Seeking Multiple Solutions: An Updated Survey on Niching Methods and Their Applications

Xiaodong Li, Michael G. Epitropakis, Kalyanmoy Deb, Andries Engelbrecht

***Abstract*—Multi-Modal Optimization (MMO) aiming to locate multiple optimal (or near-optimal) solutions in a single simulation run has practical relevance to problem solving across many fields. Population-based meta-heuristics have been shown particularly effective in solving MMO problems, if equipped with specifically-designed diversity-preserving mechanisms, commonly known as *niching* methods. This paper provides an updated survey on niching methods. The paper first revisits the fundamental concepts about niching and its most representative schemes, then reviews the most recent development of niching methods, including novel and hybrid methods, performance measures, and benchmarks for their assessment. Furthermore, the paper surveys previous attempts at leveraging the capabilities of niching to facilitate various optimization tasks (e.g., multi-objective and dynamic optimization) and machine learning tasks (e.g., clustering, feature selection, and learning ensembles). A list of successful applications of niching methods to real-world problems is presented to demonstrate the capabilities of niching methods in providing solutions that are difficult for other optimization methods to offer. The significant practical value of niching methods is clearly exemplified through these applications. Finally, the paper poses challenges and research questions on niching that are yet to be appropriately addressed. Providing answers to these questions is crucial before we can bring more fruitful benefits of niching to real-world problem solving.**

***Index Terms*—Niching methods, Multi-modal optimization, Meta-heuristics, Multi-solution methods, Evolutionary computation, Swarm intelligence.**

I. Introduction

THIS paper presents an updated survey on niching methods, which are Multi-Modal Optimization (MMO) methods aiming at locating multiple optimal solutions in a single execution run. In many real-world situations, a decision maker prefers to have multiple optimal (or close to optimal) solutions at hand before making a final decision. If one solution is not suitable, an alternative solution can be adopted immediately. A good practical example is the well-publicized *Second Toyota Paradox* [1], which shows that delaying decisions and pursuing multiple candidate solutions concurrently can produce better cars faster and cheaper during the car manufacturer's production process.

Xiaodong Li is with the School of Science (Computer Science and Software Engineering), RMIT University, VIC 3001, Melbourne, Australia, email:xiaodong.li@rmit.edu.au.

Michael G. Epitropakis is with the Data Science Institute and the Department of Management Science, Lancaster University Management School, Lancaster University, Lancaster, UK, email: m.epitropakis@lancaster.ac.uk.

Kalyanmoy Deb is with the Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824, USA, email: kdeb@egr.msu.edu.

Andries Engelbrecht is with the Department of Computer Science, School of Information Technology, University of Pretoria, Pretoria 0002, South Africa, email: engel@cs.up.ac.za.

The goal of locating multiple optimal solutions in a single run by niching methods contrasts sharply with the goal of a classic optimization method [2], which usually starts from an initial single point and iteratively improving it, before arriving at one final solution. Since it is not guaranteed that starting at different initial points will arrive at different solutions with multiple runs, classic optimization methods are not suited for the purpose of locating multiple solutions. This goal is also different from the usual single-optimum seeking mechanism employed by a standard meta-heuristic method. In literature, sometimes "*multi-modal optimization*" also refers to seeking a single optimum on a *multi-modal* fitness landscape. To avoid this confusion and to be more precise, in this paper we also refer to niching methods as "*multi-solution*" methods.

Classic niching methods, including *fitness sharing* [3] and *crowding methods* [4], were developed in the early 70s and 80s. In subsequent years, many niching methods have been proposed. Some representative examples include *deterministic crowding* [5], *derating* [6], *restricted tournament selection* [7], *parallelization* [8], *clustering* [9], *stretching and deflation* [10], [11], and *speciation* [12], [13]. Initially, niching methods were developed for Evolutionary Algorithms (EAs). However, recently niching methods were also developed for other meta-heuristic optimization algorithms [14], such as Evolution Strategies (ES), Particle Swarm Optimization (PSO), and Differential Evolution (DE).

It is interesting to note that though several subareas of meta-heuristics, such as evolutionary multi-objective optimization (EMO) and constrained optimization, have gained widespread acceptance going even beyond the meta-heuristic research community, niching methods are perceived to have failed in making a similar impact. Research on niching methods is seen by many as a byproduct of research on population diversity preservation, which is an important issue to deal with in standard meta-heuristic algorithms. It is a common perception that niching methods have limited use in real-world problem solving because of the difficulties faced when applying them (see Section V). Nevertheless, literature review suggests that research on niching methods is continuing to demonstrate remarkable success in facilitating various optimization tasks across a wide range of real-world application areas. In recent years, niching methods have been developed taking into account the unique characteristics of new meta-heuristic methods such as PSO and DE, injecting renewed vitality to this classic optimization topic. The resurgence of research interests in MMO is clearly evident from the rapidly increasing number of research publications in this area, as shown in Figure 1. Seeking multiple optimal (or good sub-optimal) solutions in a

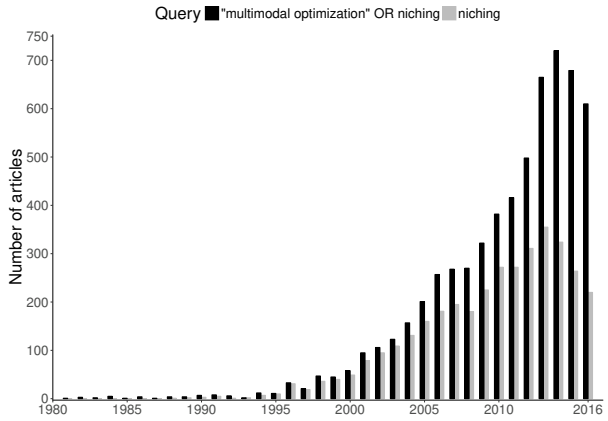

Fig. 1. Publication trends from the Elsevier Scopus library that contains (in either meta-data, or full-text) the following keyword search queries: a) "multimodal optimization" OR "niching"; and b) "niching".

single optimization run has the following benefits:

- Finding multiple solutions may help to reveal hidden properties or relations of the problem under study, e.g., the distribution of the solution set in the problem space. This provides much richer information about the problem domain than single-solution approaches.
- In some real-world problems, there may be factors that are difficult to model mathematically, e.g., degrees of difficulty in manufacturing, maintenance, and reliability. Having multiple solutions with a similar quality will give the decision maker more options for consideration, with factors that are not captured in the mathematical model.
- Finding multiple solutions with a similar quality is a step towards providing *robust* solutions, and also helping with potential sensitivity studies of the given problem.
- Seeking multiple "good" solutions may increase the probability of a meta-heuristic algorithm finding the globally optimal solution, since computational effort is not concentrated just in one area, but diverted to different regions of the search space.
- Seeking multiple "good" solutions in different regions of the search space may help with keeping a diverse population, counteracting the effect of *genetic drift*, i.e., the population losing quickly its diversity and prematurely converging to local optima.

The importance of niching methods and their applications goes beyond just meta-heuristics. There are many problem domains where the need to locate multiple solutions is prevalent, e.g., clustering, feature selection, machine learning, and numerous engineering design problems. We feel that it is about time to provide an updated survey on this classic subarea of meta-heuristics. Although a few survey papers on niching exist [15]–[18], little attention was given to niching methods applied to optimization algorithms other than EAs, neither a diverse range of real-world applications of niching methods. This updated survey will differ from these previous surveys in the following aspects:

- Instead of exhaustively covering existing niching methods in the literature, we focus on providing an updated survey of the most recent advancements in niching methods that are inspired by the development of new meta-heuristics such as PSO and DE.
- We emphasize more on revealing the intrinsic links between niching and several topics in optimization and machine learning, together with other different roles that niching can play in these areas.
- We provide a more in-depth and detailed account of niching methods on MMO benchmark test function suites, performance measures, difficulties in practical usage, and research questions yet to be addressed.
- We aim to present a more holistic view on the current state of niching methods and their applications through a list of examples of real-world niching applications. It is interesting to note that many researchers who work on various MMO problems in their respective domain areas are not aware of niching research done in other disciplinary areas. This survey hopes to increase the awareness of potential applications of niching methods across domain boundaries.

This survey begins with background information covering the fundamental concepts of niching and diversity, and their places in population-based meta-heuristics. Section III describes the two most well-known niching methods which laid the foundation to this field. Section IV then presents the most recent advancements in niching methods, in particular those from meta-heuristics other than EAs. Section V discusses the difficulties faced by niching method users. This is followed by discussions on designing benchmark functions for evaluating niching methods and performance measures, in Section VI and VII respectively. Section VIII provides a detailed account of how niching is applied across several optimization and machine learning areas, which are very revealing in terms of the influence of niching in these areas. Section IX presents a list of real-world applications. In the last two sections, we present a list of open research questions and finish with our concluding remarks.

II. Background

Both notions of *niche* and *species* can be found in natural ecosystems, where individual species must compete to survive by taking on different roles. Different species or organisms evolve to fill different *niches* (or subspaces) in the environment that can support different types of life. As remarked in [19], "A niche can be defined generally as a subset of resources in the environment. A species, on the other hand, can be defined as a type or class of individuals that takes advantage of a particular niche. Thus niches are divisions of an environment, while species are divisions of the population." In biology, a species is defined as a group of individuals of similar biological features capable of interbreeding among themselves, but not with individuals from a different group. Since each niche has a finite number of resources, which must be shared among species members occupying that niche, over time different niches and species emerge naturally in the environment. Instead of evolving a single population of individuals indifferently, natural ecosystems evolve different species (or subpopulations) to fill different niches.

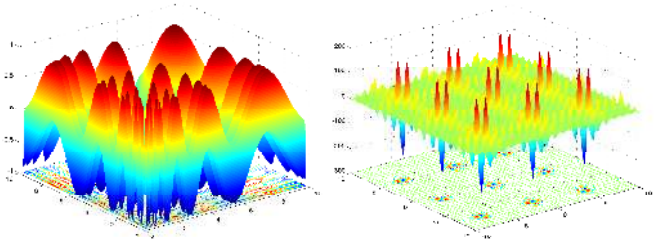

Fig. 2. Examples of fitness landscape with multiple global peaks (optima); a) Vincent 2D function with unevenly-spaced global peaks (left); b) Shubert 2D function with multiple pairs of clustered global peaks (right). More examples can be found in [21].

In optimization, we often use the term *niche* to refer to an area of the fitness landscape where only one peak resides, and *species* the subpopulation maintained around a specific peak (or niche). It is common to use some sort of distance metric to measure the closeness among individuals in the same or different species.

In the following sub-sections, we first provide a definition of MMO, then discuss the role of niching in meta-heuristics, and how it differs from maintaining population diversity.

A. Multi-modal problem formulation

Niching methods are generally designed to solve *Multi-Modal Optimization* (MMO) problems. A typical MMO problem can be expressed as follows: given a search domain $\mathcal{X}$ and an objective function f that maps elements of $\mathcal{X}$ into a real domain $\mathcal{R}$ (assuming maximization):

$$\max_{\vec{x} \in \mathcal{X}} f(\vec{x}), \tag{1}$$

where $\vec{x}$ is a n-dimensional vector $(x_1, \ldots, x_n)$. In MMO, an optimization or niching method aims to locate all possible $\vec{x}^* \in \mathcal{X}$ (not just a single $\vec{x}^*$), which obtain the maximum possible objective value:

$$f(\vec{x}^*) \geq f(\vec{x}), \forall \vec{x} \in \mathcal{X}. \tag{2}$$

The mapped f values in the immediate vicinity of an $\vec{x}^*$ should be all equal or lower than $f(\vec{x}^*)$, which maximizes the possible objective value. This is different from the notion of *local optima*: although they are surrounded in their immediate vicinity by inferior solutions, the fitness values of local optima do not exceed the highest possible value [20]. Fig. 2 shows examples of two multi-modal functions, Vincent 2D, which has a fitness landscape of multiple global peaks with vastly different basin widths, and Shubert 2D which has 9 pairs of clustered global peaks, with each pair very close to each other, but the distance between any pair being much greater.

It is also possible to relax the MMO definition to allow locating globally optimal solutions, as well as "sufficiently" good sub-optimal solutions.

B. Population diversity vs niching

In population-based meta-heuristics, *population diversity* plays an important role in maintaining the meta-heuristic algorithm's capability to explore the search space. When the population converges (or in other words the diversity loss is at its greatest), the algorithm ceases to make further progress in optimization. Usually, striking a good balance between maintaining sufficient diversity (for exploration) and refining the existing solution locally towards a good accuracy (for exploitation) is a common goal for these population-based meta-heuristic algorithms seeking to locate a single optimal solution. Several definitions were provided by Mahfoud [22] to characterize diversity in the context of EAs. In contrast, niching not only helps to maintain a more diverse population, but also helps to achieve an additional goal, that is to simultaneously locate more than one optimal solution. In retrospect, the early works in EA have been largely dominated by efforts to maintain good population diversity, and the development of early niching methods was considered only as a byproduct [23]. Niching was used largely for the purpose of preventing the best candidate solution in the population from replacing other similar quality but distant solutions.

Note that simply maintaining a high level of population diversity is inadequate for niching, since a high population diversity could be made up by random points. To induce a niching effect, a niching method must allow convergence locally to desired solutions, as well as diversity among these solutions across different regions of the search space, achieving some sort of *distributed convergence* (see Fig. 3).

III. Classic Methods

This section briefly describes two classic niching methods, *fitness sharing* and *crowding*, which had a significant influence on the development of subsequent niching methods. For other well-established niching methods, the readers are referred to [15], [16].

A. Fitness Sharing and Crowding

One classic niching method is *fitness sharing*, probably the most widely-used niching method. The sharing concept was originally introduced by Holland [24] and then adopted as a mechanism to divide the population into several subpopulations based on the similarity of the individuals in the population [3]. Fitness sharing was inspired by the notion of '*sharing*' observed in nature, where an individual has only limited resources that must be shared with other individuals occupying the same niche in the environment. Fitness sharing attempts to maintain a diverse population by degrading an individual's fitness based on the presence of other neighbouring individuals. During selection, many individuals in the same neighbourhood would degrade each other's fitness, thereby discouraging the number of individuals occupying the same niche. This in turn rewards individuals uniquely exploiting different areas of the search space. Although *fitness sharing* has proven to be a useful niching method, it has been shown that it is rather difficult to set a proper value for the niche radius σ_{share} and the scaling factor α [25], [26] without any prior knowledge of the problem. The computation of niches can also be expensive if the population size is large [15]. Later efforts in improving fitness sharing led to the development of

Algorithm 1: The pseudo-code of *deterministic crowding*.

```
Select two parents, p1 and p2 randomly, without replacement
Generate two offspring c1 and c2
if d(p1, c1) + d(p2, c2) ≤ d(p1, c2) + d(p2, c1) then
    if f(c1) > f(p1) then replace p1 with c1
    if f(c2) > f(p2) then replace p2 with c2
else
    if f(c2) > f(p1) then replace p1 with c2
    if f(c1) > f(p2) then replace p2 with c1
end if
```

several niching techniques, including *dynamic fitness sharing* [27], *dynamic niching sharing* [28], and *clearing* [12].

Whereas *fitness sharing* aims to downgrade the fitness values of overcrowded individuals, the *crowding* method relies on a competition mechanism between an offspring and its close parents to allow adjusted selection pressure in favouring individuals that are far apart and fit. The *crowding* method was initially designed only to preserve population diversity and prevent pre-mature convergence [4]. In crowding, an offspring is compared to a small random sample taken from the current population, and the most similar individual in the sample is replaced. A parameter CF (*crowding factor*) is commonly used to determine the size of the sample. Mahfoud [5] closely examined both *crowding* and *pre-selection* and found that De Jong's crowding method was unable to maintain more than two peaks of a five peaks fitness landscape due to stochastic replacement errors. Mahfoud then made several modifications to *crowding* to reduce replacement errors, restore selection pressure, and also to eliminate the crowding factor parameter. The resulting algorithm, *deterministic crowding* (DC), was able to locate and maintain multiple peaks. One merit is that DC does not assume any prior knowledge of the number of peaks or niche radius as by the sharing methods. Algorithm 1 shows the basic procedure of DC.

B. Other methods

Many other forms of niching methods have been developed, of which the most representative ones include *restricted tournament selection* (RTS) [7], *clearing* [12], *multi-national GA* [29], and *speciation* [13]. It is interesting to note that the primary goal of the early niching methods was to preserve population diversity, due to the constant battle of population diversity loss in any standard evolutionary algorithm. Using niching methods to find multiple optima was merely a byproduct of this process (see [23], p.41). Nevertheless, subsequent to early research, niching methods have been developed with the primary goal of locating multiple optimal solutions.

IV. Recent Developments

As several meta-heuristics other than EAs become increasingly popular, the properties of these new meta-heuristics have been harnessed to induce niching behaviours. This section describes two most widely-used meta-heuristics, *Particle Swarm Optimization* (PSO) and *Differential Evolution* (DE), and how they can be modified to locate multiple solutions. Another interesting development is the hybridization of niching methods with local search methods.

A. Particle Swarm Optimization

Particle Swarm Optimization (PSO) is an optimization technique inspired by the bird flocking behaviour [30]. In PSO, each particle has its own memory of the best-known position visited so far, and is able to share this information with other particles in the swarm. At each iteration, each particle is propelled towards the area defined by the stochastic average of its own known best position and the swarm best position. It was shown in [31] that basic versions of PSO do not have the ability to locate multiple solutions. This is mainly due to the loss of swarm diversity as particles converge. In order to niche, approaches have to be implemented to promote exploration and distributed convergence.

The notion of memory associated with each particle is unique to PSO, and this property can be used to induce niching behaviour: a swarm can be divided into two parts, an *explorer-swarm* consisting of the current particles, and a *memory-swarm*, comprising of only best-known positions of individual particles. The explorer-swarm tends to explore the search space more broadly, whereas the memory-swarm tends to be more stable, providing an archive of best positions found so far by the entire swarm. If a restricted communication topology (e.g., a ring topology) is mapped over the particles of the swarm, these particles will be attracted only towards its local neighbourhood best positions identified in the topological space. As search proceeds, individual niches are formed naturally around different optima, eventually leading to locating multiple optima. *Neighbourhood* can be defined either in the topological space or decision space. A few methods explicitly exploit this PSO property, e.g., the ring-topology-based niching PSO [32], and the Fitness-Euclidean distance Ratio (FER-PSO) [33], where the fitness-Euclidean distance ratio is used to drive particles towards their nearest-and-fittest neighbourhood bests. In addition, an Euclidean distance-based niching PSO, namely LIPS [34], forms niches by using the *nearest neighbours* to each personal best in the Fully Informed PSO (FIPS) [35].

Several methods proven to be useful to induce niching behaviour in the classic niching methods have also been adopted to work with PSO. For example, a *stretching* method was used in [36] to modify the fitness landscape to allow potentially good solutions to be isolated from other particles, which is similar to the effect of a *derating* method [6]. It was, however, shown in [37] that the stretching PSO introduces false local optima. The Derating Niche PSO was proposed to avoid these false local optima. The idea of speciation was also adopted in Speciation-based PSO (SPSO) [38], [39], where species (or sub-swarms) can be adaptively formed around different optima. However, a niche radius must be pre-specified in order to define a species. Species are allowed to be merged or separated into new ones at each iteration. Other similar methods include *NichePSO* [40], *nbest* PSO [41], and Multi-swarms [42].

To remove the need to pre-specify a niche radius, an Adaptive Niching PSO (ANPSO) [43] was developed to adaptively determine this parameter by calculating population statistics at each iteration. Another method that avoids the specification of a niche radius parameter is a vector-based PSO (VPSO) [44] where niche identification is done by carrying out vector operations on the vector components of the velocity update. A niche is determined by the radius value based on the distance between the swarm best and the nearest particle with a negative dot product (i.e., moving in an opposite direction). Nevertheless, these methods tend to introduce new parameters that may still be sensitive to the induction of niching effect. For further information on recent PSO niching methods, the reader can refer to [45], [46].

B. Differential Evolution

Unlike PSO, Differential Evolution (DE) [47] makes use of scaled differences between randomly sampled pairs of individuals in the population to determine how to modify individual vectors to produce offspring. Considering that the distribution of these sampled individual pairs reflects the topographic feature of the search space, DE's search behaviour to some extent is self-adaptive to the fitness landscape of the search space.

Rönkkönen introduced several interesting ideas for global and local selection in DE [48], and how to use the local selection concept for MMO [49]. Since this local selection method requires only the offspring to compete against its own parent, it is similar to *deterministic crowding* (DC) used in the EA context. Like DC, it also has the advantage of not having to specify additional niching parameters.

Further studies on the dynamics of DE [50], [51] reveal that DE individuals are inclined to cluster around either local or global optima after some iterations. A clustering tendency statistic, H-measure, was suggested in [50] to measure the varying degrees of clustering tendency that may occur for six classical DE variants. Inspired by this observation, the mutation operator in a classic DE variant DE/rand/1, was altered to induce niching behaviour without the need of adding any extra control parameter [52], namely the DE/nrand/1. More specifically, instead of using the base vector in the usual way, its nearest neighbour is always chosen as the actual base vector:

$$v_{g+1}^{i} = x_{g}^{NN_i} + F(x_{g}^{r_1} - x_{g}^{r_2}), \tag{3}$$

where $x_{g}^{NN_i}$ is the nearest neighbour of the current individual $x_{g}^{i}, r_1, r_2 \in \{1, 2, \ldots, NP\} \backslash \{i\}$ are random integers mutually different and not equal to the running index i, and F is the scaling factor. A similar nearest-neighbour idea for a niching PSO was also adopted [34]. Intuitively, such mutation scheme distributes the new offspring individuals to exploit the vicinity of their nearest neighbours, while exploration is attained by the scaled differences of randomly selected vectors. The proposed mutation modification is generic and a family of new niching DE variants can be produced, i.e., the DE/nrand family of algorithms.

One appealing aspect of DE/nrand/1 is its simplicity in implementation, requiring only addition of a few lines of codes to the standard DE source code[1]. Figure 3 illustrates a series of snapshots of a simulation run of the DE/nrand/1 on the Shubert 2D function, after 0, 20, 50 and 100 iterations. One can clearly observe a strong niching effect, i.e., clustering tendency of the distributed individuals in the immediate vicinity of the global optima (in other words, *distributed convergence*), during the simulation run.

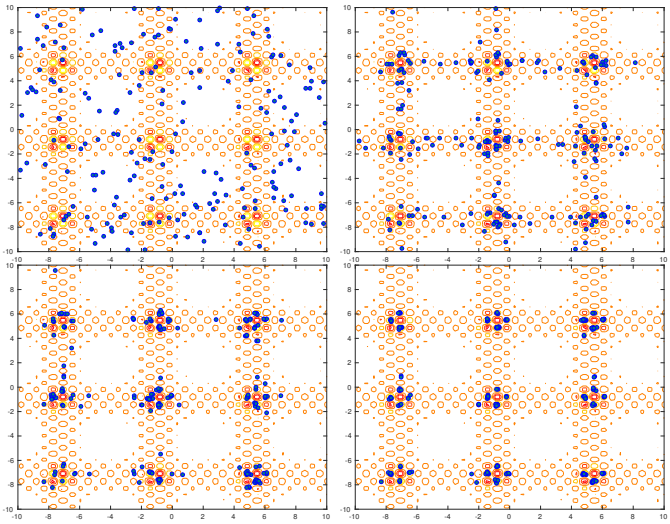

Fig. 3. Snapshots of a simulation run of DE/nrand/1 on the Shubert 2D function, at the 0th, 20th, 50th, and 100th iterations (clockwise).

The ability of a niching method to locate and maintain a large number of "optimal" solutions is heavily dependent on the chosen population size. A dynamic archive niching DE, $(dADE/nrand/1)$, was introduced in [53] to overcome this dependency issue. The dynamic archive mechanism, similar to [54], along with a re-initialization strategy, was incorporated into the DE/nrand family to achieve better efficiency in maintaining high quality solutions found during the optimization run while retaining the algorithm's exploratory search power.

In a neighbourhood-based niching DE [55], similar vectors within the neighbourhood of each base vector were used to define the DE mutation operator in order to induce niching behaviour. Here a difference vector is only generated from "similar" individuals of the DE population rather than randomly selected from the entire population. Similarity is defined for each base vector as the niche, or sub-population, that contains its m-th closest individuals in terms of Euclidean distance. As such, each individual is mutated by randomly selecting individuals within the niche of its base vector. Note that these niches are overlapped with each other. The parameter m is user-defined and needs to be fined-tuned according to the problem characteristics. Three well-known niching techniques were used in conjunction with this scheme, namely crowding [5], speciation [12] and a modified version of the fitness sharing technique [3], resulting in three different niching DE variants.

Similarly, Biswas *et al.* [56] recently proposed three niching

[1]Price & Storn's original implementation DeMat with the DE/nrand/1 modification by only using six new lines of MATLAB code. Available at: https://github.com/mikeagn/DeMatDEnrand

DE variants by incorporating a probabilistic parent selection scheme based on fitness and proximity information, known as *localized shared information*. More specifically, parent selection in the mutation stage is replaced by a probabilistic scheme to increase the probability of selecting fitter individuals that are closer to the target vector. This is a trade-off that is usually needed to induce a niching effect and minimize the tendency of changing basin of attractions without hindering the exploratory ability of the considered algorithm.

One key common characteristic of niching methods is the incorporation of proximity information of the evolving population into the search operations. Such information unavoidably increases the complexity of the niching method. To mitigate this issue, Zhang *et al.* [57] recently proposed a fast niching algorithm to calculate approximate nearest neighbours using a hashing mechanism (namely Locality Sensitive Hashing), instead of exact pairwise distances within a solution set. In this scheme, potential solutions are projected to a number of buckets by a hash function, where similar solutions have a higher probability to be assigned in the same buckets than dissimilar ones. To induce a niching effect, search operations in DE or PSO identify niches by solutions that lie in the same bucket. The complexity of this fast niching algorithm is proved to be linear to the population size. Another approach using approximate neighbourhoods is the index-based neighbourhoods in DE [51], which can also substantially decrease the complexity of the niching algorithm.

Other recently proposed DE niching variants include parent-centric mutation strategies combined with crowding [58] and ensembles of niching techniques such as speciation [59].

C. Other meta-heuristics

The previous section on classic and recent niching methods are far from complete. Obviously niching can be introduced to other meta-heuristics as well, such as Artificial Immune Systems (AIS) [60], [61], Ant Colony Optimization (ACO) [62]–[64], and Cultural Algorithms (CA) [65]. It is also possible to induce niching behaviour through probabilistic modelling building, e.g., via an Estimated Distributed Algorithm (EDA) [66]. Please refer to [17] for further information on these niching methods.

D. Hybrid methods

Hybrid methods combining meta-heuristics with local search (or hill climbing), e.g., the quasi-Newton or NelderMead simplex methods, have shown great promise for global optimization [67]. These hybrids derive their enhanced problem solving capability by harnessing both the explorative search power of the meta-heuristics and the refining capability (i.e., exploitation) offered by a local search method. They are also commonly referred to as Memetic Algorithms (MA) [68]. Attempts have been made to hybridize niching methods with local search procedures, in order to enhance convergence to multiple optima, or in other words, *distributed convergence*. For example, regression was incorporated into Speciation-based PSO (i.e., rSPSO) for improving local convergence on both static and dynamic multi-modal landscapes in [69]. The faster and more accurate local convergence is achieved by using regression computed based on only a handful of existing individuals in the population. An EA hybridizing the Nelder-Mead simplex method with clearing was proposed in [70]. Gradient descent was used in conjunction with a dynamic niche sharing algorithm applying mating restriction [71], with the results showing that this hybrid method performed better than using niching methods alone. Quasi-Newton local search was combined with a sharing GA as well as an artificial immune algorithm by Ono, *et al.* [72], which resulted in the hybridized algorithms outperforming those methods using niching alone, on high-dimensional multi-modal functions.

V. Difficulties Facing Niching

In this section we discuss several difficulties faced by the users of niching methods.

A. Maintaining found solutions

In the early days of niching algorithm development, it was observed that both sharing and crowding methods tended to have difficulty in maintaining found optima. Subsequent research aimed at designing enhanced niching methods so that they can maintain found solutions stably until the end of a run. Any loss of found optima would be considered a failure. However, most researchers now accept the fact that the population does not have to fully converge to single solutions each corresponding to a single optimum. For example, we can store the found solutions into an archive [53], [54], separate from the running EA population (similar to a Tabu list), or we can let the population reach some kind of *equilibrium* state, as shown in [32]. In an equilibrium state, some individuals of the population would keep oscillating around a stable state, never reaching complete convergence.

B. Specifying niching parameters

The difficulty in specifying niche parameters has been a major impediment to the use of niching methods in practice. The most representative one is *niche radius*, which needs to be specified to indicate how far apart the optima are from each other. For some search landscapes, using a fixed uniform niche radius is likely to fail, e.g., the Vincent function which has irregular uneven-spaced optima (see Fig. 2 (left)).

Many attempts have been made specifically to address this issue. They can be mostly categorized into the following:

- *Attempting to find a single uniform niche radius*. For example, in [73], a radius function and a cooling procedure similar to simulated annealing were adopted. However, this method GAS (S for species) introduced several new parameters that must be specified by a user. In [74], Dick proposed a *local sharing* method where the information about the fitness landscape during a run is first collected and then subsequently used to adapt the niche radius parameter value.
- *Instead of using a fixed niche radius, several studies suggest to adopt a variable niching radius*. More specifically, each niche has its own niche radius, independent

from other niches. Niching algorithms that follow this approach include multinational GA [29], forking GA [75], Dynamic Niche Clustering [76], an adaptive extension of NichePSO (ANPSO) [43], and a CMA-ES niching algorithm [23], [77]. Inspired by the notion of self-adaptation in evolution strategies, the CMA-ES niching algorithm allows each individual to adapt its own niche radius along with other adaptive strategy parameters.

- *Avoiding to specify the niche radius*. These methods include tagging [78], deterministic crowding (DC) [5], implicit fitness sharing [79], multi-national GA (MGA) [29], population index-based niching PSO [32], vector-based PSO (VPSO) [44], DE niching method using the nearest neighbour of the current individual as the new base vector [52], [53], and locally-informed PSO (LIPS) using the nearest neighbours (measured in the decision space) to each particle's personal best [34]. DC randomly selects two parents for crossover and mutation. The two offspring generated are compared with the parents. The children only replace the nearest parent if their fitness values are greater. This process does not rely on any niching parameters. In MGA [29], instead of using a radius, a *hill-valley detection* mechanism is required to take some sampling points between two individuals to see if they belong to the same peak. However, accurate detection could be an expensive exercise. Further attempts were made to improve the hill-valley detection but still required pre-specification of certain parameters [80]. Recently, a history-based topological speciation algorithm (HTS) was developed to recursively construct a sequence of sample points between two individuals using search history to determine if they belong to the same species [81].

One common issue among the above-mentioned approaches is that the removal of the niche radius parameter sometimes unavoidably introduces new parameters, some of which may still be difficult to specify in practice.

C. Scalability

Scalability of niching methods refers to two aspects: the number of dimensions and the number of optima to be located. Most of the niching methods in literature have been evaluated on low-dimensional test functions. The scalability of several niching methods including *NichePSO* has been studied [82], [83], but only functions up to 5 dimensions were used. In [23], the CMA-ES niching algorithm was only tested on functions up to 10 dimensions. In the most recent CEC'2013 niching benchmark suite [21], functions up to 20 dimensions were proposed for the competition. It can be envisaged that the performance of niching methods would degrade rapidly even though they may perform well on low-dimensional problems. Furthermore, if the number of optima increases, how does a niching method respond? Another question that needs to be answered here is whether all optima (global and good local ones) need to be found. In particular, in real-life scenarios, the number of optima is often unknown, in which case it seems unreasonable to demand a niching method to locate all the optima, e.g., millions of optima. Perhaps a niching method with an adaptive population size, identifying a pre-specified number of optima would be a more appropriate strategy.

Very few attempts have been made towards answering these questions. One noticeable work on niching scalability is given in [84]. The authors suggested that for low-dimensional problems, we should use sequential niching methods [6], [11] where collision avoidance mechanisms were implemented to avoid finding the same optima repetitively. However, for high-dimensional problems, parallel niching methods with automatic restarts should be considered. As soon as an optimum is found, it is archived, and its search capacity is reused by randomly generating it anew. The authors named this process as *niche deactivation* in [85], [86]. Using an archive seems to be an effective mechanism dealing with this situation, as also demonstrated in [54] and in [53], where adaptive archive mechanisms combined with restart techniques can substantially enhance the performance of the simple DE [53] and PSO [54] variants.

D. Measuring performance

In many real-world situations, we do not always know the true global optima, their objective values, or the number of optima. This fact renders most existing performance metrics on niching (as discussed in section VII) unusable. There are still open research questions on how to design better metrics for comparing niching methods more fairly, as well as being more informative to the decision makers.

VI. Benchmark Test Functions

To empirically evaluate and compare the strengths and weaknesses of different niching methods, it is important to use a set of multi-modal test functions representing different characteristics and various levels of difficulty. The earliest work on designing multi-modal benchmark test functions was carried out by Deb [87] in his master thesis, where, five 1 or 2-dimensional test functions, each with several peaks with varying heights, equal or unequal distances between these peaks, were defined. A more challenging multi-modal test function with millions of local optima and 32 global optima was proposed by Goldberg *et al.* [25].

Several efforts have been made to design multi-modal test function generators. In particular, Rönkkönen *et al.* [88] suggested some general guidelines when designing such multi-modal function generators. The authors suggested that the following desirable features should be considered for the function generator: ease of use and being tunable; functions transformable from separable to non-separable; regular and irregular distributions of optima; controllable number of global and local optima; scalable to different dimensions; software easily expandable and freely available; and facilitating performance measures. With these guidelines, Rönkkönen *et al.* [48], [88] developed a versatile and flexible test function generator based on several suitably-designed and tunable function families, such as the $cosine$ and $quadratic$ function families. Functions can be rotated to a random angle, and each dimension of a function can be stretched independently using Bezier curves [89] to decrease regularity. The $quadratic$

family can be used to generate completely irregular landscapes and allows the number of optima to be defined independently of the number of dimensions. Any number of global and local optima can be determined by a user. This function generator is easily tunable and can offer a wide variety of landscape characteristics and difficulties. Other efforts were also made to produce irregular landscapes by using a Gaussian density function [90] or generic hump functions [16].

Furthermore, Qu and Suganthan [91] proposed a set of multi-modal test functions derived from some early work on composition functions by Liang *et al.* [92]. These composition functions, which are constructed by combining several simple basis functions, can have a complex and rugged landscape, posing difficulties to existing niching methods. Qu *et al.* [93] constructed and adopted several simple and composition multi-modal functions, making them scalable according to dimensionality. Moreover, a suite of multi-modal constrained test functions was proposed by Deb and Saha in [94]. The technical report on "Benchmark Functions for CEC'2013 Special Session and Competition on Niching Methods for Multimodal Function Optimization" [21] is the latest effort in providing a unifying framework for evaluating and comparing niching methods. The CEC'2013 competition is the first attempt to create a common platform that encourages fair and easy comparisons between different niching methods across a range of difficulty levels. Twenty test functions ranging from simple and low-dimensional to challenging and high-dimensional were introduced. Some functions are scalable and tunable in terms of dimensionality and the number of optima. Two performance measures designed for evaluating and comparing different niching methods, i.e., *peak ratio* and *success rate*, are adopted as performance measures for this competition. Two competitions CEC'2013 and CEC'2015 have been held using this benchmark suite, which involve comparing more than 20 different participating niching algorithms. The top 5 entries are provided in Table I[2].

The winning entry of the CEC'2013 niching competition was *NEA2* (Niching with CMA-ES via NBC) [95], following the basic idea of nearest-better-clustering (NBC). NBC assumes that the best individuals in the population are usually located at different basins of attractions, further away from each other, and that the distances between them are usually larger than the average distance between all individuals and their nearest better neighbours. As such, NEA2 first creates a spanning tree among all the individuals, and then connects each individual to its nearest better neighbour (according to fitness). Next it identifies the attractors (sub-populations) via clustering done by cutting the longest edges of the graph that are larger than the average distance between all individuals and their nearest better neighbour. The connected sub-graphs (clusters) that remain are the predicted attractors (or niches). For each attractor, CMA-ES is employed (in parallel) to search its neighbourhood. NEA2 has shown promising performance. However, two parameters, i.e., the maximum number of niches and the previously mentioned scale factor, must be specified by the user.

[2]For further information on the latest activities on niching methods, please visit the *IEEE CIS Task Force on Multi-modal Optimization* website: http://www.epitropakis.co.uk/ieee-mmo/

TABLE I
TOP 5 ENTRIES FROM BOTH CEC'2013 AND 2015 NICHING METHODS FOR MULTIMODAL FUNCTION OPTIMIZATION COMPETITIONS.

Algorithm	Statistics			Friedman's Test	
	Median	Mean	St.D.	Rank	Score
NMMSO [20]	**0.9885**	**0.8221**	0.2538	**1**	16.1900
NEA2 [95]	**0.8513**	**0.7940**	0.2332	**2**	16.1150
LSEAEA [96]	**0.9030**	**0.7477**	0.3236	**4**	14.5050
dADE/nrand/1 [53]	0.7488	**0.7383**	0.3010	5	14.2450
LSEAGP [98]	**0.7900**	0.7302	0.3268	**3**	14.7550

Some algorithmic analysis of the CEC'2013 competition top-ranked entries in [20], [96] identified several techniques offering significant advantages: self-adaption of search parameters [97], dynamic niche maintenance [53], and exploitative local search [95]. Leveraging on these results, Fieldsend proposed a more competent niching algorithm, *NMMSO* (Niching Migratory Multi-swarm Optimiser) [20], winning the most recent CEC'2015 niching competition. NMMSO employs multiple swarms, each having strong local search, fine-tuning its local niche estimates. At each iteration, swarms which have improved their niche estimate are paired with their closest adjacent swarm to see if they should merge (thus preventing duplication of labour). Niches in new areas are searched and identified by splitting particles from existing swarms.

VII. PERFORMANCE MEASURES

Early studies of niching methods focused more on measuring the difference between the distribution of a final EA population from a goal-distribution [22]. Deb and Goldberg proposed a *Chi-square-like performance statistic* [99], which measures the deviation of the actual distribution of the individuals X_i from the goal distribution mean μ_i (with variance δ_i^2) in all the i sub-spaces (q niche sub-spaces plus the non-niche subspace):

$$\chi^2 = \sqrt{\sum_{i=1}^{q+1} \left(\frac{X_i - \mu_i}{\delta_i} \right)^2}, \tag{4}$$

where X_i denotes the actual number of individuals in the ith subspace (following a standard normal distribution), and μ_i denotes the ideal number of individuals in the ith subspace, with δ being the standard deviation. Both μ_i and δ can be calculated from the known optima of a multi-modal function. If the number of individuals in every niche equals the mean of that niche, the χ^2 value will be zero. The smaller χ^2 value, the better of the distribution is.

Instead of comparing two distributions such as χ^2, a metric that measures directly the quality of the final solutions as well as the number of optima is the *Maximum Peak Ratio* (MPR) [28]. Assuming maximization, the metric is defined as follows:

$$MPR = \frac{\sum_{i=1}^{q} f_i}{\sum_{i=1}^{q} F_i}, \tag{5}$$

where f_i denotes the best fitness value of the individual on the ith peak (or optimum), and F_i represents the fitness value of the ith peak. Assuming all q optima are known *a priori*.

Basically, MPR defines the ratio of the sum of fitness values of the obtained optima divided by the sum of fitness values of the actual optima. MPR can be measured over time, to see how a niching algorithm behaves in terms of *niching formation acceleration* [23]. A logistic function can be used for curve-fitting in order to obtain *niching formation acceleration*.

Success Rate (SR) can be used to measure the percentage of runs in which all the optima are located. The success rate is generally well-correlated with the MPR [23].

Both χ^2 and MPR metrics (Equation (4) and (5)) assume that the number and locations of the global optima are known *a priori*, which is very unlikely in practice. An alternative performance metric which does not make this assumption was proposed in [84]:

$$sc'(P, \theta_l, \theta_u) = \sum_{\{x_i \in P | f(x_i) > \theta_u\}} \frac{f(x_i) - \theta_l}{\theta_u - \theta_l}, \qquad (6)$$

where P denotes a set of candidate solutions. This metric allows the decision maker to select a threshold internal [θ_l, θ_u], covering a range of objective values that are regarded as *interesting*. A fitness value above the lower bound θ_l can be judged as interesting, whereas the upper bound θ_u is set to some reachable fitness value. A real-world example in bioinformatics is given in [85] to show how to set θ_l and θ_u in practice. Given the interval [θ_l, θ_u], we can compute the score sc' (using Equation (6)), which is within [0, s_{max}]. Note that s_{max} may be unknown since the number of optima is unknown.

When checking if an optimum is reached within a certain level of *accuracy*, a threshold ϵ (usually some small value) can be supplied. It is possible to evaluate a niching method over a range of such threshold values [21], [85], so that its ability to obtain optimal solutions accurately can be appropriately assessed.

It may be possible to adapt some ideas from EMO performance metrics for the purpose of MMO, since both MMO and EMO methods emphasize the need to locate a set of solutions. In [100], Preuss and Wessing provided a review on both EMO and MMO performance metrics, discussing the similarities and differences between the two. It remains unclear about what would be the appropriate number of solutions in a solution set, since in most real-world situations, neither a too large nor a too small number is preferred by a decision maker. The authors suggested a metric named *Representative 5 Selection* (R5S) to emphasize that a niching method should aim to select around 5 diverse but good solutions. Clearly, none of the studied indicators is perfect, requiring future effort to improve and fine-tune their capabilities.

Recently, Mwaura et al. [101] provided a review of niching algorithm performance measures. A derivative-based performance measure that does not require any knowledge of the number of optima nor their positions was proposed.

VIII. Niching in Specialized Tasks

Niching not only helps to provide more effective problem solving in a diverse range of tasks (as shown in Fig. 4) but also sometimes benefits itself from its interaction with these areas. This section provides some examples of such interactions.

Fig. 4. Niching methods for a diverse range of problem solving.

A. Multi-objective Optimization

Multi-objective optimization using meta-heuristics has gained great popularity in recent years. While MMO using niching methods emphasizes the aspect of locating multiple good but different solutions in the decision space, multi-objective optimization using meta-heuristics (i.e., EMO) focuses more on the aspect of producing a set of *trade-off* solutions in the objective space. What is in common here is that both approaches produce a set of solutions from which a decision maker can choose from. However, for niching methods, solutions produced are not required to be in conflict.

When using a population-based meta-heuristic algorithm, diversity maintenance is normally required for spreading out solutions in the decision space. However, in EMO, it is the objective space where solution diversity is most often and explicitly maintained. This is usually done by using some niching methods. An early example is the Niched-Pareto GA (NGPA) [102], which is a multi-objective GA using a variant of fitness sharing to maintain Pareto solution diversity in the objective space. Another example is the crowding distance metric used in NSGA-II [103]. Much attention has been given to maintaining solution diversity in the objective space. However, little attention has been given to how to maintain solution diversity in the decision space, with only a few exceptions, e.g., a probabilistic model-based EMO algorithm was designed in [104] to explicitly promote diversity of solutions in both decision and objective spaces simultaneously. Another example is the *Omni-Optimizer* [105].

1) Niching in EMO: An interesting extension of NSGA-II to a more generic optimization method, the so-called *Omni-Optimizer* [105], allows degeneration of NSGA-II into a single objective MMO method (i.e., a niching method). In this case, a variable space crowding distance metric is used to encourage distant solutions in the decision space to remain in the population (see Fig. 5). Consequently, distant solutions with similar or equal objective function values will tend to survive to the end of an optimization run. This would achieve the same effect as niching. One highly desirable feature of *Omni-Optimizer* is that it does not require any additional parameters such as niche size or radius. Furthermore, *Omni-Optimizer* can degenerate to a niching method for multi-objective MMO

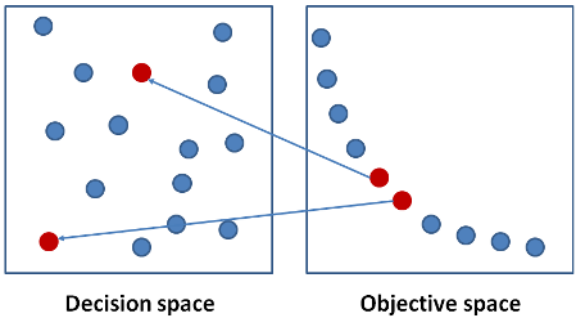

Fig. 5. An example where two solutions that are close in the objective space but their corresponding points in the decision space are further apart.

(MOMMO), capable of finding multiple Pareto-optimal fronts. The versatility of *Omni-Optimizer* is remarkable.

2) Many-objective optimization: Research on many objectives (i.e., more than 4 objectives) has been very active in recent years. One popular approach is to use an Achievement Scalarizing Function (ASF) [106] to decompose a multi-objective problem into several subproblems using a set of predefined weight vectors. A representative example of such a decomposition-based method is MOEA/D [107]. A crucial part of MOEA/D is to use a niching parameter to select two solutions associated with neighbouring weight vectors, in order to produce offspring solutions.

In the subsequently proposed NSGA-III [108], the original framework of NSGA-II is kept, but significant changes have been made to the selection operator. In NSGA-III, a set of well-spread reference points is used for maintaining population diversity. It is achieved by associating each individual in the population with a reference point. More specifically, a reference line is defined by drawing a line from a reference point to the origin of the hyper-plane, then each individual is considered to be *associated* with this reference line if the individual has the closest perpendicular distance to it. Here, niche preservation is done through favouring an individual that is the closest to the reference line of each reference point. NSGA-III does not require any additional parameter, but niching is established by ensuring at least one available population member is selected for each reference line. In another variation of NSGA-III, namely U-NSGA-III [109], which allows more than one individuals to be associated with each reference line, a niching-based tournament selection is used to introduce more selection pressure and better diversity across multiple reference lines.

3) Multi-objective formulation of a multi-modal problem: Deb and Saha demonstrated in [94], [110] that a single-objective MMO problem can be transformed into a suitable bi-objective optimization problem. Here, the second objective can be defined based on two approaches: a) the gradient information of the first objective function (to differentiate those weak Pareto-front consisting of global and local minimums, secondary derivatives are incorporated into this second objective function); b) the number of sample neighbouring solutions that are better than the current solution, which is more practical when the gradient information is unavailable. This bi-objective approach was shown to scale well with a large number of optima (up to 500) and higher dimensionality (up to 16 variables). To cut down the number of sample neighbouring points required, as the number of dimensions n grows, the Hooke-Jeeves search is adopted to evaluate only $2n$ sample neighbouring points instead of 2^n for the second objective. This is one of a few studies which proposed a number of scalable constraint multi-modal test problems. Another method for defining the second objective is to use the mean Euclidean distance of a solution from other individuals of the population as proposed in [111], which has the merit of avoiding the sampling evaluation cost. Furthermore, to avoid cases where the multiobjectification of the problem does not lead to conflicting objectives, Wang et al. [112] proposed a transformation of a multi-modal problem to a multi-objective optimization problem that always leads to conflicting objectives based only on information from its decision variables and fitness values.

4) Multi-modal formulation of a multi-objective problem: It is shown in [113] that it may be beneficial to formulate a multi/many objective optimization problem into a multi-modal scalarized single-objective problem, where each Pareto-optimal solution can be treated as an independent optimum of a multi-modal fitness landscape. More specifically, multiple reference points and weight vectors are used to obtain multiple Pareto-optimal solutions, with each solution corresponding to a single reference point and weight vector combination. A niche-based EA (MEMO) is then used to find these Pareto-optimal solutions in a single simulation run. One advantage of MEMO is that there is no need to employ non-dominated sorting, thereby making it more effective and computationally efficient on many-objective problems. MEMO was found to provide superior performance on both unconstrained and constrained multi-objective optimization problems when compared with existing state-of-the-art approaches.

5) Diversity in decision space: The popular CMA-ES [97] was also integrated with niching methods [23], and was further extended in [114] to solve multi-objective optimization problems, with a particular emphasis on promoting decision space diversity. It was shown in [114] that the multi-objective Niching-CMA method can produce a more diverse set of *efficient* solutions (i.e., solutions in the decision space), without sacrificing objective space diversity. One drawback of this method is the introduction of several user-specified parameters which may be difficult to tune. Another example is provided in [115], where a niching method is explicitly used to approximate Pareto-optimal solutions in both objective and decision spaces, resulting in finding two equivalent Pareto-subsets of solutions for the TWO-ON-ONE problem. If a standard NSGA-II was used, half of the Pareto-optimal solutions would have been neglected, as the solutions tended to converge to only one of the two niches.

B. Dynamic Optimization

In a dynamic environment where a problem itself may change over time, the key objective of a meta-heuristic algorithm is not only to locate the global optimum, but also to keep track of the optimum or relocate a new global optimum

if the problem changes over time [116]. Merely maintaining population diversity is often inadequate. Instead, some sort of *distributed convergence* is more desirable.

Niching methods can be used to track or relocate the global optimum more effectively. As observed in a number of multi-population based methods [39], [42], [117], [118], a useful strategy to ensure good tracking of the global optimum in a dynamic environment, is to maintain multiple species at all the optima found so far, regardless whether they are globally or locally optimal. This is because in a dynamic environment, optima may change in locations, heights, and/or shapes, as well as the widths of the basins. By maintaining individual species at each local optimum, it helps tremendously in cases where such a local optimum turns into a global optimum. A hierarchical clustering-based multi-population method was shown in [119] to track changing optima, even without an explicit mechanism for change detection.

To speed up local convergence, one could use individuals in the population and their fitness evaluations accumulated so far during the optimization run to estimate and predict the positions of the changing optima. A simple regression method with Speciation-based PSO (so called rSPSO) [69], [120] shows significantly better performance than several other multi-population methods such as mQSO [42]. The regression method can be substituted by other surrogate models such as Kriging [121]. Multiple surrogates can also be used to model each niche on a multi-modal fitness landscape. As demonstrated by Fieldsend [98], such localized surrogates using only local information in the vicinity of a local optimum can save a substantial amount of time than the more commonly-used method that models the entire fitness landscape. The readers are referred to [122] for more recent developments on surrogate modelling.

As shown in [123], it is also possible to make use of the directional information provided by the particles in a swarm (namely a vector-based PSO) to adaptively form niches in parallel in an effort to track multiple dynamically changing optima in a dynamic environment.

C. Bilevel Optimization

Bilevel optimization involves two levels of optimization of tasks, in which a feasible solution to the upper level optimization corresponds to an optimal solution of the lower level optimization problem. Such a nested structure of dependency makes bilevel optimization problems very challenging [124]. Niching can be done for bilevel optimization at either the upper or lower level, or both. In [125], several test functions have been constructed to show that for any given set of variables at the upper level, there may exist multiple global solutions at the lower level. Niching in bilevel optimization makes these problems extremely challenging to solve by any types of optimization problems and should be of interest to algorithm developers.

D. Clustering

The goal of clustering is to group data points into clusters such that points in each cluster have a high degree of similarity, whereas points in different clusters have a high degree of dissimilarity. A similarity metric is often based on some distance measured between these data points, e.g., Euclidean or Mahalanobis distance can be used. Since both clustering and niching share some common features, e.g., data points can be seen as individuals or clusters as niches, it is not difficult to see that clustering methods can be used to do niching, and vice versa.

1) Clustering for niching: The classic k-means clustering technique [126] can be easily incorporated into a niching method to identify niches, assuming that the number of clusters k is known *a priori*, or can be adapted. For example, an adaptive k-means clustering-based niching algorithm was developed in [9], with an aim in particular to improve the efficiency of the sharing methods. Essentially, the k-means clustering method is used to subdivide the population into clusters (or niches), but instead of having to compute the niche count parameter in the classic sharing method, the distance between an individual to the centroid of each cluster is calculated. Initial candidate points for centroids are critical here, so before applying clustering, the algorithm first sorts the population in descending order according to fitness values, giving the best-fit individuals higher preferences as initial centroids. The cluster centroids are recalculated and the number of clusters updated at each iteration. Two new parameters d_{min} and d_{max} (i.e., the minimum and maximum allowable distances between any two niche centroids) were introduced to determine an individual's membership to a niche.

Among other clustering-based niching methods are dynamic niche sharing [28], dynamic niche clustering [76], and dynamic fitness sharing [27]. The species conserving genetic algorithm (SCGA) [13] and the topological species conservation algorithm (TSC) [80] could also be considered as belonging to this category. More advanced clustering methodologies that do not need information on the number of clusters *a priori* have been combined with EAs to simultaneously locate more than one (global and/or local) optimal solutions [127], [128].

2) Niching for clustering: What is more interesting is that a clustering problem can be formulated as an MMO problem, and be handled by a niching method [129], [130]. Some early attempts at using genetic algorithms for clustering [131] indicated several challenges: the problem representation often leads to an explosion of the search space as the data set grows larger; the algorithms tend to be sensitive to initialization and noise; crossover often produces meaningless solutions. An unsupervised niche clustering algorithm (UNC) was proposed to combat these aforementioned issues [130]. Instead of formulating the clustering problem as searching through a space for multiple clusters, UNC adopted a density-based fitness function that would reach a maximum at every good cluster center. As a result, the search space is substantially reduced: if there are c clusters and S is the search space for UNC, then the previous formulation searching for all c cluster centers would have a search space of S^c.

To identify dense areas of a feature space as clusters, UNC adopts the following density-based fitness function, assuming that c_i is the location of a hypothetical cluster center, and the data set X has n features/dimensions, with N data points:

$$f_i = \frac{\sum_{j=1}^{N} w_{ij}}{\delta_i^2}, \quad (7)$$

where $w_{ij} = exp(-\frac{d_{ij}^2}{2\delta_i^2})$, and $d_{ij}^2 = ||x_j - c_i||^2$ which measures the Euclidean distance of the data point x_j (where $j = 1, \ldots, N$) to the cluster center c_i. The value of f_i will be high for points falling within the boundary of a cluster, and low for points falling outside of the cluster. δ_i^2 is a measure of dispersion for the i-th cluster. This parameter is crucial for determining the cluster boundaries. UNC measures the *goodness-of-fit* of a model to just a part of the data. Essentially this constructs a fitness landscape with multiple peaks with each at a cluster center location. Both Euclidean and Mahalanobis distance measures were used in order to more accurately estimate the different shapes and the sizes of the niches. Deterministic crowding was then used in conjunction with a restricted mating scheme, to allow no assumption of niche radii, or whether all peaks are equally distant. Through experimentation on both synthetic data and a data set of a real-world image segmentation problem, UNC was shown to be less prone than non-niching techniques to premature convergence, noise, and initialization. Several recent works adopting a similar objective function based on the compactness of points around a cluster center include [132] where a dynamic niching GA was used, and [133] where LIPS [34] was adopted. A distinct advantage of these methods is that they do not require any prior knowledge of the number of clusters, and can still perform reasonably well on several synthetic and real data sets.

Niching can be used to enhance clustering and feature selection simultaneously. It has been observed in [134] that clustering is highly multi-modal, and a direct application of a standard EA tends to result in getting stuck in local optima. Furthermore, clustering on all features is not a good strategy since not all features are relevant. It was shown in [134] that niching could help preserve population diversity allowing the EA to explore many optimal solutions in parallel, and as a result, help to prevent the algorithm from getting stuck in local optima. A unified criterion was designed as an objective function to simultaneously optimize the clustering centers and feature subset selection. In this case, a replacement group was adopted to encourage mating among similar solutions with the same number of clusters, and competition among dissimilar solutions with different numbers of clusters. The similarity measure is based on the Euclidean distance between a pair of solutions in the phenotypic space. This niching memetic algorithm, NMA_CFS [134], shows clear advantages over other methods that do not use niching or simultaneous optimization of clustering and feature subset selection.

E. Feature selection

Feature selection plays an important role in pattern recognition. Generally speaking, the aim of *feature selection* is to choose features that allow us to discriminate patterns belonging to different classes. The feature selection problem can be defined as follows [135]: given an initial set F with n features, search for a subset $S \in F$ with k features that maximize the mutual information $I(C, S)$ between the class label C and the subset S of selected features.

Feature selection algorithms are generally classified into two categories, *wrapper* and *filter* methods. The wrapper method makes use of a learning classifier's performance to evaluate the suitability of the feature subset, whereas the filter method treats the selection of feature subsets as a pre-processing step, independent from the learning classifier.

Traditional feature selection algorithms are mostly incremental methods where features are selected one at a time, according to criteria based on a single feature. This is limiting, since in many real-world problems, several features acting simultaneously may be relevant (i.e., *epistasis*), though an individual feature may not. However, selection of subsets of features can be done more efficiently by EAs. Since an optimal subset might not be unique, there is merit to obtain all such optimal subsets before making a final choice. We can consider different optimal subsets of features as different optima on a multi-modal fitness landscape, which can be searched using a niching method. For example, a subset of the selected features can be represented using a binary string where the i-th bit being 1 indicates that the i-th feature is included in the subset, whereas 0 indicates the feature is excluded. To evaluate the goodness of the subset, the binary string is fed into a learning classifier (e.g., neural network). The fitness function takes into account the classifier accuracy term and the penalty for selecting a large number of features [136].

A genetic algorithm (GA) guided normalized Mutual Information Feature Selection (GAMIFS) algorithm was developed in [137], which is a hybrid of the filter and wrapper feature selection methods employing the GA and a neural network classifier. The GA incorporated *deterministic crowding* into its procedure to encourage searching for multiple optimal feature subsets. In this niching scenario, a tournament selection is run between the offspring and its nearest parent with respect to Hamming distance. The winner is carried over to the next iteration. A mutation operator is used to allow adding a relevant feature or eliminating an irrelevant or redundant feature from the individuals in the GA population. A mutated individual survives only if it has a better fitness than that of the original individual. The niching-based GAMIFS is able to find individual relevant features as well as groups of relevant features. On 4 data sets with up to 60 features, GAMIFS outperformed those incremental search methods.

The effect of employing niching methods for solving feature selection problems was also investigated in [138]. This study combined the standard wrapper method with various niching methods such as DFS [27] and $r3pso$ [32] and their variants. These feature selection wrapper variants were evaluated on 12 UCI data sets[3], and the results were compared with the single-optimum seeking GA and memetic algorithm, showing that the niching variants outperformed the standard GA and memetic algorithm in finding multiple accurate feature subsets.

[3]http://archive.ics.uci.edu/ml/datasets.html

F. Machine learning

Machine learning (ML) plays an increasingly important role in data analytics these days because ML can make predictions by learning from data. Many real-world problems are often too large and complex to solve by a single machine learning model. An effective approach may be to employ an ensemble of learning models, each specializing in solving a subtask of a much larger problem. Meta-heuristic algorithms can be used to evolve a population of ML models, e.g., an ensemble of neural networks [139] or a set of knowledge rules [140]. In such a setting, niching methods are useful schemes for maintaining a diverse population of learning models. The ensemble approach is shown [141] not only performing much better, but also more robust and generalizing better than those employing a single ML model.

1) Evolving neural network ensembles: Several early works by Yao and Liu [139], [142], [143] showed that a population contains more information than a single individual in it. Such information can be useful for improving the generalization ability of a learning model. Furthermore, speciation (or niching) can be introduced to the population to evolve a diverse but accurate set of specialist modules which can be then combined to perform learning tasks. In [144], Liu *et al.*, proposed Evolutionary Ensembles with Negative Correlation Learning (EENCL) to automatically determine the number of individual neural networks in an ensemble. Fitness sharing was adopted to promote diversity in the ensemble. If one training example is learned correctly by n individual neural networks, then each of these n neural networks receives a fitness value $1/n$, and the remaining neural networks in the ensemble receive zero fitness. This procedure is repeated for all examples in the training set. The final fitness of an individual is determined by summing up its fitness values over all training examples. The idea is to encourage niche formation by degrading the original fitness of an individual neural network according to the presence of other similar neural networks. The output of the ensemble is normally determined by the majority of the neural networks. This series of works on neural network ensembles is nicely summarized in [141].

A PSO-based niching method (i.e., NichePSO [40]) was used to evolve neural network ensembles (NNE) [145], more specifically for training a group of neural networks for solving a set of multivariate classification and regression tasks from the UCI data sets. A typical goal for using an NNE, is to evolve different neural networks to specialize in solving complementary parts of a task, making niching methods a good fit for this purpose. In the proposed NichePSO for NNE (NPSOE), each sub-swarm is used to optimize the connection weights of each constituent neural network. During the training, the training data are fed into each neural network as the input layer, and the output is compared with those produced as the validation data is passed as the inputs. The difference (or error) is used by NichePSO as the fitness value for a particle in a sub-swarm. The weights for each neural network are set according to the best particle in the sub-swarm. In short, this research shows that NichePSO evolving an NNE, i.e., NPSOE, is capable of exploiting the multivariate nature of the multivariate classification and regression tasks. On the majority of the tested UCI classification and regression data sets, NPSOE was shown to produce much lower classification and prediction errors than those by using the typical back-propagation trained NNE.

2) Learning multiple rules from data: Data mining employs many of the same techniques developed in ML. In data mining, meta-heuristics can be used to extract knowledge such as rules and use these rules to solve classification problems [140], [146]. There are usually two different methods, the *Michigan* approach where each individual encodes a single rule, and the *Pittsburgh* approach where each individual represents multiple rules, i.e., a rule set. Since it is often difficult to capture the knowledge of a data set by a single rule, multiple rules are often required. Directly evolving multiple rule sets from scratch using the Pittsburgh approach is too challenging as the search space is vast. However, for the Michigan approach, niching methods can be used to evolve multiple different good individuals that are required to produce a rule set.

An idea of *token competition* was proposed in [140] to promote diversity in the rule population. Each record in the training set is regarded as a resource (so-called token). If an individual (or rule) is matched with a record, then this individual can seize the token. The order of receiving tokens is determined by the fitness values of the rules. A rule with a high score (original fitness) means it can cover more records, and at the same time, the other rules attempting to cover the same rules (or niche) will have their fitness decreased since they cannot compete with the stronger rule. An individual's fitness is modified according to the following:

$$f_{modified} = f_{original} * count/ideal, \tag{8}$$

where $f_{original}$ is obtained from the objective function evaluation, $count$ is the number of tokens actually seized by the rule, and $ideal$ is the total number of possible tokens that can be seized by the rule. Token competition favours stronger rules that cover more records, and weakens other rules that cannot compete with them in the same niche areas.

Unlike classic niching methods such as fitness sharing or crowding, token competition does not use a distance measure directly on the evolved rules, since it can be difficult to determine how similar two rules are (e.g., produced by using genetic programming). Token competition regards two individuals as similar if they cover similar sets of records.

Building upon the above token competition idea, a coevolution-based classification method was proposed in [147] to coevolve individual rules and rule sets concurrently in separate coevolving populations in order to further confine the search space and produce quality rule sets efficiently.

3) Game playing: In [148] an evolutionary system was proposed to automatically create a collection of specialist strategies for a game playing system, relieving humans from having to decide how to specialize. In the real-time Neuroevolution of Augmented Topologies (rtNEAT) embedded in the NERO (Neuroevolving Robotic Operative) game [149], speciation played a critical role in protecting *topological innovation* by only allowing individual topology solutions (i.e.,

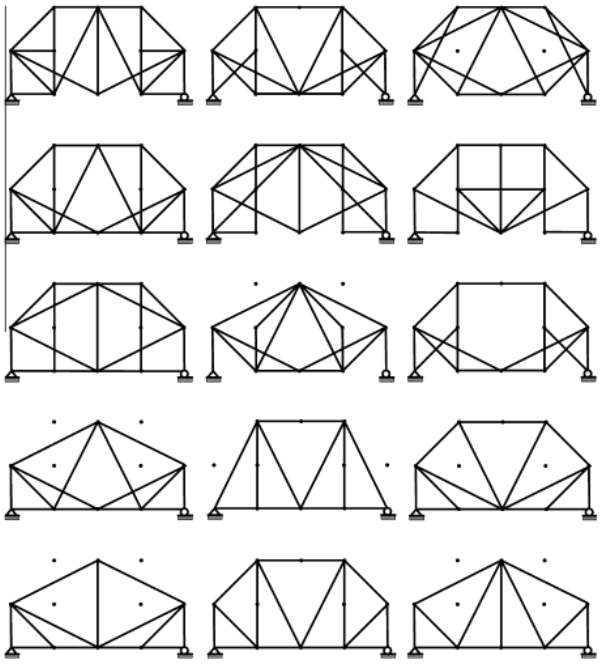

Fig. 6. Multiple optimal truss topologies obtained by using niching methods [150]. Permission for using the figure has been obtained from the first author.

neural network structures) to compete within their own niches instead of with the population at large.

IX. Examples of Real-world Applications

In this section, we provide a collection of various real-world MMO problems where niching methods have been successfully applied to (it is by no means a complete list). Each example will be briefly described, focusing more on the problem characteristics and the key motivation of adopting niching methods in its specific context. Readers can follow the references for further information:

Truss-structure optimization: Optimization of truss-structures has been a well-researched area in engineering. It usually involves finding solutions with optimal cross-sectional size, topology, and configuration so that an overall minimum weight can be achieved. Given a set of supports, concentrated loads, and node points, the optimization method needs to determine the optimal connectivity between elements, member sizing, and node positions that will lead to a least weight structure, satisfying all design constraints. An early work by Deb and Gulati [151] showed that there existed multiple different topologies with almost equal overall weight. More specifically, this truss-structure optimization problem can be considered *multi-modal* since it has a large number of possibly different topological solutions. Luh and Lin [150] adopted a two-staged approach: first topology optimization from a given ground structure, and then size and shape optimization employing the identified topology from the first stage. For the topology optimization phase, they demonstrated that a binary PSO with fitness sharing can be used to find multiple equally good truss-structure solutions, as shown in Fig. 6.

Metabolic network modelling: A metabolic network is modelled using Generalized Mass Action Kinetics formulation (GMAKr) [85]. In this case, using a classic global optimization algorithm can only find a single solution that is not biologically plausible. Kronfeld *et. al* [85] showed that this problem is highly multi-modal, and there exists a large number of high-quality solutions which can be used together for sensitivity analysis. More specifically, some parameters are more sensitive than others. This allows identification of parameter values that are less sensitive as well as producing high-quality solutions.

Drug molecule design: This problem was formulated as a multi-objective constrained optimization problem and optimized by NSGA-II [152]. A major issue here is that approximation models must be used since it is difficult to obtain accurate measures on objectives and constraints. Physical experiments are necessary, and as a result, it is desirable that after the optimization, a set of diverse solutions, i.e., molecules, can be obtained. An expert can then decide which one of the solutions should be selected. With the assistance of a niching based NSGA-II method, it is possible to substantially enhance the diversity of solutions found in the design space.

Femtosecond laser pulse shaping problem: A CMA-ES based niching method was employed to solve this Femtosecond Laser Pulse Shaping problem in the field of Quantum Control [153]. A distance metric was appropriately defined between two feasible solutions, in order to discover multiple unique pulse profiles of high quality. In this case, different niches represent the same conceptual designs. The CMA-ES based niching method achieved better alignment results than the standard evolution-strategy method.

Job shop scheduling problem (JSSP): This is a classic optimization problem studied extensively in literature. Perez *et al.* [154], [155] represents one of the very few studies on JSSP with a focus on identifying multiple solutions. JSSP are typically multi-modal, presenting an ideal case for applying niching methods. Their studies suggest that not only do niching methods help to locate multiple good solutions, but also to preserve the diversity more effectively than employing a standard single-optimum seeking genetic algorithm.

Resource constrained multi-project scheduling problems (RCMPSP): In this problem, multiple projects must be carried out and completed using a common pool of scarce resources. The difficulty is that one has to prioritize each project's tasks to optimize an objective function without violating both intra-project precedence constraints and inter-project resource constraints. A decision maker can benefit from choosing between different good scheduling solutions, instead of being limited to only one. In addition, it is also much faster than rescheduling. The *deterministic crowding* and *clearing* methods were adopted in [156] to find multiple optimal scheduling solutions for this problem. A library called *RCMPSPlib* was created by the authors to report the benchmarking instances and the multiple optima that have been found [4].

Automatic point determination: The problem of automatic determination of point correspondence between two images can be formulated as an MMO problem. A niching GA was used to determine the automatic point correspondence between

[4] http://www.eii.uva.es/elena/RCMPSPLIB.htm

two images [157]. The niching GA method was able to discover optimal solutions that are measured by the similarity between patches of two images.

Seismological inverse problem: A niching GA was applied to an inversion problem of teleseismic body waves for the source parameters of an earthquake [158]. Here a distance metric for waveform inversion was adopted for measuring the similarity between solutions. The niching GA was shown to be more efficient than a grid search in detecting several global and local optima over a range of scales, representing the fault and auxiliary planes.

Monte Carlo nonlinear filtering: Niching methods was used to improve a Monte Carlo filtering algorithm when the posterior distributions of problems are multi-modal [159]. The standard Monte Carlo filters often suffer from the issue of diversity loss due to the random nature of re-sampling. Niching in Monte Carlo filtering helps to combat *genetic drift*. In this case, the Kolmogorov-Smirnov metric was used as a measure of the distance between two probability distributions [159].

Image segmentation: A Dynamic Niching Genetic Clustering algorithm (DNGA) was developed for image segmentation without knowing the number of clusters [160]. The similarity of each data point to all other points is defined by a similarity function based on the density shape of the data points in the vicinity of the chosen point. The DNGA was shown to be insensitive to a range of niche radius values.

Clustering: A clustering-based niching EA was developed in order to reconstruct gene regulatory networks from data [161]. The niching method was used to maintain a better solution diversity and to ultimately identify multiple alternative networks. The decision maker can then make the final choice based on further design considerations. In [162], the restricted mating scheme (a classic niching method) was incorporated into a memetic algorithm to carry out the task of web document clustering. A clustering algorithm based on dynamic niching with niche migration was shown in [132] to perform well for the task of *remote sensing image clustering*. In spatial data clustering [163], it was shown that the GA-based spatial analysis technique could benefit from employing *fitness sharing* to mitigate the effect of genetic drift, and as a result, promote population diversity and encourage multiple optimal solutions to be located in a single run.

Real-time tracking of body motion: A Niching Swarm Filtering (NSF) algorithm was developed to address the problem of real-time tracking of unconstrained full-body motion [164]. In this case, multiple significant global and local solutions of the configuration distribution are found.

Competitive facilities location and design: In this facility location problem, typically multiple global solutions need to be obtained. A niching method named the Universal Evolutionary Global Optimizer (UEGO) was shown to significantly outperform simulated annealing and multi-start methods [165].

Solving systems of equations: One of the first niching PSO algorithms were developed to solve systems of linear equations [166]. Niching algorithms are suitable to solve systems of equations due to systems of equations having multiple solutions. Recently, it was shown in [167] that systems of nonlinear equations can also be solved using niching techniques.

Protein structure prediction: A protein structure prediction problem on the 3D Hydrophobic-Polar (HP) lattice model was formulated as an MMO problem in [168], and it was shown that even applying a simple niching method outperformed the state-of-the-art approaches.

Induction motor design for electric vehicle: For design optimization of induction motors (shape or structure), there is a need to identify multiple optimal profiles. A niching method with restricted tournament selection was used for this task [169]. It is interesting to note the discussion in [169] on the difficulty in formulating the problem as a multi-objective optimization problem, e.g., it is difficult to directly apply the geometrical constraints and manufacturing considerations, such as stator coil winding. Furthermore, the transient temperature rise of the stator coil could not be calculated during the optimization. A better alternative is to carry out some *post-processing*, i.e., using other criteria and the designer's experience to select the best solution from a list of optimal solutions produced by a niching method.

Electromagnetic design: In [170], an electromagnetic device design problem was reformulated into an MMO problem by deliberately not specifying a normative value for the *magnetic flux density* attribute. Several niching methods were used, locating typically 14 to 20 solutions, whereas a simple GA just found one.

Other examples include a niching method for detecting multiple nearly-optimal solutions for space mission design problems [171] and a niching PSO method for identification of static equilibria via potential energy optimization [172]. Though many more examples can be found in literature, we hope the above list suffices to demonstrate a common pattern, i.e., the importance and usefulness of niching methods going beyond the boundaries of many application areas.

X. Discussion and open questions

From the previous sections, we can see that among many real-world niching applications it is important to adopt a domain-specific tailor-made distance (or similarity) measure, as also noted in [23]. Another observation is that up to now many optimization problems have been treated as single-solution seeking problems, but actually can be reconsidered as multi-solution seeking problems, e.g., data mining problems approached by the token competition method [140]. Finding multiple solutions using a niching method helps to reveal some global properties of the problem under study, which is information a user would not normally get by observing just a single obtained solution in isolation. A decision maker can compare and study these alternative solutions before making a final choice, depending on the circumstance. Furthermore, sometimes a multi-modal formulation may be easier hence more viable than others, e.g., difficulties associated with a multi-objective formulation [169]. In such a case, niching methods can be used to first produce a set of alternative solutions, which can be subsequently *post-processed* according to criteria external to the formulation.

A common feature that can be observed among these PSO, DE, and ES niching algorithms is the need to identify *nearest*

neighbourhood best points with respect to a current point in the population, e.g., FER-PSO [33], LIPS [34], DE/nrand/1 [52], dADE/nrand/1 [53], LoINDE [56], the DE using a proximity mutation operator [50], the Local selection-based DE [49], and *NEA2* (Niching with CMA-ES via NBC) [95]. These neighbourhood best points can be subsequently used to attract individuals in their respective neighbourhoods, in order to achieve the niching effect. It is obvious that only population-based meta-heuristic algorithms can exploit this property.

Below we provide a list of open research questions on niching methods, which we think are important to address in future research:

- We need to rethink real-world problems with a view to seeking multiple solutions. Clearly, this is an important perspective that tends to be missed by many. For example, we can formulate a typical clustering problem as an MMO problem.
- Ensemble-based learning models have been shown to be promising for non-stationary environments [173]. How can we apply niching methods to maintain a good diversity of learners in an ensemble? A broader question is how to apply niching to uncertain environments.
- There is a general lack of theoretical understanding of the distributed convergence behaviour among different niching techniques, as also remarked in [17]. Such theoretical studies can provide guidance for designing effective niching techniques applicable to a wide range of problem domains.
- Many new niching methods have been introduced and revamped under different search paradigms. A common problem is that these methods introduce additional parameters for tuning and they may be problem dependent. How do we adapt and even remove these undesirable parameters without sacrificing performance?
- How do we measure the performance of niching methods in real-world settings, where the locations and number of optima are usually unknown? The existing performance metrics are clearly limited, as they make several assumptions [101]. In addition, when there exist too many solutions, it is perhaps unnecessary to find all of them (see arguments made against too many choices in [174]). Instead, it may be sufficient to attain a subset of solutions with some desirable coverage and spread. This is in some way, similar to the distribution requirement of a solution set obtained by an EMO algorithm.
- There are not enough studies on high-dimensional MMO problems, as shown in [110]. Though evidence exists that shows niching is helpful for low-dimensional problems, it is unclear how much benefit we can derive from doing niching in high-dimensional cases.
- Many real-world problems are highly constrained and of combinatorial nature. However, there lacks a systematic study on how existing niching methods, largely designed for unconstrained optimization, should cope with constraints. Deb and Saha provided a multi-modal constrained test function suite in [94], [110], which may help spur more research in this direction.
- Although there exist many studies hybridizing EA with local search [67] for locating a single optimum, it seems still rare to see niching methods hybridized with local search [71]. It can be envisaged that niching combined with local search has the potential to further enhance convergence onto multiple optima.

Many interesting and challenging research ideas have also been raised and discussed in [175], [176], which is the first book published on the topic of niching methods.

XI. Concluding Remarks

Niching methods are powerful search methods that can produce multiple good solutions for a decision maker to choose from. In this paper, we have revisited classic niching methods in EAs and reviewed recent developments of niching methods derived from other meta-heuristics. We have shown through many real-world application examples that seeking multiple good solutions is a common task across multiple disciplinary areas, and niching methods can play an important role in achieving this task. These examples of niching applications present a more holistic picture of the impact by niching methods, and hopefully this will provide a great impetus for an even more wide-spread use of niching methods. We have identified several open research questions. We hope these questions will help to rejuvenate new interest and research effort in this classic but important topic in the years to come.

Population-based optimization methods, such as evolutionary methods and other meta-heuristics methods, are attractive due to their ability to store and process multiple and diverse solutions from the search space. Maintaining diversity of a population may not be automatic in all problems and therefore the role of a niching operator becomes evident and invaluable in a population-based optimization method. However, a niching operator needs an appropriate space (genotypic or phenotypic or both) and a distance metric to be effective. Although most niching methods use at least one user-defined threshold parameter to compare the distance metric with, recent efforts have been focused on a parameter-less approach. This paper has presented many different existing niching methodologies that exploited (and reasonably so) the population approach of evolutionary and other meta-heuristics methods.

Acknowledgment

The authors would like to thank anonymous reviewers for their constructive comments which have greatly improved the quality of this paper.

References

[1] A. Ward, J. K. Liker, J. J. Cristiano, and D. K. Sobek, "The second toyota paradox: How delaying decisions can make better cars faster," *Sloan management review*, vol. 36, no. 3, p. 43, 1995.

[2] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[3] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proc. of the Second International Conference on Genetic Algorithms*, J. Grefenstette, Ed., 1987, pp. 41–49.

[4] K. A. De Jong, "An analysis of the behavior of a class of genetic adaptive systems." Ph.D. dissertation, University of Michigan, 1975.

[5] S. W. Mahfoud, "Crowding and preselection revisited," in *Parallel problem solving from nature 2*, R. Männer and B. Manderick, Eds. Amsterdam: North-Holland, 1992, pp. 27–36. [Online]. Available: citeseer.ist.psu.edu/mahfoud92crowding.html

[6] D. Beasley, D. R. Bull, and R. R. Martin, "A sequential niche technique for multimodal function optimization," *Evolutionary Computation*, vol. 1, no. 2, pp. 101–125, 1993. [Online]. Available: citeseer.ist.psu.edu/beasley93sequential.html

[7] G. R. Harik, "Finding multimodal solutions using restricted tournament selection," in *Proc. of the Sixth International Conference on Genetic Algorithms*, L. Eshelman, Ed. San Francisco, CA: Morgan Kaufmann, 1995, pp. 24–31. [Online]. Available: citeseer.ist.psu.edu/harik95finding.html

[8] M. Bessaou, A. Pétrowski, and P. Siarry, *Island Model Cooperating with Speciation for Multimodal Optimization*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 437–446.

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

[10] K. E. Parsopoulos, V. P. Plagianakos, G. D. Magoulas, and M. N. Vrahatis, "Objective function "stretching" to alleviate convergence to local minima," *Nonlinear Analysis*, vol. 47, no. 5, pp. 3419–3424, 2001.

[11] K. E. Parsopoulos and M. N. Vrahatis, "On the computation of all global minimizers through particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 211–224, June 2004.

[12] A. Pétrowski, "A clearing procedure as a niching method for genetic algorithms," in *Proc. of the 3rd IEEE International Conference on Evolutionary Computation*, 1996, pp. 798–803.

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

[14] A. P. Engelbrecht, *Computational Intelligence: An Introduction*. New York, NY, USA: Halsted Press, 2002.

[15] B. Sareni and L. Krahenbuhl, "Fitness sharing and niching methods revisited," *IEEE Transactions on Evolutionary Computation*, vol. 2, no. 3, pp. 97–106, Sep. 1998.

[16] G. Singh and K. Deb, "Comparisons of multi-modal optimization algorithms based on evolutionary algorithms," in *Proc. of the Genetic and Evolutionary Computation Conference 2006 (GECCO'06)*, Washington, USA, 2006, pp. 1305–1312.

[17] S. Das, S. Maity, B.-Y. Qu, and P. N. Suganthan, "Real-parameter evolutionary multimodal optimization - a survey of the state-of-the-art," *Swarm and Evolutionary Computation*, vol. 1, pp. 71–88, June 2011.

[18] O. M. Shir, "Niching in evolutionary algorithms," *Handbook of Natural Computing: Theory, Experiments, and Applications*, pp. 1035–1069, 2012.

[19] J. Horn, "The nature of niching: Genetic algorithms and the evolution of optimal, cooperative populations," Tech. Rep. UIUCDCS-R-97-2000, 1997. [Online]. Available: citeseer.ist.psu.edu/horn97nature.html

[20] J. E. Fieldsend, "Running up those hills: Multi-modal search with the niching migratory multi-swarm optimiser," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*, July 2014, pp. 2593–2600.

[21] X. Li, A. Engelbrecht, and M. G. Epitropakis, "Benchmark functions for cec'2013 special session and competition on niching methods for multimodal function optimization," Technical Report, Evolutionary Computation and Machine Learning Group, RMIT University, 2013.

[22] S. W. Mahfoud, "Niching methods for genetic algorithms," Ph.D. dissertation, Urbana, IL, USA, 1995. [Online]. Available: citeseer.ist.psu.edu/mahfoud95niching.html

[23] O. M. Shir, "Niching in derandomized evolution strategies and its applications in quantum control, phd thesis," Ph.D. dissertation, Natural Computing Group, LIACS, Faculty of Science, Leiden University, 2008.

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

[25] D. E. Goldberg, K. Deb, and J. Horn, "Massive multimodality, deception, and genetic algorithms," in *PPSN 2*, R. Männer and B. Manderick, Eds. Amsterdam: Elsevier Science Publishers, B. V., 1992. [Online]. Available: citeseer.ist.psu.edu/goldberg92massive.html

[26] P. J. Darwen and X. Yao, "A Dilemma for Fitness Sharing with a Scaling Function," in *Proceedings of the Second IEEE International Conference on Evolutionary Computation*. Piscataway, New Jersey: IEEE Press, 1995. [Online]. Available: citeseer.ist.psu.edu/darwen95dilemma.html

[27] A. Della Cioppa, C. De Stefano, and A. Marcelli, "Where are the niches? dynamic fitness sharing," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 4, pp. 453–465, Aug 2007.

[28] B. L. Miller and M. J. Shaw, "Genetic algorithms with dynamic niche sharing for multimodal function optimization," in *Proceedings of the 1996 IEEE International Conference onEvolutionary Computation*, May 1996, pp. 786–791.

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

[30] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*. Morgan Kaufmann, 2001.

[31] A. P. Engelbrecht, B. S. Masiye, and G. Pampard, "Niching ability of basic particle swarm optimization algorithms," in *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005.*, Jun. 2005, pp. 397–400.

[32] X. Li, "Niching without niching parameters: Particle swarm optimization using a ring topology," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 150–169, February 2010.

[33] ——, "Multimodal function optimization based on fitness-euclidean distance ratio," in *Proc. of Genetic and Evolutionary Computation Conference 2007*, D. Thierens, Ed., 2007, pp. 78–85.

[34] B. Y. Qu, P. N. Suganthan, and S. Das, "A distance-based locally informed particle swarm model for multimodal optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 3, pp. 387–402, June 2013.

[35] J. K. R. Mendes and J. Neves, "The fully informed particle swarm: simpler, maybe better," *IEEE Trans. Evol. Comput.*, vol. 8, pp. 204–210, Jun. 2004.

[36] K. E. Parsopoulos and M. N. Vrahatis, "Modification of the particle swarm optimizer for locating all the global minima," in *Artificial Neural Networks and Genetic Algorithms*, V. Kurkova, N. Steele, R. Neruda, and M. Karny, Eds. Springer, 2001, pp. 324–327.

[37] A. P. Engelbrecht, "Finding multiple solutions to unconstrained optimization problems using particle swarm optimization," in *Proceedings of the International Conference on Mathematical and Computational Models*, 2009.

[38] X. Li, "Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization," in *Proc. of Genetic and Evolutionary Computation Conference 2004(LNCS 3102)*, K. Deb, Ed., 2004, pp. 105–116.

[39] D. Parrott and X. Li, "Locating and tracking multiple dynamic optima by a particle swarm model using speciation," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 4, pp. 440–458, August 2006.

[40] R. Brits, A. P. Engelbrecht, and F. van den Bergh, "A niching particle swarm optimizer," in *Proc. of the 4th Asia-Pacific Conference on Simulated Evolution and Learning 2002(SEAL 2002)*, 2002, pp. 692–696.

[41] ——, "Solving systems of unconstrained equations using particle swarm optimizers," *Proc. of the IEEE Conference on Systems, Man, Cybernetics*, pp. 102–107, October 2002.

[42] T. Blackwell and J. Branke, "Multiswarms, exclusion, and anti-convergence in dynamic environments," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 4, pp. 459–472, 2006.

[43] S. Bird and X. Li, "Adaptively choosing niching parameters in a PSO," in *Genetic and Evolutionary Computation Conference, GECCO 2006, Proceedings, Seattle, Washington, USA, July 8-12, 2006*, M. Cattolico, Ed. ACM, 2006, pp. 3–10. [Online]. Available: http://doi.acm.org/10.1145/1143997.1143999

[44] I. L. Schoeman and A. P. Engelbrecht, "Using vector operations to identify niches for particle swarm optimization," in *Proc. of the 2004 IEEE Conference on Cybernetics and Intelligent Systems*, Singapore, 2004, pp. 361–366.

[45] J. Barrera and C. A. C. Coello, "A review of particle swarm optimization methods used for multimodal optimization," in *Innovations in Swarm Intelligence*, ser. Studies in Computational Intelligence, C. Lim, L. Jain, and S. Dehuri, Eds. Springer Berlin Heidelberg, 2009, vol. 248, pp. 9–37.

[46] X. Li, "Developing niching algorithms in particle swarm optimization," in *Handbook of Swarm Intelligence*, ser. Adaptation, Learning, and Optimization, B. Panigrahi, Y. Shi, and M.-H. Lim, Eds. Springer Berlin Heidelberg, 2011, vol. 8, pp. 67–88.

[47] K. Price, "An introduction to differential evolution," *New Ideas in Optimization*, pp. 79–108, 1999.

[48] J. Rönkkönen, *Continuous Multimodal Global Optimization with Differential Evolution Based Methods*. Acta Universitatis Lappeenrantaensis 363, 2009.

[49] J. Rönkkönen and J. Lampinen, "On determining multiple global optima by differential evolution," in *Evolutionary and Deterministic Methods for Design, Optimization and Control*, ser. Proceedings of Eurogen 2007, 2007, pp. 146–151.

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

[51] M. G. Epitropakis, V. P. Plagianakos, and M. N. Vrahatis, "Multimodal optimization using niching differential evolution with index-based neighborhoods," in *Proceedings of 2012 IEEE Congress on Evolutionary Computation (CEC'12)*, June 2012, pp. 1–8.

[52] ——, "Finding multiple global optima exploiting differential evolution's niching capability," in *Differential Evolution (SDE), 2011 IEEE Symposium on*, April 2011, pp. 1–8.

[53] M. G. Epitropakis, X. Li, and E. K. Burke, "A dynamic archive niching differential evolution algorithm for multimodal optimization," in *Evolutionary Computation (CEC), 2013 IEEE Congress on*, June 2013, pp. 79–86.

[54] Z. Zhai and X. Li, "A dynamic archive based niching particle swarm optimizer using a small population size," in *Proceedings of the Australian Computer Science Conference (ACSC 2011)*, Jan 2011, pp. 1–7.

[55] B. Y. Qu, P. N. Suganthan, and J. J. Liang, "Differential evolution with neighborhood mutation for multimodal optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 5, pp. 601–614, Oct 2012.

[56] S. Biswas, S. Kundu, and S. Das, "Inducing niching behavior in differential evolution through local information sharing," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 2, pp. 246–263, April 2015.

[57] Y. Zhang, Y. j. Gong, H. Zhang, T. L. Gu, and J. Zhang, "Towards Fast Niching Evolutionary Algorithms: A Locality Sensitive Hashing-Based Approach," *IEEE Transactions on Evolutionary Computation*, vol. PP, no. 99, pp. 1–1, 2016.

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

[59] S. Hui and P. N. Suganthan, "Ensemble and arithmetic recombination-based speciation differential evolution for multimodal optimization," *IEEE Transactions on Cybernetics*, vol. 46, no. 1, pp. 64–74, Jan 2016.

[60] S. Forrest, B. Javornik, R. Smith, and A. Perelson, "Using genetic algorithms to explore pattern recognition in the immune system," *Evolutionary Computation*, vol. 1, no. 3, pp. 191–211, Sept 1993.

[61] L. N. de Castro and J. Timmis, "An artificial immune network for multimodal function optimization," in *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, vol. 1, May 2002, pp. 699–704.

[62] Q. Yang, W. N. Chen, Z. Yu, T. Gu, Y. Li, H. Zhang, and J. Zhang, "Adaptive Multimodal Continuous Ant Colony Optimization," *IEEE Transactions on Evolutionary Computation*, vol. PP, no. 99, pp. 1–1, 2016.

[63] M. Guntsch and M. Middendorf, *Applying Population Based ACO to Dynamic Optimization Problems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002, pp. 111–122.

[64] D. Angus, *Niching for Ant Colony Optimisation*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 165–188.

[65] M. Z. Ali and N. H. Awad, "A novel class of niche hybrid cultural algorithms for continuous engineering optimization," *Information Sciences*, vol. 267, pp. 158–190, 2014.

[66] Q. Yang, W. N. Chen, Y. Li, C. L. P. Chen, X. M. Xu, and J. Zhang, "Multimodal estimation of distribution algorithms," *IEEE Transactions on Cybernetics*, vol. PP, Available Online, no. 99, pp. 1–15, 2016.

[67] R. Chelouah and P. Siarry, "Genetic and neldermead algorithms hybridized for a more accurate global optimization of continuous multiminima functions," *European Journal of Operational Research*, vol. 148, no. 2, pp. 335–348, 2003, sport and Computers.

[68] X. Chen, Y. S. Ong, M. H. Lim, and K. C. Tan, "A multi-facet survey on memetic computation," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 5, pp. 591–607, Oct 2011.

[69] S. Bird and X. Li, *Computational Intelligence in Expensive Optimization Problems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, ch. Improving Local Convergence in Particle Swarms by Fitness Approximation Using Regression, pp. 265–293.

[70] L. Wei and M. Zhao, "A niche hybrid genetic algorithm for global optimization of continuous multimodal functions," *Applied Mathematics and Computation*, vol. 160, no. 3, pp. 649–661, 2005.

[71] J. X. Peng, S. Thompson, and K. Li, "A gradient-guided niching method in genetic algorithm for solving continuous optimisation problems," in *Intelligent Control and Automation, 2002. Proceedings of the 4th World Congress on*, vol. 4, 2002, pp. 3333–3338 vol.4.

[72] S. Ono, Y. Hirotani, and S. Nakayama, "Multiple solution search based on hybridization of real-coded evolutionary algorithm and quasi-newton method," in *2007 IEEE Congress on Evolutionary Computation*, Sept 2007, pp. 1133–1140.

[73] M. Jelasity and J. Dombi, "GAS, a concept on modeling species in genetic algorithms," *Artificial Intelligence*, vol. 99, no. 1, pp. 1–19, 1998. [Online]. Available: citeseer.ist.psu.edu/jelasity98gas.html

[74] G. Dick, "Automatic identification of the niche radius using spatially-structured clearing methods," in *IEEE Congress on Evolutionary Computation*, July 2010, pp. 1–8.

[75] S. Tsutsui, J. Suzuri, and A. Ghosh, "Forking gas: Gas with search space division schemes," *Evolutionary Computation*, vol. 5, no. 1, pp. 61–80, 1997.

[76] J. Gan and K. Warwick, "Dyanmic niche clustering: a fuzzy variable radius niching technique for multimodal optimisation in gas," in *Proc. of the 2001 Congress on Evolutionary Computation*. IEEE Press, 2001, pp. 215–222.

[77] O. M. Shir and T. Bäck, "Niche radius adaptation in the cms-es niching algorithm," in *Parallel Problem Solving from Nature - PPSN IX, 9th International Conference (LNCS 4193)*. Reykjavik, Iceland: Springer, 2006, pp. 142–151.

[78] W. M. Spear, "Simple subpopulation schemes," in *Proc. of 3rd Annual Conf. on Evolutionary Programming*. World Scientific, 1994, pp. 296–307.

[79] P. J. Darwen and X. Yao, "Every niching method has its niche: Fitness sharing and implicit sharing compared," in *Proceedings of the 4th International Conference on Parallel Problem Solving from Nature*, ser. PPSN IV. London, UK, UK: Springer-Verlag, 1996, pp. 398–407.

[80] C. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu, "Multimodal optimization by means of a topological species conservation algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 6, pp. 842–864, Dec 2010.

[81] L. Li and K. Tang, "History-based topological speciation for multimodal optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 1, pp. 136–150, Feb 2015.

[82] R. Brits, A. P. Engelbrecht, and F. van den Bergh, "Scalability of niche pso," in *Swarm Intelligence Symposium, 2003. SIS '03. Proceedings of the 2003 IEEE*, April 2003, pp. 228–234.

[83] ——, "Locating multiple optima using particle swarm optimization," *Applied Mathematics and Computation*, vol. 189, pp. 1859–1883, 2007.

[84] M. Kronfeld and A. Zell, "Towards scalability in niching methods," in *Proceedings of the 2010 IEEE Congress on Evolutionary Computation (CEC'10)*, July 2010, pp. 1–8.

[85] M. Kronfeld, A. Dräger, M. Aschoff, and A. Zell, "On the benefits of multimodal optimization for metabolic network modeling," in *German Conference on Bioinformatics (GCB 2009), LNCS*, ser. Lecture Notes in Informatics, S. P. F. S. Ivo Grosse, Steffen Neumann and P. Stadler, Eds., vol. P-157, no. 978-3-88579-251-2. Halle (Saale), Germany: German Informatics Society, Sep. 2009, pp. 191–200.

[86] F. Streichert, G. Stein, H. Ulmer, and A. Zell, "A clustering based niching ea for multimodal search spaces," in *Artificial Evolution*, ser. Lecture Notes in Computer Science, P. Liardet, P. Collet, C. Fonlupt, E. Lutton, and M. Schoenauer, Eds. Springer Berlin Heidelberg, 2004, vol. 2936, pp. 293–304.

[87] K. Deb, "Genetic algorithms in multimodal function optimization (master thesis and tcga report no. 89002)," Ph.D. dissertation, Tuscaloosa: University of Alabama, The Clearinghouse for Genetic Algorithms, 1989.

[88] J. Rönkkönen, X. Li, V. Kyrki, and J. Lampinen, "A framework for generating tunable test functions for multimodal optimization," *Soft Computing*, vol. 15, no. 9, pp. 1689–1706, 2011.

[89] P. Bezier, *The Mathematical Basis of the UNISURF CAD System*. Newton, MA, USA: Butterworth-Heinemann, 1986.

[90] M. Gallagher and B. Yuan, "A general-purpose tunable landscape generator," *Evolutionary Computation, IEEE Transactions on*, vol. 10, no. 5, pp. 590–603, Oct 2006.

[91] B.-Y. Qu and P. N. Suganthan, "Novel multimodal problems and differential evolution with ensemble of restricted tournament selection," in *Evolutionary Computation (CEC), 2010 IEEE Congress on*, July 2010, pp. 1–7.

[92] J. J. Liang, P. N. Suganthan, and K. Deb, "Novel composition test functions for numerical global optimization," in *Swarm Intelligence*

Symposium, 2005. SIS 2005. Proceedings 2005 IEEE, June 2005, pp. 68–75.

[93] B. Y. Qu, J. J. Liang, Z. Y. Wang, Q. Chen, and P. N. Suganthan, "Novel benchmark functions for continuous multimodal optimization with comparative results," *Swarm and Evolutionary Computation*, vol. 26, pp. 23–34, 2016.

[94] K. Deb and 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*, ser. GECCO '10. New York, NY, USA: ACM, 2010, pp. 447–454.

[95] M. Preuss, "Niching the cma-es via nearest-better clustering," in *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation*, ser. GECCO '10. New York, NY, USA: ACM, 2010, pp. 1711–1718.

[96] J. E. Fieldsend, "Using an adaptive collection of local evolutionary algorithms for multi-modal problems," *Soft Computing*, vol. 19, no. 6, pp. 1445–1460, Jun. 2015.

[97] A. Auger and N. Hansen, "Performance evaluation of an advanced local search evolutionary algorithm," in *The 2005 IEEE Congress on Evolutionary Computation, CEC 2005*, vol. 2, Sept 2005, pp. 1777–1784 Vol. 2.

[98] J. E. Fieldsend, "Multi-modal optimisation using a localised surrogates assisted evolutionary algorithm," in *Computational Intelligence (UKCI), 2013 13th UK Workshop on*, Sept 2013, pp. 88–95.

[99] K. Deb and D. E. Goldberg, "An investigation of niche and species formation in genetic function optimization," in *Proc. of the Third International Conference on Genetic Algorithms*, J. Schaffer, Ed., 1989, pp. 42–50.

[100] M. Preuss and S. Wessing, "Measuring multimodal optimization solution sets with a view to multiobjective techniques," in *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation IV*, ser. Advances in Intelligent Systems and Computing, M. e. a. Emmerich, Ed. Springer International Publishing, 2013, vol. 227, pp. 123–137.

[101] J. Mwaura, A. P. Engelbrecht, and F. V. Nepomuceno, "Performance Measures for Niching Algorithms," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2016.

[102] J. Horn, N. Nafpliotis, and D. E. Goldberg, "A Niched Pareto Genetic Algorithm for Multiobjective Optimization," in *Proc. of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, vol. 1. Piscataway, New Jersey: IEEE Service Center, 1994, pp. 82–87. [Online]. Available: citeseer.ist.psu.edu/horn94niched.html

[103] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, Apr 2002.

[104] A. Zhou, Q. Zhang, and Y. Jin, "Approximating the set of pareto-optimal solutions in both the decision and objective spaces by an estimation of distribution algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 1167–1189, Oct 2009.

[105] K. Deb and S. Tiwari, "Omni-optimizer: A generic evolutionary algorithm for single and multi-objective optimization." *European Journal of Operational Research*, vol. 185, no. 3, pp. 1062–1087, 2008.

[106] A. P. Wierzbicki, "The use of reference objectives in multiobjective optimization," in *Multiple Criteria Decision Making Theory and Application*, ser. Lecture Notes in Economics and Mathematical Systems, G. Fandel and T. Gal, Eds. Springer Berlin Heidelberg, 1980, vol. 177, pp. 468–486.

[107] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *Evolutionary Computation, IEEE Transactions on*, vol. 11, no. 6, pp. 712–731, Dec 2007.

[108] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, part i: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.

[109] H. Seada and K. Deb, "Effect of selection operator on nsga-iii in single, multi, and many-objective optimization," in *2015 IEEE Congress on Evolutionary Computation (CEC)*, May 2015, pp. 2915–2922.

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

[111] A. Basak, S. Das, and K. C. Tan, "Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 5, pp. 666–685, Oct 2013.

[112] Y. Wang, H. X. Li, G. G. Yen, and W. Song, "Mommop: Multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems," *IEEE Transactions on Cybernetics*, vol. 45, no. 4, pp. 830–843, April 2015.

[113] C. Tutum and K. Deb, "A multimodal approach for evolutionary multi-objective optimization (memo): Proof-of-principle results," in *Evolutionary Multi-Criterion Optimization*, ser. Lecture Notes in Computer Science, A. Gaspar-Cunha, C. Henggeler Antunes, and C. C. Coello, Eds. Springer International Publishing, 2015, vol. 9018, pp. 3–18.

[114] O. M. Shir, M. Preuss, B. Naujoks, and M. Emmerich, "Enhancing decision space diversity in evolutionary multiobjective algorithms," in *Proceedings of the 5th International Conference on Evolutionary Multi-Criterion Optimization*, ser. EMO '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 95–109.

[115] O. Kramer and H. Danielsiek, "Dbscan-based multi-objective niching to approximate equivalent pareto-subsets," in *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '10. New York, NY, USA: ACM, 2010, pp. 503–510.

[116] J. Branke, *Evolutionary Optimization in Dynamic Environments*. Norwell, MA, USA: Kluwer Academic Publishers, 2001.

[117] X. Li, J. Branke, and T. Blackwell, "Particle swarm with speciation and adaptation in a dynamic environment," in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '06. New York, NY, USA: ACM, 2006, pp. 51–58.

[118] T. Blackwell, J. Branke, and X. Li, *Swarm Intelligence: Introduction and Applications*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, ch. Particle Swarms for Dynamic Optimization Problems, pp. 193–217.

[119] C. Li and S. Yang, "A general framework of multipopulation methods with clustering in undetectable dynamic environments," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 4, pp. 556–577, Aug 2012.

[120] S. Bird and X. Li, "Using regression to improve local convergence," in *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, Sept 2007, pp. 592–599.

[121] J. Sacks, W. J. Welch, T. J. Mitchell, and H. P. Wynn, "Design and analysis of computer experiments," *Statistical Science*, vol. 4, no. 4, pp. 409–423, 11 1989.

[122] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 61–70, 2011.

[123] I. Schoeman and A. Engelbrecht, "Niching for dynamic environments using particle swarm optimization," in *Simulated Evolution and Learning*, ser. Lecture Notes in Computer Science, T.-D. Wang, X. Li, S.-H. Chen, X. Wang, H. Abbass, H. Iba, G.-L. Chen, and X. Yao, Eds. Springer Berlin Heidelberg, 2006, vol. 4247, pp. 134–141.

[124] J. F. Bard, *Practical Bilevel Optimization: Algorithms and Applications (Nonconvex Optimization and Its Applications)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[125] A. Sinha, P. Malo, and K. Deb, "Test problem construction for single-objective bilevel optimization," *CoRR*, vol. abs/1401.1942, 2014. [Online]. Available: http://arxiv.org/abs/1401.1942

[126] J. MacQueen, "Some methods for classification and analysis of multivariate observations," Berkeley, Calif., pp. 281–297, 1967.

[127] D. K. Tasoulis, V. P. Plagianakos, and M. N. Vrahatis, "Clustering in evolutionary algorithms to efficiently compute simultaneously local and global minima," in *2005 IEEE Congress on Evolutionary Computation*, vol. 2, Sept 2005, pp. 1847–1854.

[128] V. P. Plagianakos, "Unsupervised clustering and multi-optima evolutionary search," in *2014 IEEE Congress on Evolutionary Computation (CEC)*, July 2014, pp. 2383–2390.

[129] D. Zaharie, "Density based clustering with crowding differential evolution," in *2011 13th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC 2005)*. Los Alamitos, CA, USA: IEEE Computer Society, Sept 2005, pp. 343–350.

[130] O. Nasraoui, E. Leon, and R. Krishnapuram, "Unsupervised niche clustering: Discovering an unknown number of clusters in noisy data sets," in *Evolutionary Computation in Data Mining*, ser. Studies in Fuzziness and Soft Computing, A. Ghosh and L. Jain, Eds. Springer Berlin Heidelberg, 2005, vol. 163, pp. 157–188.

[131] V. V. Raghavan and K. Birchard, "A clustering strategy based on a formalism of the reproductive process in natural systems," in *Proceedings of the 2Nd Annual International ACM SIGIR Conference on Information Storage and Retrieval: Information Implications into the Eighties*, ser. SIGIR '79. New York, NY, USA: ACM, 1979, pp. 10–22.

[132] D.-X. Chang, X.-D. Zhang, C.-W. Zheng, and D.-M. Zhang, "A robust dynamic niching genetic algorithm with niche migration for automatic clustering problem," *Pattern Recognition*, vol. 43, no. 4, pp. 1346–1360, 2010.

[133] H.-L. Ling, J.-S. Wu, Y. Zhou, and W.-S. Zheng, "How many clusters? a robust pso-based local density model," *Neurocomputing*, vol. 207, pp. 264–275, 2016.

[134] W. Sheng, X. Liu, and M. Fairhurst, "A niching memetic algorithm for simultaneous clustering and feature selection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 7, pp. 868–879, July 2008.

[135] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE Transactions on Neural Networks*, vol. 5, no. 4, pp. 537–550, Jul. 1994.

[136] F. Z. Brill, D. E. Brown, and W. N. Martin, "Fast generic selection of features for neural network classifiers," *IEEE Transactions on Neural Networks*, vol. 3, no. 2, pp. 324–328, Mar 1992.

[137] P. A. Estevez, M. Tesmer, C. A. Perez, and J. M. Zurada, "Normalized mutual information feature selection," *IEEE Transactions on Neural Networks*, vol. 20, no. 2, pp. 189–201, Feb 2009.

[138] S. Kamyab and M. Eftekhari, "Feature selection using multimodal optimization techniques," *Neurocomputing*, vol. 171, pp. 586–597, 2016.

[139] Y. Liu and X. Yao, "Simultaneous training of negatively correlated neural networks in an ensemble," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 29, no. 6, pp. 716–725, Dec 1999.

[140] M. L. Wong and K. S. Leung, *Data Mining Using Grammar-Based Genetic Programming and Applications*. Norwell, MA, USA: Kluwer Academic Publishers, 2000.

[141] X. Yao and M. M. Islam, "Evolving artificial neural network ensembles," *IEEE Computational Intelligence Magazine*, vol. 3, no. 1, pp. 31–42, Feb. 2008.

[142] X. Yao and Y. Liu, "Making use of population information in evolutionary artificial neural networks," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 28, no. 3, pp. 417–425, Jun 1998.

[143] ——, "A new evolutionary system for evolving artificial neural networks," *IEEE Transactions on Neural Networks*, vol. 8, no. 3, pp. 694–713, May 1997.

[144] Y. Liu, X. Yao, and T. Higuchi, "Evolutionary ensembles with negative correlation learning," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, pp. 380–387, Nov 2000.

[145] C. Castillo, G. Nitschke, and A. Engelbrecht, "Niche particle swarm optimization for neural network ensembles," in *Advances in Artificial Life. Darwin Meets von Neumann*, ser. Lecture Notes in Computer Science, G. Kampis, I. Karsai, and E. Szathmary, Eds. Springer Berlin Heidelberg, 2011, vol. 5778, pp. 399–407.

[146] K. C. Tan, Q. Yu, C. M. Heng, and T. H. Lee, "Evolutionary computing for knowledge discovery in medical diagnosis," *Artificial Intelligence in Medicine*, vol. 27, pp. 129–154, 2003.

[147] K. C. Tan, Q. Yu, and J. H. Ang, "A coevolutionary algorithm for rules discovery in data mining," *International Journal of Systems Science*, vol. 37, no. 12, pp. 835–864, 2006.

[148] P. J. Darwen and X. Yao, "Speciation as automatic categorical modularization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 2, pp. 101–108, Jul 1997.

[149] K. O. Stanley, B. D. Bryant, and R. Miikkulainen, "Real-time neuroevolution in the nero video game," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 6, pp. 653–668, Dec 2005.

[150] G.-C. Luh and C.-Y. Lin, "Optimal design of truss-structures using particle swarm optimization," *Computers and Structures*, vol. 89, no. 23-24, pp. 2221–2232, Dec. 2011.

[151] K. Deb and S. Gulati, "Design of truss-structures for minimum weight using genetic algorithms," *Finite Elements in Analysis Design*, vol. 37, pp. 447–465, 2001.

[152] J. W. Kruisselbrink, A. Aleman, M. T. M. Emmerich, A. P. IJzerman, A. Bender, T. Baeck, and E. van der Horst, "Enhancing search space diversity in multi-objective evolutionary drug molecule design using niching," in *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO'09. New York, NY, USA: ACM, 2009, pp. 217–224.

[153] O. M. Shir, C. Siedschlag, T. Bäck, and M. J. J. Vrakking, "Niching in evolution strategies and its application to laser pulse shaping," in *Proceedings of the 7th International Conference on Artificial Evolution*, ser. EA'05. Berlin, Heidelberg: Springer-Verlag, 2006, pp. 85–96.

[154] E. Pérez, F. Herrera, and C. Hernández, "Finding multiple solutions in job shop scheduling by niching genetic algorithms," *Journal of Intelligent Manufacturing*, vol. 14, no. 3–4, pp. 323–339, 2003.

[155] E. Prez, M. Posada, and F. Herrera, "Analysis of new niching genetic algorithms for finding multiple solutions in the job shop scheduling," *Journal of Intelligent Manufacturing*, vol. 23, no. 3, pp. 341–356, 2012.

[156] E. Pérez, M. Posada, and A. Lorenzana, "Taking advantage of solving the resource constrained multi-project scheduling problems using multi-modal genetic algorithms," *Soft Computing*, vol. 20, no. 5, pp. 1879–1896, 2016.

[157] K. Delibasis, P. A. Asvestas, and G. K. Matsopoulos, "Multimodal genetic algorithms-based algorithm for automatic point correspondence," *Pattern Recognition*, vol. 43, no. 12, pp. 4011–4027, Dec. 2010.

[158] K. D. Koper and M. E. Wysession, "Multimodal function optimization with a niching genetic algorithm: A seis-mological example," *Bulletin of the Seismological Society of America*, vol. 89, pp. 978–988, 1999.

[159] A. Bienvenüe, M. Joannides, J. Bérard, E. Fontenas, and O. François, "Niching in monte carlo filtering algorithms," in *Artificial Evolution*, ser. Lecture Notes in Computer Science, P. Collet, C. Fonlupt, J.-K. Hao, E. Lutton, and M. Schoenauer, Eds. Springer Berlin Heidelberg, 2002, vol. 2310, pp. 19–30.

[160] D. Chang, Y. Zhao, and Y. Xiao, "A robust dynamic niching genetic clustering approach for image segmentation," in *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '11. New York, NY, USA: ACM, 2011, pp. 1077–1084.

[161] C. Spieth, F. Streichert, N. Speer, and A. Zell, "Clustering-based approach to identify solutions for the inference of regulatory networks," in *Proc. of the IEEE Congress onEvolutionary Computation 2005 (CEC'05*, vol. 1. Edinburgh, UK: IEEE Press, Sept 2005, pp. 660–667.

[162] C. Cobos, C. Montealegre, M. F. Mejia, M. Mendoza, and E. Leon, "Web document clustering based on a new niching memetic algorithm, term-document matrix and bayesian information criterion," in *IEEE Congress on Evolutionary Computation*, July 2010, pp. 1–8.

[163] R. Sahajpal, G. V. Ramaraju, and V. Bhatt, "Applying niching genetic algorithms for multiple cluster discovery in spatial analysis," in *Intelligent Sensing and Information Processing, 2004. Proceedings of International Conference on*, 2004, pp. 35–40.

[164] Z. Zhang and H.-S. Seah, "Real-time tracking of unconstrained full-body motion using niching swarm filtering combined with local optimization," in *2011 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, June 2011, pp. 23–28.

[165] J. L. Redondo, J. Fernández, I. García, and P. M. Ortigosa, "Solving the multiple competitive facilities location and design problem on the plane," *Evolutionary Computation*, vol. 17, no. 1, pp. 21–53, Mar. 2009.

[166] R. Brits, A. P. Engelbrecht, and F. v. d. Bergh, "Solving systems of unconstrained equations using particle swarm optimization," in *2002 IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, Oct. 2002, pp. 6 pp. vol.3–.

[167] W. Song, Y. Wang, H. X. Li, and Z. Cai, "Locating multiple optimal solutions of nonlinear equation systems based on multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 3, pp. 414–431, June 2015.

[168] K.-C. Wong, K.-S. Leung, and M.-H. Wong, "Protein structure prediction on a lattice model via multimodal optimization techniques," in *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation*, ser. GECCO '10. New York, NY, USA: ACM, 2010, pp. 155–162.

[169] D.-H. Cho, H.-K. Jung, and C.-G. Lee, "Induction motor design for electric vehicle using a niching genetic algorithm," *IEEE Transactions on Industry Applications*, vol. 37, no. 4, pp. 994–999, Jul 2001.

[170] B. Sareni, L. Krahenbuhl, and A. Nicolas, "Niching genetic algorithms for optimization in electromagnetics. i. fundamentals," *IEEE Transactions on Magnetics*, vol. 34, no. 5, pp. 2984–2987, Sep 1998.

[171] O. Schutze, A. Lara, C. Coello Coello, and M. Vasile, "On the detection of nearly optimal solutions in the context of single-objective space mission design problems," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 225, no. 11, pp. 1229–1242, 2011.

[172] S. Spreng, "Identification of static equilibria via potential energy optimization, master thesis," 2010.

[173] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in non-stationary environments: A survey," *IEEE Computational Intelligence Magazine*, vol. 10, no. 4, pp. 12–25, Nov 2015.

[174] B. Schwartz, *The Paradox of Choice: Why More Is Less*. Harper Perennial, 2004.

[175] M. Preuss, *Multimodal Optimization by Means of Evolutionary Algorithms*, ser. Natural Computing Series. Springer International Publishing, 2016.
[176] ——, "Review of "multimodal optimization by means of evolutionary algorithms" by mike preuss," *SIGEVOlution*, vol. 8, no. 3, pp. 8–9, Mar. 2016.

Xiaodong Li (M03-SM07) received his B.Sc. degree from Xidian University, Xi'an, China, and Ph.D. degree in information science from University of Otago, Dunedin, New Zealand, respectively. Currently, he is a Professor at the School of Science (Computer Science and Software Engineering), RMIT University, Melbourne, Australia. His research interests include evolutionary computation, neural networks, data analytics, multi-objective optimization, multi-modal optimization, and swarm intelligence. He serves as an Associate Editor of the IEEE Transactions on Evolutionary Computation, Swarm Intelligence (Springer), and International Journal of Swarm Intelligence Research. He is a founding member of IEEE CIS Task Force on Swarm Intelligence, a vice-chair of IEEE Task Force on Multi-modal Optimization, and a former chair of IEEE CIS Task Force on Large Scale Global Optimization. He is the recipient of 2013 ACM SIGEVO Impact Award and 2017 IEEE CIS "IEEE Transactions on Evolutionary Computation Outstanding Paper Award".

Michael G. Epitropakis received his B.Sc., M.Sc. and Ph.D. degrees from the Department of Mathematics, University of Patras, Greece. Currently, he is a Lecturer (Assistant Professor) at the Data Science Institute, and the Department of Management Science, Lancaster University Management School, Lancaster University, UK. His current research interests include operational research, computational intelligence, evolutionary computation, computational search and optimization, multi-objective and multi-modal optimization, and search-based software engineering. He has published more than 35 journal and conference papers, and serves as a reviewer for numerous high-impact journals and first tier conferences. His research has currently attracted over 580 citations with h-index 11. He is a founding member and chair of the IEEE CIS Task Force on Multi-modal Optimization.

Kalyanmoy Deb is Koenig Endowed Chair Professor at Department of Electrical and Computer Engineering in Michigan State University, USA. Prof. Deb received his Bachelor's degree from Indian Institute of Technology Kharagpur in 1985 and masters and doctoral degrees from University of Alabama, Tuscaloosa, USA in 1989 and 1991, respectively. Prof. Deb's research interests are in evolutionary optimization and their application in multi-criterio optimization, modeling, and machine learning. He has been a visiting professor at various universities across the world including IITs in India, Aalto University in Finland, University of Skovde in Sweden, Nanyang Technological University in Singapore. He was awarded Infosys Prize, TWAS Prize in Engineering Sciences, CajAstur Mamdani Prize, Distinguished Alumni Award from IIT Kharagpur, Edgeworth-Pareto award, Bhatnagar Prize in Engineering Sciences, and Bessel Research award from Germany. He is fellow of IEEE, ASME, and three Indian science and engineering academies. He has published over 450 research papers with Google Scholar citation of over 92,000 with h-index 100. He is in the editorial board on 20 major international journals. More information about his research contribution can be found from http://www.egr.msu.edu/~kdeb.

Andries Engelbrecht received the Masters and PhD degrees in Computer Science from the University of Stellenbosch, South Africa, in 1994 and 1999 respectively. He is Professor in Computer Science at the University of Pretoria, and serves as Head of the department. He holds the position of South African Research Chair in Artificial Intelligence, and leads the Computational Intelligence Research Group. His research interests include swarm intelligence, evolutionary computation, neural networks, artificial immune systems, and the application of these paradigms to data mining, games, bioinformatics, finance, and difficult optimization problems. He has published over 300 papers in these fields and is author of two books, Computational Intelligence: An Introduction and Fundamentals of Computational Swarm Intelligence.

Prof Engelbrecht is very active in the international community, annually serving as reviewer for over 20 journals and 10 conferences. He is an Associate Editor of the IEEE Transactions on Evolutionary Computation, IEEE Transactions on Neural Networks and Learning Systems, the Swarm Intelligence Journal, and Engineering Applications of Artificial Intelligence. He was co-guest editor of special issues of the IEEE Transactions on Evolutionary Computation and of the Swarm Intelligence journal. He served on the international program committee and organizing committee of a number of conferences, organized special sessions, presented tutorials, and took part in panel discussions. He was the founding chair of the South African chapter of the IEEE Computational Intelligence Society. He is a member of the Evolutionary Computation Technical Committee and the Neural Networks Technical Committee, and serves as member of a number of task forces.