Differential Evolution with Neighborhood Mutation for Multimodal Optimization

B. Y. Qu, P. N. Suganthan, *Senior Member, IEEE,* and J. J. Liang

***Abstract*—In this paper, a neighborhood mutation strategy is proposed and integrated with various niching differential evolution (DE) algorithms to solve multimodal optimization problems. Although variants of DE are highly effective in locating a single global optimum, no DE variant performs competitively when solving multi-optima problems. In the proposed neighborhood based differential evolution, the mutation is performed within each Euclidean neighborhood. The neighborhood mutation is able to maintain the multiple optima found during the evolution and evolve toward the respective global/local optimum. To test the performance of the proposed neighborhood mutation DE, a total of 29 problem instances are used. The proposed algorithms are compared with a number of state-of-the-art multimodal optimization approaches and the experimental results suggest that although the idea of neighborhood mutation is simple, it is able to provide better and more consistent performance over the state-of-the-art multimodal algorithms. In addition, a comparative survey on niching algorithms and their applications are also presented.**

***Index Terms*—Crowding, differential evolution, multimodal optimization, neighborhood mutation, niching algorithm, sharing, speciation.**

I. Introduction

MANY OPTIMIZATION problems contain several high quality global or local solutions which have to be identified and the most appropriate solution should be chosen. These problems are known as multimodal optimization problems. Classical evolutionary algorithms (EAs) were originally designed to locate single globally optimal solution. To solve multimodal optimization problems, numerous techniques commonly known as "niching" methods have been developed [1]–[5], [89], [90]. A niching method generally modifies the behavior of a classical EA in order to maintain multiple groups within a single population in order to locate multiple optima. The niching techniques include crowding [6], fitness sharing [7], restricted tournament selection [8], and speciation [9]. Besides multimodal problems, different niching techniques are also frequently used to solve multiobjective and dynamic optimization problems [10].

Differential evolution (DE) is a simple yet powerful global optimization technique with successful application in various areas [11], [12]. Unlike other EAs, DE modifies individuals by using differences of randomly sampled pairs of individual vectors from the population [13]. This mutation method is suitable for solving single global optimum optimization problems but is inappropriate for solving multimodal optimization. Various niching methods are integrated with DE to make it suitable for multimodal optimization. However, few work focuses on DE's mutation operation which is critical for efficiently solving multimodal optimization problems. Motivated by these observations, a neighborhood based mutation strategy is proposed and integrated with different niching DE algorithms. The method limits the mutation within a number of distance based neighborhood solutions, thereby making it effective in locating multiple optima.

Neighborhood concept has been widely used in evolutionary algorithms. Generally, neighborhoods can be divided into two main categories, namely index-based and distance-based. In a single global peak optimization scenario, index-based neighborhood is commonly used, especially in particle swarm optimization (PSO) [46]. In [47], different topology-based PSO algorithms were presented and compared. These topologies are based on the indices of the population. In 2004, Mendes [48] proposed a fully informed PSO, which also used topologies and index-based neighborhood as the basic structure. Index-based neighborhood was also adopted in DE. One of the earliest topological index-based neighborhood DE works was carried out by Tasoulis [51]. The algorithm divides the population into different subpopulations and uses ring topology to exchange information between different subpopulations. The method can be considered as "discrete" topological populations as each subpopulation runs as a separated DE in the offspring production phase. This method was modified and further improved by Weber *et al.* [52], [53]. In [49] and [50], Das *et al.* utilized the concept of index-based neighborhood of each population member to improve the performance of DE. The idea is the same of the community of the PSO algorithms and it balances the

Manuscript received November 18, 2010; revised March 15, 2011 and May 30, 2011; accepted June 9, 2011. Date of publication February 10, 2012; date of current version September 27, 2012. This work was supported in part by the National Natural Science Foundation of China, under Grant 60905039. This paper was recommended by Associate Editor B. Sendhoff.

B. Y. Qu is with the School of Electrical and Electronic Engineering, Nanyang Technological University, 639798, Singapore, and also with the School of Electrical Engineering, Zhengzhou University, Zhengzhou, Henan 450001, China (e-mail: e070088@ntu.du.sg).

P. N. Suganthan is with the School of Electrical and Electronic Engineering, Nanyang Technological University, 639798, Singapore (e-mail: epnsugan@ntu.edu.sg).

J. J. Liang is with the School of Electrical Engineering, Zhengzhou University, Zhengzhou, Henan 450001, China (e-mail: liangjing@zzu.edu.cn).

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the author. It contains Tables XIV–XXIII mentioned in Section VI as well as descriptions of the test problems. This material is 138 kB in size.

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/TEVC.2011.2161873

TABLE I
TAXONOMY OF NEIGHBORHOOD BASED EVOLUTIONARY ALGORITHMS

Global Optimization		Multimodal Optimization	
Index-Based Neighborhood	Distance-Based Neighborhood	Index-Based Neighborhood	Distance-Based Neighborhood
Topologies-based PSO [47] Fully informed particle swarm [48] Parallel DE [51] Distributed DE [52], [53] Global and local neighborhoods DE [49], [50]	DE with proximity-based mutation operators [54]	Ring topology PSO [28]	Crowding DE [6] Sharing DE [6] Species-based DE [13] Species-based PSO [28] Fitness Species conserving genetic algorithm [18] Fitness-Euclidean distance ratio PSO [28] Sharing GA [7] Clearing GA [17]

exploration and exploitation abilities of DE. This technique can be viewed as "continuous" topological populations as there is only one population involved. Compared with the extensive investigations into index-based neighborhoods, few works used the concept of distance-based neighborhood for single global optimum optimization. Recently, Epitropakis *et al.* [54] proposed a proximity-based mutation operator which selects the vectors to perform mutation operation using a distance related probability.

As, in multimodal optimization, the objective is locating multiple optima, diversity is one of the major issues. To maintain high diversity in the population, distance-based neighborhood is generally used to form different niches in multimodal optimization. Thomsen [6] proposed the Crowding DE which limits the competition between nearest (Euclidean distance) members to maintain the diversity. Species-based DE (SDE) [13] and speciation PSO (SPSO) [28] which form species based on Euclidean distance were introduced by Li. In contrast to the distance based neighborhoods, index-based neighborhood is only used by Li [28] in ring-topology based PSO for multi-modal optimization. Table I presents a taxonomy based on neighborhoods.

The remainder of this paper is organized as follows. Section II reviews various multimodal optimization algorithms. Section III discusses the differential evolution with niching. Section IV introduces the neighborhood based differential evolution algorithms. Experimental setup and numerical results are presented in Sections IV and V while Section VI concludes this paper.

II. MULTIMODAL OPTIMIZATION AND NICHING TECHNIQUES

When an optimization problem requires more than one optimal solution, it can be considered as a multimodal optimization problem. The objective of locating different optima in a single run makes it more complicated than locating a single global optimal solution. Different niching methods were introduced to enable EAs to maintain a diverse population and locate multiple solutions. The concept of niching is inspired by the way organisms evolve in nature. When integrated with EAs, niching involves the formation of subpopulations within a population. Each subpopulation aims to locate one optimal solution and together the whole population is expected to locate multiple optimal solutions in a single run. Various niching methods were proposed in the literature. Generally, the niching methods can be divided into two major categories: sequential niching and parallel niching. Sequential niching generally runs an algorithm iteratively to obtain multiple optima [20]. As sequential niching methods are time-consuming and the performance is relatively limited, we focus on parallel niching methods in this paper. In the literature, most multimodal evolutionary algorithms adopt parallel schemes. In the following section, several commonly used niching methods are briefly introduced.

A. Crowding and Restricted Tournament Selection

Crowding concept is one of the earliest and simplest niching techniques used to solve multimodal optimization problems.

1) *Original Crowding:* The crowding method introduced by De Jong in 1975 [14] allows competition for limited resources among similar individuals in the population. Hence, the competition is within each niche. This approach will maintain the diversity of the whole population. Generally, the similarity is measured using distance between individuals. The algorithm compares an offspring with some randomly sampled individuals from the current population. The most similar individual will be replaced if the offspring is a superior solution. A parameter CF called crowding factor is used to control the size of the sample. CF is generally set to 2 or 3. The computational complexity of crowding is equal to $O(N)$, where N is the population size. A major advantage of crowding is its simplicity. However, replacement error is the main disadvantage of crowding.

2) *Deterministic Crowding:* Due to the replacement error, the original crowding had a limited success in solving multimodal optimization problems [15]. To overcome this problem, Mahfoud introduced a deterministic crowding as an improvement to the original crowding [15]. It eliminates the CF, reduces replacement errors, and restores selection pressure. The main merit of this method is that no niching parameter is required. This method also faces the problem of loss of niches as it also uses localized tournament selection between similar individuals.

3) *Probabilistic Crowding:* To prevent the loss of niches with lower fitness or loss of local optima, Mengshoel [55] proposed probabilistic crowding. This method uses a probabilistic replacement rule which replaces the lower fitness individuals by higher fitness individuals in proportion to their finesses. The

TABLE II
SUMMARY OF DIFFERENT NICHING ALGORITHMS

Methods	Niching Parameter	Merits	Drawback	Complexity	Popular Algorithms	Applications
Crowding/RTS	Crowding factor/Window size	Simple and easy to run	Replacement errors/loss of niches	$O(N)/O(N \times w)$	Crowding DE [6]	[69], [77]
Fitness sharing	Sharing radius	Able to form and maintain stable niches	Selection of sharing radius and computational expensive	$O(N^2)$	Sharing GA [7] Sharing DE [6]	[75], [82]
Clearing	Clearing radius	Simple and able to preserve good solutions form good niches	Slow convergence rate and poor ability to locate local optima	O(c N)	Clearing GA [17]	[60]
Speciation	Radius	High diversity and able to maintain stable niches	Difficult to select the radius parameter	Best case-$O(N)$ Worst case-$O(N^2)$	Species conserving GA [18] Species-based DE [13]	[56]

advantage of this technique is the high diversity maintained by the probabilistic selection. On the other hand, this method suffers from slow convergence and poor fine searching ability.

4) *Restricted Tournament Selection:* The restricted tournament selection (RTS) [8] concept is very similar to crowding. The algorithm chooses a random sample of w (window size) individuals from the population and determines which one is the nearest to the offspring, by either Euclidean (for real variables) or Hamming (for binary variables) distance measure. The nearest member within the w individuals will compete with the offspring and the one with higher fitness will survive in the next generation. The complexity of RTS is $O(N \times w)$, where w is the window size. The advantage of RTS is the high diversity maintained by the competition between similar individuals while the disadvantage is the selection of window size and the replacement error.

B. Fitness Sharing

The fitness sharing was introduced by Holland [16] and extended by Goldberg and Richardson [7]. The concept is to divide the population into different subgroups according to the similarity of the individuals. An individual must share its information with other individuals within the same niche. The shared fitness for the ith individual can be represented as follows:

$$f_{\text{shared}}(i) = \frac{f_{\text{original}}(i)}{\sum_{j=1}^{N} sh(d_{ij})} \tag{1}$$

where the sharing function is calculated as

$$sh(d_{ij}) = \begin{cases} 1 - \left(\dfrac{d_{ij}}{\sigma_{\text{share}}}\right)^{\alpha}, & \text{if } d_{ij} < \sigma_{\text{share}} \\ 0, & \text{otherwise.} \end{cases}$$

d_{ij} is the distance between individuals i and j, σ_{share} is the sharing radius, N is the population size, and α is a constant called sharing level. The complexity of fitness sharing is $O(N^2)$. The advantage of sharing is its ability to form and maintain stable subpopulation/niches. Sharing also encourages the search in unexplored regions of the space by increasing the diversity of the population. One of the drawbacks of sharing is the usage of the niching parameter σ_{share}. Specifying this parameter requires prior knowledge of how far apart the optima are. However, in real world applications, such information is seldom available. Another disadvantage is the computational complexity of sharing. Sharing method is computationally more expensive compared to the other commonly used niching techniques [85].

C. Clearing

Clearing [17] is another widely used niching method. Different from fitness sharing, clearing removes the bad individuals and keeps only the best individual (or a few top individuals) within each niche. The algorithm first sorts the population in descending order according to the fitness values. Then it picks one individual at a time from the top and removes all the individuals with worse fitness than the selected one and falling within the specified clearing radius. This step will be repeated until all the individuals in the population are either selected or removed. Clearing eliminates similar individuals and maintains the diversity among the selected individuals. Similar to sharing, clearing also needs a user specified parameter σ_{clear} called clearing radius. This parameter is used as a dissimilarity threshold. The complexity of clearing is $O(cN)$, where c is the number of niches maintained during the generations. The advantage of clearing is its simplicity compared with sharing. Clearing is also able to preserve the best elements of the niches during the generations. However, clearing can be slow to converge and may not locate local optima effectively.

D. Speciation

The idea of speciation is commonly used in multimodal optimization [9], [18], [19]. This method also depends on a radius parameter r_s, which measures Euclidean distance from the center of a species to its boundary. The center of a species is called species seed. Each of the species is built around the dominating species' seed. All individuals that fall within the radius from the species seed are indentified as the same species. In this way, the whole population is classified into different groups according to their similarity. The complexity of speciation is $O(N)$ in the best case and $O(N^2)$ in the worst case. The main advantage of speciation is its ability to maintain high diversity and stable niches over generations while the main disadvantage is the selection of the radius parameter r_s.

In order to provide an overview of different niching methods, Table II summarizes the niching techniques discussed above. In addition to the methods listed above, there are other niching methods such as multi-population [39], clustering [21], and localized niching [22].

E. Applications of Niching Techniques

1) *Dynamic Optimization:* Niching algorithms are often used to maintain the diversity of the population and track moving peaks in dynamic optimization. In [56], Parrott and Li used the speciation technique to track multiple peaks in a dynamic environment. Subsequently in 2006, Li *et al.* [57] used SPSO to tackle dynamic problem by using detection and response. The method is designed for solving problems with primarily unknown numbers of peaks. Lung and Dumitrescu [58] used crowding DE to maintain diversity and combined it with PSO, called collaborative evolutionary-swarm optimization to solve dynamic optimization problems. In 2009, Lung and Dumitrescu [59] further improved and extended their work by introducing one more Crowing DE population that acted as a memory for the main population. Yu and Suganthan [60] presented a clearing technique to maintain the diversity of the archive used in their evolutionary algorithm to solve dynamic problems.

2) *Multiobjective Optimization:* Niching techniques are also commonly used in multiobjective optimization to adjust the fitness values in order to distribute individuals uniformly in the objective space [61], [62]. In [63] and [64], crowding concept is used to maintain diversity. Hiroyasu *et al.* [65] introduced the sharing technique into distributed genetic algorithms to increase the diversity of the solutions. Chen and Hsu [66] proposed a multiobjective optimization solver using rank-niche evolution strategy. This method also makes use of sharing concept to maintain uniformly distributed Pareto front. In 2008, Li *et al.* [67] introduced an adaptive niche multiobjective particle swarm optimization algorithm.

3) *Solving Real-World Problems:* Niching and multimodal optimization algorithms have been applied to solve problems in engineering [68], [74]. Zaharie [69] applied crowding differential evolution to solve unsupervised clustering problems. This approach treats the clustering problem as a multimodal optimization, which is similar to that of unsupervised niche clustering introduced by Nasraoui *et al.* [70]. Sheng *et al.* [77] used a modified deterministic crowding to solve the clustering problem. In [78], a niching memetic algorithm is used for simultaneous clustering and feature selection. Ling *et al.* [71] applied crowding differential evolution for solving robust optimal design problems. The crowding DE concept is used to solve market-based transmission expansion planning problems [72], and Nash equilibria detection problems in multiplayer games [73]. Dileetoso and Salerno [75] used a self-adaptive niching genetic algorithm for designing of electromagnetic devices. Perez [76] applied various niching algorithms for solving job shop scheduling problems. Redondo *et al.* [79] applied multimodal evolutionary algorithms for solving the multiple competitive facilities location and design problem. In 2002, Boughanem and Tamine [80] applied niching technique for query optimization in document retrieval. Li [81] used a niching genetic algorithm to discover fuzzy rules. Niching algorithms and multimodal optimization have also been used for dangerous areas identification in a grounding system [82]. Beside these, pattern matching and recognition is also a typical area in which multimodal optimization was used [83], [84].

III. Differential Evolution and Niching

A. Differential Evolution

The DE algorithm was proposed by Storn and Price in 1995 [23] to solve unconstrained single-objective optimization problems. Although the idea of DE is simple, it is efficient in solving global optimization problems [24], [45], [88]. DE is a population-based stochastic global optimization technique. The four main steps in DE are initialization, mutation, recombination and selection of parents for next generation from the current parent and offspring. Although different strategies have been suggested [25], [26], this paper uses DE/rand/1 strategy represented as follows:

$$v_i = x_{r1} + F \cdot (x_{r2} - x_{r3}) \tag{2}$$

where v_i is the mutation vector, $r1$, $r2$, $r3$ are random and mutually different integers drawn from the set of population indices, which should also be different from the current target vector x_i. F is a scale factor in [0, 2] used for scaling the differential vector.

Crossover or recombination is applied after mutation process to obtain the trial vector $\mu_{i,j}$

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if } rand_j \leq CR \text{ or } j = k \\ X_{i,j}, & \text{otherwise} \end{cases} \tag{3}$$

where CR is a control parameter of DE that decides in a comparison with a random number $rand_j$ in the range [0, 1] whether components are copied from v_i or x_i, respectively, into trial vector u_i [27]. The selection process is based on a simple competition between the corresponding parent and offspring in the case of single objective global optimization.

B. Differential Evolution for Niching

Several DE-based niching algorithms have been proposed in the literature to solve multimodal problems [6], [13], [40]–[44]. Thomsen integrated the fitness sharing concept with DE to form the sharing DE [6]. Thomsen also proposed to extend DE with a crowding scheme called crowding DE (CDE) [6] to allow it to tackle multimodal optimization problems. CDE which has a crowding factor equal to the population size has outperformed the sharing DE. In CDE, when an offspring is generated, it will only compete with the most similar (measured by Euclidean distance) individual in the current population. The offspring will replace this individual if it has a better fitness value. The CDE is summarized in Table III.

SDE [13] is another commonly used DE niching algorithm. The concept is the same as speciation described in Section II. Different niches are formed around the species seed. The mutation is carried out within each species. The details of SDE are presented in Table IV.

TABLE III
CROWDING DE (CDE)

Step 1	Randomly generate NP number of initial trial solutions.
Step 2	For $i = 1$ to NP Produce an offspring u_i using the standard DE. Calculate the Euclidean distance of u_i to the other individuals in the DE population. Compare the fitness of u_i with the most similar individual and replace it if the u_i has a better fitness value. Endfor
Step 3	Stop if a termination criterion is satisfied. Otherwise go to Step 2.

TABLE IV
SPECIES-BASED DE (SDE)

Step 1	Randomly generate NP number of initial trial solutions.
Step 2	Sort all individuals in descending order of their fitness values.
Step 3	Determine the species seeds for the current population: the most-fit individual will be set as the first species seed. Then all individuals are checked in turn from most-fit to least-fit against the species seeds found so far. If an individual does not fall in the radius of any seeds, it will be indentified as another species seed.
Step 4	For each species, execute a global DE variant: 4.1 If a species has less than m individuals, then randomly generate new individuals within the radius of the species seed. 4.2 If the child's fitness is the same as its species seed, replace this child with a randomly generated new individual.
Step 5	Keep only the NP fitter individuals from the combined population.
Step 6	Stop if a termination criterion is satisfied. Otherwise go to Step 4.

TABLE V
MODIFIED FITNESS SHARING METHOD

Input:	The population with original fitness value
Step 1	Sort the population in descending order of original fitness While the population is not empty
Step 2	Find the best unprocessed solution in the population and set it as the new niche center.
Step 3	Identify the solutions within the new niche using the specified niching radius.
Step 4	Penalize the solutions (except the niche center) within the niche using the original fitness sharing formula (Section II-B). Note that the best solution at the current niche center uses the original fitness value.
Step 5	Remove the niche center and the solutions within the niche from the population.
	Endwhile
Output:	The population with shared fitness computed using the modified fitness sharing method.

In this paper, a modified version of fitness sharing DE is also introduced. The process of calculating the modified shared fitness and the steps of the modified fitness sharing DE are presented in Tables V and VI, respectively.

IV. NEIGHBORHOOD BASED DIFFERENTIAL EVOLUTION

In the standard DE, the mutation is performed between randomly picked individuals from the entire population. This will allow any two members to generate the difference vector in DE/rand/1, even if the two members are far apart from each other. This mutation is efficient when searching for single global solution. It prevents premature local convergence and ensures global convergence in the final stage as all individuals in general evolve to one optimal point. However, when solving a multimodal problem, multiple optima need to be located simultaneously. If the global version of the mutation is used, it will not be efficient for multiple localized convergences at the final search stage as required for multimodal optimization. At final search stage, the whole population is distributed around different optimal regions. If the parameter space distances between different optima are large, efficient convergence to any of the optima will become impossible as the difference vectors can be generated using individuals from different optimal regions with relatively large magnitudes. Although some of the niching techniques can limit the mutation within each niche (SDE), they all depend on certain

TABLE VI
MODIFIED FITNESS SHARING DE

Step 1	Randomly initialize the population and external archive. The archive size is two times of the population size.
Step 2	Modify the fitness value of the solutions in archive using the modified fitness sharing method (Table V).
Step 3	Sort the solutions in archive in decreasing order of modified fitness.
Step 4	The first NP (population size) solutions in archive are used as parents to produce offspring using neighborhood mutation.
Step 5	Update the offspring with the nearest (Euclidean distance) member in the archive.
Step 6	Stop if a termination criterion is met. Otherwise go to Step 2.

TABLE VII
STEPS OF GENERATING OFFSPRING USING NEIGHBORHOOD MUTATION

Input	A population of solutions of current generation (current parents)
Step 1	For $i = 1$ to NP (population size)
	1.1 Calculate the Euclidean distances between individual i and other members in the population.
	1.2 Select m smallest Euclidean distance members to individual i and form a subpopulation ($subpop$) using these m members.
	1.3 Produce an offspring u_i using DE equations within $subpop_i$, i.e., pick r_1, r_2, r_3 from the subpopulation.
	2.3 Reset offspring u_i within the bounds if any of the dimensions exceed the bounds.
	2.4 Evaluate offspring u_i using the fitness function.
	Endfor
Step 2	Selection NP fitter solutions for next generation according to the strategies of different niching algorithm.
Output	A population of solutions for next generation

niching parameters such as the species radius. Choosing a proper niching parameter itself is almost impossible and the algorithm is greatly affected by the niching parameter.

In [86], Neri and Tirronen stated that DE needed some extra moves to improve the performance and the lack of these search moves can cause stagnation. In [87], Brest and Maucec also stated the importance of reducing population size in order to improve the performance of DE. Inspired by these observations, a neighborhood based mutation is proposed and integrated with three different DE niching algorithms, namely CDE, SDE, and sharing DE. Our proposed neighborhood concept allows a higher exploitation of the areas piloting the moves, thereby facilitating multiple convergences. In neighborhood mutation, difference vector generation is limited to a number (parameter m) of similar individuals as measured by Euclidean distance. The steps of generating offspring using neighborhood mutation are presented in Table VII. In this way, each individual is evolved toward its nearest optimal point and the possibility of between niche difference vector generation is reduced. The neighborhood based CDE (NCDE), neighborhood based SDE (NSDE), and neighborhood based sharing DE (NShDE) are presented in Tables VIII, IX, and X.

In neighborhood mutation, there is only one parameter m which is the neighborhood size. This parameter controls how many individuals are selected in each subpopulation. Generally, m should be chosen between 1/20 of the population size and 1/5 of the population size. It can also be dynamically set, from a relatively large value to a small value. Different

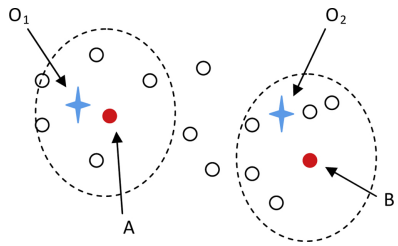

Fig. 1. Illustration of neighborhood mutation.

from other niching parameters, neighborhood size is easy to choose as it can be made proportional to the population size. Moreover, the performance of algorithm is not sensitive to the change of the neighborhood size as evidenced in Table XXI (NCDE).

To further illustrate the detection of local optima by neighborhood mutation, Fig. 1 is presented. As can be seen, O_1 and O_2 are the two peaks to be located. If solutions A and B are considered, it can be easily seen that two separated niches are formed by Euclidean distance neighborhood concept. The mutation vectors are generated within the subpopulation which is likely to move solution A toward O_1 and move solution B toward O_2.

V. EXPERIMENTAL SETUP

The algorithms were implemented in MATLAB 7.1 and executed using a Pentium 4 computer with 2.99 GHz CPU

TABLE VIII
NCDE ALGORITHM

Step 1	Randomly generate NP number of initial trial solutions.
Step 2	For i = 1 to NP
	2.1 Find the parametrically most similar (the similarity is measured in terms of Euclidean distances of parameter space) m individuals of solution i to form a subpopulation $subpop_i$.
	2.2 Produce an offspring u_i using DE within $subpop_i$, i.e., pick r_1, r_2, r_3 from the ith subpopulation.
	2.3 Calculate the Euclidean distance of u_i to the other individuals in the entire population.
	2.4 Compare the fitness of u_i with the most similar (in Euclidean distance) individual and replace the most similar individual if the u_i has a better fitness.
	Endfor
Step 3	Stop if a termination criterion is satisfied. Otherwise go to Step 2.

TABLE IX
NSDE ALGORITHM

Step 1	Randomly generate NP number of initial trial solutions.
Step 2	Sort all individuals in descending order of their fitness values.
Step 3	While the sorted population is not empty Determine the species seed which is the best (fitness value) unprocessed individual. Find the parametrically most similar (in Euclidean distance) m individuals of the species seed and set them as one species. Remove the processed members for the current populations. Endwhile
Step 4	For each species, execute a global DE variant: If the child's fitness is the same as its species seed, replace this child with a randomly generated new individual.
Step 5	Keep only the NP fitter (objective value) individuals from the combined population.
Step 6	Stop if a termination criterion is satisfied. Otherwise go to Step 4.

and 2 GB RAM. The operating system is Microsoft Windows XP. The DE parameters used in this paper are as follows: $F = 0.9$ $CR = 0.1$. Two experiments are conducted. In the first experiment, the proposed algorithm is compared with a number of commonly used niching algorithms, while the second experiment compares the performance of neighborhood based DE against one of the most recently reported multimodal algorithms. In Experiment Two, we use the same test functions and settings in [38] instead of reprogramming the algorithms in [38]. In total, 18 different multimodal algorithms are considered in our experiments:

1) NCDE: the neighborhood based crowding DE;
2) NSDE: the neighborhood based speciation DE;
3) NShDE: the neighborhood based sharing DE;
4) CDE [6]: the original crowding DE;
5) SDE [13]: speciation-based DE;
6) ShDE: modified sharing DE;
7) FERPSO [28]: fitness-Euclidean distance ratio PSO;
8) SPSO [28]: speciation-based PSO;
9) *r*2pso [28]: a *lbest* PSO with a ring topology, each member interacts with only its immediate member to its right;
10) *r*3pso [28]: a *lbest* PSO with a ring topology, each member interacts with its immediate member on its left and right;
11) *r*2pso-*lhc* [28]: the same as *r*2pso, but with no overlapping neighborhoods;
12) *r*3pso-*lhc* [28]: the same as *r*3pso, but with no overlapping neighborhoods;
13) CMA-ES [35]: niching covariance matrix adaptation evolution strategy;
14) SCMA-ES [35]: self-adaptive niching CMA-ES;
15) TSC [36]: topological species conservation;
16) SCGA [18]: species conserving genetic algorithm;
17) DFS [37]: dynamic fitness sharing;
18) TSC2 [38]: topological species conservation 2.

A. Test Functions

In Experiment One, two sets of test functions are used in order to demonstrate the superior performance of neighborhood mutation based DE. These functions are widely used in multimodal optimization and possess diverse characteristics. Set 1 contains 14 basic multimodal problems while set 2 has 15 composite multimodal problems. Table XI presents the problems used in Experiment One.

In Experiment Two, the test functions and results in [38] are used. Table XII shows the test functions. More details can be found in [38]. The experimental procedure is adopted from [38]. For E1-F1, E1-F2, E1-F6, E1-F7, E1-F10 to E1-F13 the target is to locate all the peaks (global and local), while for the

TABLE X
NSHDE ALGORITHM

Step 1	Randomly initialize the population and external archive. The archive size is two times of the Population size.
Step 2	Modify the fitness value of the solutions in archive using the modified fitness sharing method (Table III).
Step 3	Sort the solutions in archive in decreasing of the modified fitness order.
Step 4	The first NP (population size) solutions in archive are used as parents to produce offspring.
Step 5	For $i = 1 : NP$
	5.1 Find the parametrically most similar (in Euclidean distance) m individuals of solution i to form a subpopulation $subpop_i$.
	5.2 Produce an offspring u_i using DE within $subpop_i$, i.e., pick r_1, r_2, r_3 from the ith subpopulation.
	5.3 Calculate the Euclidean distance of u_i to the other individuals in the entire external archive.
	5.4 Compare the fitness of u_i with the most similar individual and replace the most similar individual if the u_i has a better fitness.
	Endfor
Step 6	Stop if a termination criterion is met. Otherwise go to Step 2.

TABLE XI
TEST FUNCTIONS FOR EXPERIMENT ONE (CF: COMPOSITION FUNCTION)

Test Function Set 1		Test Function Set 2 [33]	
Test Function Name	Number of Global Peaks/Dimension	Test Function Name	Number of Global Peaks/Dimension
E1-F1: two-peak trap [29]	1/1	E1-F15: CF 1	8/10
E1-F1: central two-peak trap [29]	1/1	E1-F16: CF 2	6/10
E1-F3: five-uneven-peak trap [18]	2/1	E1-F17: CF 3	6/10
E1-F4: equal maxima [30]	5/1	E1-F18: CF 4	6/10
E1-F5: decreasing maxima [30]	1/1	E1-F19: CF 5	6/10
E1-F6: uneven maxima [30]	5/1	E1-F20: CF 6	6/10
E1-F7: uneven decreasing maxima [30]	1/1	E1-F21: CF 7	6/10
E1-F8: Himmelblau's function [30]	4/2	E1-F22: CF 8	6/10
E1-F9: six-hump camel back [31]	2/2	E1-F23: CF 9	6/10
E1-F10: Shekel's foxholes [14]	1/2	E1-F24: CF 10	6/10
E1-F11: 2-D inverted Shubert function [18]	18/2	E1-F25: CF 11	8/10
E1-F12: 1-D inverted Vincent function [32]	6/1	E1-F26: CF 12	8/10
E1-F13: 2-D inverted Vincent function [32]	36/2	E1-F27: CF 13	10/10
E1-F14: 3-D inverted Vincent function [32]	216/3	E1-F28: CF 14	10/10
		E1-F29: CF 15	10/10

rest the objective is to search for the global optimum/optima and to escape the local peaks.

B. Population Size and Maximal Number of Evaluations

In Experiment One for test function set 1, the level of accuracy, niching radius, population size, and maximal number of function evaluations are shown in Table XIII. For test function set 2, a population size of 600 is used with the maximal number of function evaluations set at 300 000. The level of accuracy and niching radius are set as 0.5 and 1, respectively, for all the problems in test function set 2. The detailed setting for experiments two can be found in [38].

C. Performance Measure

1) *Experiment One:* To compare the performance of different multimodal algorithms, in this experiment, a level of accuracy (typically $0<\varepsilon<1$) indicating how close the computed solutions to the known global peaks are, needs to be specified [28]. If the difference from a computed solution to a known global optimum is below ε, then the peak is considered to have been found. The performance of all multimodal algorithms is measured in terms of the following two criteria:

a) Success rate (the percentage of runs in which all global peaks are successfully located);
b) Average number of optima found [34].

Note that the success rate depends on the specified level of accuracy. For a more relaxed level of accuracy, an algorithm is more likely to have a higher success rate [28]. All performances are calculated and averaged over 25 independent runs to deal with the effect due to the random initialization.

2) *Experiment Two:* In this experiment, two criteria in [38] are used.

a) *Peak accuracy:* For each desired peak to be located the closest individual x in the population is taken and absolute difference in objective values is calculated. If

TABLE XII
TEST FUNCTIONS FOR EXPERIMENT TWO [38]

Test Function Name	Number of Global Peaks/Dimension	Test Function Name	Number of Global Peaks/Dimension
E2-F1: waves	10/2	E2-F8: Shubert	18/2
E2-F2: six-hump camel back	2/2	E2-F9: Ackley	1/2
E2-F3: sphere	1/2	E2-F10: Michalewicz	1/2
E2-F4: shifted Rastrigin	1/2	E2-F11: Ursem F1	1/2
E2-F5: rotated hybrid composition function	1/2	E2-F12: Ursem F3	1/2
E2-F6: rescaled six-hump camel back	2/2	E2-F13: Ursem F4	1/2
E2-F7: Branin RCOS	3/2		

TABLE XIII
PARAMETER SETTINGS FOR E1-F11 TO E1-F14 (SET 1)

Function No.	ε	r	Population Size	No. of Function
E1-F1	0.05	0.5	50	10 000
E1-F2	0.05	0.5	50	10 000
E1-F3	0.05	0.5	50	10 000
E1-F4	0.000001	0.01	50	10 000
E1-F5	0.000001	0.01	50	10 000
E1-F6	0.000001	0.01	50	10 000
E1-F7	0.000001	0.01	50	10 000
E1-F8	0.0005	0.5	50	10 000
E1-F9	0.000001	0.5	50	10 000
E1-F10	0.00001	0.5	50	10 000
E1-F11	0.05	0.5	250	100 000
E1-F12	0.0001	0.2	100	20 000
E1-F13	0.001	0.2	500	200 000
E1-F14	0.001	0.2	1000	400 000

the objective value of individual x is denoted by $f(x)$, the peak accuracy is calculated using the following equation:

$$peak\ accuracy = \sum_{i=1}^{\#peaks} |f(peak_i) - f(x)|.$$

b) *Distance accuracy:* peak accuracy may lead to erroneous results, if the peaks are close to each other or with identical height. The distance accuracy is used to avoid this error. It is calculated the same way as peak accuracy, with the only change that the fitness values are replaced by the Euclidean distance [38].

VI. EXPERIMENTAL RESULTS

A. Experiment One

This section presents the experimental results and analyses of Experiment One. All algorithms were run until all known peaks were found or the maximum number of function evaluations was exhausted. To save space, all the results (i.e., tables) are presented in supplementary files, which can be downloaded from http://ieeexplore.ieee.org or the author's web page http://www.ntu.edu.sg/home/epnsugan/ and then clicking on "Publications".

1) *Results on Test Function Set 1:* The results of test function set 1 are shown in Tables XIV and XV (see supplementary file). The second and third columns of these tables indicate the level of accuracy and niche radius (for SDE and SPSO) used in the experiments. Success rates with ranks (inside the bracket) are listed in Table XIV (see supplementary file) while the average number of optima found is listed in Table XV (see supplementary file). From these two tables, we can see that the proposed algorithm outperforms the other algorithms in terms of both criteria. For these 14 problems, the proposed algorithms ranks top three among all the compared state-of-the-art algorithms. If we compare NCDE with the original CDE, we can see that NCDE improved 11 problems out of 14 while the remaining 3 cases perform the same as both methods are able to locate all optima for all runs. The superior performance is due to the neighborhood mutation, which makes it easier for the proposed algorithms to converge to different optima.

In order to determine the statistical significance of the advantage of neighborhood mutation based DE over other methods, t-test is applied on the average number of peaks found and the results are shown in Table XVI (see supplementary file). The numerical values 1, 0 represent that other methods are statistically inferior to, equal to the best algorithm. From the results, we can observe that the neighborhood based DE methods always perform better or equal when compared with other niching methods on all test functions.

As stated by Mahoud [15], a good niching algorithm must be able to locate global optima and maintain them for an exponential to infinite time period, with respect to population size. Maintaining found optima is crucial for multimodal optimization algorithms. The neighborhood based DE algorithms are able to find and maintain the found optima to the end of

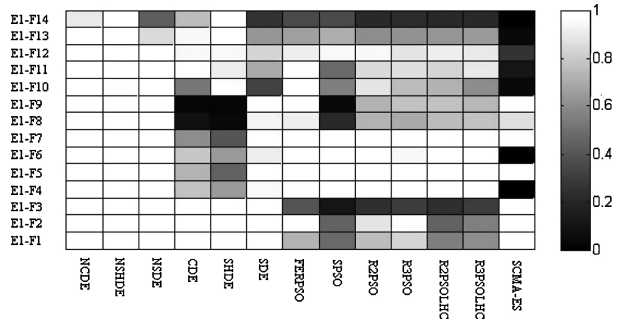

Fig. 2. Overview of average number of peaks found by each algorithm (Set 1). Results are normalized for each problem so that 1 (white) refers to the best and 0 (black) to the worst algorithm.

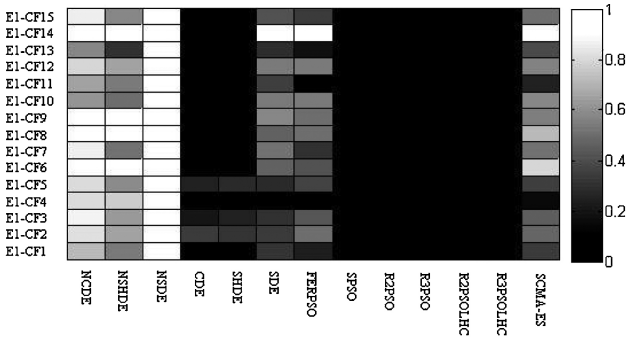

Fig. 3. Overview of average number of peaks found by each algorithm (Set 2). Results are normalized for each problem so that 1 (white) refers to the best and 0 (black) to the worst algorithm.

the run. This is because of the replacement strategies. The replacement is either within the subpopulation or between most similar individuals. Therefore, once a niche is formed around one global peak, it can maintain this peak until the end of the run.

2) *Results on Test Function Set 2:* All 15 functions in test function set 2 are composition functions. These composition functions are much more complex than the functions in set 1. Among these test problems, no algorithm is able to generate a nonzero success rate except the proposed algorithms (NCDE and NSDE). NCDE is able to generate the success rates of 0.8, 0.1, and 0.4 on $F3$, $F4$, and $F5$, respectively, while NSDE is able to generate the success rates of 1, 0.5, and 0.9 on these test functions. Therefore, the average number of global optima found is used as the sole criterion. The results are shown in Table XVII (see supplementary file). NSDE ranks the best out of the 13 algorithms on every problem. The statistical test results are shown in Table XVIII (see supplementary file). By comparing the results of benchmark sets 1 and 2, we can conclude that as the complexity of the problems is increased, the proposed neighborhood mutation increases its advantage over all competing state-of-the-art algorithms. Figs. 2 and 3 present a simplified visual comparison based on average number of peaks found by all algorithms.

3) *Locating Local Optima:* In order to test the ability of locating local optima, five test functions (E1-F1, E1-F2, E1-F3, E1-F5, E1-F10) from Set 1 are used. The results are shown in Tables XIX and XX (see supplementary file). It can

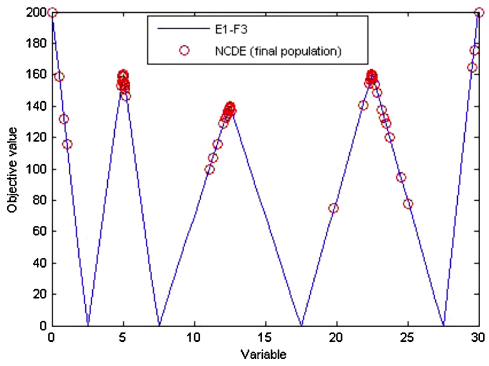

Fig. 4. Final population of NCDE for E1-F3.

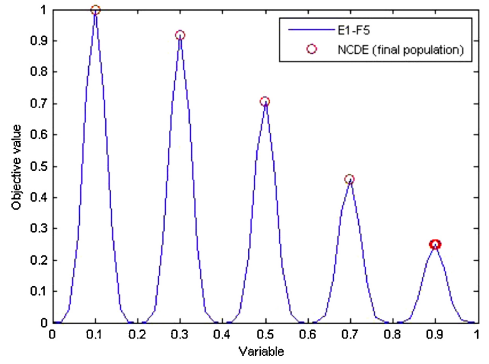

Fig. 5. Final population of NCDE for E1-F5.

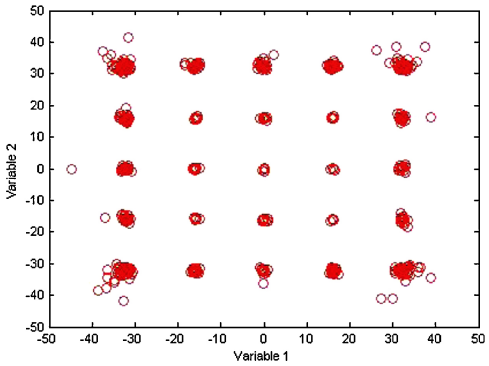

Fig. 6. Final population of NCDE for E1-F10.

be observed that with neighborhood mutation the ability of locating both global and local optima is greatly improved. To give a clearer view, the final population of NCDE is also plotted (Figs. 4–6) for E1-F3, E1-F5, and E1-F10. Note that the population size and maximum number of function evaluations are set to 500 and 100 000 for E1-F10. The settings of remaining parameters are kept unchanged.

4) *Performance of Different Algorithms on the Test Problems:* This section presents a brief summary on the performance of different algorithms on the tested benchmark functions.

a) *The three novel algorithms (NCDE, NSHDE, and NSDE):* As we can see from the above tables, all the three algorithms are able to generate consistent and satisfactory performance over a large number of test functions.

b) *CDE, SHDE, and SPSO:* They can generate acceptable results over simple and low dimensional functions (test function set 1), but their fine tuning abilities are limited. Their performances over difficult and high dimensional problems (test function set 2) are poor.

c) *SDE:* Although SDE is always able to find certain number of global peaks and the fine tuning ability over the found optima is good, it fails to find most of the local optima. SDE algorithm also shows very poor performance if the number of global peaks is high. Therefore, if the user's target is finding a few global peaks with high accuracy, SDE could be a good choice. Otherwise, SDE may not be able to generate satisfactory results.

d) *FERPSO:* Compared to the algorithms proposed in the literature, FERPSO is able to generate relatively satisfactory results over many test functions. However, the ability of locating local optima is very poor. It cannot be used to locate both global and local optima.

e) *R2PSO and R3PSO:* These two methods perform well on simple problems and fail on difficult test functions. Their ability to locate local optima is also poor. These algorithms do not require any of the niching parameters (same as the proposed algorithm) which can be claimed as one of the advantages, as the performance of other algorithms may vary according to the assigned values of the niching parameters.

f) *R2PSOLHC and R3PSOLHC:* The performances are similar to R2PSO and R3PSO. The only difference is that these two algorithms demonstrated slightly better ability in locating local optima.

g) *SCMA-ES:* SCMA-ES is one of the most complicated multimodal algorithms. Its performance on complex and high dimensional problems is good. However, this algorithm does not possess good fine searching ability. Therefore, it may fail if the level of accuracy is high when solving even simple problems.

5) *Effect of Varying the Neighborhood Size:* Although the performance is not sensitive to the neighborhood size m, it can still affect the performance. In order to test the effect of varying the neighborhood size, three problems (E1-F1, E1-F4, and E1-F11) from test function set 1 are examined using NCDE. The results are shown in Table XXI (see supplementary file). For E1-F1 and E1-F4, m is set at 5 (1/10 of population size) and 10 (1/5 of the population size). As we can see, there is no difference in performance. For E1-F11, the m is varied from 20 (2/25 of the population size) to 50 (1/5 of the population size). As we can see, only when m = 20, the performance is reduced by a small margin.

6) *Advantage of Neighborhood Mutation:* To show the advantage of the neighborhood mutation, the NCDE is compared with the original CDE. The test compares the percentage of the algorithm that can produce an offspring which is within the same niche or targeting on the same peak as its parent. If the offspring generated is within the same niche of the parents, the offspring and parents are searching for the same peak. In this case, the chance of refining this peak is increased. In each generation, the nearest peak to parent and the nearest peak to its offspring are indentified. If the peaks are the same, it is consider within the same niche and the percentage is recorded.

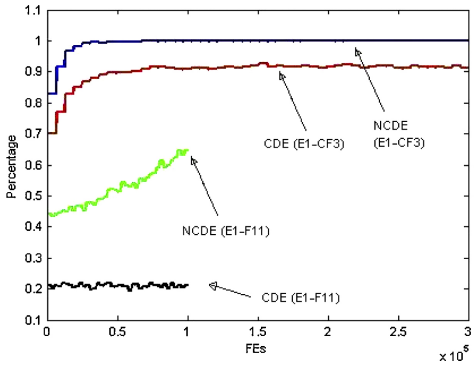

Fig. 7. Percentage of local mutation.

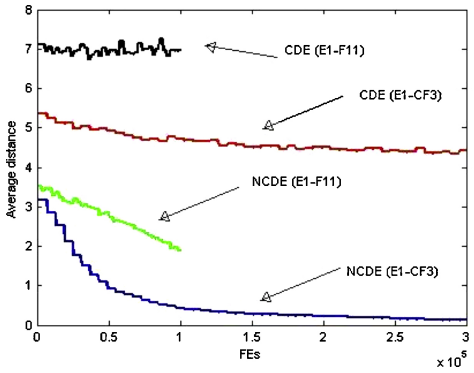

Fig. 8. Average distance between parents and offspring.

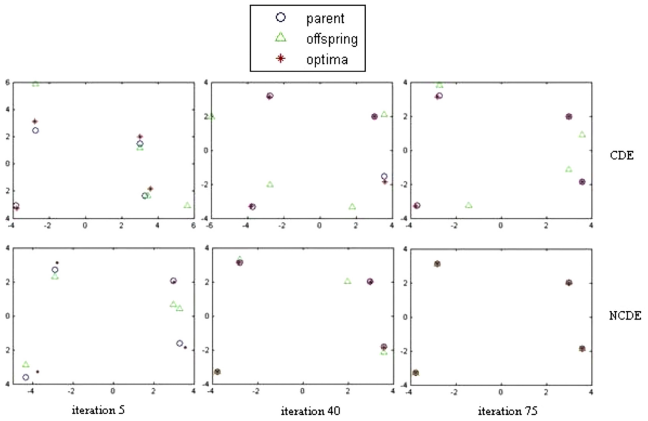

Fig. 9. Best parents and its offspring of CDE and NCDE in different iterations on E1-F8.

Fig. 7 shows the percentage (the percentage that an offspring is within the same niche as its parent) versus the function evaluation for F11 and CF3. As can be seen, NCDE always generates a higher percentage than CDE. In order to give a clearer view, the average distance between the parents and their offspring in each generation is plotted in Fig. 8. The results show that NCDE always has a smaller distance than CDE. To further demonstrate the merit of neighborhood mutation, The best parents and their offspring of CDE and NCDE in different iterations (E1-F8) are plotted in Fig. 9. As can be seen from the plot, the best parents of NCDE are always able to produce offspring around them, which means within the same niche and targeting the same optimal solution. However, the offspring produced are generally far away from their parents for CDE

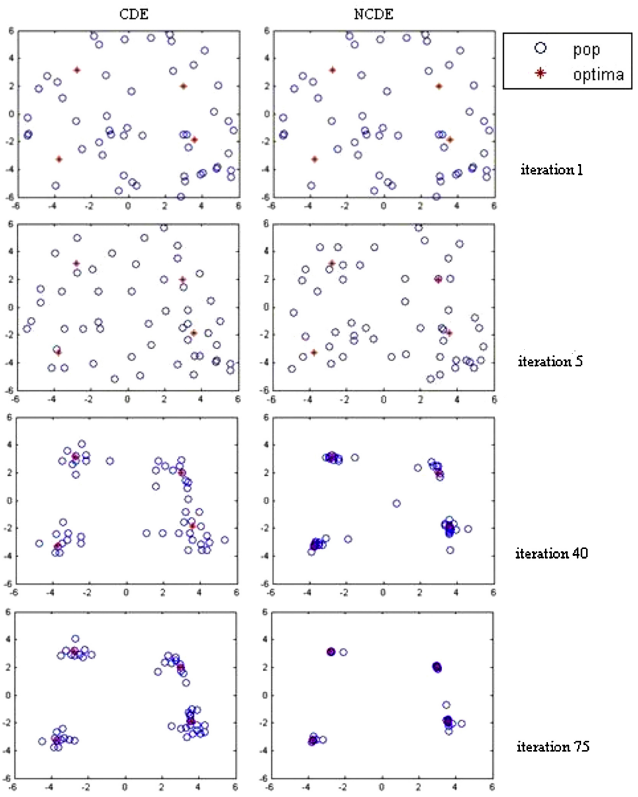

Fig. 10. Niching behavior of CDE and NCDE on E1-F8 (with identical initial solutions).

which may cause slow convergence and poor fine-searching ability. The distributions of population of CDE and NCDE in different iterations (E1-F8) are also presented in Fig. 10, which clearly shows NCDE converges faster than the original CDE.

B. Experiment Two

In this section, the test functions/results reported in [38] are used and compared with the proposed neighborhood based DE algorithms. The results for the two criterions (peak accuracy/distance accuracy) are shown in Tables XXII and XXIII (see supplementary file). From Table XXII (see supplementary file), we can see that the proposed algorithm performs the best except F8. However, the superior performance of DFS on F8 is because of the peak accuracy error as explained in Section V-C on performance measures. The table of distance accuracy can truly reflect the performance of different algorithms where the neighborhood mutation based DEs rank on top for all test functions. Note that NSDE performs very well, if the target is to locate multiple global optima only. However, the performance decreases greatly if both local and global optima need to be found. This is because of the selection strategy used in SDE and NSDE. The selection method may discard some of the local peaks during the evolution process.

VII. Conclusion

DE has been widely used as an efficient optimization algorithm. Although different niching techniques have been integrated with DE, niching DE algorithms' performance is unsatisfactory on multimodal optimization problems. This paper proposed a neighborhood-based mutation and integrated it with various niching DE algorithms to solve multimodal optimization problems. Neighborhood mutation is able to restrict the production of offspring within a local area or the same niche as their parents. This method ensures that the algorithms converge faster with a high accuracy. We demonstrated that the neighborhood mutation is able to induce stable niching behavior. The neighborhood based DE algorithm is able to locate multiple global optima and maintain them. The results of experimental studies suggest that the proposed algorithms can provide a better and more consistent performance than numerous state-of-the-art multimodal optimization algorithms on a large number of test problems. Out of 29 problems, the proposed method improved almost all of them except a few test instances that both proposed and the original algorithms are able to solve them perfectly.

References

[1] S. W. Mahfoud, "A comparison of parallel and sequential niching methods," in *Proc. 6th Int. Conf. Genet. Algorithms*, Jul. 1995, pp. 136–143.

[2] J. E. Vitela and O. Castanos, "A real-coded niching memtic algorithm for continuous multimodal function optimization," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2008, pp. 2170–2177.

[3] O. Mengsheol and D. Goldberg, "Probabilistic crowding: Deterministic crowding with probabilistic replacement," in *Proc. GECCO*, Jul. 1999, pp. 409–416.

[4] B. Sareni and L. Krahenbuhl, "Fitness sharing and niching methods revisited," *IEEE Trans. Evol. Comput.*, vol. 2, no. 3, pp. 97–106, Sep. 1998.

[5] G. Singh and K. Deb, "Comparison of multimodal optimization algorithms based on evolutionary algorithms," in *Proc. Genet. Evol. Comput. Conf.*, 2006, pp. 1305–1312.

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

[7] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proc. 2nd Int. Conf. Genet. Algorithms*, 1987, pp. 41–49.

[8] G. R. Harik, "Finding multimodal solutions using restricted tournament selection," in *Proc. 6th Int. Conf. Genet. Algorithms*, 1995, pp. 24–31.

[9] A. Petrowski, "A clearing procedure as a niching method for genetic algorithms," in *Proc. 3rd IEEE Congr. Evol. Comput.*, May 1996, pp. 798–803.

[10] L. T. Bui, J. Branke, and H. A. Abbass, "Multiobjective optimization for dynamic environments," in *Proc. IEEE Congr. Evol. Comput.*, Sep. 2005, pp. 2349–2356.

[11] D. Zaharie, "Density based clustering with crowding differential evolution," in *Proc. 7th Int. Symp. Symbolic Numeric Algorithms for Sci. Comput.*, 2005, pp. 343–352.

[12] K. Price, R. Storn, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Berlin, Germany: Springer, 2005.

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

[14] K. A. De Jong, "An analysis of the behavior of a class of genetic adaptive systems," Doctoral dissertation, Comput. Commun. Sci., Univ. Michigan, Ann Arbor, MI, 1975.

[15] S. Mahfoud, "Niching methods for genetic algorithms," Doctoral dissertation, Comput. Sci., Univ. Illinois, Urbana, IL, 1995.

[16] J. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: University of Michigan Press, 1975.

[17] A. Pétrowski, "A clearing procedure as a niching method for genetic algorithms," in *Proc. IEEE Int. Conf. Evol. Comput.*, May 1996, pp. 798–803.

[18] 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.

[19] A. Pétrowski. (1997). "An efficient hierarchical clustering technique for speciation," Instit. Natl. Telecommun., Evry, France, Tech. Rep. [Online]. Available: http://etna.int-evry.fr/ap

[20] D. Beasley, D. R. Bull, and R. R. Martin, "A sequential niche technique for multimodal function optimization," *Evol. Comput.*, vol. 1, no. 2, pp. 101–125, 1993.

[21] X. Yin and N. Germay, "A fast genetic algorithm with sharing scheme using cluster analysis methods in multi-modal function optimization," in *Proc. Int. Conf. Artif. Neural Nets Genet. Algorithms*, 1993, pp. 450–457.

[22] G. Dick and P. Whigham, "Spatially-structured sharing technique for multimodal problems," *J. Comput. Sci. Technol.*, vol. 23, no. 1, pp. 64–76, 2008.

[23] R. Storn and K. V. Price, "Differential evolution: A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optimization*, vol. 11, no. 4, pp. 341–359, 1995.

[24] V. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.

[25] K. Price, R. Storn, and J. Lampinen, *Differential Evolution*, vol. 143. Berlin, Germany: Springer, 2008.

[26] X. Q. Chen, Z. X. Hou, and J. X. Liu, "Multiobjective optimization with modified Pareto differential evolution," in *Proc. ICICTA*, 2008, pp. 90–95.

[27] K. Price, "An introduction to differential evolution," in *New Ideas in Optimization*, D. Corne, M. Dorigo, and F. Glover, Eds. New York: McGraw-Hill, 1999, pp. 79–108.

[28] 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.

[29] D. Ackley, "An empirical study of bit vector function optimization," in *Genetic Algorithms Simulated Annealing*. London, U.K.: Pitman, 1987, pp. 170–204.

[30] K. Deb, "Genetic algorithms in multimodal function optimization, the Clearing house for genetic algorithms," M.S. thesis and Rep. 89002, Dept. Eng. Math., Univ. Alabama, Tuscaloosa, AL, 1989.

[31] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. New York: Springer-Verlag, 1996.

[32] O. Shir and T. Back, "Niche radius adaptation in the CMS-ES niching algorithms," in *Proc. 9th Int. Conf. PPSN*, LNCS 4193. Reykjavik, Iceland: Springer, 2006, pp. 142–151.

[33] B. Y. Qu and P. N. Suganthan, "Novel multimodal problems and differential evolution with ensemble of restricted tournament selection," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2010, pp. 1–7.

[34] J. Gan and K. Warwick, "A variable radius niching technique for speciation in genetic algorithms," in *Proc. GECCO*, 2000, pp. 96–103.

[35] O. M. Shir, M. Emmerich, and T. Back, "Adaptive niche radii and niche shapes approaches for niching with the CMA-ES," *Evol. Comput.*, vol. 18, no. 1, pp. 97–126, 2010.

[36] C. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu, "Disburdening the species conservation evolutionary algorithm of arguing with radii," in *Proc. GECCO*, 2007, pp. 1420–1427.

[37] A. D. Cioppa, C. D. Stefano, and A. Marcelli, "Where are the niches? Dynamic fitness sharing," *IEEE Trans. Evol. Comput.*, vol. 11, no. 4, pp. 453–465, Aug. 2007.

[38] 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.

[39] D. Zaharie, "A multipopulation differential evolution algorithm for multimodal optimization," in *Proc. 10th MENDEL Int. Conf. Soft Comput.*, 2004, pp. 17–24.

[40] J. Ronkkonen and J. Lampinen, "On determining multiple global optima by differential evolution," in *Proc. Eurogen Evol. Deterministic Methods Design Optimization Control*, Jun. 2007, pp. 146–151.

[41] J. Ronkkonen, "Continuous multimodal global optimization with differential evolution-based methods," Ph.D. thesis, Dept. Information Technol., Lappeenranta Univ. Technol., Lappeenranta, Finland, 2009.

[42] Z. Hendershot, "A differential evolution algorithm for automatically discovering multiple global optima in multidimensional, discontinuous spaces," in *Proc. 15th Midwest Artif. Intell. Cognit. Sci. Conf.*, Apr. 2004, pp. 92–97.

[43] J. Rumpler and F. Moore, "Automatic selection of subpopulations and minimal spanning distances for improved numerical optimization," in *Proc. CEC*, vol. 1. 2001, pp. 38–43.

[44] B. Rigling and F. Moore, "Exploitation of subpopulations in evolutionary strategies for improved numerical optimization," in *Proc. 11th MAICS*, 1999, pp. 80–88.

[45] 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.

[46] J. Kennedy and R. Eberhart, *Swarm Intelligence*. San Francisco, CA: Morgan Kaufmann Academic Press, 2001.

[47] J. Kennedy and R. Mendes, "Topological structure and particle swarm performance," in *Proc. 4th CEC*, 2002, pp. 1671–1676.

[48] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: Simpler, maybe better," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 204–210, Jun. 2004.

[49] S. Das, A. Konar, U. K. Chakraborty, and A. Abraham, "Differential evolution with a neighborhood based mutation operator: A comparative study," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 526–553, Jun. 2009.

[50] U. K. Chakraborty, S. Das, and A. Konar, "Differential evolution with local neighborhood," in *Proc. IEEE CEC*, Sep. 2006, pp. 2042–2049.

[51] D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, "Parallel differential evolution," in *Proc. Congr. Evol. Comput.*, 2004, pp. 142–151.

[52] M. Weber, F. Neri, and V. Tirronen, "Distributed differential evolution with explorative-exploitative population families," in *Genetic Programming and Evolvable Machines*, vol. 10. Berlin, Germany: Springer, 2009, pp. 343–471.

[53] M. Weber, V. Tirronen, and F. Neri, "Scale factor inheritance mechanism in distributed differential evolution," in *Soft Computing: A Fusion of Foundations, Methodologies and Applications*, vol. 14. Berlin, Germany: Springer, 2010, pp. 1187–1297.

[54] M. G. Epitropakis, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, "Enhancing differential evolution utilizing proximity-based mutation operators," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 99–119, Feb. 2011.

[55] O. Mengsheol and D. Goldberg, "Probabilistic crowding: Deterministic crowding with probabilistic replacement," in *Proc. GECCO*, 1999, pp. 409–416.

[56] D. Parrott and X. Li, "A particle swarm model for tracking multiple peaks in a dynamic environment using speciation," in *Proc. IEEE CEC*, Jun. 2004, pp. 105–116.

[57] X. Li, J. Branke, and T. Blackwell, "Particle swarm with speciation and adaptation in a dynamic environment," in *Proc. GECCO*, 2006, pp. 51–58.

[58] R. Lung and D. Dumitrescu, "A collaborative model for tracking optima in dynamic environments," in *Proc. IEEE CEC*, Sep. 2007, pp. 564–567.

[59] R. Lung and D. Dumitrescu, "Evolutionary swarm cooperative optimization in dynamic environments," *Natural Comput.*, vol. 9, no. 1, pp. 83–94, Mar. 2010.

[60] E. L. Yu and P. N. Suganthan, "Evolutionary programming with ensemble of external memories for dynamic optimization," in *Proc. IEEE Congr. Evol. Comput.*, May 2009, pp. 431–438.

[61] L. Costa and P. Oliveira, "An adaptive sharing elitist evolution strategy for multiobjective optimization," *Evol. Comput.*, vol. 11, no. 4, pp. 417–438, 2003.

[62] J. Horn, N. Nafpliotis, and D. E. Goldberg, "A niched Pareto genetic algorithm for multiobjective optimization," in *Proc. IEEE World Congr. Comput. Intell.*, vol. 1. Jun. 1994, pp. 82–87.

[63] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA–II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[64] X. Li, "A non-dominated sorting particle swarm optimizer for multiobjective optimization," in *Proc. GECCO*, 2003, pp. 37–48.

[65] T. Hiroyasu, M. Miki, and S. Watanabe, "Distributed genetic algorithms with a new sharing approach in multiobjective optimization problems," in *Proc. CEC*, vol. 1. 1999, pp. 69–76.

[66] T. Y. Chen and Y. S. Hsu, "A multiobjective optimization solver using rank-niche evolution strategy," *Adv. Eng. Softw.*, vol. 37, no. 10, pp. 684–699, Oct. 2006.

[67] Y. Li, J. Zhou, H. Qin, Y. Lu, and J. Yang, "Adaptive niche multiobjective particle swarm optimization algorithm," in *Proc. 4th Int. Conf. Natural Comput.*, 2008, pp. 418–422.

[68] Q. Ling, G. Wu, and Q. Wang, "Restricted evolution based multimodal function optimization in holographic grating design," in *Proc. IEEE CEC*, Sep. 2005, pp. 789–794.

[69] D. Zaharie, "Density based clustering with crowding differential evolution," in *Proc. 7th Int. Symp. Symbolic Numeric Algorithms Sci. Comput.*, 2005, pp. 343–352.

[70] O. Nasraoui, E. Leon, and R. Krishnapuram, "Unsupervised niche clustering: Discovering and unknown number of clusters in noisy data sets," in *Evolutionary Computing in Data Mining*. Berlin, Germany: Springer-Verlag, 2005, pp. 157–186.
[71] Q. Ling, G. Wu, Z. Yang, and Q. Wang, "Robust optimal design under standard crowding differential evolution framework," in *Proc. 6th World Congr. Intell. Control Autom.*, 2006, pp. 3173–3177.
[72] P. S. Georgilakis, "Market-based transmission expansion planning by improved differential evolution," *Int. J. Electr. Power Energy Syst.*, vol. 32, no. 5, pp. 450–456, Jun. 2010.
[73] R. I. Lung, T. D. Mihoc, and D. Dumitrescu, "Nash equilibria detection for multi-player games," in *Proc. IEEE CEC*, Jul. 2010, pp. 1–5.
[74] 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.
[75] 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.
[76] E. Perez, F. Herrera, and C. Hernandez, "Finding multiple solutions in job shop scheduling by niching genetic algorithms," *J. Intell. Manuf.*, vol. 14, nos. 3–4, pp. 323–339, Jun. 2003.
[77] W. Sheng, A. Tucker, and X. Liu, "Clustering with niching genetic K-means algorithm," in *Proc. Genet. Evol. Comput.*, 2004, pp. 162–173.
[78] W. Sheng, X. Liu, and M. Fairhurst, "A niching memetic algorithm for simultaneous clustering and feature selection," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 7, pp. 868–879, Jul. 2008.
[79] J. L. Redondo, J. Fernandez, I. Garcia, and P. M. Ortigosa, "Solving the multiple competitive facilities location and design problem on the plane," *Evol. Comput.*, vol. 17, no. 1, pp. 21–53, 2009.
[80] M. Boughanem and L. Tamine, "A study on using genetic niching for query optimization in document retrieval," in *Proc. Adv. Inform. Retrieval*, 2002, pp. 93–100.
[81] Y. Li, "Using niche genetic algorithm to find fuzzy rules," in *Proc. Int. Symp. Web Inform. Syst. Applicat.*, 2009, pp. 64–67.
[82] G. Delvecchio, C. Lofrumento, F. Neri, and M. S. Labini, "A fast evolutionary-deterministic algorithm to study multimodal current fields under safety level constraints," *Int. J. Comput. Math. Electr. Electron. Eng.*, vol. 25, no. 3, pp. 599–608, 2006.
[83] J. Yao, N. Kharma, and P. Grogono, "Multipopulation genetic algorithm for robust and fast ellipse detection," *Pattern Anal. Applicat.*, vol. 8, no. 1, pp. 149–162, 2005.
[84] Y. K. Wang and K. C. Fan, "Applying genetic algorithms on pattern recognition: An analysis and survey," in *Proc. 13th Int. Conf. Pattern Recognit.*, vol. 2. 1996, pp. 740–744.
[85] B. Sareni and L. Krahenbuhl, "Fitness sharing and niching methods revisited," *IEEE Trans. Evol. Comput.*, vol. 2, no. 3, pp. 97–106, Sep. 1998.
[86] F. Neri and V. Tirronen, "Recent advances in differential evolution: A review and experimental analysis," *Artif. Intell. Rev.*, vol. 33, no. 1, pp. 61–106, Feb. 2010.
[87] J. Brest and M. S. Mauc̀ec, "Population size reduction for the differential evolution algorithm," *Appl. Intell.*, vol. 29, no. 3, pp. 228–247, 2008.
[88] B. Y. Qu and P. N. Suganthan, "Multiobjective evolutionary algorithms based on the summation of normalized objectives and diversified selection," *Inform. Sci.*, vol. 180, no. 17, pp. 3170–3181, 2010.
[89] E. L. Yu and P. N. Suganthan, "Ensemble of niching algorithms," *Inform. Sci.*, vol. 180, no. 15, pp. 2815–2833, Aug. 2010.
[90] S. Das, S. Maity, B.-Y. Qu, and P. N. Suganthan, "Real-parameter evolutionary multimodal optimization: A survey of the state-of-the-art," *Swarm Evol. Comput.*, vol. 1, no. 2, pp. 71–88, Jun. 2011.

B. Y. Qu received the B.E. degree with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, in 2007. Since 2008, he has been working toward the Ph.D. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University.

He is currently a Research Staff Member and Lecturer with the School of Electrical Engineering, Zhengzhou University, Zhengzhou, Henan, China. His current research interests include machine learning, neural networks, genetic and evolutionary algorithms, swarm intelligence, and multiobjective optimization.

P. N. Suganthan (S'91–M'92–SM'00) received the B.A. degree, the Postgraduate Certificate, and the M.A. degree in electrical and information engineering from the University of Cambridge, Cambridge, U.K., in 1990, 1992, and 1994, respectively, and the Ph.D. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.

He was a Pre-Doctoral Research Assistant with the Department of Electrical Engineering, University of Sydney, Sydney, Australia, from 1995 to 1996, and was a Lecturer with the Department of Computer Science and Electrical Engineering, University of Queensland, Brisbane, Australia, from 1996 to 1999. Since 1999, he has been with the School of Electrical and Electronic Engineering, Nanyang Technological University, where he was an Assistant Professor and is currently an Associate Professor. His current research interests include evolutionary computation, pattern recognition, multiobjective evolutionary algorithms, bioinformatics, applications of evolutionary computation, and neural networks.

Dr. Suganthan is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, INFORMATION SCIENCES, PATTERN RECOGNITION, and the *International Journal of Swarm Intelligence Research* journals. He is a Founding Co-Editor-in-Chief of *Swarm and Evolutionary Computation*, an Elsevier journal. His SaDE (April, 2009) paper won the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Award.

J. J. Liang received the B.Eng. degree from the Harbin Institute of Technology, Harbin, China, and the Ph.D. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.

She is currently an Associate Professor with the School of Electrical Engineering, Zhengzhou University, Zhengzhou, Henan, China, Her current research interests include evolutionary computation, swarm intelligence, multiobjective optimization, and applications of evolutionary computation.