Swarm and Evolutionary Computation 1 (2011) 71–88

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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

Survey paper

Real-parameter evolutionary multimodal optimization — A survey of the state-of-the-art

Swagatam Das [a], Sayan Maity [a], Bo-Yang Qu [b], P.N. Suganthan [b,*]

[a] *Department of Electronics and Telecommunication Engg., Jadavpur University, Kolkata 700 032, India*
[b] *School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore*

ARTICLE INFO

Article history:
Received 4 February 2011
Received in revised form
5 May 2011
Accepted 30 May 2011
Available online 22 June 2011

Keywords:
Multimodal optimization
Niching
Crowding
Speciation
Evolutionary algorithms
Particle swarm optimization

ABSTRACT

Multimodal optimization amounts to finding multiple global and local optima (as opposed to a single solution) of a function, so that the user can have a better knowledge about different optimal solutions in the search space and as and when needed, the current solution may be switched to another suitable one while still maintaining the optimal system performance. Evolutionary Algorithms (EAs), due to their population-based approaches, are able to detect multiple solutions within a population in a single simulation run and have a clear advantage over the classical optimization techniques, which need multiple restarts and multiple runs in the hope that a different solution may be discovered every run, with no guarantee however. Numerous evolutionary optimization techniques have been developed since late 1970s for locating multiple optima (global or local). These techniques are commonly referred to as "niching" methods. Niching can be incorporated into a standard EA to promote and maintain formation of multiple stable subpopulations within a single population, with an aim to locate multiple globally optimal or suboptimal solutions simultaneously. This article is the first of its kind to present a comprehensive review of the basic concepts related to real-parameter evolutionary multimodal optimization, a survey of the major niching techniques, a detailed account of the adaptation of EAs from diverse paradigms to tackle multimodal problems, benchmark problems and performance measures.

1. Introduction

In practical optimization problems, it is often desirable to simultaneously locate multiple global and local optima of a given objective function. For real-world problems due to physical (and/or cost) constraints, the best results cannot be realized always. Under such conditions, if multiple solutions (local and global) are known, the implementation can be quickly switched to another solution without much interrupting the optimal system performance. Multiple solutions could also be analyzed to discover hidden properties (or relationships) of the concerned functional landscape. Thus, as the name suggests, *multimodal optimization* refers to the task of finding multiple optimal solutions and not just one single optimum, as it is done in a typical optimization study. As a practical example consider the problem of locating the resonance points in a mechanical or electrical system [1]. If the fitness function gives the resonant amplitude of the system under particular conditions, one may be interested in detecting all resonant frequencies with amplitudes above a particular threshold and not simply the frequency of greatest resonance. Frequencies of large resonance need to be identified because the designer generally wishes to minimize or maximize all such resonances, depending on the application.

In a multimodal function landscape if the peaks are of equal value (height), convergence to every peak is desirable. Apart from knowing the best solutions, one may also be interested in knowing other optimal solutions that are present in the search space. If a point-by-point classical optimization approach is used for multimodal optimization, the approach must have to be applied several times, every time with the expectation of finding a different optimal solution. Evolutionary Algorithms (EAs) [1,2], due to their population-based search method, provide a natural advantage over classical optimization techniques. They maintain a population of possible solutions, which are processed every generation, and if the multiple solutions can be preserved over all these generations, then on termination of the algorithm, we will have multiple good solutions rather than only the best solution. Note that, this is against the natural tendency of EAs that will always converge to the best solution or a suboptimal solution (in a rugged, not so well-posed function). Detection and maintenance of multiple solutions are wherein the challenge of using EAs for multimodal optimization lies. *Niching* [3–5] is a generic term referred to as

* Corresponding author.
E-mail addresses: swagatamdas19@yahoo.co.in (S. Das), sayanmaity.10@gmail.com (S. Maity), E070088@ntu.edu.sg (B.-Y. Qu), epnsugan@ntu.edu.sg (P.N. Suganthan).

2210-6502/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.swevo.2011.05.005

the technique of finding and preserving multiple stable *niches*, or favorable parts of the solution space possibly around multiple solutions, so as to prevent convergence to a single solution.

Cavicchio's dissertation [6] was probably one of the first studies attempting to induce niching behavior in a GA in the form of *preselection*. He proposed three schemes for choosing an offspring generated from two parents depending on the fact that whether the offspring could outperform none of the parents, both of the parents, or only a single parent. The preselection schemes were later generalized by De Jong in another method called *crowding* [7] that was initially devised to preserve population diversity. Goldberg and Richardson [8], in their landmark paper of 1987, nicely showed how a niche-preserving technique can be introduced in a standard Genetic Algorithm (GA) and multiple optimal solutions can be obtained. Since that study, many researchers have suggested methodologies of introducing niche-preserving techniques so that, for each optimum solution, a niche gets formed in the population of an EA. Currently the most popular niching techniques used in conjunction with the evolutionary computation community include crowding [9], fitness sharing [10], restricted tournament selection [8], and speciation [11]. Besides multimodal problems, niching techniques are also frequently employed for solving multi-objective and dynamic optimization problems [12,13].

Although there exist a few significant survey articles on the application of EAs to multi-objective, constrained, and dynamic optimization problems [14,15], to the best of our knowledge, no such review article capturing the entire horizon of the research on real-parameter evolutionary multimodal optimization has so far been published. In this direction, recently Barerra and Coello Coello [16] came up with a review of the Particle Swarm Optimization (PSO) based methods developed for tackling the multimodal optimization problems. However, the article did not focus on other powerful EAs like Genetic Algorithms (GAs), Evolution Strategies (ESs), Differential Evolution (DE) etc. and also on the most prominent niching techniques devised to solve multimodal problems. In this paper we attempt to provide an extensive survey of the most representative evolutionary approaches developed so far for detecting multiple optima at a single run. We provide a detailed overview of the existing niching methods and then explore how powerful EAs like GA, ES, DE, PSO etc. could be adopted for multimodal problems in a comprehensive style. We also focus on how multimodal optimization could be performed with Clustering techniques, Memetic Algorithms (MAs) and Multi-objective Optimization (MO) approaches.

2. Basics of niching

For enabling an EA to properly search over multimodal fitness landscapes, researchers have turned to techniques that drive subsets of a population toward different areas in the search space and promote distributed convergence to multiple peaks simultaneously. These mechanisms, commonly referred to as niching methods, are designed to create and maintain diverse subpopulations of individuals within the global population. Literally in ecology, *niching* means the formation of distinct species exploiting different niches (resources) in the ecosystem. Niching algorithms can be categorized based on the way the niches are located. Four categories can be identified [17]:

- *Sequential niching* (or *temporal niching*) techniques locate niches over time. They find multiple niches iteratively/temporarily.
- *Parallel niching* (or *spatial niching*) methods locate all niches in parallel. They detect and maintain multiple niches simultaneously in a single population.
- *Quasi-sequential niching* techniques locate niches sequentially and the search for new niches continues while the already found niches are maintained in parallel.
- *Hierarchical niching* [18] is a continuous version of temporal niching together with spatial niching designed to overcome the limitation of spatial niching.

In [19], Mahfoud undertook a detailed comparison of parallel and sequential niching methods. Parallel Niching (PN) method follows a *parallel hillclimbing* approach, which more or less resembles a binary search technique. The hillclimbing method starts with a large step size and each population element continues to hillclimb until they can no longer improve. Then the step size is divided into half of the previous and the hillclimbing approach is again carried out. The process continues till it uses a predefined smallest possible step size 'ε'. The Sequential Niching (SN) [20] method is practically an extension of the iterating GAs that maintain the best solution of each run off-line. To avoid converging to particular optima more than once, whenever SN finds an optimum, it depresses the search space at all points that fall within some threshold radius, called niche radius [10] of the solution. It is not too easy to determine the stopping criterion of SN; generally after finding out all desired peaks the iterations are terminated. By applying these techniques on various multimodal problems, authors in [19] concluded that *parallel hillclimbing* is best to use for the easier problem and also works reasonably well for problems with intermediate complexity but fails for highly complex problems. On the other hand SN is weak on an easy problem and remains unable to solve harder problems either.

2.1. Existing niching techniques

Primarily niching techniques have been developed to reduce the effect of genetic drift resulting from the selection operator of the classical GAs. Thus they also aim at maintaining genetic diversity in the population and making GAs able to find multiple optima in parallel. A niching method must have to form and maintain multiple and diverse final solutions within the search and it should also be able to maintain these multiple solutions for a large enough number of iterations. In what follows, we discuss some of the most prominent niching techniques available in the existing literature:

2.1.1. Clearing

Petrowski [21,7] presented the clearing procedure that draws inspiration from the principle of sharing of limited resources to the best population of each niche and eliminate other individuals in the same niche unlike 'sharing' method (to be discussed in Section 2.1.3), where resources are shared among all individuals within the subpopulations. In [21] it has been shown that, clearing procedure has a lower complexity than that of the sharing method. The genetic drift due to selection noise is significantly reduced in the case of clearing and also the population size required, may be much smaller than it is needed for a sharing method.

A simplified form of the clearing procedure is presented below as a pseudo-code. '*P*' and '*Np*' are global variables and σ is the clearing radius; '*C*' being the capacity of each niche. '*n_Win*' is the number of winners of the subpopulation within the current niche. Population '*P*' can be considered as an array of '*Np*' individuals i.e. in other words the population size is'*Np*'. In this pseudo-code the functions used are:

Sort_Fitness(P): Sorts population '*P*'according to the individual fitness values in a descending order.

Fitness$P[i]$: Returns the fitness of the '*i*'th element of population '*P*'.

Distance($P[i]$, $P[j]$): Returns the distance between the 'i'th and 'j'th population of 'P'.

```
Pseudo-code for Clearing technique
function Clearing(σ, C) /* Function definition*/
begin
Sort_Fitness (P) /* Sort the population in descending or-
der according to their fitness*/
 for i = 0 to (Np − 1)
        if (Fitness(P[i]) > 0)
         n_Win = 1
         for j = (i + 1) to(Np − 1)
          if (Fitness(P[j]) > 0 AND Distance(P[i], P[j]) < σ)
            if n_Win < C
            n_Win = n_Win + 1
            else
            Fitness (P[j]) = 0
            endif
          endif
         endfor
        endif
  endfor
 end
```

2.1.2. Crowding

Crowding [7] is motivated by the analogy with the competition for limited resources among similar members of a natural population. To determine similarity, in crowding method a distance metric should be utilized in either genotypic or phenotypic form. In genotypic distance sharing, the distance function 'd' is simply the Hamming distance between two strings (where the strings are binary coded). In phenotypic distance sharing, the distance function 'd' is defined using some problem-specific knowledge of the phenotype, the most common choice for a phenotypic distance function being the Euclidean distance. De Jong used a simple method where the newly generated individuals replace the similar individuals. This method is quite similar to a GA except that only a fraction of the global population indicated by the percentage of *generation gap* (G) reproduces and dies in each generation. Crowding follows from the analogy that dissimilar individuals tend to reside in different niches such that they typically do not compete. The final result is that in a fixed-size population at equilibrium, new members of a particular species replace older members of that species to maintain preexisting diversity of a population. Unlike sharing methods, crowding methods do not assign individuals to peak fitness. Instead, the number of individuals assembling about a peak is largely determined by the size of that peak's basin of attraction under crossover. A random sample of CF individuals is taken from the population, where CF is called the *crowding factor*. For lower CF values, an offspring replaces another individual that may not be similar to the offspring resulting into replacement error. To overcome this problem, CF should be very large or equal to the number of individuals in the population. Because of a large number of replacement errors, the initial crowding of De Jong was shown to be of limited usefulness in multimodal function optimization [3,22].

2.1.2.1. Deterministic crowding. In [3,9,23] Mahfoud reviewed De Jong's [7] crowding factor technique and indicated its inability to maintain more than two peaks of a multimodal objective function due to replacement errors that result from genetic drift. Deterministic crowding [3,9,23] is a modification of the technique, first introduced by De Jong [7] and its objective is to maintain diverse population, eliminate parameter requirements, reduce replacement error, and restore selection pressure. Here, at first, the algorithm randomly selects two parents from the current population and performs crossover and mutation to generate two offspring. Then the children replace the nearest parent if they are of greater fitness. In case of a tie, parents are preferred. The following procedure should be performed $Np/2$ times (where Np is the population size):

Pseudo-code for deterministic crowding

1. Select two parents p_1, p_2 randomly with no replacement.
2. Perform a crossover between them yielding offspring c_1, c_2.
3. Apply mutation operator to generate c_1', c_2'.
4. if $[d(p_1, c_1') + d(p_2, c_2') \le d(p_1, c_2') + d(p_2, c_1')]$
   - if $f(c_1') \ge f(p_1)$ replace p_1 with c_1'
   - if $f(c_2') \ge f(p_2)$ replace p_2 with c_2'

   else
   - if $f(c_2') \ge f(p_1)$ replace p_1 with c_2'
   - if $f(c_1') \ge f(p_2)$ replace p_2 with c_1'

Thus, Deterministic Crowding (DC) results in two sets of tournaments: (*parent* 1 *against child* 1 and *parent* 2 *against child* 2) or (*parent* 1 *against child* 2 and *parent* 2 *against child* 1). The set of tournaments that yields the closest competitions is held. Similarity is computed using preferably the phenotypic distance.

2.1.2.2. Probabilistic crowding. The crowding techniques discussed so far, replaces the upfront selection pressure with selection pressure at the replacement stage through some form of localized tournaments between similar individuals. Since a deterministic tournament is used, such methods will always prefer individuals with higher fitness over individuals with lower fitness. This finally leads to a loss of niches, whenever the tournaments between global and local niches are played. To modify the deterministic nature of the algorithm and thus to provide a restorative pressure in such cases, Mengshoel [24] proposed a probabilistic crowding technique. Under this scheme, a probabilistic replacement rule was proposed that permitted individuals with higher fitness to win over individuals with lower fitness in proportion to their fitness. This allows a restorative pressure and prevents the loss of niches of lower fitness. This algorithm basically uses the deterministic crowding with a probabilistic replacement operator. In probabilistic crowding, two similar individuals X and Y compete through a probabilistic tournament where the probability of X winning the tournament is given by:

$$p(X) = \frac{f(X)}{f(X) + f(Y)}, \tag{1}$$

where f is the fitness function.

2.1.3. Sharing

Sharing method marks the first attempt to deal directly with the locations and preservation of multiple solutions among all the niching techniques. It was originally introduced by Goldberg [25, p. 164] and then improved by Goldberg and Richardson [6]. 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. Fitness sharing modifies the search space by reducing the payoff in densely populated regions. It decreases each individual's fitness by an amount nearly equal to the number of similar individuals in the population. Typically, the shared fitness f_i' of an individual i with fitness f_i is:

$$f_i' = \frac{f_i}{m_i}, \tag{2}$$

where m_i is known as niche count that measures the approximate number of members with whom the individual has to share its fitness. This niche count is measured by summing the sharing function over all individuals of the population i.e.

$$m_i = \sum_{j=1}^{Np} sh(d_{ij}), \tag{3}$$

where Np represents the population size and d_{ij} denotes the distance between the 'i'th individual and the 'j'th individual. Thus, the sharing function (sh) measures the similarity level between two population elements. It returns '1' if the individuals are identical, '0' if their distance is higher than a threshold of dissimilarity i.e. niche radius, and an intermediate value at intermediate level of dissimilarity. Here $sh(d_{ij})$ represents the sharing function which is the power function of the form:

$$sh(d_{ij}) = \begin{cases} 1 - \left(\dfrac{d_{ij}}{\sigma_{\text{share}}}\right)^{\alpha}, & d_{ij} < \sigma_{\text{share}} \\ 0, & \text{otherwise}, \end{cases} \tag{4}$$

where σ_{share} represents the threshold of dissimilarity (also the distance cut off or the niche radius) and α is the constant parameter which regulates the shape of the sharing function. α is commonly set to one with the resulting sharing function referred to as the triangular sharing function [25]. The distance d_{ij} between two individuals is characterized by a similarity metric based on either genotypic or phenotypic similarity. Deb and Goldberg [22] have shown that sharing based on phenotypic similarity may give slightly better results than sharing with genotypic similarity. Goldberg and Wang [26] proposed an alternative sharing scheme known as the coevolutionary sharing. It surpasses the limitations of fixed sharing schemes by allowing the locations and niche radius to adapt to complex landscapes, as well as permitting a better distribution of solutions in problems with many poorly spaced optima. The principle behind the implementation of coevolutionary sharing is the model of *monopolistic competition* in economics, which utilizes a set of two populations - a population of businessmen and a population of customers. Here the locations of the businessmen stand for niche locations and the locations of customers are analogous to solutions. Thus individuals in both of the populations search for maximizing their individual interests thereby evolving suitably spaced niches consisting of the mostly fit individuals.

2.1.4. Restricted tournament selection

Restricted Tournament Selection (RTS), introduced by Harik [27,8], is a modified tournament selection for multimodal optimization. RTS scheme allows the GAs to choose which individuals will be replaced to insert a pair of elements. As in deterministic crowding, RTS randomly selects two parents from the population and yields two offspring by applying crossover and mutation operators. After that, for each offspring, the algorithm chooses a random sample of 'w' (window size analogous to *CF* in Crowding) individuals from the population and determines which one is the nearest to the offspring, by applying the similarity distance measure that can be either Euclidean (for real variables) or Hamming (for binary coded variables). The nearest member within the 'w' individuals will compete with the offspring to determine the one with higher fitness. If the offspring wins, it is allowed to enter the population by replacing its opponent. This type of tournament will restrict an element of the population from competing with others that are too dissimilar w.r.t. it. Harik [8] tested his model on several multimodal real-world problems with the number of peaks varying from 5 to 32. The algorithm could also maintain individuals at all the peaks, though some of the peaks gradually lost a number of elements. It was able to maintain all the global optima in the multimodal problems. Roy and Parmee presented an Adaptive RTS (ARTS) [28] integrated with a GA for multimodal optimization. ARTS differ from RTS with respect to the fact that the former requires no prior knowledge about the modality of the fitness landscape to distribute the final population on different peaks.

2.1.5. Clustering

Yin [29] proposed a clustering-based niching scheme to help the formation of the niches and avoid the need for estimation of σ_{share} needed in sharing technique. The fitness is calculated based on the distance $d_{i,c}$ between the ith individual and its niche centroid. This significantly reduces the time complexity. The formation of the niches is based on the adaptive Macqueen's K-means clustering algorithm. The algorithm begins with a fixed number (k) of seed points taken as the best k individuals. Using a minimum allowable distance $d_{\min}$ between niche centroids, a few clusters are formed from the seed points. The remaining population members are then added to these existing clusters or are used to form new clusters based on $d_{\min}$ and $d_{\max}$. These computations are performed in each generation. The final fitness of an individual is calculated using the relation:

$$F_i = \frac{f_i}{n_c\left(1 - \left(d_{i,c}/2d_{\max}\right)^{\alpha}\right)}, \tag{5}$$

where n_c is the number of individuals in the niche containing the individual i, $d_{\max}$ is the maximum distance allowed between an individual and its niche centroid, and α is a constant.

2.1.6. Species conservation

Li et al. [30,31] introduced a recent technique called *species conservation* for evolving parallel subpopulations that achieves niching by exploiting the notion of species. The technique is based on the concept of separating the population into several species according to their similarity. Each of these species is built around a dominating individual called the species seed. Species seeds found in the current generation are maintained (conserved) by carrying them into the next generation. According to Li et al. [30] this technique has proven itself to be very effective in finding multiple solutions of multimodal optimization problems. The GA using species conservation (SCGA) makes no distinction between genotypes and phenotypes. Thus, the genetic operators are applied directly to individuals represented by arrays of real numbers and this increases simplicity of the technique. To define a species as well as the operation of the SCGA, we have to define a parameter known as species distance, which we denote by σ_s. The species distance specifies the upper bound on the distance between two individuals for which they are considered to be similar.

Li et al. [30] also proposed that the species distance should be used to determine which individuals are worth preserving from one generation to the next. A species is defined with respect to a finite population $P_N = \{x_1, x_2, \ldots, x_N\}$. The species in a population is partitioned around certain prevailing individuals known as *dominating individuals* or *species seeds*. A species is a subset, in which the distance between any two individuals is less than the species distance; this does not mean that any two individuals satisfying the condition that the distance between them is less than the species distance belong to the same species. To find out the individuals that will be maintained in the next generation, we need to partition the current population into a set of dominated species and determine the dominating individual in each of these species. Here X_s denotes the set of species seeds found in a certain generation. If X_s does not contain any seed that is closer than half the species distance i.e. $(\sigma_s/2)$ to the individual considered, then the individual will be added to X_s. When all the species have been found, the new population is generated by applying the usual genetic operators: selection, crossover, and mutation. Since some species may not survive following these operations, we copy them into the new population and thus enable them to survive. In the species conservation method, after the creation of the new population in the next generation a search for individuals belonging to the same species is performed. The species seeds of the previous generation are replaced with these individuals of the same species in the next generation if they are worst than those of the previous generation. If no individuals of the same species are found in the next generation, then the worst unmarked individual of the new population will be replaced by the species seed. However, since the species seed are being taken from the previous generation, the number of species is always less than the population size.

2.1.7. Niching memetic algorithm

Sequential Niching Memetic Algorithm (SNMA) [32] is the extension of *sequential niching technique* of Beasley et al. [20] to locate multiple optima and it introduces a local search to improve the accuracy. SNMA incorporates a gradient-based local search process that makes use of a derating function along with niching and clearing techniques. It penalizes individuals that may stay into regions near previously located optima to promote the occupation of different niches in the function to be optimized. The SNMA method requires the use of a niche radius but unlike other algorithms, in which it is difficult to identify the appropriate value of this radius or the species distance, the performance results are not highly sensitive to the values of this very parameter. This is an advantage in problems where the number and distribution of the optima are unknown.

It is recognized by the EC community that the combination of GA with local search, commonly known as Memetic Algorithms (MAs) [32], greatly improves the accuracy of EAs in locating the optimal solutions for function optimization problems. MAs concentrate on locating a promising area in the search space and then use different local search techniques to strengthen the search within that region. In SNMA, first we have to initialize population of the sequence with randomly generated individuals. Here the total number of optimal solutions, both local and global, is given by J_{Total}. A new generation is obtained after the application of the genetic operators, i.e. evaluation, selection, reproduction, and mutation, to all members in the current population and like the GAs, the population size is kept constant from generation to generation. At each generation every individual in the population moves toward its nearest peak following a hillclimbing gradient-based algorithm. If at some point, during this process, an individual leaves the pre-specified search space, then the corresponding variable takes the boundary value assigned. It has been assumed [33] that the population consists of M individuals, and expects J_{Total} optimal solutions within the search space. If J optimal solutions have been already located (with $J < J_{\text{Total}}$) the distances d_m from each individual in the population to their nearest optimal solution are determined. These distances together with the niche radius 'R' are used to assign an effective fitness function to each individual in the population. The niche radius R is identified with the width of the inverted Gaussian function. Thus, the effective fitness of an individual will be closer to zero as it will approach any of the previously found optima.

The individuals in the population are now ordered according to their effective fitness in decreasing order from $m = 1$ to M. Then a roulette wheel *selection* is used according to a probability of survival in which *clearing* is introduced. This *selection* operator assigns larger survival probabilities to individuals with larger effective fitness; however, the probabilities assigned are not proportional to the effective fitness of the individuals, instead, they decrease linearly (except for *clearing*) following the order position. Because of *clearing* the SNMA has the important characteristic that it eliminates individuals lying within a niche radius from any of the previously located optima promoting the occupation of niches not yet found by the MA. *Recombination* is implemented through the parent centric PBX-crossover [34]. *Mutation* is applied with probability P_m to all members of the population.

The performance of the SNMA is not highly sensitive to the selection of the niche radius R, a feature advantageous especially when the number and distribution of the optima are unknown. An advantage of SNMA over other algorithms is that it does not need to maintain permanent populations around each optimal found; it is only necessary to store the location of these peaks.

Table 1
Complexity of various niching techniques.

Algorithm	Complexity order
Clearing	$O(C \times Np^2)$
Deterministic crowding	$O(Np)$
Probabilistic crowding	$O(Np)$
Sharing	$O(Np^2)$
RTS	$O(Np \times w)$
Clustering	$O(C \times N_c \times Np)$
SCGA	Best case—$O(Np)$
	Worst case—$O(Np^2)$
SNMA	$O(Np)$

2.2. Complexity of different niching techniques

The approximate order of the computational complexities of different niching techniques [5] has been presented in Table 1. For clearing technique, the computational complexity is $O(c.Np^2)$ where 'c' is the number of subpopulations and Np is the population size. When Np increases, c tends toward a limit deter the fitness landscape, σ and C. These parameters should be chosen in such a way that the limit of c is of the same order of the desirable peak number. For Deterministic Crowding and Probabilistic Crowding the computational complexity is $O(Np)$. For the sharing approach, the complexity is $O(Np^2)$ that comes from calculating the niche count value. The complexity for Restricted Tournament Selection is $O(Np \times w)$ where 'w' is window size (i.e. the niching parameter in RTS) since for every individual the distance is computed 'w' times. In case of clustering algorithm, if N_c is the number of clusters, C is a capacity of each niche and Np is population size, then its computational complexity can be expressed as $O(C \times N_c \times Np)$. For the best case, the complexity of SCGA is $O(Np)$, and it is $O(Np^2)$ for the worst case. The complexity of SNMA is computed as of the order of $O(Np)$.

2.3. Localized niching

In the previous sections we discussed several techniques for promoting speciation within a population. While each technique is different, they can all be categorized into two major categories: techniques that promote speciation through genetic information and those that use geographic isolation to induce niches. A different approach was taken in [35] for the use of genetic-based speciation techniques within the limits of a spatially structured population. This concept mainly follows from Spatially Structured EAs (SSEAs) [36]. SSEAs impose a spatial structure on the overall population in an EA. SSEAs are often implemented in one, two or three dimensional space. Each individual in the global population occupies a different location in the search space. Subpopulations are formed by collecting the individuals from closely related locations known as 'deme'. Genetic operations like selection and crossover are limited to within the deme. Thus, an individual can be included in more than one deme. This enables communication of genetic information throughout the global population. In [37] Spears proposed a technique that integrated both a sharing-like niching method with a ring topology where an improvement in performance on some problems have observed. This gave birth to the idea of hybridization of two different niching techniques. The global, genetic-based niching techniques need suitably large population of individuals to form reasonable stable subpopulations. Conversely, SSEAs require small, tightly clustered demes in order to construct species within the population. Hybridizing individual-based niching with SSEAs introduces the following benefits:

- *Reduced complexity,* global genetics-based niching carry on a large numbers of comparisons between individuals. In worst cases, fitness sharing requires each individual to be compared with every other population member, which results in an computational complexity of $O(Np^2)$ here by giving a small enough deme size d, we can easily reduce the complexity to $O(d.Np)$.
- *Parallel implementation,* Usually SSEAs are considered as natural candidates for implementation on parallel machines for its fine-grained nature of space, that may results in significant reductions in physical execution time.
- *Behavioral changes*: Author in [35] hopes that by changing the size and shape of subpopulations presented to the genetic-based niching methods will obviously alter the system's behavior that results reduction in computation required.

Local Clearing (LC), [38] is a particular realization of this localized niching approach, which takes the traditionally global operation of clearing and applies it to each location in space. The particular spatial structure, which is used by LC is the ring topology. The most visible modification of LC over global clearing is in the application of clearing; in place of a single instance of clearing being applied to the whole population, LC implement clearing to every deme at every location in space. This introduces a location-dependent component to clearing. An individual participates in multiple demes and must go through the clearing process for each deme. Another extension of this localized niching has been adapted on elitism.

3. Genetic Algorithms (GAs) for multimodal optimization

Genetic Algorithms (GAs) [1,25] are adaptive, randomized search techniques founded on the simulation of the Darwinian evolution and natural genetics. They are efficient, adaptive and robust search processes, producing near-optimal solutions and offer a large amount of implicit parallelism. In our ecosystem natural evolutionary process basically maintains a variety of species, each occupying a different ecological niche, whereas classical GAs rapidly push the artificial population toward convergence i.e., all individual population soon gather and become more or less identical. Even when multiple optima exist in a problem, classical GAs used to locate only one optimum.

Niching methods are embedded in a GA [3,39] to make it capable of detecting multiple optimal solutions by using a single population. The basic motivation of including niching methods was, in fact, to promote diversity in classical GAs. Diversity serves two purposes in a GA, first one is to delay the convergence in order to vigorously explore the search space so that a better single solution can be located and the second one is to finally locate multiple solutions. In [3] it has been concluded that three important factors that reasonably take part in the loss of diversity are *selection pressure*, *selection noise*, and *operator disruption*. Four niching techniques were tested in [3] with various difficulty levels on GA. Parallel hillclimbing is the best for the easiest problem and has some success on the problems with intermediate difficulty which often fails on complex optimization problem. Sequential niching is obviously the weakest method to handle most of the difficult multimodal optimization problems available in the literature. Sharing and deterministic crowding give fair enough result on several kinds of test functions with different difficulty levels. However, the sharing technique has a limitation over those optimization problems containing a number of extraneous optima that are of similar fitness of the desirable peak. Crossover probability should be less than one for sharing, however, some minimum probability is required to maintain sufficient exploration of the search space. Deterministic crowding faces serious trouble to maintain local optima that lie on the crossover path to global optima. In general the niching methods having exponential to infinite drift time with respect to the population size are preferred. Ursem proposed a Multinational Evolutionary Algorithm (MEA) [40] for detecting global and local optima on a function landscape. The method tends to adapt itself to the problem by catching some topological features of the fitness landscape under consideration. The main concept is to employ the topological information for grouping the whole population into subpopulations each of which will cover a part of the fitness landscape. Ursem further extended the work in [41] by proposing Multinational GAs (MGAs) to tackle multimodal optimization problems in dynamic environments. MGA is basically a self-organizing GA that partitions the population into subpopulations based on a technique of detecting valleys in the landscape. Walter et al. applied a Multi-Niche crowding GA [42] to find multiple peaks in a multimodal dynamic landscape where the locations and heights of peaks changes with time.

Yang et al. proposed a density clustering-based niching method [43] in 2005. In this method, to prevent the loss of diversity, the global selection pressure within a single population is replaced by local selection of a multipopulation strategy. The subpopulations representing species specialized on niches are dynamically identified using density-based clustering algorithm on a primordial population. The algorithm also includes a method for automatically calculating the clustering threshold.

Recently Yao et al. presented a Bi-objective Multipopulation GA (BMPGA) [44] for locating all global and local optima on a real–valued, differentiable, multimodal function. BMPGA is distinguished by its use of two separate but complementary fitness objectives designed to enhance the diversity of the overall population and exploration of the search space – features that make it able to overcome various drawbacks such as inferior performance on irregular multimodal surfaces [40,45], the inability to preserve species as well as the reliance of the method on particular landscapes and parameter settings [5,40,45]. The first objective is the objective function itself i.e. $f(\vec{X})$ and the second objective is formed by taking an average over the D components of the function gradient for a D-dimensional function as:

$$g(\vec{X}) = \frac{\sum_{i=1}^{D} \frac{\partial f}{\partial x_i}}{D}. \tag{6}$$

The algorithm includes a multipopulation scheme that promotes diversity and speciation within the population and helps in undertaking a detailed search of areas of potential optima. It performs clustering for identifying reliable and robust subpopulations based on three major operations: migration, splitting, and merging. BMPGA is shown to be better than other GA variants over a six function test-suite in terms of overall effectiveness, applicability, and reliability.

4. Particle swarm optimization for multimodal optimization

The concept of particle swarm, although initially introduced for simulating social behaviors commonly observed in the animal kingdom, has become very popular these days, as an efficient means for intelligent search and optimization. Since its advent in 1995, the Particle Swarm Optimization (PSO) [46,47] algorithm has attracted the attention of a lot of researchers all over the world resulting into a huge number of variants of the basic algorithm as well as many parameter selection/control strategies, comprehensive surveys of which can be found in [48,49]. In PSO, the particles are conceptual mathematical entities, which accelerate simultaneously along two directions – the best positions of the search space individually experienced by each of them at some point of time and the globally best position found by a neighborhood (geographical or social) of the current particle so far. Thus the particles have a tendency to fly toward the better and better regions of the search space over time, which results in the fast convergence of the search. PSO requires no gradient information of the function to be optimized is very easy to implement in any standard programming language and uses only primitive mathematical operators throughout.

For years, PSO has remained a favorite choice of the researchers working on multimodal optimization problems. Susana et al. introduced a non-uniform mutation operator along with PSO [50] that has been shown to significantly improve PSO's performance for multimodal function optimization. Based on the principle that mutation is a powerful diversity maintenance mechanism, in [50] two simple neighborhood-based PSO model have been hybridized using a non-uniform mutation operator and the results have been compared with standard PSO model over multimodal functions to prove its efficiency. Parsopoulos et al. [51] used the objective function "stretching" as a sequential PSO niching technique, similar to that of Beasley et al. [20]. Once the PSO algorithm has identified a local maximum $f(\vec{X}^*)$, (through comparing particle objective function values to a minimum threshold value) the objective function is *stretched*, such that for each point $\vec{X}$, if $f(\vec{X}) > f(\vec{X}^*)$, the point remains unaltered. But all other points, for which $f(\vec{X}) \leq f(\vec{X}^*)$ holds, are stretched so that $\vec{X}^*$becomes a local minimum. All particles are then repositioned randomly. Thus, a potentially good solution is isolated once it is found and then the fitness landscape is "stretched" to keep other particles away from this area of the search space [52,53]. Parsopoulos et al. [53] also introduced a threshold value ε such that when $f(\vec{X}_i) < \varepsilon$ for the ith particle, it is removed from the swarm and labeled as one of the potential global solution. If the removed particle's fitness is not up to the desired level, the solution can be refined by searching the surrounding function landscape with the addition of more particles. The objective function's landscape is then stretched to keep other particles from exploring this area in the search space. Van den Bergh points out that the stretching technique may alter the search space by introducing false maxima, preventing PSO from discovering all possible solutions [54]. Also the stretching function requires several new parameters to be specified and that is disadvantageous from a practical point of view.

In 2002, Britts et al. [55] proposed NichePSO where multiple subswarms are grown from an initial swarm by monitoring the fitness of individual particles. The subswarms can merge together or absorb particles from the main swarm. NichePSO tracks the variance of a particle over generations for monitoring its fitness. If the change in a particle's fitness over iterations is not very high, a subswarm is created with the particle's closest neighbor. It also utilizes the concept of Guaranteed Convergence PSO (GCPSO) algorithm [54,56] to evolve each subswarm. The success of NichePSO depends on the proper initial distribution of particles throughout the search space. To ensure uniform distribution, *Faure*-sequences [57] were used to initialize particle positions in the search space. The main swarm is trained using the *cognition only* model [58] as shown in Eq. (7).

$$v_{i,j}(t+1) = wv_{i,j}(t) + c_1 r_{1,j}(t)(y_{i,j}(t) - x_{i,j}(t)). \quad (7)$$

In this model, only a conscience factor, in the form of a personal best, is considered when updating particle positions. Therefore no social information in the form of a *global best* solution will influence position updates. Engelbrecht and Loggerenberg [59] pointed out that in NichePSO, the subswarm merging technique and the particle absorbing strategies are premature and they restrict exploration in the main swarm. The authors proposed a few novel merging (*Directional Based Merging, Scatter Merging*) and absorption techniques (*Directional Based Absorption, Euclidean Diversity Absorption*) and demonstrated that the fine tuning of these processes improves the performance of the NichePSO algorithm on multimodal functions of lower dimensionality. In [60], Brits et al. proposed an *nbest* PSO model, where the neighborhood best of a particle is determined by taking the average of the positions of all particles in its neighborhood. Based on the Euclidean distance among the particles, the neighborhood of a particle can be defined by its k closest particles, k being a user defined parameter.

Li proposed the Speciation-based PSO (SPSO) [61], by adopting the idea of determining species and the dominant particles in these species from [31]. In SPSO [61,62] each species and its corresponding species seed (i.e. the dominant particle) form a separate subpopulation, which can evolve as a particle swarm itself. As species are adaptively formed around different optima, over successive iterations, it is expected that multiple global optima will be detected in parallel. SPSO requires a niching radius to be specified in order to define the size of a species (niche). This parameter may depend on the distance among the peaks and their orientation in the fitness landscape and such information are not always available beforehand for a practical multimodal problem. Bird and Li proposed two methods based on population statistics [63] and a time-based convergence measure [64] to enhance the robustness of the SPSO algorithm to the niche radius value. Nickabadi et al. proposed a Dynamic Niching PSO (DNPSO) [65] technique where instead of determining the radius of the subswarm, the number of particles belonging to each niche is restricted. In DNPSO the particles are divided into small subswarms and a group of free particles is also maintained. The subswarms are created and destroyed dynamically in each iteration. A unique feature of DNPSO is the use of free particles. The group of free particles are very successful both in exploration and exploitation of the search space. Dynamicity of the subswarm enables them to absorb new particles as soon as they found new local or global optima.

Li proposed a PSO-based on Fitness-Euclidean distance Ratio (FER-PSO) [66] for multimodal optimization. In FER-PSO, a *memory-swarm* is formed by utilizing the personal bests of the particles. It provides a stable network for retaining the best points found so far by the population while the current positions of particles act as parts of an *explorer-swarm* to explore large areas of the search space. These stored solutions in the *memory-swarm* are further improved by moving toward its "fittest and closest" neighbors, recognized by calculating a particle's FER value. The FER value is calculated based on the ratio of the fitness difference and the Euclidean distance between a solution's personal best and other personal best of the solution's in the population, scaled by a factor (α) so that neither the fitness nor the Euclidean distance becomes too dominating. With a sufficiently large population size, using FER-PSO can reliably locate all the global optima as the search particles naturally create niches around the global optima in the search space. Although FER-PSO does not require the specification of niching parameters, it introduces a new parameter α, which needs to be determined from the search range of each variable. Schoeman and Engelbrecht presented a vector-based PSO [67,68] that depends on a niche radius defined by the distance between the current particle and the closest particle that moves in opposite direction. However, as a downside of the algorithm the distance calculation can be expensive, as every particle has to be compared with all others.

Alamil et al. presented a fuzzy clustering-based PSO [69] that does not require any prior information about the niche radius and number of optima. The basic idea of this technique is to maintain and promote the formation of the parallel subswarms using fuzzy clustering technique. Since the cluster radii are dynamically adapted, a fine local tuning is used to improve the solution during the evolution of the process. Passaro and Starita [70] used k-means clustering technique in conjunction with PSO for identifying niches within the swarm. The PSO with Craziness and hillclimbing (CPSO) algorithm [71] uses a random walk component and a hill climber to enhance the exploration and exploitation capabilities of PSO, respectively. The basic CPSO technique consist of two phases is used: *sampling* and *acceptance*. As a sampling technique either a random walk (craziness) or the PSO algorithm itself is invoked, depending on the mode of operation. In acceptance strategy, only improving moves of the particles are admitted. CPSO with its enhanced exploration and exploitation capabilities based on craziness and hillclimbing has a good performance especially in locating multiple global optima.

Recently Li used PSO with ring topology as a parameter-free niching algorithm [72]. For classical niching techniques the performance is very often dependent on niching parameters, a priori knowledge of which may not be available while solving a real-world multimodal problem. A classical niching algorithm, depending on a fixed niche radius value to determine an individual's membership in a niche, will in general face difficulty while working on various kinds of multimodal fitness landscapes. To locate and maintain all the global and local peaks, a niching EA would have to set its niche radius extremely small so that the closest two peaks can be distinguished. However, doing so would form several small niches, with possibly too few individuals in each of them. Thus, these niches tend to prematurely converge. On the other hand, if the niche radius is set too large, peaks with a distance between them smaller than this value will not be distinguished. In short, it is likely that there is no generalized optimal value for the niche radius parameter. In [72] the author demonstrated that, given a reasonably large population uniformly distributed in the search space, the *lbest* PSOs with ring topology can form stable niches across different local neighborhoods, eventually converging to multiple global/local optima in a distributed fashion. The complexity of these niching methods was shown to be O(Np), Np being the population size. A detailed survey has been done by Li [73] where some recent research undertaken to solve multimodal optimization problems by using PSO can be found.

In recent time Julio et al. introduced a concept [74] of configuring a simple and configurable set of test functions to assess the performance of PSO in multimodal search space. The authors designed scalable test functions with several local optima, but only one global optimum by using linear transformation and function composition into homogeneous coordinates. According to [74] test functions developed using this technique overcome the regularities such as symmetry, uniform location of the optima, exponential increase in the number of global optima with respect to the increase in the number of decision variables etc. which are most commonly found in the multimodal test functions currently available in the literature.

5. Evolution strategies for multimodal optimization

Evolution Strategies (ESs) [75,76] are nature-inspired optimization techniques built around the concept of the *evolution of evolution*—practically biological processes are optimized by evolution and on the other hand evolution is itself a biological process, thus it may be conclude that evolution optimizes itself. ES emphasizes on the phenotypic behaviors of individuals or search agents that constitute a population. Each individual consists of a set of decision (search) variables and strategy parameters. Evolution then amounts to changing both the decision variables and strategy parameters and the evolution of decision variables is controlled by the strategy parameters. As far as real-valued search spaces are concerned, mutation is normally performed by adding a normally distributed random value to each vector component. The step size or mutation strength (i.e. the standard deviation of the normal distribution) is often governed by self-adaptation. Individual step sizes for each coordinate or correlations between coordinates are either governed by self-adaptation or by covariance matrix adaptation (as done in CMA-ES [77]). The canonical versions of the ES are denoted by $(\mu/\rho, \lambda) - ES$ and $(\mu/\rho + \lambda) - ES$ respectively. Here μ denotes the number of parents, $\rho \leq \mu$ the mixing number (i.e., the number of parents involved in the procreation of an offspring), and λ the number of offspring. The parents are *deterministically* selected (i.e., deterministic survivor selection) from the (multi-)set of either the offspring, referred to as *comma-selection* ($\mu < \lambda$ must hold), or both the parents and offspring, referred to as *plus-selection*.

Shir and Bäck proposed the ES dynamic niching algorithm [78,79] in which for the first time a niching technique was incorporated within the framework of ES. The algorithm first mutates the population members, using a single step size (that is self-adapted as for traditional ESs) per individual. Fitness is then evaluated. Next the various fitness-peaks are identified dynamically by applying the Dynamic Peak Identification (DPI) algorithm [80]. The algorithm needs an estimate of the niche radius ρ based on which all the individuals are assigned to those peaks (i.e. they are engaged in populating those niches). At this point a mating restriction scheme is applied to allow competitive mating only within the niches: every niche can produce a defined number of offspring, following a fixed mating resources concept. In this manner the best niche is prevented from taking over the population's resources and flood the next generation with its offspring. Considering a uniform distribution of resources, each niche is allowed to have $\tilde{\mu}$ parents and produce $\tilde{\lambda}$ offspring in every generation, such that $\left(\tilde{\lambda} = \frac{\lambda}{q}\right)$ and $\left(\tilde{\mu} = \frac{\mu}{q}\right)$, where q is the expected number of niches. The procedure of creating $\tilde{\lambda}$ offspring consist of following steps: first using a line breeding mechanism [80] to choose the two parents—one by applying the tournament selection and the other would be the best individual in the niche except the first one. Then a standard recombination technique is applied. Finally a predefined number of individuals (η) from the offspring and the best individuals from the parent ($\delta = \tilde{\mu} - \eta$) population are chosen ($\tilde{\mu}$) for the next iteration of dynamic peak identification algorithm. A disadvantageous feature of the algorithm is that it needs an estimate of the number of peaks and a lower bound on the distance among the peaks, both of which may not be available beforehand for a practical problem.

In [81], the authors used CMA-ES (Covariance Matrix Adaptation based ES) in conjunction with the dynamic niching method proposed in [79]. The CMA-ES is composed of two adaptation phases: a Cumulative Steps Adaptation (CSA) mechanism, which is based upon the path length control, as well as the actual covariance matrix adaptation mechanism, which is based upon the evolution path. The fundamental property of this method is the exploitation of information obtained from previous successful mutation operations. Given an initial search point $\vec{X}^0$offspring are sampled from it by applying the mutation operator. The best search point out of those λ offspring is chosen to become the parent of the next generation. A candidate solution at generation g is mutated as:

$$\vec{X}^{g+1} = \vec{X}^{g} + \delta \cdot \mathbf{B} \cdot \vec{z}, \tag{8}$$

where δ is the adaptive global step size and $\vec{z}$ is a vector of random variables drawn from a *multivariate normal distribution*. The matrix **B** is the key element of this process and it is composed of the eigenvectors of the covariance matrix with the appropriate scaling of the eigenvalues defining the distribution of a sequence of successful mutation points. The CMA-ES based dynamic niching algorithm performed better than the classical ES-based niching especially on the low-dimensional problems. Shir et al. demonstrated the success of the ES-based niching algorithms on a challenging problem of quantum control concerning the *femto second laser pulse shaping* [82]. In that application, the niching technique was shown to be clearly qualitatively superior in comparison to multiple restart runs with a single population, in locating highly-fit unique optima that had not been obtained otherwise, and represented different conceptual designs. The authors took a radius based approach where choice of the radius value was based on theoretical considerations about the application problem.

In order to circumvent the problem of specifying a fixed niche radius, Shir and Bäck extended their CMA-ES based niching algorithm in [83,84] by allowing each individual to adapt its own niche radius value along with the adaptation of other strategy parameters. For this, two new schemes were implemented in [84]. The first technique exploits the cumulative step size adaptation of CMA-ES and couples the individual niche radius to it and the second approach employs the *Mahalanobis distance metric* with the covariance matrix mechanism for the distance calculation and for obtaining niches with more complex geometrical shapes. Although the adaptation of individual niching radii provided better performances than the niching methods using a fixed niche radius, the adaptation itself introduced some new tunable parameters e.g. the number of expected optima and learning coefficients.

Shir and Bäck proposed a framework for niching techniques based on the derandomized Evolution Strategies (ES) in [85,86]. The authors surveyed five variants of derandomized ESs, based on the fixed niche radius approach. The core mechanisms range from the very first derandomized approach [87] to self-adaptation of ES to the sophisticated $(1 + /, \lambda)$ covariance matrix adaptation. The algorithms were applied to artificial as well as real-world multimodal continuous landscapes, of different levels of difficulty and various dimensions, and compared on the basis of the Maximum Peak Ratio (MPR) performance analysis tool.

6. Differential evolution for multimodal optimization

Differential Evolution (DE) [88–90] is arguably one of the most powerful stochastic real-parameter optimization algorithms of current interest. DE has been frequently adopted to tackle multi-objective, constrained, dynamic, large scale, and multimodal optimization problems and the resulting variants have been achieving top ranks in various competitions held under the IEEE CEC (Congress on Evolutionary Computation) conference series (e.g. see http://www3.ntu.edu.sg/home/epnsugan/index_files/cec-benchmarking.htm). In DE community, the individual trial solutions (which constitute a population) are called *parameter vectors* or *genomes*. DE operates through the same computational steps as employed by a standard EA. However, unlike traditional EAs, DE employs difference of the parameter vectors to explore the objective function landscape. Like other population-based search techniques, DE generates new points (trial solutions) that are perturbations of existing points, but these deviations are not samples from a predefined probability density function, like those in ESs. Instead, DE perturbs current generation vectors with the scaled difference of two randomly selected population vectors. In its simplest form, DE adds the scaled, random vector difference to a third randomly selected population vector to create a *donor* vector corresponding to each population vector (also known as *target* vector). Next the components of the target and donor vectors are mixed through a crossover operation to produce a *trial* vector. In the selection stage, the trial (or offspring) vector competes against the population vector of the same index, i.e. the parent vector. Once the last trial vector has been tested the survivors of all the pair wise competitions become parents for the next generation in the evolutionary cycle. A detailed survey on the state-of-the-art research on DE as well as its applications to different kinds of optimization problems can be found in [90].

Thomsen integrated the fitness sharing concept with DE to form the sharing DE [91]. Sharing DE utilizes the classical sharing technique described in Eqs. (3) and (4), using the Euclidean distance as the distance metric. In each generation the number of offspring generated is equal to the parent population size. Thus after *Np* trial vectors have been generated from *Np* parents, the sharing function is used to calculate the fitness for each individual and the worst half of the population is purged. The algorithm provides elitism by always preserving the individual with best un-scaled fitness. The algorithm requires defining σ_{share} that represents the threshold of dissimilarity or niche radius. Thomsen also proposed to extend DE with a crowding Scheme (Crowding DE) [91] to allow it to tackle multimodal optimization problems. In Crowding DE (CDE) when an offspring is generated by using he standard DE, it competes only 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. To avoid replacement error in CDE, the crowding factor *CF* is taken equal to the population size *Np*.

Li proposed the Species-based DE (SDE) algorithm, built around the notion of speciation [92], for solving multimodal optimization problems. The Species-based DE (SDE) is capable of locating multiple global optima simultaneously through adaptive formation of multiple species (or subpopulations) in a DE population at each iteration step. The DE population is partitioned into species according to an Euclidean distance based similarity metric. Steps of SDE are summarized in the form of a pseudo-code below:

Pseudo-code for species-based differential evolution

1. *Np* Individuals are randomly generated to create initial population.
2. Evaluate the fitness of all individuals of the population.
3. Sort all the individuals of the population in descending order of their fitness values.
4. Find out the species seed by means of the species seed determining algorithm.
5. For each species as identified by its species seed, run the classical DE
    a. If no. of individuals in a particular species is less than 3 we have to generate new individual within the radius of the species seed until it becomes more or equal to 3.
    b. If the fitness of an offspring is equal to that of the species seed, replace the redundant child by a randomly generated offspring.
6. Keep only *Np* no. of fittest individual from the total population.
7. Go to step 2 and continue till the termination criterion satisfies.

A first approach that employs DE to evolve subpopulations for achieving simultaneous convergence to multiple optima of a multimodal function can be traced in [93]. The proposed algorithm implements a mating restriction, so that the variation operations are performed only inside each subpopulation. Additionally, a penalty is applied to members of each subpopulation that are too close to members of different subpopulations to drive each subpopulation toward a different optimum. The method requires definition of the number of subpopulations, a penalty term, and the minimum spanning distance to be maintained among the subpopulations, which are all problem-dependent parameters and thus form a major difficulty for the use of the method. Rumbler and Moore [94] attempted to overcome these limitations by suggesting a NewEDE method for determining the optimal values for these parameters. The idea is to simply run the algorithm repeatedly with different parametric setups to determine suitable values and at the same time keep record of the found solutions. Of course the repeated runs increase the computational complexity of the whole process.

Zaharie [95] proposed a Multi-resolution Multipopulation DE (MMDE), which divides the population to c equally sized subpopulations. The search is divided into epochs, between which the subpopulations are reinitialized using a finer separation of the domain, so that the number of sub-domains increases by c after each epoch. The best solution in each subpopulation is stored in an archive after each epoch. To prevent redundant solutions from entering the archive, the Euclidean distance between each new entree is calculated for each existing solution in the archive, and too similar solutions are discarded. A hill–valley detection method [40] is used to exclude solutions belonging to the same peak. MMDE does not require the definition of the niche radius parameter, but introduces a set of new parameters, the number and size of the subpopulations, as well as the number and length of the epochs. Hendershot took a similar approach in his MultiDE algorithm [96] by considering equal-sized subpopulations. However, in MultiDE the number of subpopulations is kept variable i.e. subpopulations can appear and disappear. MultiDE uses a structure similar to archive in MMDE, called *population 0*. When an element from a subpopulation is similar to an element from population 0, the former is no longer considered for further evolutions. The similarity is based on a *precision parameter* to be controlled by the user. MultiDE employs a *minimum spanning distance* to encourage the search for different optima. It also introduces another time delay regarding the tunable parameter operation i.e. the number of generations after which a subpopulation is eliminated if it fails to discover a new optimum. Zaharie devised a multipopulation crowding DE [97] by integrating a crowding based niching technique along with the multipopulation DE algorithm. Under this scheme subpopulation reinitialization is no more necessary as each subpopulation is capable of locating multiple optima. The crowding computation is kept limited to subpopulations so that a global processing step can be avoided.

DE with Local Selection (DELS) [89,98] is a variant of DE where the target vector and the base vector for mutation are kept same. In the selection phase each population member is always compared to its own mutant. When the selection is local, evolution of a vector only depends on the current set of vector differences, and not directly on the parameter values of vectors other than those of the target. In effect, local selection partitions the population into *Np* niches, each of which is inhabited by a single vector that evolves in isolation. In order to employ the potential of DELS for solving multimodal problems, Rönkkönen and Lampinen divided the mutation operation of DELS into two parts [99]: local mutation and global mutation. The resulting algorithm, DELS using local mutation (DELL) [99,100], also adopts the "either/or" concept [89, p. 117], which uses only one variation operator for generating each trial. The selection between two possible operators is done probabilistically for each trial using the *PX* parameter to control the probabilities for using either. While it would be possible to use the traditional uniform crossover operation also with DELS, it would destroy the rotational invariance of the approach, and thus the crossover has been removed from the proposed algorithm. More recently, Qu and Suganthan [101] proposed a neighborhood-based DE mutation for multimodal optimization. The experimental results suggested that the performances of the DE-niching algorithms are greatly improved with the introduction of neighborhood mutation.

7. Clustering-based multimodal evolutionary approaches

Several research works have been reported for solving multimodal problems by incorporating clustering methods in some classical EAs. Different clustering techniques have been suitably used to overcome the inability of the traditional EAs to locate multiple solutions and to enhance their efficiency, comprehensiveness, and robustness. Ling et al. [102] proposed Crowding Clustering Genetic Algorithm (CCGA) where they utilize clustering strategy to eliminate the genetic drift that is introduced by the crowding strategy. Authors introduced a peak detection concept to combine the clustering and crowding techniques. Basically to create multiple niches in the given landscape the standard crowding strategy is employed in CCGA. Clusters formed by the crowding model can coexist in the same niche and lead to the same optimal solution. As both standard and deterministic crowding have the tendency to converge to numerous potential solutions and to create genetic drift, clustering operation is used to remove this genetic drift by introducing an inter-cluster competition and to stimulate exploration in the entire search space. Felix et al. proposed a Clustering-Based Niching (CBN) [103] method for Evolutionary Algorithms (EAs) to identify multiple global and local optima's in a multimodal environment. The principle behind the CBN implementation is to apply the biological concept of species in separate ecological niches to EA for preserving diversity that results advantageous to find out niches of arbitrary size, shape and spacing.

Alami et al. proposed a Multipopulation Cultural algorithm using fuzzy clustering [104,105] that was shown to be competent for optimizing high dimensional multimodal functions. Under this scheme, the entire population is divided into small subpopulations by using fuzzy clustering technique. These subpopulations and their belief spaces are kept isolated and handled by their own local cultural algorithm. Thus the cultural exchange concept has been implemented to follow the social environment principle. Experimental results [105] indicate that Multipopulation Cultural algorithm using fuzzy clustering perform better than fitness sharing technique in high dimensional multimodal search space.

Shin et al. proposed a framework for multimodal function optimization by incorporating clustering concept into Genetic Algorithm (GA) [25]. In Adaptive Isolation Model (AIM) using data clustering for multimodal function optimization [106] authors proposed a novel idea to identify the clusters in the GA population to help to recognize the optimal solution in the entire landscape. Populations in the fitness landscape of the multimodal problems are spread and form numerous clusters across different optimal regions. The main approach of AIM is to detect these clusters and to isolate them from the entire search space. These isolated clusters are later optimized independently. Also the objective function of the GA is adapted to suppress the information of the isolated clusters from the entire population. This way, the optimal solutions lying in these isolated clusters are also removed from the fitness function. The technique increases the efficiency of the search by eliminating the distance parameter and by reducing the time complexity. If a translation or rotation has been performed on a single cluster it leads to create different genotypes for similar structures. To overcome this certain problem known as cluster geometry optimization problem Leitão et al. analyze the behavior of Spatially Dispersed Genetic Algorithm [107] its application on the cluster geometry optimization.

8. Population-based niching algorithms

Different from genetic operator based niching algorithms; population-based niching algorithms use parallel population to locate different peaks. Generally, the mutation and offspring production are within the individual population. Each population is in charge of searching for one peak.

In 1999, the *multinational GA* (MNGA) was proposed by Ursem [40]. The algorithm maintains a number of parallel populations. Each population evolve toward one potential peaks in the search space. The diversity is maintained through the division of the population (subpopulations). The mating and production of offspring are restricted within individual subpopulation. Either global/weighted selection or locally/national selection are used to choose the fitter individuals for the subsequent generation. For global selection, the fitness of the two selected individuals is divided by the number of individuals in their respective nations and competes with each. On the other hand, individuals only compete with other individuals from the same nation if national selection is adopted.

In 2002, Siarry et al. proposed a multipopulation genetic algorithm [108], which searches multiple peaks by maintaining subpopulations. The algorithm divides a traditional GA into two processes: exploration (or diversification) and exploitation (or intensification). Tsutsui et al. [109] proposed the *forking GA* (fGA) which partitions the search space into various sub-spaces by using the convergence status of the current population and the solutions obtained so far. Lung [110] proposed a roaming technique to handle multimodal optimization problems. Subpopulations are used to detect multiple optima. A stability measure is defined for subpopulations which used to determine the location of the local peaks.

9. Ensemble method for solving multimodal optimization problems

Recently, ensemble idea was adopted in evolutionary algorithms to solve multimodal optimization problems. Yu and Suganthan [111] proposed an ensemble of niching algorithms (ENA) which use four parallel populations. Each population is associated with one niching method. All four populations use genetic algorithm as the search method. The offspring produced by all four populations are combined together and subsequently added to the four populations separately. Each population will select parents for the next generation from each combined population according to the niching method used. In this way, each algorithm always keeps the best offspring according to the selection rules of the associated niching method.

Recently Bo-yang and Suganthan proposed DE with an Ensemble of Restricted Tournament Selection (ERTS-DE) and tested their algorithm on 15 newly designed scalable benchmark multimodal optimization problems [112]. Obviously there is only one key parameter w (window size) that controls the performance of the restricted tournament selection. Since it is practically difficult to find a universally good value of this parameter, the authors used two populations with two different window sizes. Additional populations can also be employed, if further different parameters or different niching techniques are used. Each respective population will generate its own offspring-population and they have to compete with not only its own population but also with others. Thus ERTS-DE maintains the offspring that were generated by the more suitable parameter leading to a better performance. In [113], Qu et al. used a similar idea and proposed a dynamic grouping crowding differential evolution (DGCDE) with ensemble of parameter values. In the proposed algorithm, the population is dynamically regrouped into three subgroups every few generations. Each of the subgroups is assigned with a set of DE parameters to generate offspring. The algorithm retains the best offspring generated by the most suitable parameters.

10. Miscellaneous evolutionary approaches

Multi-objective Optimization (MO) problems involve multiple objectives, which should be optimized simultaneously and that often are in conflict with each other. This results in a group of alternative solutions (Pareto optimal set) that must be considered equivalent in the absence of information concerning the relevance of the others. Many EAs were formulated by the researchers to tackle multi-objective problems in recent past [114,115]. Recently Deb and Saha [116,117] took an MO approach for solving the multimodal optimization problems. They converted the single-objective multimodal problem into a suitable bi-objective optimization problem so that all optimal solutions become members of the resulting weak Pareto optimal set. One of the objectives was the objective function of the multimodal optimization problem and the authors made a number of suggestions for choosing the other objective. Starting with the gradient-based approaches (demonstrating the foundation of the bi-objective approach), more pragmatic neighborhood count based approaches were developed for this purpose. The authors modified the NSGA-II (Non-dominated Sorting Genetic Algorithm II) [118] algorithm in order to find weak non-dominated solutions. Using the Hooke–Jeeves based neighborhood count method [119] for the second objective, the proposed EMO procedure reportedly solved 16-variable, 48-optima problems having a combination of global and local optima. Rajeev et al. introduced a multimodal function optimization by reformulating it into a multi-objective framework [120] that does not include an explicit niching/sharing. As for most of the real-world problems the analytical form is unknown thus the prior estimation of the niche formation parameters is not possible thus in [120] the multimodal problem has been solved by using a steady-state multi-objective genetic algorithm in which diverse sampling of the Pareto front have been done to preserve diversity without niching.

Stoean et al. presented a Topological Species Conservation Algorithm (TSCA) [121] that integrates the conservation of the best successive local individuals (as done in SCGA [30]) with topological subpopulations separation (as done in the multinational genetic algorithms [41]) instead of the common but problematic radius-triggered manner. In order to detect if two individuals belong to the same subpopulation, the algorithm utilizes their fitness evaluations and of those of some intermediary assigned candidates to provide an overview on their position. More importantly, this alternative triggers flexibility in context to the formation of the species within attraction basins of different sizes. Several distinct solutions are preserved to maintain multiple optima. Each species is concentrated on a seed, which represents the fittest individual of the species. The seeds from all species are copied from one generation to another so that no important regions are lost through selection and variation operators. The species masters are then updated at each cycle, by once more appointing their fittest inner individuals.

De Castro et al. extended the concept of Artificial immune network (aiNet) algorithm, which is a discrete immune network algorithm developed for data compression and clustering by means of clonal selection and immune network theory of biology immune system, to design an optimization version of aiNet (opt-aiNet) [122], utilized to solve multimodal optimization problem. This is a mutation-based evolutionary search procedure with a population size dynamically adjusted and is able to maintain the trade-off between the exploitation and exploration of the search space for new and better solutions, locate multiple optima's, maintain many local and the global optima solutions. Although the performance of opt-aiNet was good on the simple bi-dimensional functions with multiple optimal solutions but to improve its performance in the problems needed high number of function evaluations and in dynamic environment some modification have been proposed by Fabricio et al. that results into dopt-aiNet [123]. To make dopt-aiNet capable to track the optima while it changed its position through time in a dynamic environment two new mutation operators introduced to fine-tune each cell with a modified selection method along with a cell line suppression mechanism which has been incorporated over a limited size population. Apart from dopt-aiNet another improvement of opt-aiNet have been put-forward by Qingzheng et al. known as predication based immune network (PiNet) [124]. To overcome the limitation of opt-aiNet like indeterminate direction of local search, lacking of efficient regulation mechanism between local search and global search the proposed PiNet includes two main features with opt-aiNet. To determine the direction of local search and to adjust the balance between local and global search the information of antibodies in continuous generations is utilized. Thus the performance of PiNet of finding multiple peaks from a multimodal search space improved drastically in successful rate, convergence speed, etc.

Ant Colony Optimization (ACO) [125] inspired by the foraging behavior of biological ants has also come into play to solve multimodal optimization problems. Daniel Angus incorporated the idea of sharing and crowding into population-based Ant Colony Optimization (PACO) [126] to generate Fitness Sharing PACO (FS-PACO) and Simple Crowding PACO algorithm (SC-PACO) respectively [127] to handle multimodal optimization problem. The niching methods help to maintain the diversity which is expected to improve the PACO algorithm to make it robust to solve complex multimodal problems such as dynamic problems. Both of the FS-PACO and SC-PACO algorithms are able to locate and maintain multiple, spatially distributed, near-optimal solutions rather than just the single best solution found overall. Recently another effective approach has been taken by Chao-Yang Panget al. [128] to solve extremum problem in multimodal functions. To find all the extreme points in a multimodal search space is known as extremum problem, which is a well studied complex problem in numerical optimization.

To enhance the speed of searching the local and global optima in multimodal search space, Preuss [129] suggests that if basin of attraction for each individual niche is approximately equal then Monte Carlo Simulation will outperform the EAs.

11. Benchmark problems and performance measures

Many multimodal benchmark test functions have been proposed in the past few decades (as listed in Appendix A). However, these problems are not complex to challenge the powerful search algorithms. Many multimodal optimization algorithms are able to solve them easily. Recently, Qu and Suganthan [112] proposed a new set of multimodal benchmark problems consisting of many composition problems. An overview of these functions is presented in Table 2, while mathematical descriptions are presented in Appendix B. More details can be found in [112]. Note that the dimensions of these test problems can be increased.

To compare different multimodal optimization algorithms, various performance measures have been proposed. These performance measures are presented in Table 3.

12. Practical applications of real-parameter multimodal optimization

Practical real-parameter multimodal optimization problems are yet to get the attention of the researchers from the field of niching and evolutionary multimodal optimization. Currently most of the papers in this area publish test results on benchmark problems. In this subsection we discuss a few practical multimodal

Table 2
The overview of the benchmark functions.

Test function	Dimension	Number of optima	Number of composition functions
CF1	10D	8	4
CF2	10D	6	3
CF3	10D	6	3
CF4	10D	6	3
CF5	10D	6	3
CF6	10D	6	3
CF7	10D	6	3
CF8	10D	6	3
CF9	10D	6	3
CF10	10D	6	3
CF11	10D	8	4
CF12	10D	8	4
CF13	10D	10	5
CF14	10D	10	5
CF15	10D	10	5

problems that arise from the diverse domains of science and engineering. Alami and Imrani [130] report the application of a multipopulation cultural algorithm to solve the multimodal dielectric composite design problem. The objective of this problem is to determine the structure of a dielectric composite so that its macroscopic effective permittivity ε_{eff} in the direction of the field applied is equal to ε_{obj}.

This problem is evident since an analytical expression of the effective permittivity does exist. This expression can be obtained by considering association in series of two dielectrics and given below [130]:

$$\varepsilon_{\text{eff}} = \frac{\varepsilon_1 \varepsilon_2}{\varepsilon_2 f + \varepsilon_1 (1 - f)}, \tag{9}$$

where ε_1 and ε_2 indicate the permittivity of dielectrics 1 and 2 respectively, and f represents the concentration of material 1. To produce a composite of effective permittivity macroscopic $\varepsilon_{\text{obj}} = 1.5$ from a range of material 1 of permittivity varying between 10 and 30, the permittivity of material 2 being imposed ($\varepsilon_2 = 1$) and the concentration of material 1 can vary between 0.1 and 0.9. The problem of optimization to be solved can be expressed in the following way according to the two parameters ε_1 and f,

$$\begin{cases} \min F_{\text{obj}}(f, \varepsilon_1) = |\varepsilon_{\text{eff}} - 1.5| \\ 0.1 \le f \le 0.9 \\ 10 \le \varepsilon_1 \le 30. \end{cases} \tag{10}$$

In [131], Wong et al. considers the protein structure prediction problem as a multimodal optimization problem. In particular, they applied multimodal EAs for solving de novo protein structure prediction problems on the 3D Hydrophobic-Polar (HP) lattice model, based on the observation that there can be multiple conformations for each energy value.

In [132], Lee et al. applied a niching GA with restrictive competition selection to the optimal design of induction motors for Electric Vehicles (EV). Multiple solutions to the design problem are needed so that the designer may select the best for EV traction motor among alternative designs using certain evaluation criteria. Im et al. [133] applied an ES-based multimodal optimizer to the design of a deflection yoke for color display tubes. The objective functions were the beam shadow neck (BSN), trilemma, and sensitivity of horizontal coils. Seo et al. [134] used a PSO-based multimodal optimization algorithm to the structural optimization of a short-time rating Interior Permanent-Magnet Synchronous Motor (IPMSM) [6] having four poles and 18 slots. Seo et al. [135] applied an improved PSO-based niching algorithm to the structural optimization of an IPMSM for 42 V Integrated Starter-Generator (ISG). The proposed multimodal PSO variant was termed as Auto-Tuning Multi-Grouped PSO (AT-MGPSO) algorithm and it imitates

Table 3
Performance measures.

Measure	Description
Success rate	Success rate is the percentage of runs in which all the desired peaks are successfully located.
Average number of optima found	This criterion is used to compare the average number of peaks found over different runs.
Success performance	The success performance is calculated using the following equation: Success performance $= \frac{\text{Average number of function evaluations}}{\text{success rate}}$ Note that the success performance can be obtained only if the success rate is not zero.
Maximum Peak Ratio (MPR)	The maximum peak ratio is defined as follows: (assuming a maximization problem) $\mathrm{MPR} = \frac{\sum_{i=1}^{q} f_i}{\sum_{i=1}^{q} F_i}$ where q is the number of optima, $\{f_i\}_{i=1}^{q}$ are the fitness values of the optima in the final population while $\{F_i\}_{i=1}^{q}$ are the values of real optima of the objective function. All values are assumed to be positive. Note that larger MPR value indicates a better performance of the particular algorithm.

the natural phenomena in ecosystem such as the territorial dispute between different group members and immigration of weak groups, resulting in automatic determination of the size of each group's territory and robust convergence. An interesting application of the Niching GAs to benchmark electromagnetic design problems (TEAM Workshop Problem 22 (SMES), for the discrete case) can be found in [136].

Shir et al. [137] used the dynamic niching with CMA-ES algorithm for optimizing the alignment of a sample of generic diatomic molecules undergoing irradiation by a shaped femtosecond laser. The authors proposed a new approach for the problem-specific diversity measurement among the niches, with an objective of obtaining different conceptual laser pulse designs.

13. Conclusions

This article is the first of its kind to present a detailed survey of the research on and with EAs specifically tailored to tackle the real-parameter multimodal optimization problems. It started with a detailed review of several kinds of niching techniques like crowding, clearing, sharing, clustering, probabilistic and deterministic crowding, speciation, restricted tournament selection, parallel and sequential niching, and localized niching. It then thoroughly explored the variants of most prominent evolutionary computing techniques like GAs, PSO, DE, and ESs that were modified over past few decades for solving multimodal problems. It also provided an account of some of the uncommon approaches to multimodal optimization like the ensemble of niching strategies and the multi-objective formulation of the multimodal problems.

Although research on evolutionary approaches to multimodal optimization started almost four decades earlier, several challenges still exist before the researchers and new application areas are continually emerging. An EA devised for multimodal optimization must answer two crucial questions in order to guarantee some success on a given task: How to most unboundedly distinguish between the different basins of attraction on the fitness landscape? And how to most accurately safeguard the consequently discovered solutions? It is important that an EA does not introduce new parameters in doing so, such that the performance of the EA becomes sensitive to a problem-dependent tuning of these parameters. Li's work as reported in [72] takes a bold step toward making the PSO-based niching algorithms free of tunable parameters. It would be beneficial to attempt designing parameter-free niching methods based on other powerful EAs like DE, ES etc. There is a great lack of theoretical understanding of the distributed convergence behavior of the niching techniques and an analysis of the effect of the niching parameters on such convergence. Such studies can provide important guidelines for estimating the niching parameters that will support a statistically superior performance over a wide variety of problems. Also much more investigation is necessary with EAs to make them capture multiple peaks in dynamic and noisy environments. The content of the article indicates the fact that the topic of evolutionary multimodal optimization will continue to remain a vibrant and active field of multi-disciplinary research in the years to come.

Appendix A. Classical benchmark test functions

F1: Two-peak trap [138]

$$f_1(x) = \begin{cases} \dfrac{160}{15}(15 - x), & \text{for } 0 \le x \le 15 \\ \dfrac{200}{5}(x - 15), & \text{for } 15 \le x \le 20. \end{cases}$$

Range: $0 \le x \le 20$.

F2: Central two-peak trap [138]

$$f_2(x) = \begin{cases} \dfrac{160}{10}x, & \text{for } 0 \le x \le 10 \\ \dfrac{160}{5}(15 - x) & \text{for } 10 \le x \le 15 \\ \dfrac{200}{5}(x - 15), & \text{for } 15 \le x \le 20. \end{cases}$$

Range: $0 \le x \le 20$.

F3: Five-uneven-peak trap [30]

$$f_3(x) = \begin{cases} 80(2.5 - x) & \text{for } 0 \le x < 2.5 \\ 64(x - 2.5) & \text{for } 2.5 \le x < 5 \\ 64(7.5 - x) & \text{for } 5 \le x < 7.5 \\ 28(x - 7.5) & \text{for } 7.5 \le x < 12.5 \\ 28(17.5 - x) & \text{for } 12.5 \le x < 17.5 \\ 32(x - 17.5) & \text{for } 17.5 \le x < 22.5 \\ 32(27.5 - x) & \text{for } 22.5 \le x < 27.5 \\ 80(x - 27.5) & \text{for } 27.5 \le x \le 30. \end{cases}$$

Range: $0 \le x \le 20$.

F4: Equal maxima [139]

$$f_4(x) = \sin^6(5\pi x).$$

Range: $0 \le x \le 1$.

F5: Decreasing maxima [139]

$$f_5(x) = \exp\left[-2\log(2)\cdot\left(\frac{x - 0.1}{0.8}\right)^2\right]\cdot\sin^6(5\pi x).$$

Range: $0 \le x \le 1$.

F6: Uneven maxima [139]

$$f_6(x) = \sin^6(5\pi(x^{3/4} - 0.05)).$$

Range: $0 \le x \le 1$.

F7: Uneven decreasing maxima [139]

$$f_7(x) = \exp\left[-2\log(2)\cdot\left(\frac{x-0.08}{0.854}\right)^2\right] \times \sin^6(5\pi(x^{3/4}-0.05)).$$

Range: $0 \le x \le 1$.

F8: Himmelblau's function [139]

$$f_8(x,y) = 200-(x^2+y-11)^2-(x+y^2-7)^2.$$

Range: $-6 \le x, y \le 6$.

F9: Six-hump camel back [140]

$$f_9(x,y) = -4\left[\left(4-2.1x^2+\frac{x^4}{3}\right)x^2+xy+(-4+4y^2)y^2\right].$$

Range: $\begin{matrix}-1.9 \le x \le 1.9;\\ -1.1 \le y \le 1.1.\end{matrix}$

F10: Shekel's foxholes [7]

$$f_{10}(x,y) = 500-\frac{1}{0.002+\sum_{i=0}^{24}\frac{1}{1+i+(x-a(i))^6+(y-b(i))^6}}$$

where $a(i) = 16(i \bmod 5)-2$, and $b(i) = 16(\lfloor(i/5)\rfloor-2)$.

Range: $-65.536 \le x,\ y \le 65.535$.

F11: 2D inverted Shubert function [30]

$$f_{11}(\vec{x}) = -\prod_{i=1}^{2}\sum_{j=1}^{5} j\cos[(j+1)x_i+j].$$

Range: $-10 \le x_1,\ x_2 \le 10$.

F12: Inverted Vincent function [141]

$$f(\vec{x}) = \frac{1}{n}\sum_{i=1}^{n}\sin(10.\log(x_i))$$

where n is the dimension of the problem.

Range: $0.25 \le x_i \le 10$.

F13: Branin RCOS [30]

$$f_{17}(x,y) = \left(y-\frac{5.1}{4\pi^2}x^2+\frac{5}{\pi}x-6\right)^2 + 10\left(1-\frac{1}{8\pi}\right)\cos(x)+10.$$

Range: $-5 \le x \le 10, 0 \le y \le 15$.

F14: Michalewicz [91]

$$f_{19}(x,y) = \sin(x)\sin^{20}\left(\frac{x^2}{\pi}\right)+\sin(y)\sin^{20}\left(\frac{2y^2}{\pi}\right).$$

Range: $0 \le x,\ y \le \pi$.

F15: Ursem F1 [40]

$$f_{20}(x,y) = \sin(2x-0.5\pi)+3\cos(y)+0.5x.$$

Range: $-2.5 \le x \le 3,\ \ -2 \le y \le 2$.

F16: Ursem F3 [40]

$$f_{21}(x,y) = \sin(2.2\pi x+0.5\pi)\cdot\frac{2-|y|}{2}\cdot\frac{3-|x|}{2} + \sin(0.5\pi y^2+0.5\pi)\cdot\frac{2-|y|}{2}\cdot\frac{2-|x|}{2}.$$

Range: $-2.5 \le x \le 3,\ \ -2 \le y \le 2$.

Appendix B. Composition benchmark test functions

$F(x)$: new composition function
$f_i(x)$: ith basic function used to construct the composition function.
n: number of basic functions (number of optima)
D: dimensions (can be chosen from 1 to 100)
M_i: linear transformation matrix for each $f_i(x)$
o_i: new shifted optima position for each $f_i(x)$

$$F(x) = \sum_{i=1}^{n}\{w_i * [f_i'((x-o_i)/\lambda_i * M_i)]\}$$

w_i: weight value for each $f_i(x)$, calculated as follow:

$$w_i = \exp\left(-\frac{\sum_{k=1}^{D}(x_k-o_{ik})}{2D\sigma_i^2}\right)$$

$$w_i = \begin{cases} w_i & w_i = \max(w_i) \\ w_i^*(1-\max(w_i).\hat{}10) & w_i \ne \max(w_i). \end{cases}$$

Then normalize the weight $w_i = w_i/\sum_{i=1}^{n} w_i$.
σ_i: used to control each $f_i(x)$'s coverage range.
λ_i: used to stretch compress the function.
$f_i'(x) = C * f_i(x)/|f_{\max i}|$, C is a predefined constant.
$|f_{\max i}|$ is estimated using: $|f_{\max i}| = f_i((x'/\lambda_i) * M_i), x' = [5, 5, \ldots, 5]$.

Composition function 1 ($n = 8$)

$f_{1-2}(x)$: Rastrigin's function

$$f_i(x) = \sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$$

$f_{3-4}(x)$: Weierstrass function

$$f_i(x) = \sum_{i=1}^{D}\left(\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k(x_i+0.5))] - D\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k\cdot 0.5)]\right)$$

$a = 0.5, b = 3, k_{\max} = 20$.

$f_{5-6}(x)$: Griewank's function

$$f_i(x) = \sum_{i=1}^{D}\frac{x_i^2}{4000}-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$$

$f_{7-8}(x)$: Sphere function

$$f_i(x) = \sum_{i=1}^{D}x_i^2$$

$\sigma_i = 1$ for all i
$\lambda = [1, 1, 10, 10, 5/60, 5/60, 5/32, 5/32]$.

M_i: are all identity matrices.

These formulas are basic functions; shift and rotation should be added to these functions. Take f_1 as an example, the following function should be evaluated:

$$f_i(z) = \sum_{i=1}^{D}(z_i^2-10\cos(2\pi z_i)+10)$$

where $z = ((x-o_i)/\lambda_1) * M_1$.

Composition function 2 ($n = 6$)

$f_{1-2}(x)$: Griewank's function
$f_{3-4}(x)$: Weierstrass function

$f_{5-6}(x)$: Sphere Function
$\sigma_i = 1$ for all i
$\lambda = [1, 1, 10, 10, 5/60, 5/60]$
M_i: are all identity matrices.

Composition function 3 ($n = 6$)

$f_{1-2}(x)$: Rastrigin's function
$f_{3-4}(x)$: Griewank's function
$f_{5-6}(x)$: Sphere function
$\sigma_i = 1$ for all$i$
$\lambda = [1, 1, 10, 10, 5/60, 5/60]$
M_i: are all identity matrices.

Composition function 4 ($n = 6$)

$f_{1-2}(x)$: Rastrigin's function
$f_{3-4}(x)$: Weierstrass function
$f_{5-6}(x)$: Griewank's function
$\sigma_i = 1$ for all i
$\lambda = [1, 1, 10, 10, 5/60, 5/60]$
M_i: are all identity matrices.

Composition function 5 ($n = 6$)

$f_{1-2}(x)$: Rastrigin's function
$f_{3-4}(x)$: Weierstrass function
$f_{5-6}(x)$: Sphere function
$\sigma_i = 1$ for all i
$\lambda = [1, 1, 10, 10, 5/60, 5/60]$
M_i: are all identity matrices.

Composition function 6 ($n = 6$)

$f_{1-2}(x)$: F8F2 function

$$F8(x) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

$$F2(x) = \sum_{i=1}^{D-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$$

$$\begin{aligned} f_i(x) \ = \ & F8(F2(x_1, x_2)) + F8(F2(x_2, x_3)) + \cdots \\ & + F8(F2(x_{D-1}), x_D) + F8(F2(x_D, x_1)). \end{aligned}$$

$f_{3-4}(x)$: Weierstrass function
$f_{5-6}(x)$: Griewank's function

$\sigma = [1, 1, 1, 1, 1, 2]$,

$\lambda = [5 * 5/100; 5/100; 5 * 1; 1; 5 * 1; 1]$.

M_i: are all orthogonal matrix.

Composition function 7 ($n = 6$)

$f_{1-2}(x)$: Rotated expanded Scaffer's F6 Function

$F(x, y) = 0.5 + \frac{(\sin^2(\sqrt{x^2+y^2}) - 0.5)}{(1+0.001(x^2+y^2))^2}$

$f_i(x) = F(x_1, x_2) + F(x_2, x_3) + \cdots + F(x_{D-1}, x_D) + F(x_D, x_1)$

$f_{3-4}(x)$: F8F2 function
$f_{5-6}(x)$: Weierstrass function

$\sigma = [1, 1, 1, 1, 1, 2]$,

$\lambda = [5; 10; 5; 1; 5 * 5/100; 5/100]$

M_i: are all orthogonal matrix.

Composition function 8 ($n = 6$)

$f_{1-2}(x)$: Rotated expanded Scaffer's F6 function
$f_{3-4}(x)$: F8F2 function
$f_{5-6}(x)$: Griewank's function

$\sigma = [1, 1, 1, 1, 1, 2]$,

$\lambda = [5 * 5/100; 5/100; 5 * 1; 1; 5 * 1; 1]$.

M_i: are all orthogonal matrix.

Composition function 9 ($n = 6$)

$f_{1-2}(x)$: Rotated expanded Scaffer's F6 function
$f_{3-4}(x)$: Weierstrass function
$f_{5-6}(x)$: Griewank's function

$\sigma = [1, 1, 1, 1, 1, 2]$,

$\lambda = [5; 10; 5 * 5/100; 5/100; 5; 1]$

M_i: are all orthogonal matrix.

Composition function 10 ($n = 6$)

$f_{1-2}(x)$: Rastrigin's function
$f_{3-4}(x)$: F8F2 function
$f_{5-6}(x)$: Weierstrass function

$\sigma = [1, 1, 1, 1, 1, 2]$,

$\lambda = [5; 10; 5 * 5/100; 5/100; 5; 1]$

M_i: are all orthogonal matrix.

Composition function 11 ($n = 8$)

$f_{1-2}(x)$: Rastrigin's function
$f_{3-4}(x)$: F8F2 function
$f_{5-6}(x)$: Weierstrass function
$f_{7-8}(x)$: Griewank's function

$\sigma = [1, 1, 1, 1, 1, 2, 2, 2]$,

$\lambda = [5; 1; 5; 1; 50; 10; 5 * 5/200; 5/200]$

M_i: are all orthogonal matrix.

Composition function 12 ($n = 8$)

$f_{1-2}(x)$: Rotated expanded Scaffer's F6 function
$f_{3-4}(x)$: F8F2 function
$f_{5-6}(x)$: Weierstrass function
$f_{7-8}(x)$: Griewank's function

$\sigma = [1, 1, 1, 1, 1, 2, 2, 2]$,

$\lambda = [5 * 5/100; 5/100; 5; 1; 5; 1; 50; 10]$

M_i: are all orthogonal matrix.

Composition function 13 ($n = 10$)

$f_{1-2}(x)$: Rotated expanded Scaffer's F6 function
$f_{3-4}(x)$: Rastrigin's function
$f_{5-6}(x)$: F8F2 function
$f_{7-8}(x)$: Weierstrass function
$f_{9-10}(x)$: Griewank's function

$\sigma = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]$,

$\lambda = [5 * 5/100; 5/100; 5; 1; 5; 1; 50; 10; 5 * 5/200; 5/200]$

M_i: are all orthogonal matrix.

Composition function 14 ($n = 10$)

All settings are the same as F13, except M_i's condition numbers are [10 20 50 100 200 1000 2000 3000 4000 5000].

Composition function 15 ($n = 10$)

$f_1(x)$: Weierstrass function
$f_2(x)$: Rotated expanded Scaffer's F6 function
$f_3(x)$: F8F2 function
$f_4(x)$: Ackley's function

$$f_i(x) = -20\exp(-0.2\sqrt{\tfrac{1}{D}\textstyle\sum_{i=1}^{D} x_i^2}) - \exp(\tfrac{1}{D}\textstyle\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e f_5(x)\text{: Rastrigin's function}$$

$f_6(x)$: Griewank's function

$f_7(x)$: Non-continuous expanded Scaffer's F6 function

$$F(x, y) = 0.5 + \frac{(\sin^2(\sqrt{x^2+y^2}) - 0.5)}{(1 + 0.001(x^2+y^2))^2}$$

$$f_i(x) = F(y_1, y_2) + F(y_2, y_3) + \cdots + F(y_{D-1}, y_D) + F(y_D, y_1)$$

$$y_i = \begin{cases} x_j & |x_j| < 1/2 \\ \text{round}(2x_j)/2 & |x_j| > 1/2 \end{cases} \quad \text{for } j = 1, 2, \ldots, D$$

$$\text{round}(x) = \begin{cases} a - 1 & \text{if } x \le 0 \text{ and } b \ge 0.5 \\ a & \text{if } b < 0.5 \\ a + 1 & \text{if } x > 0 \text{ and } b \ge 0.5. \end{cases}$$

$f_8(x)$: Non-continuous Rastrigin's function

$$f_i(x) = \sum_{i=1}^{D} (y_i^2 - 10\cos(2\pi y_i) + 10)$$

$$y_i = \begin{cases} x_j & |x_j| < 1/2 \\ \text{round}(2x_j)/2 & |x_j| > 1/2 \end{cases} \quad \text{for } j = 1, 2, \ldots, D.$$

$f_9(x)$: High conditioned elliptic function

$$f(x) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} x_i^2$$

$f_{10}(x)$: Sphere function with noise in fitness

$$f_i(x) = \left(\sum_{i=1}^{D} x_i^2\right)(1 + 0.1|N(0, 1)|)$$

$$n = 10$$

$$\sigma_i = 2 \text{ for all } i$$

$$\lambda = [10; 5/20; 1; 5/32; 1; 5/100; 5/50; 1; 5/100; 5/100]$$

M_i are all rotation matrices, condition number are [100 50 30 10 5 5 4 3 2 2];

References

[1] T. Bäck, D. Fogel, Z. Michalewicz, Handbook of Evolutionary Computation, Oxford Univ. Press, 1997.

[2] A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing, Springer, 2003.

[3] S. Mahfoud, Niching method for genetic algorithms, Doctoral Dissertation, Technical Report, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA, Illinois Genetic Algorithms Laboratory, IlliGAL, Report No. 95001, 1995.

[4] B. Sareni, L. Krahenbuhl, Fitness sharing and niching methods revisited, IEEE Transactions on Evolutionary Computation 2 (3) (1998) 97–106.

[5] G. Singh, K. Deb, Comparison of multimodal optimization algorithms based on evolutionary algorithms, in: Proceedings of the Genetic and Evolutionary Computation Conference, ACM Press, Seattle, WA, 2006, pp. 1305–1312.

[6] D. Cavicchio, Adapting search using simulated evolution, Ph.D. Dissertation, Univ. Michigan, Ann Arbor, 1970.

[7] K.A. De Jong, An analysis of the behavior of a class of genetic adaptive systems, Ph.D. Dissertation, Univ. Michigan, Ann Arbor, MI, 1975.

[8] G. Harik, Finding multi-modal solutions using restricted tournament selection, in: Proceedings of the Sixth International Conference on Genetic Algorithms, ICGA-95, 1997, pp. 24–31.

[9] S. Mahfoud, Crowding and preselection revisited, in: Parallel Problem Solving from Nature, vol. 2, 1992, pp. 27–37.

[10] D.E. Goldberg, J. Richardson, Genetic algorithms with sharing for multimodal function optimization, in: Proceedings of the second International Conference on Genetic Algorithms, 1987, pp. 41–49.

[11] A. Pétrowski, An efficient hierarchical clustering technique for speciation, Tech. Rep., Institute National des Telecommunications, Evry, France, 1997.

[12] J. Horn, N. Nafpliotis, D.E. Goldberg, A niched Pareto genetic algorithm for multiobjective optimization, in: Proceedings of the First IEEE Conference on Evolutionary Computation, Orlando, FL, USA, 1994.

[13] I. Schoeman, A.P. Engelbrecht, Niching for dynamic environments using particle swarm optimization, in: Simulated Evolution and Learning, SEAL, in: LNCS, vol. 4247-2006, Springer, 2006, pp. 134–141.

[14] Y. Jin, J. Branke, Evolutionary optimization in uncertain environments—a survey, IEEE Transactions on Evolutionary Computation 9 (3) (2005) 303–317.

[15] C.A. Coello Coello, An updated survey of GA-based multiobjective optimization techniques, ACM Computing Surveys 32 (2) (2000) 109–143. ACM Press.

[16] J. Barrera, C.A. Coello Coello, A review of particle swarm optimization methods used for multimodal optimization, in: L.C. Jain, S. Dehuri, C.P. Lim (Eds.), Swarm Intelligence for Knowledge-Based Systems, Springer-Verlag, 2010.

[17] A.P. Engelbrecht, Computational Intelligence: An Introduction, John Wiley & Sons, 2007.

[18] J. Hu, E. Goodman, Robust and efficient genetic algorithms with hierarchical niching and a sustainable evolutionary computation model, in: Proceedings of the Conference on Genetic and Evolutionary Computation, GECCO 2004, in: Lecture Notes in Computer Science, vol. 3102, Springer-Verlag, 2004, pp. 1220–1232.

[19] S.W. Mahfoud, A comparison of parallel and sequential niching method, in: Proceedings of the Sixth International Conference on Genetic Algorithms, ICGA, 1995, pp. 136–143.

[20] D. Beasley, D.R. Bull, R.R. Martin, A sequential niche technique for multimodal function optimization, Evol. Comput. 1 (2) (1993) 101–125.

[21] A. Pétrowski, A clearing procedure as a niching method for genetic algorithms, in: Proceedings of Third IEEE International Conference on Evolutionary Computation, ICEC'96, IEEE Press, Piscataway, NJ, 1996, pp. 798–803.

[22] K. Deb, D.E. Goldberg, An investigation of niche and species formation in genetic function optimization, in: J.D. Schaffer (Ed.), Proceedings of the Third, in: International Conference on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989, pp. 42–50.

[23] S. Mahfoud, Simple analytical models of genetic algorithms for multi-modal function optimization, Technical Report, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA, Illinois Genetic Algorithm Laboratory Report No. 94005, 1993.

[24] O. Mengsheol, D. Goldberg, Probabilistic crowding: deterministic crowding with probabilistic replacement, in: Proceedings of the Genetic and Evolutionary Computation Conference, 1999, GECCO 99, 1999, pp. 409–416.

[25] D.E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, New York, 1989.

[26] D.E. Goldberg, L. Wang, Adaptive niching via coevolutionary sharing, in: D. Quagliarella, J. Périaux, C. Poloni, G. Winter (Eds.), Genetic Algorithms and Evolution Strategies in Engineering and Computer Science, in: Recent Advances and Industrial Applications, John Wiley and Sons, 1998, pp. 21–38.

[27] G. Harik, Finding multi-modal solutions in problems of bounded difficulty, Technical Report, Illinois Genetic Algorithms Laboratory, Report No. 94002, 1994.

[28] R. Roy, I.C. Parmee, Adaptive restricted tournament selection for the identification of multiple sub-optima in a multi-modal function, in: T.C. Fogarty (Eds.), Selected Papers from AISB Workshop on Evolutionary Computing, April 01–02, 1996, Lecture Notes in Computer Science, vol. 1143, Springer-Verlag, London, 1996, pp. 236–256.

[29] X. Yin, N. Germay, A fast genetic algorithm with sharing scheme using cluster analysis methods in multi-modal function optimization, in Proceedings of the International Conference on Artificial Neural Nets and Genetic Algorithms, 1993, pp. 450–457.

[30] J.-P. Li, M.E. Balazs, G.T. Parks, P.J. Clarkson, A species conserving genetic algorithm for multi-modal function optimization, Evolutionary Computation 10 (3) (2002) 207–234.

[31] J.-P. Li, A. Wood, Random search with species conservation for multimodal functions, in: Proceedings of the Eleventh Conference on Congress on Evolutionary Computation, Trondheim, Norway, 2009, pp. 3164–3171.

[32] P. Moscato, On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms, CalTech Concurrent Computation Program, Tech. Rep. C3P, Report 826, 1989.

[33] J.E. Vitela, O. Castanos, A real-coded niching memetic algorithm for continuous multimodal function optimization, in: Proceedings of the Congress on Evolutionary Computation 2008, 2008, pp. 2170–2177.

[34] M. Lozano, F. Herrera, N. Krasnogor, D. Molina, Real-coded memetic algorithms with crossover hill-climbing, Evolutionary Computation 12 (3) (2004) 273–302.

[35] G. Dick, A comparison of localised and global niching methods, in: Proceedings of the 17th Annual Colloquium of the Spatial Information Research Centre, New Zealand, 2005.

[36] K.A. De Jong, J. Sharma, On decentralizing selection algorithms, in: L.J. Eshelman (Ed.), Proceedings of the Sixth International Conference on Genetic Algorithms, ICGA'95, Morgan Kaufmann Publishers San Francisco, California, 1995, pp. 17–23.

[37] W.M. Spears, Simple subpopulation schemes, in: A.V. Sebald, L.J. Fogel (Eds.), Evolutionary Programming: Proc. of the Third Annual Conference, World Scientific Press, Singapore, 1994, pp. 296–307.

[38] G. Dick, P. Whigham, The behaviour of genetic drift in a spatially-structured evolutionary algorithm, in: Proceedings of the Congress on Evolutionary Computation 2005 IEEE, IEEE Press, 2005, pp. 1855–1860.

[39] O.J. Mengshoel, D.E. Goldberg, The crowding approach to niching in genetic algorithms, Evolutionary Computation 16 (2008) 315–354.

[40] R.K. Ursem, Multinational evolutionary algorithms, in: Proceedings of the Congress on Evolutionary Computation, vol. 3, 1999, pp. 1633–1640.

[41] R.K. Ursem, Multinational GAs: multimodal optimization techniques in dynamic environments, in: Proceedings of the Second Genetic and Evolutionary Computation Conference, GECCO, Morgan Kaufmann, 2000.

[42] W. Cedeño, V.R. Vemuri, On the use of niching for dynamic landscapes, in: Proceedings of the 1997 IEEE International Conference on Evolutionary Computation, Piscataway, New Jersey, April, 1997, pp. 361–366.

[43] H.Z. Yang, F.C. Li, C.M. Wang, A density clustering based niching genetic algorithm for multimodal optimization, in: Proceedings of 2005 International Conference on Machine Learning and Cybernetics, vol. 3, August 2005, pp. 1599–1604.

[44] J. Yao, N. Kharma, P. Grogono, Bi-objective multipopulation genetic algorithm for multimodal function optimization, IEEE Transactions on Evolutionary Computation 14 (1) (2010).

[45] J. Gan, K. Warwick, Dynamic niche clustering: a fuzzy variable radius niching technique for multimodal optimization in GAs, in: Proceedings of the Congress on Evolutionary Computation, Seoul, Korea, 2001.

[46] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: Proc. IEEE Int. Conf. Neural Netw., ICNN, vol. 4, November 1995, pp. 1942–1948.

[47] J. Kennedy, R.C. Eberhart, Y. Shi, Swarm Intelligence, Morgan Kaufmann, San Francisco, CA, 2001.

[48] A.P. Engelbrecht, Fundamentals of Computational Swarm Intelligence, John Wiley & Sons, 2006.

[49] Y. del Valle, G.K. Venayagamoorthy, S. Mohagheghi, J.-C. Hernandez, R.G. Harley, Particle swarm optimization: basic concepts, variants and applications in power systems, IEEE Transactions on Evolutionary Computation 12 (2) (2008) 171–195.

[50] S.C. Esquivel, C.A.C. Coello, On the use of particle swarm optimization with multimodal functions, CEC'2003, in: Proceedings of 2003 IEEE Congress on Evolutionary Computation, vol. 2, IEEE Press, Canberra, Australia, 2003, pp. 1130–1136.

[51] K.E. Parsopoulos, V.P. Plagianakos, G.D. Magoulas, M.N. Vrahitis, Stretching technique for obtaining global minimizers through particle swarm optimization, in: Proceedings of the Particle Swarm Optimization Workshop, Indianapolis, USA, 2001, pp. 22–29.

[52] K. Parsopoulos, M. Vrahatis, On the computation of all global minimizers through particle swarm optimization, IEEE Transactions on Evolutionary Computation 8 (3) (2004) 211–224.

[53] K. Parsopoulos, M. Vrahatis, Modification of the particle swarm optimizer for locating all the global minima, in: Artificial Neural Networks and Genetic Algorithms, Springer, 2001, pp. 324–327.

[54] F. van den Bergh, An analysis of particle swarm optimizers, Ph.D. Thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa, 2002.

[55] R. Brits, A. Engelbrecht, F. van den Bergh, A niching particle swarm optimizer, in: Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning 2002, SEAL 2002, 2002, pp. 692–696.

[56] F. van den Bergh, A.P. Engelbrecht, A new locally convergent particle swarm optimizer, in: The Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Hammamet, Tunisia, October 2002.

[57] N.Q. Uy, N.X. Hoai, R.I. Maccay, P.M. Tuan, Initializing PSO with randomized low-discrepancy sequences: the comparative results, in: Proceedings of the Congress on Evolutionary Computation, 2007, pp. 1985–1992.

[58] J. Kennedy, The particle swarm: social adaptation of knowledge, in: Proceedings of the International Conference on Evolutionary Computation, 1997, pp. 303–308.

[59] A.P. Engelbrecht, L.N.H. Van Loggerenberg, Enhancing the niche PSO, in: Proceedings of the Congress on Evolutionary Computation, 2007.

[60] R. Brits, A.P. Engelbrecht, F. van den Bergh, Solving systems of unconstrained equations using particle swarm optimizers, in: The Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Pretoria, South Africa, October 2002, pp. 102–107.

[61] X. Li, Adaptively choosing neighborhood bests using species in a particle swarm optimizer for multimodal function optimization, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO, in: LNCS, vol. 3102, 2004, pp. 105–116.

[62] D. Parrott, X. Li, Locating and tracking multiple dynamic optima by a particle swarm model using speciation, IEEE Transactions on Evolutionary Computation 10 (4) (2006) 440–458.

[63] S. Bird, X. Li, Adaptively choosing niching parameters in a PSO, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO'06, ACM, Seattle, WA, 2006, pp. 3–10.

[64] S. Bird, X. Li, Enhancing the robustness of a speciation-based PSO, in: Proceedings of the Congress on Evolutionary Computation 2006, IEEE Press, Vancouver, BC, Canada, 2006, pp. 16–21.

[65] A. Nickabadi, M.M. Ebadzadhe, R. Safabakhsh, A dynamic niching particle swarm optimizer for multi-modal optimization, in: Proceedings of the Congress on Evolutionary Computation, CEC 2008, Hong Kong, 2008, pp. 26–32.

[66] X. Li, Multimodal function optimization based on fitness-euclidean distance ratio, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO, 2007, pp. 78–85.

[67] I. Schoeman, A.P. Engelbrecht, Using vector operations to identify niches for particle swarm optimization, in: Cybernetics and Intelligent Systems, 2004 IEEE Conference on, Singapore, vol. 1, December 2004, pp. 361–366.

[68] I. Schoeman, A.P. Engelbrecht, Effect of particle initialization on the performanceee of particle swarm niching algorithms, in: Proceedings of Swarm Intelligence. 7th International Conference, ANTS 2010, in: Lecture Notes in Computer Science, vol. 6234, Springer, Brussels, Belgium, 2010, pp. 560–561.

[69] J. Alami, L. Benameur, A.A. El Imrani, A fuzzy clustering based PSO for multi modal optimization, International Journal of Computational Intelligence Research (2009) 96–107.

[70] A. Passaro, A. Starita, Particle swarm optimization for multimodal functions: a clustering approach, Journal of Artificial Evolution and Applications, 2008, Article ID 482032, Hindawi Publishing Corporation.

[71] E. Özcan, M. Yılmaz, Particle swarms for multimodal optimization, in: Proceedings of the 8th International Conference on Adaptive and Natural Computing Algorithms, Part I, in: Lecture Notes in Computer Science, vol. 4431, Springer-Verlag, Berlin, Heidelberg, Warsaw, Poland, 2007, pp. 366–375.

[72] X. Li, Niching without niching parameters: particle swarm optimization using a ring topology, IEEE Transactions on Evolutionary Computation 14 (1) (2010).

[73] X. Li, Developing niching algorithms in particle swarm optimization, in: B.K. Panigrahi, Y. Shi, M.-H. Lim (Eds.), Handbook on Swarm Intelligence—Concepts, Principles and Applications, Springer-Verlag, ISBN: 978-3-642-17389-9, 2011, pp. 67–88.

[74] J. Barrera, C.A.C. Coello, Test function generators for assessing performanceee of PSO algorithms in multimodal optimization, in: B.K. Panigrahi, Y. Shi, M.-H. Lim (Eds.), Handbook on Swarm Intelligence—Concepts, Principles and Applications, Springer-Verlag, ISBN: 978-3-642-17389-9, 2011, pp. 89–117.

[75] H.P. Schwefel, (1974): Numerische optimierung von computer-modellen, Ph.D. Thesis. Reprinted by Birkhäuser, 1977.

[76] H.G. Beyer, H.-P. Schwefel, Evolution strategies: a comprehensive introduction, Natural Computing 1 (1) (2002) 3–52.

[77] N. Hansen, A. Ostermeier, Completely derandomized self-adaptation in evolution strategies, Evolutionary Computation 9 (2) (2001) 159–195.

[78] O.M. Shir, T. Bäck, Niching in evolution strategies, in: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO, New York, 2005, pp. 915–916.

[79] O.M. Shir, T. Bäck, Niching in evolution strategies, TR 2005-01, LIACS, Leiden University, 2005.

[80] B.L. Miller, M.J. Shaw, Genetic algorithms with dynamic niche sharing for multimodal function optimization, in: Proceedings of the 1996 IEEE International Conference on Evolutionary Computation, ICEC'96, IEEE, New York, NY, USA, 1996.

[81] O.M. Shir, T. Bäck, Dynamic niching in evolution strategies with covariance matrix adaptation, in: Proceedings of the Congress on Evolutionary Computation 2005, 2005, pp. 2584–2591.

[82] O.M. Shir, C. Siedschlag, T. Bäck, M.J. Vrakking, Niching in evolution strategies and its application to laser pulse shaping, in: Proceedings of the International Conference Evolution Artificielle, in: Lecture Notes in Computer Science, vol. 3871, Springer, Berlin, 2006, pp. 85–96.

[83] O.M. Shir, T. Bäck, Niche radius adaptation in the CMA-ES niching algorithm, in: Parallel Problem Solving from Nature, PPSN IX, in: Lecture Notes in Computer Science, vol. 4193, Springer, Berlin, 2006, pp. 142–151.

[84] O.M. Shir, M. Emmerich, T. Bäck, Adaptive niche radii and niche shapes approaches for niching with the CMA-ES, Evolutionary Computation 18 (1) (2010) 97–126. Spring.

[85] O.M. Shir, T. Bäck, Performance analysis of niching algorithms based on derandomized-ES variants, in: Proceedings of the Ninth Genetic and Evolutionary Computation Conference, GECCO, London, UK, July 2007, pp. 705–712.

[86] O.M. Shir, T. Bäck, Niching with derandomized evolution strategies in artificial and real-world landscapes, Natural Computing 8 (2009) 171–196.

[87] R. Storn, K.V. Price, Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces, Technical Report TR-95-012, ICSI, 1995. http://http.icsi.berkeley.edu/~storn/litera.html.

[88] R. Storn, K.V. Price, Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11 (4) (1997) 341–359.

[89] K.V. Price, R. Storn, J. Lampinen, Differential Evolution—A Practical Approach to Global Optimization, Springer, Berlin, 2005.

[90] S. Das, P.N. Suganthan, Differential evolution—a survey of the state-of-the-art, IEEE Transactions on Evolutionary Computation 15 (1) (2011) 4–31.

[91] R. Thomsen, Multimodal optimization using crowding-based differential evolution, in: Proceedings of the Congress on Evolutionary Computation 2004, Portland, vol. 2, June 2004, pp. 1382–1389.

[92] X. Li, Efficient differential evolution using speciation for multimodal function optimization, in: Proceedings of the Conference on Genetic and Evolutionary Computation, GECCO 2005, Washington DC, USA, 2005, pp. 873–880.

[93] B. Rigling, F. Moore, Exploitation of subpopulations in evolutionary strategies for improved numerical optimization, in: Proceedings of the Eleventh Midwest Artificial Intelligence and Cognitive Science Conference, MAICS 1999, AAAI Press, 1999, pp. 80–88.

[94] J. Rumpler, F. Moore, Automatic selection of subpopulations and minimal spanning distances for improved numerical optimization, in: Proceedings of the Congress on Evolutionary Computation, CEC 2001, vol. 1, 2001, pp. 38–43.

[95] D. Zaharie, A multipopulation differential evolution algorithm for multimodal optimization, in: Proceedings of 10th MENDEL International Conference on Soft Computing, Brno, Czech Republic, June 2004, pp. 17–22.

[96] Z. Hendershot, A differential evolution algorithm for automatically discovering multiple global optima in multidimensional discontinuous spaces, in: Proceedings of the Fifteenth Midwest Artificial Intelligence and Cognitive Sciences Conference, Chicago, April 2004, pp. 92–97.
[97] D. Zaharie, Extensions of differential evolution algorithms for multimodal optimization, in: Proceedings of SYNASC'04, 6th International Symposium of Symbolic and Numeric Algorithms for Scientific Computing, 2004, pp. 523–534.
[98] K.V. Price, J. Rönkkönen, Comparing the uni-modal scaling performanceee of global and local selection in mutation-only differential evolution algorithm, in: Proceedings of 2006 IEEE World Congress on Computational Intelligence, Vancouver, Canada, 16–21 July 2006, pp. 7387–7394.
[99] J. Rönkkönen, J. Lampinen, On determining multiple global optima by differential evolution, in: Evolutionary and Deterministic Methods for Design, Optimization and Control, Proceedings of EUROGEN 2007, Jyyvaskyla, Finland, 11–13 June 2007, pp. 146–151.
[100] J. Rönkkönen, Continuous multimodal global optimization with differential evolution-based methods, Ph.D. Thesis, Lappeenranta University of Technology, Lappeenranta, Finland, 2009.
[101] B.Y. Qu, P.N. Suganthan, Differential evolution with neighborhood mutation for multi-modal optimization, IEEE Transactions on Evolutionary Computation (2011) (in press).
[102] L. Qing, W. Gang, Y. Zaiyue, W. Qiuping, Crowding clustering genetic algorithm for multimodal function optimization, Applied Soft Computing 8 (2008) 88–95.
[103] F. Streichert, G. Stein, H. Ulmer, A. Zell, A clustering based niching method for evolutionary algorithms, in: Proceedings of the Conference on Genetic and Evolutionary Computation, GECCO 2003, in: Lecture Notes in Computer Science, vol. 2723, Springer, 2003, pp. 644–645.
[104] R.G. Reynolds, An introduction to cultural algorithms, in: Proceedings of the 3rd Annual Conference on Evolutionary Programming, World Scientific Publishing, 1994, pp. 131–139.
[105] J. Alami, A.Al. Imrani, A. Bouroumi, A multipopulation cultural algorithm using fuzzy clustering, Applied Soft Computing 7 (2) (2007) 506–519.
[106] S. Ando, J. Sakuma, S. Kobayashi, Adaptive isolation model using data clustering for multimodal function optimization, in: Proceedings of the Conference on Genetic and Evolutionary Computation, GECCO 2005, Washington, DC, USA, June 25–29, 2005, pp. 1417–1424.
[107] T. Leitão, F.B. Pereira, J. Tavares, E. Costa, Niching techniques: a study on the cluster geometry optimization problem, in: Proceedings of the Conference on Genetic and Evolutionary Computation, GECCO 2007, ACM Press, London, UK, 2007, p. 1524.
[108] P. Siarry, A. Pétrowski, M. Bessaou, A multipopulation genetic algorithm aimed at multimodal optimization, Advances in Engineering Software 33 (2002) 207–213.
[109] S. Tsutsui, Y. Fujimoto, A.R. Ghosh, Forking genetic algorithms: GAs with search space division schemes, Evolutionary Computation 5 (1997) 61–80.
[110] R.L. Lung, A sub-population stability based evolutionary technique for multimodal optimization, in: Proc. GECCO, 2004.
[111] E.L. Yu, P.N. Suganthan, An ensemble of niching algorithms, Information Sciences 180 (15) (2010) 2815–2833. Elsevier.
[112] B.Y. Qu, P.N. Suganthan, Novel multimodal problems and differential evolution with ensemble of restricted tournament selection, IEEE Congress on Evolutionary Computation (2010) Barcelona, Spain.
[113] B.Y. Qu, P. Gouthanan, P.N. Suganthan, Dynamic grouping crowding differential evolution with ensemble of parameters for multi-modal optimization, in: Proc. of Swarm, Evolutionary, and Memetic Computing, in: LNCS, 2010, pp. 19–28. doi:10.1007/978-3-642-17563-3_3.
[114] K. Deb, Multi-Objective Optimization using Evolutionary Algorithms, John Wiley & Sons, 2001.
[115] C.A. Coello Coello, G.B. Lamont, D.A. Van Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems, Springer, 2007.
[116] K. Deb, A. Saha, Multimodal optimization using a bi-objective evolutionary algorithm, KanGAL Report No. 2009006, IIT Kanpur, December 2009.
[117] K. Deb, A. Saha, Finding multiple solutions for multimodal optimization problems using a multi-objective evolutionary approach, in: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, GECCO '10, ACM, New York, NY, Portland, Oregon, USA, 2010, pp. 447–454.
[118] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002).
[119] G.V. Reklaitis, A. Ravindran, K.M. Ragsdell, Engineering Optimization Methods and Applications, Wiley, New York, 1983.
[120] R. Kumar, P. Rockett, Effective evolutionary multimodal optimization by multiobjective reformulation without explicit niching/sharing, in: Proceedings of Applied Computing, Second Asian Applied Computing Conference, AACC 2004, in: Springer Lecture Notes in Computer Science vol. 3285, Kathmandu, Nepal, 2004 pp. 1–8.
[121] C. Stoean, M. Preuss, R. Stoean, D. Dumitrescu, Multimodal optimization by means of a topological species conservation algorithm, IEEE Transactions on Evolutionary Computation 14 (6) (2010) 842–864.
[122] L.N. de Castro, J. Timmis, An artificial immune network for multimodal function optimization, in: Proceedings of the Congress on Evolutionary Computation, vol. 1, IEEE Press, Piscataway, NJ, 2002, pp. 674–699.
[123] F.O. de França, F.J. Von Zuben, L.N. de Castro, An artificial immune network for multimodal function optimization on dynamic approach, in: Proceedings of the Conference on Genetic and Evolutionary Computation, GECCO 2005, ACM, 2005, pp. 289–296.
[124] Q. Xu, L. Wang, J. Si, Predication based immune network for multimodal function optimization, Engineering Applications of Artificial Intelligence 23 (2010) 495–504.
[125] M. Dorigo, G. Di Caro, Ant colony optimization: a new meta-heuristic, in: P.J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, A. Zalzala (Eds.), IEEE Congress on Evolutionary Computation—CEC'99, IEEE Press, Piscataway, NJ, 1999, pp. 1470–1477.
[126] M. Guntsch, M. Middendorf, Applying population based ACO to dynamic optimization problems, in: ANTS'02: Proceedings of the Third International Workshop on Ant Algorithms, Springer-Verlag, London, UK, 2002, pp. 111–122.
[127] D. Angus, Niching for ant colony optimization, in: A. Lewis, S. Mostaghim, M. Randall (Eds.), Biologically-Inspired Optimization Methods, Springer, ISBN: 978-3-642-01261-7, 2009, pp. 165–188.
[128] C.-Y. Pang, X. Li, Hui Liu, Y.-F. Wang, B.-Q. Hu, Applying ant colony optimization to search all extreme points of function, in: 2010 the 5th IEEE Conference on Industrial Electronics and Applications, ICIEA, 2010, pp. 1517–1521.
[129] M. Preuss, Niching prospects, in: B. Filipi, J. Silc (Eds.), Bio-Inspired Optimization Methods and their Applications, Jozef Stefan Institute, 2006, pp. 25–34.
[130] J. Alami, A. El Imrani, Dielectric composite multimodal optimization using a multipopulation cultural algorithm, in: Intelligent Data Analysis, vol. 12, IOS Press, 2008, pp. 359–378. 359.
[131] K.-C. Wong, K.-S. Leung, M.-H. Wong, Protein structure prediction on a lattice model via multimodal optimization techniques, in: Genetic and Evolutionary Computation Conference, GECCO 2010, July 7–11, Portland, Oregon, USA, 2010, pp. 155–162.
[132] C.-G. Lee, D.-H. Cho, H.-K. Jung, Niching genetic algorithm with restricted competition selection for multimodal function optimization, IEEE Transactions on Magnetics 35 (3) (1999) 1722–1725.
[133] C.-H. Im, H.-K. Kim, H.-K. Jung, K. Choi, A novel algorithm for multimodal function optimization based on evolution strategy, IEEE Transactions on Magnetics 40 (2) (2004) 1224–1227.
[134] J.-H. Seo, C.-H. Im, C.-G. Heo, J.-K. Kim, H.-K Jung, C.-G. Lee, Multimodal function optimization based on particle swarm optimization, IEEE Transactions on Magnetics 42 (4) (2006) 1095–1098.
[135] J.-H. Seo, C.-H. Im, S.-Y. Kwak, C.-G. Lee, H.-K. Jung, An improved particle swarm optimization algorithm mimicking territorial dispute between groups for multimodal function optimization problems, IEEE Transactions on Magnetics 44 (6) (2008) 1046–1049.
[136] E. Dilettoso, N. Salerno, A self-adaptive niching genetic algorithm for multimodal optimization of electromagnetic devices, IEEE Transactions on Magnetics 42 (4) (2006) 1203–1206.
[137] O.M. Shir, M. Emmerich, Th. Bäck, M.J.J. Vrakking, Conceptual designs in laser pulse shaping obtained by niching in evolution strategiesm in: P. Neittaanmäki, J. Periaux, T. Tuovinen (Eds.), Evolutionary Methods for Design, Optimization and Control, Barcelona, Spain, 2007.
[138] D. Ackley, An empirical study of bit vector function optimization, in: Genetic Algorithms Simulated Annealing, Pitman, London, UK, 1987, pp. 170–204.
[139] K. Deb, Genetic algorithms in multimodal function optimization, the clearing house for genetic algorithms, M.S. Thesis and Rep. 89002, Univ. Alabama, Tuscaloosa, 1989.
[140] Z. Michalewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, New York, 1996.
[141] O. Shir, T. Back, Niche radius adaptation in the cms-es niching algorithms, in: Proc.9th Int. Conf. Parall. Prob. Solv. Nat., PPSN, in: LNCS, vol. 4193, Springer, Reykjavik, Iceland, 2006.