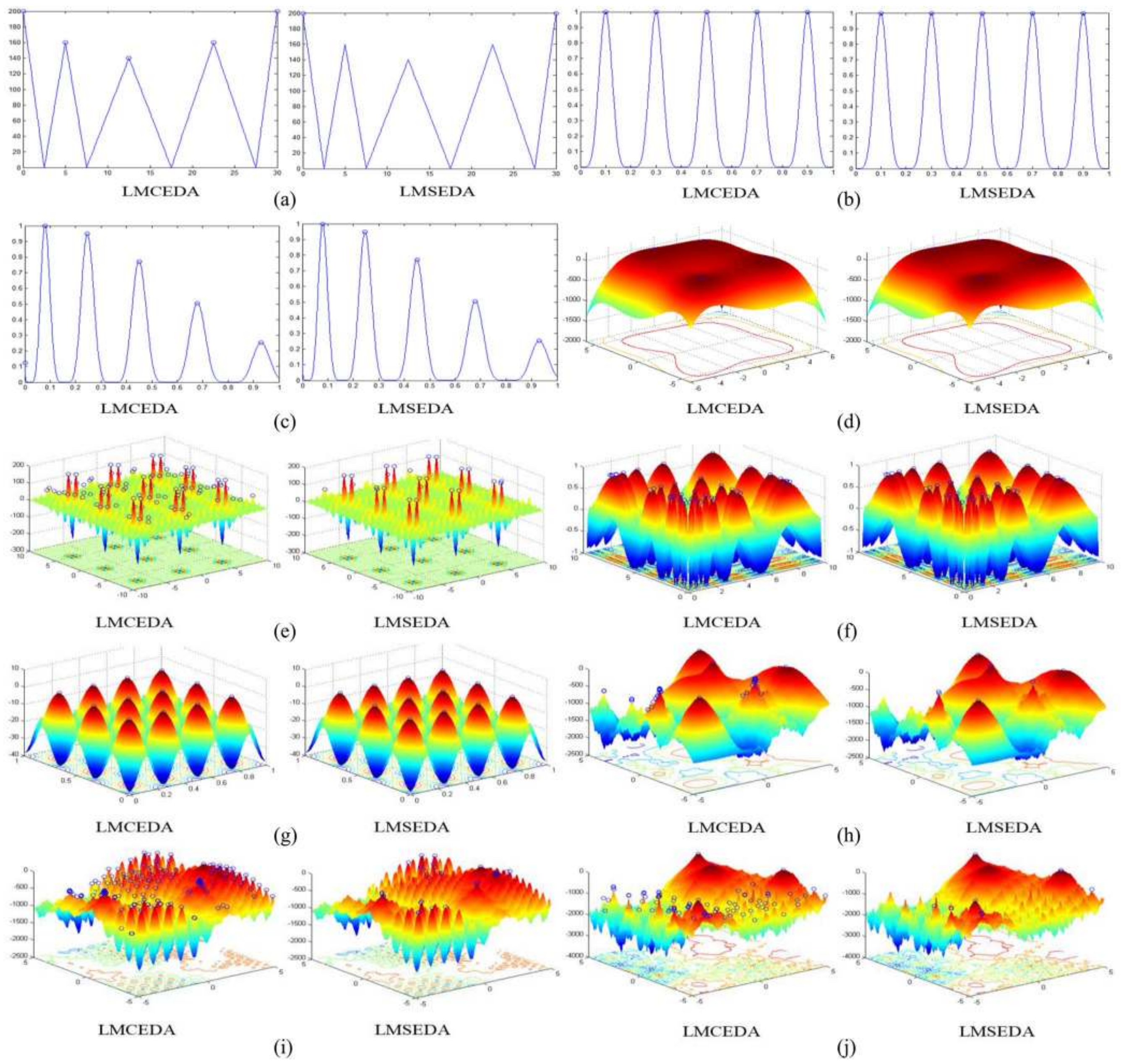

Fig. 2. Comparison results in final fitness landscapes between LMCEDA and LMSEDA on ten functions that can be visualized. (a) F_1. (b) F_2. (c) F_3. (d) F_4. (e) F_6. (f) F_7. (g) F_{10}. (h) F_{11}. (i) F_{12}. (j) F_{13}.

at the five accuracy levels. In these tables, each compared algorithm is associated with two columns, of which the left are the results compared with LMCEDA and the right are the results compared with LMSEDA. In addition, the critical value of Wilcoxon rank-sum test with respect to the rank sum for 51 samples is 2866. Therefore, the number larger than 2866 indicates that our algorithm is significantly better than the compared algorithm, and the number smaller than 2387 indicates our algorithm is significantly worse. Other cases mean our algorithm is equivalent to the compared algorithm. According to this standard, the grayed units in Tables SVIII–SXII (in the supplemental material) mean LMCEDA or LMSEDA is significantly better than the compared algorithm, while the bolded values indicate that LMCEDA or LMSEDA is significantly worse. Other cases suggest LMCEDA or LMSEDA performs similarly to the compared method. On the basis of these, the last row (*w/l/t*) of these tables counts the number of functions on which LMCEDA or LMSEDA significantly wins, significantly loses, and ties the competitions when compared with corresponding counterparts.

Observing Tables SIII–SXII in the supplement material, we can draw the following conclusions.

1) From the perspective of bprs, based on Tables SIII–SVII (in the supplemental material), we can see that with the accuracy level increasing, the performance of all methods degrades drastically, except for LMSEDA, whose performance is very stable. In details, when the accuracy level changes from ε = 1.0E-01 to ε = 1.0E-02, the bprs of CDE, NCDE, Self_CCDE, LoICDE, and

PNPCDE, whose performance is comparable to both LMCEDA and LMSEDA at accuracy level $\varepsilon = 1.0\text{E-}01$, decreases rapidly from 13, 15, 13, 15, and 13 to 8, 8, 10, 8, and 6, respectively, while that of LMSEDA keeps nearly unchanged. As the accuracy level changes from $\varepsilon = 1.0\text{E-}02$ to $\varepsilon = 1.0\text{E-}05$, LMSEDA always keeps its dominant position and performs significantly better than the other algorithms. When it comes to the last accuracy level, $\varepsilon = 1.0\text{E-}05$, we can observe that both LMCEDA and LMSEDA outperform all the compared algorithms.

2) When we take a further look at Tables SIII–SVII (in the supplemental material), both LMCEDA and LMSEDA achieve the best performance on F_{15}–F_{20}, except for F_{20}, on which LMSEDA obtains very similar result with NCDE, while LMCEDA receives poor performance. Such observation becomes more and more evident with the accuracy level increasing, which substantiates the competitive efficiency and superiority of LMEDAs in dealing with larger and more complex search spaces, where masses of local optima exist.

3) When it reaches the comparison in CS, since it makes no sense to evaluate this standard on functions where there is no successful run for one algorithm and on algorithms whose PR results are not comparable, we compare LMCEDA and LMSEDA only with CDE, NCDE, Self_CCDE, LoICDE, and PNPCDE on F_1–F_5, where they achieve comparable performance. At the first two accuracy levels, both LMCEDA and LMSEDA show no evident superiority to these five compared algorithms. However, when it arrives at the last three accuracy levels, we find that both LMCEDA and LMSEDA present their dominance to the five compared algorithms in CS. Take accuracy level $\varepsilon = 1.0\text{E-}04$ for example. From Table SVI (in the supplemental material), it can be found that LMCEDA achieves faster CS than CDE, NCDE, Self_CCDE, LoICDE, and PNPCDE on $3(F_2, F_3$, and $F_5)$, $3(F_1$–$F_3)$, $3(F_1$–$F_3)$, $2(F_2$ and $F_3)$, and $3(F_2, F_3$, and $F_5)$ functions, respectively, while LMSEDA converges faster than them on $3(F_2, F_3$, and $F_5)$, $3(F_1$–$F_3)$, $4(F_1$–F_3 and $F_5)$, $3(F_2, F_3$, and $F_5)$, and $3(F_2, F_3$, and $F_5)$ functions, respectively. This observation potentially shows that LMEDAs can achieve a competitive convergence speed to locate global optima for multimodal optimization.

4) From Tables SVIII–SXII (in the supplemental material), we find that the dominance of both LMCEDA and LMSEDA to the counterparts becomes more and more obvious as the accuracy level increases. Specially, at the last accuracy level, $\varepsilon = 1.0\text{E-}05$, both LMCEDA and LMSEDA are significantly better than all the compared algorithms on more than ten functions, except for NCDE and Self_CCDE. However, LMCEDA and LMSEDA still beat NCDE and Self_CCDE down on 7 and 8 functions, which is much more than the number of functions where LMCEDA and LMSEDA are defeated by the two compared algorithms, respectively. Particularly, we find both LMCEDA and LMSEDA are significantly superior to SDE, NSDE, and LoISDE on more than 17 functions. All these evidently verify the superiority of LMEDAs to the state-of-the-art multimodal algorithms.

Comprehensively, we can conclude that with the accuracy level increasing, both LMCEDA and LMSEDA become more and more outstanding when compared with the state-of-the-art multimodal algorithms. This observation demonstrates that both LMCEDA and LMSEDA achieve consistent superiority at most accuracy levels, which can be attributed to the powerful exploration ability and exploitation ability of LMEDAs, resulted from the three techniques proposed in this paper: 1) the dynamic cluster sizing; 2) the alternative usage of two distributions to generate offspring; and 3) the local search around seeds. Equipped with these strategies, LMEDAs can make a good balance between exploration and exploitation, which results in their efficiency and effectiveness in dealing with multimodal optimization problems.

V. Conclusion

This paper has developed MEDAs to locate multiple global optima for multimodal optimization problems. Distribution estimation and niching are effectively utilized to realize the proposed algorithms. Specially, the clustering-based niching tactics for crowding and speciation are incorporated, leading to crowding-based and speciation-based MEDAs, named MCEDA and MSEDA, respectively. Further, they are enhanced with local search, forming LMCEDA and LMSEDA, respectively. Their superior performance on multimodal problems is brought about by three techniques developed in this paper: 1) the dynamic cluster sizing; 2) the alternative usage of two distributions to generate offspring; and 3) the local search around seeds.

The niching methods of MEDAs are improved from those in the literature through developing a dynamic cluster-sizing strategy to afford a potential balance between exploration and exploitation, whereby relieving MEDAs from the sensitivity to the cluster size. Differing from classical EDAs to estimate the probability distribution of the whole population, MEDAs focus on the estimation of distribution at the niche level, and all individuals in each niche participate in the estimation of distribution of that niche. Further, the alternative usage of Gaussian and Cauchy distributions to generate offspring takes the advantages of both distributions, and potentially offers a balance between exploration and exploitation. Finally, the solution accuracy is enhanced through a new local search scheme based on Gaussian distribution with probabilities self-adaptively determined according to fitness values of seeds.

The influence of these three techniques has been fully tested in the experiments. The comparison results with respect to PR, SR, and CS between MEDAs and the state-of-the-art multimodal algorithms demonstrate the superiority and efficiency of MEDAs developed in this paper.

However, Tables SIII–SVII (in the supplemental material) also indicate that even though MEDAs can locate more global optima than the state-of-the-art multimodal algorithms, there is still room to improve the performance such that all global optima may be located when the dimensionality of the problem increases or there exist an excessive number of local optima. This directs certain future work in the development of MEDAs.

References

[1] K.-C. Wong, K.-S. Leung, and M.-H. Wong, "Protein structure prediction on a lattice model via multimodal optimization techniques," in *Proc. Conf. Genet. Evol. Comput.*, Portland, OR, USA, 2010, pp. 155–162.

[2] Q. Ling, G. Wu, and Q. Wang, "Restricted evolution based multimodal function optimization in holographic grating design," in *Proc. IEEE Congr. Evol. Comput.*, Edinburgh, U.K., 2005, pp. 789–794.

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

[4] Y. Li, "Using niche genetic algorithm to find fuzzy rules," in *Proc. Int. Symp. Web Inf. Syst. Appl.*, Nanchang, China, 2009, pp. 64–67.

[5] M. Boughanem and L. Tamine, "A study on using genetic niching for query optimisation in document retrieval," in *Advances in Information Retrieval*. Heidelberg, Germany: Springer, 2002, pp. 135–149.

[6] E. Dilettoso and N. Salerno, "A self-adaptive niching genetic algorithm for multimodal optimization of electromagnetic devices," *IEEE Trans. Magn.*, vol. 42, no. 4, pp. 1203–1206, Apr. 2006.

[7] D.-K. Woo, J.-H. Choi, M. Ali, and H.-K. Jung, "A novel multimodal optimization algorithm applied to electromagnetic optimization," *IEEE Trans. Magn.*, vol. 47, no. 6, pp. 1667–1673, Jun. 2011.

[8] J. Kennedy, "Bare bones particle swarms," in *Proc. IEEE SIS*, Indianapolis, IN, USA, 2003, pp. 80–87.

[9] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[10] A. Zhou, J. Sun, and Q. Zhang, "An estimation of distribution algorithm with cheap and expensive local search methods," *IEEE Trans. Evol. Comput.*, vol. 19, no. 6, pp. 807–822, Dec. 2015.

[11] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm Evol. Comput.*, vol. 1, no. 3, pp. 111–128, 2011.

[12] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of Machine Learning*. New York, NY, USA: Springer, 2010, pp. 760–766.

[13] Z. Ren, A. Zhang, C. Wen, and Z. Feng, "A scatter learning particle swarm optimization algorithm for multimodal problems," *IEEE Trans. Cybern.*, vol. 44, no. 7, pp. 1127–1140, Jul. 2014.

[14] K. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Heidelberg, Germany: Springer, 2006.

[15] Y. L. Li *et al.*, "Differential evolution with an evolution path: A DEEP evolutionary algorithm," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1798–1810, Sep. 2015.

[16] N. Chen *et al.*, "An evolutionary algorithm with double-level archives for multiobjective optimization," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1851–1863, Sep. 2015.

[17] W.-J. Yu *et al.*, "Differential evolution with two-level parameter adaptation," *IEEE Trans. Cybern.*, vol. 44, no. 7, pp. 1080–1099, Jul. 2014.

[18] S.-Y. Park and J.-J. Lee, "Stochastic opposition-based learning using a beta distribution in differential evolution," *IEEE Trans. Cybern.*, in press.

[19] W. Du, S. Y. S. Leung, Y. Tang, and A. V. Vasilakos, "Differential evolution with event-triggered impulsive control," *IEEE Trans. Cybern.*, in press.

[20] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey," *Computer*, vol. 27, no. 6, pp. 17–26, Jun. 1994.

[21] X.-Y. Zhang *et al.*, "Kuhn–Munkres parallel genetic algorithm for the set cover problem and its application to large-scale wireless sensor networks," *IEEE Trans. Evol. Comput.*, in press.

[22] W.-N. Chen *et al.*, "Particle swarm optimization with an aging leader and challengers," *IEEE Trans. Evol. Comput.*, vol. 17, no. 2, pp. 241–258, Apr. 2013.

[23] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 4. Perth, WA, Australia, 1995, pp. 1942–1948.

[24] Y.-J. Gong *et al.*, "Genetic learning particle swarm optimization," *IEEE Trans. Cybern.*, in press.

[25] W.-N. Chen *et al.*, "A novel set-based particle swarm optimization method for discrete optimization problems," *IEEE Trans. Evol. Comput.*, vol. 14, no. 2, pp. 278–300, Apr. 2010.

[26] A. Pétrowski, "A clearing procedure as a niching method for genetic algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Nagoya, Japan, 1996, pp. 798–803.

[27] G. R. Harik, "Finding multimodal solutions using restricted tournament selection," in *Proc. Int. Conf. Genet. Algorithms*, Pittsburgh, PA, USA, 1995, pp. 24–31.

[28] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proc. Int. Conf. Genet. Algorithms*, Cambridge, MA, USA, 1987, pp. 41–49.

[29] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, vol. 2. Portland, OR, USA, 2004, pp. 1382–1389.

[30] X. Li, "Efficient differential evolution using speciation for multimodal function optimization," in *Proc. Conf. Genet. Evol. Comput.*, Washington, DC, USA, 2005, pp. 873–880.

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

[32] C. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu, "Multimodal optimization by means of a topological species conservation algorithm," *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 842–864, Dec. 2010.

[33] C. L. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu, "Disburdening the species conservation evolutionary algorithm of arguing with radii," in *Proc. Conf. Genet. Evol. Comput.*, London, U.K., 2007, pp. 1420–1427.

[34] J. Gan and K. Warwick, "Dynamic niche clustering: A fuzzy variable radius niching technique for multimodal optimisation in GAs," in *Proc. IEEE Congr. Evol. Comput.*, Seoul, Korea, 2001, pp. 215–222.

[35] Y. Jie, N. Kharma, and P. Grogono, "Bi-objective multipopulation genetic algorithm for multimodal function optimization," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 80–102, Feb. 2010.

[36] L. Li and K. Tang, "History-based topological speciation for multimodal optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 1, pp. 136–150, Feb. 2015.

[37] 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.

[38] 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.

[39] 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.

[40] 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.

[41] B. Y. Qu, P. N. Suganthan, and S. Das, "A distance-based locally informed particle swarm model for multimodal optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 3, pp. 387–402, Jun. 2013.

[42] 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.

[43] X. Liang, H. Chen, and J. A. Lozano, "A Boltzmann-based estimation of distribution algorithm for a general resource scheduling model," *IEEE Trans. Evol. Comput.*, vol. 19, no. 6, pp. 793–806, Dec. 2015.

[44] S. Shahraki and M. R. A. Tutunchy, *Continuous Gaussian Estimation of Distribution Algorithm* (Advances in Intelligent Systems and Computing). Heidelberg, Germany: Springer, 2013, pp. 211–218.

[45] P. Yang, K. Tang, and X. Lu, "Improving estimation of distribution algorithm on multimodal problems by detecting promising areas," *IEEE Trans. Cybern.*, vol. 45, no. 8, pp. 1438–1449, Aug. 2015.

[46] D. Weishan and Y. Xin, "NichingEDA: Utilizing the diversity inside a population of EDAs for continuous optimization," in *Proc. IEEE Congr. Evol. Comput.*, Hong Kong, 2008, pp. 1260–1267.

[47] 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.

[48] 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.

[49] K. Deb and A. Saha, "Multimodal optimization using a bi-objective evolutionary algorithm," *Evol. Comput.*, vol. 20, no. 1, pp. 27–62, 2012.

[50] R. K. Ursem, "Multinational evolutionary algorithms," in *Proc. IEEE Congr. Evol. Comput.*, vol. 3. Washington, DC, USA, 1999, p. 1640.

[51] R. K. Ursem, "Multinational GAs: Multimodal optimization techniques in dynamic environments," in *Proc. Conf. Genet. Evol. Comput.*, Las Vegas, NV, USA, 2000, pp. 19–26.

[52] J. Yao, N. Kharma, and Y. Q. Zhu, "On clustering in evolutionary computation," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 1752–1759.

[53] W. Dong, T. Chen, P. Tino, and X. Yao, "Scaling up estimation of distribution algorithms for continuous optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 6, pp. 797–822, Dec. 2013.

[54] P. Larrañaga and J. A. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. New York, NY, USA: Springer, 2002.

[55] M. Pelikan and H. Mühlenbein, "The bivariate marginal distribution algorithm," *Advances in Soft Computing*. London, U.K.: Springer, 1999, pp. 521–535.

[56] J. S. De Bonet, C. L. Isbell, and P. Viola, "MIMIC: Finding optima by estimating probability densities," in *Proc. Conf. Adv. Neural Inf. Process. Syst.*, Denver, CO, USA, 1997, pp. 424–430.

[57] C. Tianshi, T. Ke, C. Guoliang, and Y. Xin, "Analysis of computational time of simple estimation of distribution algorithms," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 1–22, Feb. 2010.

[58] C. W. Ahn, J. An, and J.-C. Yoo, "Estimation of particle swarm distribution algorithms: Combining the benefits of PSO and EDAs," *Inf. Sci.*, vol. 192, pp. 109–119, Jan. 2012.

[59] H. Karshenas, R. Santana, C. Bielza, and P. Larranaga, "Multiobjective estimation of distribution algorithm based on joint modeling of objectives and variables," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 519–542, Aug. 2014.

[60] L. Wang, C. Fang, C.-D. Wu, and M. Liu, "A Pareto-archived estimation-of-distribution algorithm for multiobjective resource-constrained project scheduling problem," *IEEE Trans. Eng. Manag.*, vol. 60, no. 3, pp. 617–626, Aug. 2013.

[61] 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://goanna.cs.rmit.edu.au/∼xiaodong/cec13-lsgo/competition/

[62] M. Sebag and A. Ducoulombier, "Extending population-based incremental learning to continuous search spaces," in *Proc. Parallel Probl. Solving Nat. V*, Amsterdam, The Netherlands, 1998, pp. 418–427.

[63] P. A. Bosman and D. Thierens, "Expanding from discrete to continuous estimation of distribution algorithms: The IDEA," in *Parallel Problem Solving from Nature PPSN VI*. Heidelberg, Germany: Springer, 2000, pp. 767–776.

[64] P. Larrañaga, R. Etxeberria, J. A. Lozano, and J. M. Peña, "Optimization in continuous domains by learning and simulation of Gaussian networks," in *Proc. Conf. Genet. Evol. Comput.*, San Francisco, CA, USA, 2000, pp. 201–204.

[65] P. A. N. Bosman and D. Thierens, "Continuous iterated density estimation evolutionary algorithms within the IDEA framework," in *Proc. Optim. Build. Using Probabilist. Models OBUPM Workshop GECCO*, San Francisco, CA, USA, 2001, pp. 197–200.

[66] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.

[67] X. Li and X. Yao, "Cooperatively coevolving particle swarms for large scale optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 2, pp. 210–224, Apr. 2012.

Qiang Yang (S'14) received the M.S. degree from Sun Yat-sen University, Guangzhou, China, in 2014, where he is currently pursuing the Ph.D. degree.

His current research interests include evolutionary computation algorithms and their applications on real-world problems, large scale optimization algorithms, multimodal optimization algorithms, distribute evolutionary algorithms and their applications on real-world problems, like intelligent transportation.

Wei-Neng Chen (S'07–M'12) received the bachelor's and Ph.D. degrees from the Department of Computer Science, Sun Yat-sen University, Guangzhou, China, in 2006 and 2012, respectively.

He was a Lecturer and then an Associate Professor with Sun Yat-sen University, from 2012 to 2016. He is currently a Professor with the School of Computer Science and Engineering, South China University of Technology, Guangzhou. He has published 50 papers in international journals and conferences. His current research interests include swarm intelligence algorithms and their applications on cloud computing, operations research, and software engineering.

Dr. Chen was a recipient of the IEEE Computational Intelligence Society Outstanding Dissertation Award for his doctoral thesis in 2016, the Natural Science Foundation for Distinguished Young Scientists of Guangdong Province, China, in 2015, the "Guangdong Special Support Program" for Outstanding Young Scientists in 2015, and the Pearl River New Star in Science and Technology in 2014.

Yun Li (S'87–M'90) received the B.Sc. degree in radio electronics science from Sichuan University, Chengdu, China, in 1984, the M.Sc. degree in electronic engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, in 1987, and the Ph.D. degree in computing and control engineering from the University of Strathclyde, Glasgow, U.K., in 1990.

From 1989 to 1990, he was with National Engineering Laboratory and Industrial Systems and Control Ltd., Glasgow, U.K. He joined the University of Glasgow, Glasgow, as a Lecturer in 1991, served as the Founding Director with the University of Glasgow Singapore, Singapore, from 2011 to 2013, and was an Interim/the Founding Director of the University's first joint programme in China, in 2013, with the University of Electronic Science and Technology (UESTC), Chengdu, China. He has been a Visiting Professor with Kumamoto University, Kumamoto, Japan, UESTC, and Sun Yat-sen University, Guangzhou, China. He has supervised over 20 Ph.D. students and has over 200 publications.

Prof. Li established Evolutionary Computation workgroups for the IEEE Control System Society and European Network of Excellence in Evolutionary Computing (EvoNet) in 1998 and served on the Management Board of EvoNet from 2000 to 2005. He is a Chartered Engineer in the U.K.

C. L. Philip Chen (S'88–M'88–SM'94–F'07) received the M.S. degree in electrical engineering from the University of Michigan, Ann Arbor, MI, USA, in 1985, and the Ph.D. degree in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1988.

He was a Tenured Professor, the Department Head, and an Associate Dean in two different universities at U.S. for 23 years. He is currently the Dean with the Faculty of Science and Technology, University of Macau, Macau, China, and a Chair Professor with the Department of Computer and Information Science. His current research interests include systems and cybernetics.

Dr. Chen is a fellow of the American Association for the Advancement of Science and the Chinese Association of Automation. He was the President of the IEEE Systems, Man, and Cybernetics Society from 2012 to 2013. He has been the Editor-in-Chief of the IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS since 2014, and an Associate Editor of several IEEE TRANSACTIONS. He is also the Chair of TC 9.1 Economic and Business Systems of International Federation of Automatic Control. He is an Accreditation Board of Engineering and Technology Education Program Evaluator for Computer Engineering, Electrical Engineering, and Software Engineering programs.

Xiang-Min Xu (M'13) received the Ph.D. degree from the South China University of Technology, Guangzhou, China.

He is a Full Professor with the School of Electronic and Information Engineering, South China University of Technology, Guangzhou, China. His current research interests include image/video processing, human–computer interaction, computer vision, and machine learning.

Jun Zhang (M'02–SM'08) received the Ph.D. degree in electrical engineering from the City University of Hong Kong, Hong Kong, in 2002, under the supervision of Prof. H. Chung.

Since 2004, he has been with Sun Yat-sen University, Guangzhou, China, where he is a Cheung Kong Chair Professor. His current research interests include computational intelligence, cloud computing, high performance computing, data mining, wireless sensor networks, operations research, and power electronic circuits. He has authored seven research books and book chapters, and over 100 technical papers in the above areas.

Dr. Zhang was a recipient of the China National Funds for Distinguished Young Scientists from the National Natural Science Foundation of China in 2011 and the First-Grade Award in Natural Science Research from the Ministry of Education, China, in 2009. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, and the IEEE TRANSACTIONS ON CYBERNETICS. He is the Founding and the Current Chair of the IEEE Guangzhou Subsection, the IEEE Beijing (Guangzhou) Section Computational Intelligence Society Chapters, and the ACM Guangzhou Chapter.