History-Based Topological Speciation for Multimodal Optimization

Lingxi Li and Ke Tang, *Senior Member, IEEE*

***Abstract*—Evolutionary algorithms integrating various niching techniques have been widely used to find multiple optima of an optimization problem. In recent years, an increasing amount of research has been focused on the design and application of speciation-based niching techniques. These techniques rely on speciation to partition a population into subpopulations (species) such that each occupies a different region of attraction (niche) on the fitness landscape. Existing speciation methods are either distance-based or topology-based. Topology-based methods are more flexible and have fewer assumptions than distance-based methods. However, existing topology-based methods all require sampling and evaluating new individuals in order to capture the landscape topography. This incurs additional fitness evaluations (FEs), which is a drawback, especially when the FE budget is limited. In this paper, a new topology-based speciation method named history-based topological speciation (HTS) is proposed. It relies exclusively on search history to capture the landscape topography and, therefore, does not require any additional FEs to be performed. To the best of our knowledge, HTS is the only parameter-free speciation method at the moment. Both theoretical and empirical analyses have been conducted. Theoretical analysis shows that HTS incurs acceptable computational overhead. In the experimental study, HTS outperformed existing topology-based methods on benchmark functions in up to 32-D space and with as many as 50 optima, and the time overhead was practically negligible if a single FE took seconds.**

***Index Terms*—Multimodal optimization, niching, search history, speciation, topology.**

I. Introduction

PROBLEMS possessing multiple optima[1] widely exist in real-world applications. In solving such multimodal problems, one may not be interested in finding just a single optimum, but rather in finding multiple optima [1]–[3]. On the one hand, obtaining multiple optima provides the user with a range of choices. On the other hand, by studying these optima, important insights into the target problem may be revealed.

Through the years, evolutionary algorithms (EAs) [4] have been shown to be a generic and effective approach to optimization, particularly in dealing with black-box problems that cannot be solved otherwise using traditional methods. Utilizing a population, EAs provide the possibility of locating multiple optima in a single run. To achieve this, a certain level of diversity needs to be maintained in the population [5]. However, canonical EAs tend to gradually lose diversity during evolution and converge toward a single optimum [6]–[8]. To overcome this difficulty, numerous methods, commonly known as niching techniques [9], were developed.

Existing niching techniques mainly fall into two categories according to whether they utilize speciation. In general, speciation refers to the partitioning of a population into subpopulations such that each of them occupies a different region of attraction on the fitness landscape. That is, each subpopulation is expected to independently cover a peak and its surrounding region (without loss of generality, maximization problems are assumed). In analogy with natural ecosystems [10], each subpopulation is called a species and each region of attraction is called a niche.

Nonspeciation-based niching techniques try to maintain diversity at the individual level. Fitness sharing [11] and crowding [12] are the two most famous examples. In fitness sharing, a shared fitness is computed for each individual based on its original fitness and distance to other individuals. The shared fitness, which penalizes individuals that cluster together, is then used for selection. In crowding, each offspring only competes with its most similar individual in a randomly chosen subpopulation (called a crowd). In a more recent method, namely deterministic crowding [13], each offspring only competes with its most similar parent. Both the original crowding and deterministic crowding methods select better individuals deterministically. Probabilistic crowding [14], on the other hand, performs selection probabilistically to give less fit individuals a better chance to survive. A portfolio of deterministic and probabilistic crowding was also proposed [15]. Recently, Galán and Mengshoel [16] further proposed the generalized crowding that encompasses both deterministic and probabilistic crowding, and allows better control over the selection pressure.

Speciation-based niching techniques are able to work at the higher species level [17]. In recent years, they have been attracting an increasing amount of attention. For example,

Manuscript received July 9, 2013; revised January 18, 2014; accepted January 31, 2014. Date of publication February 17, 2014; date of current version January 28, 2015. This work was supported in part by the 973 Program of China under Grant 2011CB707006, in part by the National Natural Science Foundation of China under Grants 61175065 and 61329302, in part by the Program for New Century Excellent Talents in University under Grant NECT-12-0512, in part by the Science and Technological Fund of Anhui Province for Outstanding Youth under Grant 1108085J16, and in part by the European Union Seventh Framework Programme under Grant 247619. *(Corresponding author: Ke Tang.)*

The authors are with the University of Science and Technology of China (USTC)–Birmingham Joint Research Institute in Intelligent Computation and Its Applications, School of Computer Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, China (e-mail: lilingxi@mail.ustc.edu.cn; ketang@ustc.edu.cn).

Digital Object Identifier 10.1109/TEVC.2014.2306677

[1]In this paper, by the term *optimum alone*, we mean either a global or local optimum.

species conservation techniques take explicit measures to preserve identified species from one generation to another [18]–[21]. Species-wise local evolution techniques evolve different species independently to facilitate convergence to their respective optima [22]–[24]. The formation of species also enables fitness sharing to be performed within each species instead of the whole population [25], [26]. Most notably, two recently proposed state-of-the-art EAs for multimodal optimization, namely biobjective multipopulation genetic algorithm (BMPGA) [27] and topological species conservation version 2 [28], are both speciation-based.

Speciation-based niching techniques require the population to be partitioned into species in the first place. Existing speciation methods mainly differ in the way they determine whether two individuals are of the same species, and can thereby be categorized into distance-based [29], [30] and topology-based [31], [32].

Distance-based methods rely on the intuition that closer individuals are more likely to be of the same species. Typically, a fixed distance threshold called the niche radius is specified [11]. Two individuals are determined to be of the same species if their distance is below this threshold. The niche radius setting has a strong impact on performance, and is difficult to tune [33], [34]. Deb and Goldberg [35] proposed a formula to estimate this parameter, but it takes the number of optima as input, which is typically not known *a priori* in practice. Furthermore, by using a fixed niche radius setting, one implicitly assumes equally sized and spherically shaped niches [27], [28].

Some adaptation schemes were later suggested in the literature. In the CMA-ES[2] niching algorithm, there is a niche radius for each individual, which is dynamically adapted according to the step size of that individual [36]. This allows the niches to be of different radii. To relax the assumption of spherically shaped niches, Shir *et al*. [37] proposed to use Mahalanobis distance based on the self-adapted covariance matrix of each individual in CMA-ES. This enables the niches to be of a more general ellipsoidal shape. An important advantage of these distance-based methods is that they rely on distance calculations only and do not refer to the fitness landscape for any information. Consequently, they do not require performing any new fitness evaluations (FEs). This is of great value in applications where FEs are expensive and limited.

Topology-based methods rely on the intuition that two individuals should be of different species if there exists a valley between them on the fitness landscape. This methodology makes weaker assumptions about the fitness landscape than the distance-based one, and is able to form species of varying sizes and shapes. Topology-based methods do not suffer from the difficulty of fine tuning the niche radius parameter, since the radius concept is not used at all. Despite the advantages, however, existing topology-based methods all require sampling and evaluating new individuals to capture the landscape topography, incurring extra FE cost [38]. The larger the sample size, the more accurate the speciation process, and the higher the FE cost will be. The advantages come at a price, and a tradeoff must be made. When the FE budget (maximum number of FEs that can be made) is limited, the advantages provided may not be justified considering the FE consumption.

In this paper, we propose a new topology-based speciation method named history-based topological speciation (HTS). Different from existing topology-based methods, HTS relies exclusively on search history (previously evaluated individuals and their corresponding fitness values) to capture the landscape topography, and therefore incurs no extra FE cost at all. As such, it is particularly suitable for applications with a limited FE budget (e.g., expensive optimization problems [39]). Although being FE-free, the proposed method has certain non-FE time and space cost. The time overhead incurred should be negligible in most practical applications. Particularly, in the experimental study, the time overhead was well negligible when a single FE took seconds. With respect to the space cost, theoretical analysis shows that it is totally acceptable on modern personal computers. Last but not the least, to the best of our knowledge, HTS is the only parameter-free speciation method at the moment.

The rest of this paper is organized as follows. Section II presents the generic main speciation procedure that partitions a population into species. Existing topology-based methods to determine whether two individuals are of the same species are then reviewed in Section III. Section IV elaborates on the design and implementation of HTS and presents a theoretical analysis of its computational complexity. Experimental setup and results are given in Sections V and VI, respectively. Finally, conclusions are drawn in Section VII with suggested future work.

II. Main Speciation Procedure

The main speciation procedure that partitions a population into species is generic and has been used by many different speciation methods [18], [24], [26]. The basic idea can be dated back to the dynamic peak identification algorithm proposed by Miller and Shaw [40].

Fig. 1 shows the main speciation procedure. The input is a population $P = \langle \boldsymbol{p_1}, \ldots, \boldsymbol{p_i}, \ldots, \boldsymbol{p_{|P|}} \rangle$, and the output is a set of subpopulations (i.e., species) $S = \{S_1, \ldots, S_j, \ldots, S_{|S|}\}$ that constitutes a partition of P. For each S_j, $S_j.\boldsymbol{seed}$ is the best individual in S_j. The best individual in a species is called a species seed and serves as the representative of the corresponding species. In the very beginning, no species are formed yet, and S is initialized to an empty set (line 1). After that, P is sorted in a fitness descending order for further processing (line 2). Then comes the main loop which is a scan over the individuals in P in the sorted order (lines 3–14). Within each iteration, the current individual $\boldsymbol{p_i}$ is either assigned to an existing species (lines 4–9) or forms a new species itself (lines 10–14). Specifically, $\boldsymbol{p_i}$ is arranged to be tested against each existing species S_j for membership, which is done by determining whether $\boldsymbol{p_i}$ and $S_j.\boldsymbol{seed}$ are of the same species. $\boldsymbol{p_i}$ is assigned to the first S_j for which the membership is confirmed. There may be the case that $\boldsymbol{p_i}$ does not belong to any existing species, which is indicated by the Boolean variable found being false (line 10). When this happens, $\boldsymbol{p_i}$ itself forms a new species S_{new} that is added to

[2]CMA-ES stands for covariance matrix adaptation evolution strategy.

```
speciation(P)
1   S ← ∅;
2   sort P in fitness descending order;
3   for i ← 1 to |P|
4       found ← false;
5       for each S_j ∈ S
6           if p_i and S_j.seed are of the same species
7               S_j ← S_j ∪ {p_i};
8               found ← true;
9               break;
10      if not found
11          new ← |S| + 1;
12          S_new ← {p_i};
13          S_new.seed ← p_i;
14          S ← S ∪ {S_new};
15  return S;
```

Fig. 1. Main speciation procedure that partitions a population P into a set of species S.

S. Since the individuals are processed in a fitness descending order, no subsequent individual is better than the current one. $\boldsymbol{p_i}$ is thus the seed of this newly formed species (line 13). At the time all individuals are properly processed, the procedure terminates by returning S (line 15).

We analyze the time complexity of the main speciation procedure in the number of the crucial tests performed at line 6. The worst case happens when no two individuals are determined to be of the same species. In this case, each individual is identified as a species seed and is tested against each other individual for membership. Consequently, there is a test for each unordered pair of individuals and the time complexity is $\binom{|P|}{2} = |P| \cdot (|P| - 1)/2 = O(|P|^2)$.

As already mentioned, the procedure is generic. Different speciation methods mainly differ in the way the test at line 6 is performed. In Section III, existing topology-based methods to perform the test are reviewed.

III. Existing Topology-Based Methods

There are mainly two existing topology-based methods to determine whether two individuals are of the same species, namely Hill-Valley and recursive middling (RM). Both of them were used in the comparative experimental study. This section reviews the two methods in detail. For both of them, the underlying principle is first explained and a working implementation is presented thereafter.

A. Hill-Valley

Hill-Valley was originally proposed by Ursem [22], [31] and integrated in his multinational genetic algorithm. Since then, it was employed by a number of other EAs for multimodal optimization, including Gan and Warwick's [17] dynamic niche clustering (DNC), Ling *et al.*'s [41] crowding clustering genetic algorithm, and Stoean *et al.*'s [28], [38] topological species conservation versions 1 and 2. Hill-Valley is probably the most widely used topology-based method so far.

```
hill-valley(a, b)
1   for i ← 1 to K
2       β ← i/(K + 1);
3       c ← a + (b − a) · β;
4       if f(c) < min{f(a), f(b)} then return false;
5   return true;
```

Fig. 2. Hill-Valley method to determine whether $\boldsymbol{a}$ and $\boldsymbol{b}$ are of the same species.

Individuals are points in the solution space (location space of the fitness landscape). To determine whether two points $\boldsymbol{a}$ and $\boldsymbol{b}$ are of the same species, Hill-Valley examines the landscape topography along the line segment connecting $\boldsymbol{a}$ and $\boldsymbol{b}$. If there exists a third point $\boldsymbol{c}$ on the line segment whose fitness is lower than that of both $\boldsymbol{a}$ and $\boldsymbol{b}$, a valley that separates $\boldsymbol{a}$ and $\boldsymbol{b}$ to different hills is then identified, and the two points are determined to be of different species. Mathematically, denoting the fitness function by f, $\boldsymbol{a}$ and $\boldsymbol{b}$ are determined to be of different species only if

$$\exists \boldsymbol{c} \in \boldsymbol{ab} : f(\boldsymbol{c}) < \min\{f(\boldsymbol{a}), f(\boldsymbol{b})\} \tag{1}$$

where

$$\boldsymbol{c} \in \boldsymbol{ab} \Leftrightarrow \exists \beta \in [0, 1] : \boldsymbol{c} = \boldsymbol{a} + (\boldsymbol{b} - \boldsymbol{a}) \cdot \beta. \tag{2}$$

In solving black-box optimization problems, an analytic mathematical expression of f is not available. In this case, the condition cannot be tested exactly, but can only be tested approximately. A straightforward way is to sample a number of quantile points on $\boldsymbol{ab}$, and check these sample points for the evidence of a valley. Fig. 2 shows this procedure, where $K > 0$ is the sample size. The procedure returns true when $\boldsymbol{a}$ and $\boldsymbol{b}$ are determined to be of the same species, and false otherwise. It is assumed throughout this paper that $\boldsymbol{a}$ and $\boldsymbol{b}$ are already evaluated. As a result, each call of this procedure costs at most K FEs.

B. Recursive Middling

RM was originally proposed by Yao *et al.*[32] as an improvement on Hill-Valley. They demonstrated its effectiveness by integrating it into DNC, an EA that had originally used Hill-Valley for speciation. Later, RM was also used in BMPGA [27], [42].

RM aims to fix a defect of the condition tested by Hill-Valley. A 1-D example is given in Fig. 3. Clearly, no point on $\boldsymbol{ab}$ has a fitness lower than that of $\boldsymbol{a}$. According to (1), $\boldsymbol{a}$ and $\boldsymbol{b}$ are then determined to be of the same species, whereas they actually fall in different niches on the left and right, respectively. To fix this defect, RM introduces a new rule that alternatively determines $\boldsymbol{a}$ and $\boldsymbol{b}$ to be of different species only if

$$\exists \boldsymbol{a'}, \boldsymbol{b'} \in \boldsymbol{ab}, \boldsymbol{c} \in \boldsymbol{a'b'} : f(\boldsymbol{c}) < \min\{f(\boldsymbol{a'}), f(\boldsymbol{b'})\}. \tag{3}$$

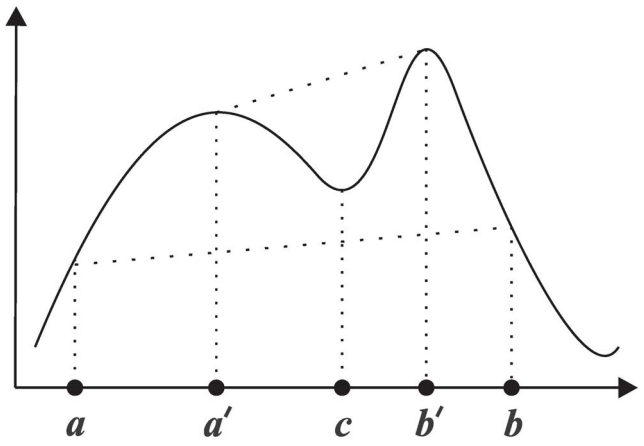


Fig. 3. 1-D example illustrating the defect of the condition tested by Hill-Valley and how the new condition introduced by RM fixed it.

In other words, $\boldsymbol{a}$ and $\boldsymbol{b}$ are determined to be of different species if for some points $\boldsymbol{a}'$ and $\boldsymbol{b}'$ on $\boldsymbol{ab}$, there exists yet another point $\boldsymbol{c}$ on $\boldsymbol{a}'\boldsymbol{b}'$, whose fitness is lower than that of both $\boldsymbol{a}'$ and $\boldsymbol{b}'$.

The difference between the two conditions can be summarized as follows. For the new condition, a valley between the inner points $\boldsymbol{a}'$ and $\boldsymbol{b}'$ is also considered a valley between the outer points $\boldsymbol{a}$ and $\boldsymbol{b}$, whereas for the old condition, this is not necessarily true. The new condition is more conservative than the old one in making the decision that two points are of the same species.

In solving black-box optimization problems, the condition can only be tested approximately. As a matter of fact, RM refers to the particular implementation that recursively samples and checks the midpoint as shown in Fig. 4. Within each recursive call, the (Euclidean) distance $dis(\boldsymbol{a}, \boldsymbol{b})$ between $\boldsymbol{a}$ and $\boldsymbol{b}$ is first computed. If the distance is below the given threshold ϵ, they are then determined to be of the same species by default, and the procedure returns true (line 1). Otherwise, the midpoint $\boldsymbol{c}$ of $\boldsymbol{ab}$ is sampled and evaluated. If it turns out to be inferior to both $\boldsymbol{a}$ and $\boldsymbol{b}$, a valley is then identified, and the procedure determines $\boldsymbol{a}$ and $\boldsymbol{b}$ to be of different species by returning false (line 3). If not, a decision cannot be made right away, and the procedure is invoked recursively on pairs $\langle \boldsymbol{a}, \boldsymbol{c} \rangle$ and $\langle \boldsymbol{c}, \boldsymbol{b} \rangle$, respectively (line 4). $\boldsymbol{a}$ and $\boldsymbol{b}$ are determined to be of the same species only if both of the recursive calls returned true, i.e., when $\boldsymbol{a}$ and $\boldsymbol{c}$ are determined to be of the same species and so are $\boldsymbol{c}$ and $\boldsymbol{b}$. In the worst case, each call of this procedure costs approximately $dis(\boldsymbol{a}, \boldsymbol{b})/\epsilon$ FEs. Most often, ϵ is set to very small values, resulting in highly accurate and expensive speciation processes.

IV. History-Based Topological Speciation

In the evolutionary computation community, there has been a growing interest in archiving search history and putting it to good use in optimization. For example, in robust optimization, Branke [43] utilized history to help estimate the expected fitness of an individual. History was also used in developing efficient EAs that adaptively mutate and never revisit [44], [45]. The evolution path technique employed by CMA-ES is yet another fine example of exploiting history [46].

$recursive\_middling(\boldsymbol{a}, \boldsymbol{b})$

1 if $dis(\boldsymbol{a}, \boldsymbol{b}) < \epsilon$ then return true;
2 $\boldsymbol{c} \leftarrow (\boldsymbol{a} + \boldsymbol{b})/2$;
3 if $f(\boldsymbol{c}) < \min\{f(\boldsymbol{a}), f(\boldsymbol{b})\}$ then return false;
4 return $recursive\_middling(\boldsymbol{a}, \boldsymbol{c}) \wedge recursive\_middling(\boldsymbol{c}, \boldsymbol{b})$;

Fig. 4. RM method to determine whether $\boldsymbol{a}$ and $\boldsymbol{b}$ are of the same species.

In black-box optimization, search history essentially contains all the knowledge acquired so far about the fitness landscape of the target problem. Without sampling and evaluating new points to gain more knowledge, fully exploiting what is currently available is the best one can do. This section elaborates on HTS—a novel topological speciation method based entirely on search history.

A. General Idea

The task is to determine whether $\boldsymbol{a}$ and $\boldsymbol{b}$ are of the same species. First of all, a sequence that consists of all points from $\boldsymbol{a}$ to $\boldsymbol{b}$ on $\boldsymbol{ab}$ can be defined as

$$\bar{X} = \langle \boldsymbol{x_1} = \boldsymbol{a}, \ldots, \boldsymbol{x}_i, \ldots, \boldsymbol{x}_{|\bar{X}|} = \boldsymbol{b} \rangle. \tag{4}$$

The sequence is meant to exactly represent $\boldsymbol{ab}$. It is therefore continuous with the difference between consecutive points being infinitesimal. The cardinality $|\bar{X}|$ can be thought of as ∞. Based on this definition, the conditions tested by Hill-Valley and RM can be expressed in a different mathematical form. In particular, condition (3) for RM can be restated as

$$\exists 1 \leq i < j < k \leq |\bar{X}| : f(\boldsymbol{x}_j) < \min\{f(\boldsymbol{x}_i), f(\boldsymbol{x}_k)\} \tag{5}$$

which is equivalent to the cleaner form

$$\exists 1 < i < |\bar{X}| : f(\boldsymbol{x}_i) < \min\{f(\boldsymbol{x}_{(i-1)}), f(\boldsymbol{x}_{(i+1)})\}. \tag{6}$$

Since there are an infinite number of points in $\bar{X}$, it is practically infeasible to test the sequence exactly by enumerating each point from $\boldsymbol{x_2}$ to $\boldsymbol{x}_{(|\bar{X}|-1)}$. Some sort of approximation has to be made. The approach adopted by RM is to sample $\bar{X}$ and test the resulting sample sequence in place of the original one. The sample sequence is discrete and finite, and can therefore be tested via enumeration. This approach is simple and straightforward. However, any point sampled between $\boldsymbol{a}$ and $\boldsymbol{b}$ is a new point with unknown fitness. Testing the sample sequence necessitates evaluating these new points, resulting in extra FE cost. In order to avoid this, we alternatively propose to approximate $\bar{X}$ with a sequence that consists solely of evaluated history points

$$\bar{Y} = \langle \boldsymbol{y_1} = \boldsymbol{a}, \ldots, \boldsymbol{y}_i, \ldots, \boldsymbol{y}_{|\bar{Y}|} = \boldsymbol{b} \rangle, \quad \forall \boldsymbol{y} \in \bar{Y} : \boldsymbol{y} \in H \tag{7}$$

where H is the set of history points. Clearly, $\bar{Y}$ can be tested without incurring extra FEs. Consequently, how to construct

$\bar{Y}$ to effectively approximate $\bar{X}$ lies at the heart of HTS. Since the issue is trivial in 1-D space, in the following, we only consider cases with a dimensionality larger than one.

B. Formulating Approximate Sequence Construction

In this section, the problem of constructing the approximate sequence $\bar{Y}$ is formally defined. The formulation is based on a fundamental property of the original sequence $\bar{X}$. Given two points $\boldsymbol{u}$ and $\boldsymbol{v}$ in D-dimensional space, a point set that constitutes an open hypercube region can be defined as

$$R(\boldsymbol{u}, \boldsymbol{v}) = \{\boldsymbol{e} = \langle e_1, \ldots, e_d, \ldots, e_D\rangle \mid \forall 1 \leq d \leq D \\ \min\{u_d, v_d\} < e_d < \max\{u_d, v_d\}\}. \quad (8)$$

With the R region definition, the property can be stated as

$$\forall 1 < i < |\bar{X}| : \boldsymbol{x_i} \in R(\boldsymbol{x_{(i-1)}}, \boldsymbol{x_{(i+1)}}). \quad (9)$$

In order to obtain a reasonable approximate sequence, we require that a feasible $\bar{Y}$ also satisfy this property.

To get an idea of the above property, it is easiest to examine the two intuitive subproperties it implies. The first subproperty says that all points in the sequence, except $\boldsymbol{a}$ and $\boldsymbol{b}$, lie in $R(\boldsymbol{a}, \boldsymbol{b})$. The second subproperty says the sequence is monotonic in the sense that

$$\forall 1 \leq i < j \leq |\bar{X}| : dis(\boldsymbol{x_i}, \boldsymbol{b}) > dis(\boldsymbol{x_j}, \boldsymbol{b}). \quad (10)$$

In other words, by moving along the sequence from $\boldsymbol{a}$ to $\boldsymbol{b}$, one is always making positive progress by getting closer and closer to $\boldsymbol{b}$. Empirical studies suggested that both of the two sub-properties are of great significance, and failing to satisfy either of them may lead to a bad approximate sequence that is practically useless. Two examples are given in Fig. 5. For both of them, only one sub-property is satisfied. These infeasible approximate sequences should not be accepted at all. Fig. 6, on the other hand, illustrates a feasible approximate sequence that successfully satisfies both sub-properties.

Given a feasible $\bar{Y}$, its goodness of approximation should be measurable. For this purpose, it is noted that $\bar{X}$ is well approximated by a feasible $\bar{Y}$ so long as each of its points is well represented. How well a point $\boldsymbol{x} \in \bar{X}$ is represented can be inversely measured by the distance to its nearest point in $\bar{Y}$ as

$$dis_{ps}(\boldsymbol{x}, \bar{Y}) = \min_{\boldsymbol{y} \in \bar{Y}} dis(\boldsymbol{x}, \boldsymbol{y}). \quad (11)$$

The name dis_{ps} denotes point-sequence (thus the ps subscript) distance. In the spirit of worst-case analysis, the goodness of approximation of the whole sequence can be measured according to the least-well-represented point as

$$dis_{ss}(\bar{X}, \bar{Y}) = \max_{\boldsymbol{x} \in \bar{X}} dis_{ps}(\boldsymbol{x}, \bar{Y}). \quad (12)$$

The name dis_{ss} denotes sequence-sequence (thus the ss subscript) distance. T he smaller the value, the better the approximation is.

Putting them all together, the approximate sequence construction can be formally formulated as a constrained combinatorial optimization problem

$$\text{minimize}_{\bar{Y}}\ dis_{ss}(\bar{X}, \bar{Y}) \\ \text{subject to } \forall 1 < i < |\bar{Y}| : \boldsymbol{y_i} \in R(\boldsymbol{y_{(i-1)}}, \boldsymbol{y_{(i+1)}}) \\ \text{where } \bar{Y} = \langle \boldsymbol{y_1}, \ldots, \boldsymbol{y_i}, \ldots, \boldsymbol{y_{|\bar{Y}|}}\rangle, \forall \boldsymbol{y} \in \bar{Y} : \boldsymbol{y} \in H. \quad (13)$$

The problem is combinatorial for the solution space comprises all sequences made up from the history points in H. It makes sense to always begin and terminate the approximate sequence with $\boldsymbol{a}$ and $\boldsymbol{b}$, respectively, which has been made explicit in (7). Also note that the trivial two-point sequence $\langle \boldsymbol{a}, \boldsymbol{b}\rangle$ is always a feasible solution to the problem. Solving the problem gives $\bar{Y}$, which is then tested to determine whether $\boldsymbol{a}$ and $\boldsymbol{b}$ are of the same species.

C. Divide-and-Conquer Algorithm

The evaluation of $dis_{ss}(\bar{X}, \bar{Y})$ involves solving a geometric problem. It is difficult, if not impossible, to establish a systematic computational method to evaluate $dis_{ss}(\bar{X}, \bar{Y})$ for all possible input. It is therefore not suitable to solve (13) using optimization methods (e.g., EAs) that depend on evaluating the objective function.

Alternatively, a specialized divide-and-conquer algorithm is designed to solve the problem efficiently. The algorithm is built on the key observation that $dis_{ss}(\bar{X}, \bar{Y})$ is easy to compute for $\bar{Y}$'s that consist of only three points. Such an approximate sequence can be denoted by $\langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b}\rangle$. The derivation of $dis_{ss}(\bar{X}, \langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b}\rangle)$ is demonstrated through the 2-D example shown in Fig. 7, where $\boldsymbol{su}$ and $\boldsymbol{tv}$ are perpendicular bisectors of $\boldsymbol{am}$ and $\boldsymbol{mb}$, respectively. It follows directly that $\|\boldsymbol{sa}\| = \|\boldsymbol{sm}\|$, $\|\boldsymbol{tm}\| = \|\boldsymbol{tb}\|$, and

$$dis_{ps}(\boldsymbol{x}, \langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b}\rangle) \begin{cases} \leq \|\boldsymbol{sa}\|, & \boldsymbol{x} \in \boldsymbol{as} \\ = \|\boldsymbol{sa}\|, & \boldsymbol{x} = \boldsymbol{s} \\ \leq \max\{\|\boldsymbol{sa}\|, \|\boldsymbol{tb}\|\}, & \boldsymbol{x} \in \boldsymbol{st} \\ = \|\boldsymbol{tb}\|, & \boldsymbol{x} = \boldsymbol{t} \\ \leq \|\boldsymbol{tb}\|, & \boldsymbol{x} \in \boldsymbol{tb}. \end{cases} \quad (14)$$

Therefore

$$dis_{ss}(\bar{X}, \langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b}\rangle) = \max_{\boldsymbol{x} \in \boldsymbol{ab}} dis_{ps}(\boldsymbol{x}, \langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b}\rangle) \\ = \max\{\|\boldsymbol{sa}\|, \|\boldsymbol{tb}\|\}. \quad (15)$$

Using trigonometry, it is derivable that

$$\begin{cases} \|\boldsymbol{sa}\| = \dfrac{\|\boldsymbol{am}\|^2 \|\boldsymbol{ab}\|}{2 \cdot \boldsymbol{am} \cdot \boldsymbol{ab}} \\ \|\boldsymbol{tb}\| = \dfrac{\|\boldsymbol{bm}\|^2 \|\boldsymbol{ba}\|}{2 \cdot \boldsymbol{bm} \cdot \boldsymbol{ba}}. \end{cases} \quad (16)$$

As a result, the computation of $dis_{ss}(\bar{X}, \langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b}\rangle)$ is reduced to several inner products. This derivation and the obtained results also hold in higher dimensions.

Based on the above results, an optimal three-point approximate sequence $\bar{Y} = \langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b}\rangle$ can be obtained by enumerating each candidate $\boldsymbol{m} \in H \cap R(\boldsymbol{a}, \boldsymbol{b})$, and selecting the one that minimizes $dis_{ss}(\bar{X}, \langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b}\rangle)$. However, it should be noticed

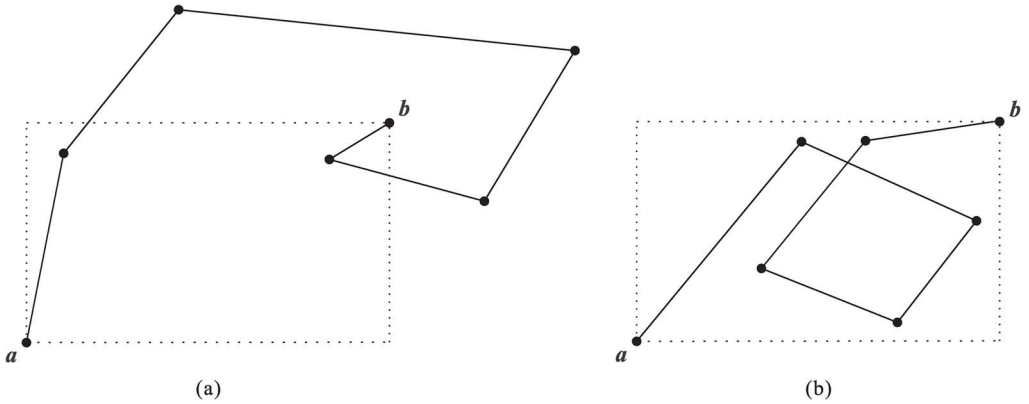


Fig. 5. 2-D examples illustrating infeasible approximate sequences that satisfy only one subproperty. Dotted rectangle marks $R(\boldsymbol{a}, \boldsymbol{b})$. (a) First subproperty is violated. (b) Second subproperty is violated.

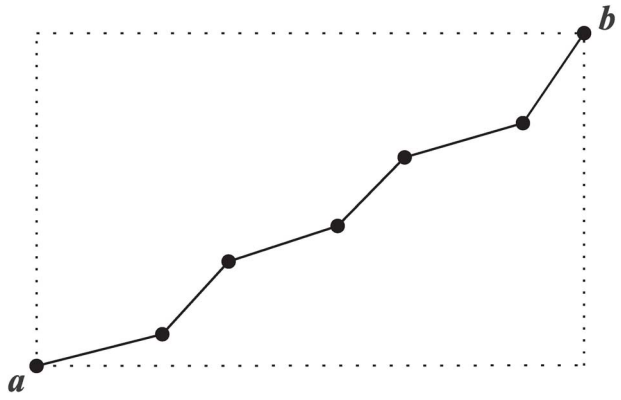


Fig. 6. 2-D example illustrating a feasible approximate sequence that satisfies both subproperties. Dotted rectangle marks $R(\boldsymbol{a}, \boldsymbol{b})$.

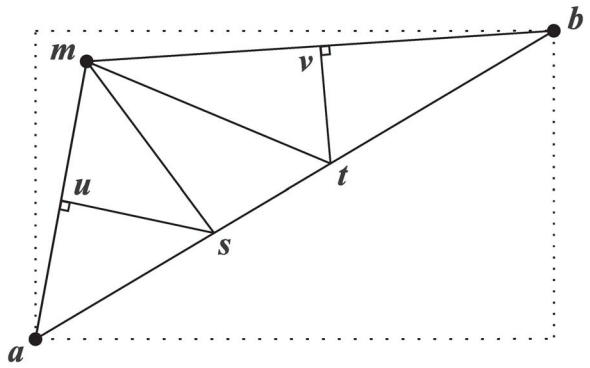


Fig. 7. 2-D example illustrating the derivation of $dis_{ss}(\bar{X}, \bar{Y})$ for $\bar{Y} = \langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b} \rangle$. Dotted rectangle marks $R(\boldsymbol{a}, \boldsymbol{b})$. $\boldsymbol{u}$ and $\boldsymbol{v}$ are the midpoints of $\boldsymbol{am}$ and $\boldsymbol{mb}$, respectively.

that a feasible $\bar{Y}$ with three or more points may not exist at all. This happens when no history points in H lie in $R(\boldsymbol{a}, \boldsymbol{b})$. In this case, an optimal $\boldsymbol{m}$ cannot be found and the overall best approximate sequence is the trivial two-point sequence $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$. If this is not the case, we find $\boldsymbol{m}$ and apply the same procedure recursively to pairs $\langle \boldsymbol{a}, \boldsymbol{m} \rangle$ and $\langle \boldsymbol{m}, \boldsymbol{b} \rangle$, respectively, trying to improve the obtained sequence $\langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b} \rangle$ by adding more suitable points to it. Ultimately, an approximate sequence with more than three points could be obtained.

$construct\_\bar{Y}(\boldsymbol{a}, \boldsymbol{b}, H)$

1 $H' \leftarrow H \cap R(\boldsymbol{a}, \boldsymbol{b})$;
2 if $H' = \emptyset$ then return $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$;
3 $\boldsymbol{m} \leftarrow \arg\min_{\boldsymbol{m} \in H'} dis_{ss}(\bar{X}, \langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b} \rangle)$;
4 $\bar{Y} \leftarrow construct\_\bar{Y}(\boldsymbol{a}, \boldsymbol{m}, H')$;
5 remove the last point of $\bar{Y}$;
6 return $\bar{Y} + construct\_\bar{Y}(\boldsymbol{m}, \boldsymbol{b}, H')$;

Fig. 8. Divide-and-conquer algorithm that constructs the approximate sequence $\bar{Y}$.

The complete procedure of the divide-and-conquer algorithm is shown in Fig. 8. It takes as input two points $\boldsymbol{a}$ and $\boldsymbol{b}$ that mark the original sequence $\bar{X}$, and a set of history points H. The first step finds the history points in $R(\boldsymbol{a}, \boldsymbol{b})$, obtaining the refined set H' (line 1). In order for the constructed $\bar{Y}$ to be feasible, all points except $\boldsymbol{a}$ and $\boldsymbol{b}$ have to be selected from H'.[3] If H' turns out to be empty, then no feasible approximate sequences with more than two points exist, and the best approximate sequence is the trivial two-point sequence $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$ which is returned by the procedure (line 2). Otherwise, an optimal $\boldsymbol{m}$ is found within H' (line 3), and we try to refine the obtained three-point sequence $\langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b} \rangle$ with recursive calls on pairs $\langle \boldsymbol{a}, \boldsymbol{m} \rangle$ and $\langle \boldsymbol{m}, \boldsymbol{b} \rangle$, respectively (lines 4–6). The former refines the part from $\boldsymbol{a}$ to $\boldsymbol{m}$, while the latter refines the part from $\boldsymbol{m}$ to $\boldsymbol{b}$. For efficiency consideration, H' instead of the original H is passed to the recursive calls. The final result is obtained by concatenating (denoted by operator "+") the two sub-sequences returned (line 6). There is a subtlety here. The last point of the first subsequence (namely $\boldsymbol{m}$) returned by the recursive call $construct\_\bar{Y}(\boldsymbol{a}, \boldsymbol{m})$ is the same as the first point of the second subsequence returned by the recursive call $construct\_\bar{Y}(\boldsymbol{m}, \boldsymbol{b})$, whereas a decent feasible approximate sequence should contain no duplicate points. Therefore, before concatenating the two sub-sequences, the last point of the first subsequence is removed (line 5). Alternatively, the first point of the second subsequence could have been removed.

[3]By definition, $R(\boldsymbol{a}, \boldsymbol{b})$ is an open region. $\boldsymbol{a}$ and $\boldsymbol{b}$ are therefore not in $H' = H \cap R(\boldsymbol{a}, \boldsymbol{b})$.

```
hts(a, b, H)
1  H' ← H ∩ R(a, b);
2  if H' = ∅ then return true;
3  m ← arg min_{m∈H'} dis_ss(X̄, ⟨a, m, b⟩);
4  if f(m) < min{f(a), f(b)} then return false;
5  return hts(a, m, H') ∧ hts(m, b, H');
```

Fig. 9. HTS method to determine whether $\boldsymbol{a}$ and $\boldsymbol{b}$ are of the same species.

D. Integrated Implementation

Conceptually, HTS is a two-step procedure. In the first step, an approximate sequence is constructed. In the second step, the obtained sequence is tested to determine whether two individuals are of the same species. In practice, however, it is usually not necessary to first explicitly construct a complete approximate sequence and test it thereafter. As described later in this section, the test can be performed online during the implicit recursive construction of $\bar{Y}$, leading to a more concise and efficient implementation of HTS.

Fig. 9 shows the integrated implementation. The input is the same as the procedure that constructs the approximate sequence. The output is a Boolean variable indicating whether $\boldsymbol{a}$ and $\boldsymbol{b}$ are determined to be of the same species. The first step computes the refined set H' as usual (line 1). If H' turns out to be empty, the best approximate sequence is then $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$. In this case, $\boldsymbol{a}$ and $\boldsymbol{b}$ are determined to be of the same species by default, and the procedure returns true (line 2). Otherwise, an optimal $\boldsymbol{m}$ is found (line 3). The fitness of $\boldsymbol{m}$ is then compared against that of $\boldsymbol{a}$ and $\boldsymbol{b}$. If $\boldsymbol{m}$ turns out to be inferior to both, a valley is then identified, and the procedure determines $\boldsymbol{a}$ and $\boldsymbol{b}$ to be of different species by returning false (line 4). If not, a decision cannot be made right away, and further tests are performed with recursive calls on pairs $\langle \boldsymbol{a}, \boldsymbol{m} \rangle$ and $\langle \boldsymbol{m}, \boldsymbol{b} \rangle$, respectively (line 5). $\boldsymbol{a}$ and $\boldsymbol{b}$ are determined to be of the same species only if $\boldsymbol{a}$ and $\boldsymbol{m}$ are determined to be of the same species and so are $\boldsymbol{m}$ and $\boldsymbol{b}$, i.e., when both recursive calls returned true. With this implementation, it is possible that a final decision be reached before an approximate sequence is fully constructed.

E. Complexity Analysis

HTS is FE-free, but nonetheless incurs a certain amount of other non-FE computational cost. In this subsection, a thorough theoretical analysis of the computational time and space cost of HTS is presented based on the integrated implementation in Fig. 9.

1) *Time Complexity:* Determining whether a point lies in $R(\boldsymbol{a}, \boldsymbol{b})$ costs $O(D)$ time in D-dimensional space. The computation of H' at line 1 therefore costs $O(|H| \cdot D)$ time. The test at line 2 is trivial, costing constant time. According to (15) and (16), the computation of $dis_{ss}(\bar{X}, \langle \boldsymbol{a}, \boldsymbol{m}, \boldsymbol{b} \rangle)$ for a candidate $\boldsymbol{m} \in H'$ amounts to a constant number of inner products of D-dimensional vectors, costing $O(D)$ time. Finding an optimal $\boldsymbol{m}$ at line 3 thus costs $O(|H'| \cdot D)$ time. Since $f(\boldsymbol{a})$, $f(\boldsymbol{m})$ and $f(\boldsymbol{b})$ are already known, the test at line 4 is trivial and costs only constant time. Together, lines 1–4 cost $O((|H| + |H'|) \cdot D)$ time in total. Denoting the time complexity of HTS by $T(|H|)$, the two recursive calls at line 5 cost $2 \cdot T(|H'|)$ time. Finally, we reach the recursive relation

$$T(|H|) = 2 \cdot T(|H'|) + O((|H| + |H'|) \cdot D). \tag{17}$$

There is a chance that the above equation be simplified to an elegant non-recursive one if there exists a fixed relation between $|H'|$ and $|H|$ throughout the recursion. Unfortunately, such a relation does not exist in general. Therefore, we make

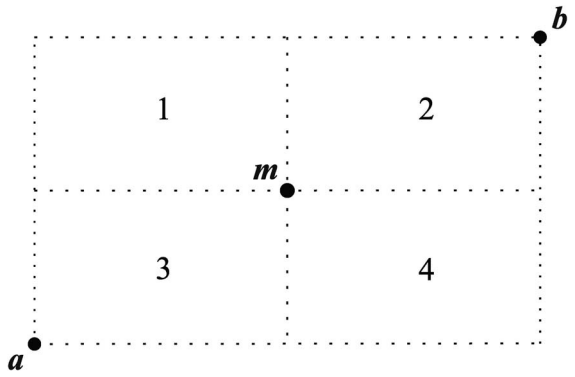


Fig. 10. In 2-D space, an optimal $\boldsymbol{m}$ lying at the center of $R(\boldsymbol{a}, \boldsymbol{b})$ (outer dotted rectangle) divides it into $2^2 = 4$ smaller sub-R regions (inner dotted rectangles) of equal size.

```
cde_with_sc()
1  initialize population P;
2  while termination criterion not satisfied
3      SEEDS ← species seeds of P;
4      O ← offspring population of P;
5      update P with O using crowding method;
6      P ← conserve_seeds(P, SEEDS);
7  return P;
```

Fig. 11. CDE with species conservation technique.

```
conserve_seeds(P, SEEDS)
1  unmark all p ∈ P;
2  for each seed ∈ SEEDS
3      find the worst unmarked p ∈ P
       that is of the same species as seed;
4      if p exists
5          if f(p) < f(seed) then p ← seed;
6      else
7          find the worst unmarked p ∈ P;
8          p ← seed;
9      mark p;
10 return P;
```

Fig. 12. Procedure that conserves the species seeds in *SEEDS* to population *P*.

TABLE I
BENCHMARK FUNCTIONS

F1: Waves
$f_1(x,y) = (0.3x)^3 - (y^2 - 4.5y^2)xy - 4.7\cos\left(3x - y^2(2+x)\right)\sin(2.5\pi x), \quad -0.9 \le x \le 1.2, -1.2 \le y \le 1.2$
10 optima
$(-0.60571004115674230\text{E}{+0}, -1.1775494178290760\text{E}{+0})$ $(+1.19999678072395180\text{E}{+0}, +1.19988474867710160\text{E}{+0})$
$(+0.61757890211528010\text{E}{+0}, +0.8944968298697576\text{E}{+0})$ $(+0.20884852339665191\text{E}{+0}, +1.19990131990056610\text{E}{+0})$
$(+0.87882895378340430\text{E}{+0}, +1.19996380111675570\text{E}{+0})$ $(+1.00614231754130250\text{E}{+0}, +9.67968724649948700\text{E}{-4})$
$(-0.17263899349392534\text{E}{+0}, -2.55036172265160080\text{E}{-5})$ $(+0.58656841319383790\text{E}{+0}, -0.77679193736320280\text{E}{+0})$
$(-0.60941272259790070\text{E}{+0}, +0.80687953596111050\text{E}{+0})$ $(+0.16160561928698120\text{E}{+0}, -1.19993805345059750\text{E}{+0})$
F2: Six-hump camelback
$f_2(x,y) = -\left(\left(4 - 2.1x^2 + \frac{x^4}{3}\right)x^2 + xy + (-4 + 4y^2)y^2\right), \quad -1.9 \le x \le 1.9, -1.1 \le y \le 1.1$
6 optima
$(-0.0898, 0.7126)$ $(0.0898, -0.7126)$ $(-1.7036, 0.7961)$ $(1.7036, -0.7961)$ $(-1.6071, -0.5687)$ $(1.6071, 0.5687)$
F3: Branin RCOS
$f_3(x,y) = -\left(\left(y - \frac{5.1}{4\pi^2}x^2 + \frac{5}{\pi}x - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x + 10\right), \quad -5 \le x \le 10, 0 \le y \le 15$
3 optima
$(-\pi, 12.275)$ $(\pi, 2.275)$ $(9.42478, 2.475)$
F4: Michalewicz
$f_4(x,y) = \sin x \sin^{20}\left(\frac{x^2}{\pi}\right) + \sin y \sin^{20}\left(\frac{2y^2}{\pi}\right), \quad 0 \le x, y \le \pi$
2 optima
$(+2.20287623766878900\text{E}{+0}, +1.57079564832159600\text{E}{+0})$ $(+2.20284650071490340\text{E}{+0}, +2.71156127633579300\text{E}{+0})$
F5: Ursem 1
$f_5(x,y) = \sin(2x - 0.5\pi) + 3\cos y + 0.5x, \quad -2.5 \le x \le 3, -2 \le y \le 2$
2 optima
$(1.697, 0)$ $(-1.444, 0)$
F6: Ursem 4
$f_6(x,y) = 3\sin(0.5\pi x + 0.5\pi)\frac{2 - \sqrt{x^2 + y^2}}{4}, \quad -2 \le x, y \le 2$
5 optima
$(+4.18045982780976500\text{E}{-6}, -6.89579106855229800\text{E}{-6})$ $(+1.99997853931785550\text{E}{+0}, +1.99969965302390800\text{E}{+0})$
$(-1.99991044693437960\text{E}{+0}, -1.99998361745823130\text{E}{+0})$ $(+1.99938220972925750\text{E}{+0}, -1.99976040930306000\text{E}{+0})$
$(-1.99985423140364070\text{E}{+0}, +1.99924168232428960\text{E}{+0})$
F7: $-$MMP(4)
$f_7(\boldsymbol{x}) = -\sum_{i=1}^{4}\left(10(1 + \cos(2\pi k_i x_i)) + 2k_i x_i^2\right), k_1 = 2, k_2 = 2, k_3 = 3, k_4 = 4, \quad \boldsymbol{x} \in [0,1]^4$
48 optima, see text
F8: $-$MMP(8)
$f_8(\boldsymbol{x}) = -\sum_{i=1}^{8}\left(10(1 + \cos(2\pi k_i x_i)) + 2k_i x_i^2\right), k_2 = 2, k_4 = 2, k_6 = 3, k_8 = 4,$ otherwise $k_i = 1, \quad \boldsymbol{x} \in [0,1]^8$
48 optima, see text
F9: $-$MMP(16)
$f_9(\boldsymbol{x}) = -\sum_{i=1}^{16}\left(10(1 + \cos(2\pi k_i x_i)) + 2k_i x_i^2\right), k_4 = 2, k_8 = 2, k_{12} = 3, k_{16} = 4,$ otherwise $k_i = 1, \quad \boldsymbol{x} \in [0,1]^{16}$
48 optima, see text
F10: $-$MMP(32)
$f_{10}(\boldsymbol{x}) = -\sum_{i=1}^{32}\left(10(1 + \cos(2\pi k_i x_i)) + 2k_i x_i^2\right), k_8 = 2, k_{16} = 2, k_{24} = 3, k_{32} = 4,$ otherwise $k_i = 1, \quad \boldsymbol{x} \in [0,1]^{32}$
48 optima, see text
F11: Composite*
$f_{11}(\boldsymbol{x}) = \max_{i=1..50} -\boldsymbol{c_i}^{\mathrm{T}}(\boldsymbol{z_i} \circ \boldsymbol{z_i}), \boldsymbol{z_i} = \boldsymbol{M_i}(\boldsymbol{x} - \boldsymbol{o_i}),$ see text for $\boldsymbol{c_i}, \boldsymbol{M_i},$ and $\boldsymbol{o_i}, \quad \boldsymbol{x} \in [0,1]^{32}$
50 optima, see text

*Column vectors are assumed.

two assumptions about the root procedure call. First, the points in H' are uniformly distributed in $R(\boldsymbol{a}, \boldsymbol{b})$ (the uniformity property). Second, $|H'| = |H|/2^D$. The fact is, if the two properties hold in the root call, they also hold in subsequent recursive calls by induction. This can be briefly explained as follows. Due to the uniformity property, the optimal $\boldsymbol{m}$ found at line 3 would lie at the center of $R(\boldsymbol{a}, \boldsymbol{b})$ on average, dividing it into 2^D smaller sub-R regions of equal size. A 2-D example is shown in Fig. 10. Likewise, the point set H' is divided into 2^D smaller sub-point sets of equal size, each of which contains $|H'|/2^D$ points uniformly distributed in one of the 2^D sub-R regions. The H and H' in the two direct recursive calls at line 5 correspond to the H' and one of the sub-point sets in the current call, respectively. As a result, the uniformity property and the equality that $|H'| = |H|/2^D$ also hold in the direct recursive calls. With this recursive invariant, (17) becomes

$$T(|H|) = 2 \cdot T(|H|/2^D) + O((1 + 1/2^D) \cdot |H| \cdot D) \tag{18}$$

which simplifies to

$$T(|H|) = \begin{cases} O(|H| \log_2 |H|), & D = 1 \\ O((1 + 1/2^D) \cdot |H| \cdot D), & D \ge 2 \end{cases} \tag{19}$$

using the master theorem [47].

Given $D \ge 2$, $T(|H|)$ is at most $O(5/4 \cdot |H| \cdot D)$ which is linear in $|H|$. Recalling that Hill-Valley costs at least one FE

TABLE II
OPTIMAL x_i VALUES FOR DIFFERENT k_i VALUES

k_i	1	2	3	4
x_i	0.49498	0.24874	0.16611	0.12468
	–	0.74622	0.49832	0.37405
	–	–	0.83053	0.62342
	–	–	–	0.87279

and RM usually costs much more than that, whether HTS is more time consuming than them mainly depends on the time cost of a single FE. Since the time complexity of a single FE is no less than $O(D)$ and associates with huge time constants for expensive optimization problems, HTS can be expected faster in such cases.

2) *Space Complexity:* The space cost of HTS mainly concerns the storage of history points (D-dimensional vectors) and their corresponding fitness values (scalars). It is simple to see that the space complexity for archiving search history is

$$M(|H|) = O(|H| \cdot (D + 1)). \tag{20}$$

To get a practical feeling of this complexity, assume an $\mathbb{R}^D$ solution space with real fitness values. Until now, most problems attempted in the evolutionary computation community have a dimensionality no larger than 1023. For these problems, a single history entry (a history point plus its fitness value) consists of no more than 2^{10} real numbers that amounts to 2^{13} bytes using double-precision floating-point representation. With vacant memory space of 2GB, which is common for modern personal computers, at least 2^{18} entries can be loaded at the same time. This is sufficient for most real-world problems. Therefore, the space required for archiving search history should not be a problem in general.

V. EXPERIMENTAL SETUP

To empirically assess the efficacy of HTS, comparative experimental study was carried out. In this section, the design and parameter settings of the experiments are presented.

A. Algorithms

In total, four speciation methods were compared, namely HTS, the distance-based method using a fixed niche radius setting, Hill-Valley, and RM. In order to acquire some kind of performance upper bound for topology-based speciation, a fictional method was made up and compared additionally. It works exactly the same as RM, but with all its FE cost being artificially neglected. In other words, it was meant to be the FE-free version of RM. For clarity, RM and this fictional method are denoted by RM and RM*, respectively.

Different speciation methods should be integrated into the same EA framework for a fair comparison. To this end, crowding differential evolution (CDE) [48] with species conservation technique [18] was employed. CDE is a typical differential evolution algorithm, but with the survival selection scheme being replaced by the crowding method where the subpopulation size is set to the whole population size. The species conservation technique-employed implements a kind of elitism that preserves identified species seeds of one generation to the next.

Figs. 11 and 12 together outline this EA framework. The species conservation technique is introduced in the main procedure (Fig. 11) via lines 3 and 6. The population P is initialized with random individuals uniformly distributed in the search space (line 1). At line 3, the species seeds are identified by means of speciation. At line 4, the offspring population O is produced from the parental population P using the standard DE/rand/1/bin variant [49] of differential evolution. Any offspring individuals generated outside the box-constrained search space during this process are fixed using the reflection method [50]. Survival selection is then performed on the parental and offspring populations at line 5, using the crowding method. The step at line 6 makes sure that the species seeds identified at line 3 are properly conserved in P. The main procedure terminates by returning the final generation of P (line 7).

B. Benchmark Functions

The selection of benchmark functions was based on the goal of optimization. In general, for a moderate number of optima,[4] one may aim to find all of them. For an excessive number of optima, however, the common practice is to seek just a few, say N_{best}, of the best optima while ignoring the rest [28], [40]. To test the speciation methods to the full extent, we aimed to find all optima. It should be noted that the speciation methods, including HTS, also apply to the case when only the best N_{best} optima are targeted. To achieve this, in the main speciation procedure, once the first (also the best) N_{best} species (seeds) are identified, assign each of the remaining individuals directly to its closest species seed without any membership checks.

Since all optima were targeted, all optima of each benchmark function had to be known *a priori*. Subject to this requirement, six 2-D functions used in [28] were first selected (none of the others meet this requirement). They are numbered as F1–F6 in Table I. In the experiments, the search space was normalized to $[0, 1]^D$ for all functions. The box constraints shown in Table I for F1–F6 correspond to the original search spaces of these functions specified in [28].

In order to study the performance of the speciation methods in higher dimensions with a larger number of optima to find, the MMP function proposed by Deb and Saha [5] was also employed. The general form of an MMP(D) function is

$$f(\boldsymbol{x}) = \sum_{i=1}^{D} \left(10(1 + \cos(2\pi k_i x_i)) + 2k_i x_i^2\right), \boldsymbol{x} \in [0, 1]^D$$
$$\text{where } k_i \in \{1, 2, 3, 4\} \text{ for all } i. \tag{21}$$

MMP functions are similar to Rastrigin's functions, but with all optima known *a priori*. Specifically, there are k_i optimal values for x_i as summarized in Table II. Therefore, there are $\prod_{i=1}^{D} k_i$ optima in total. Deb and Saha defined a specific

[4]Deb and Saha [5] once tried to find all 500 optima of a 2-D benchmark function.

TABLE III

MEANS AND STANDARD DEVIATIONS (ITALIC) OF THE DISTANCE ERROR OBTAINED. THE FICTIONAL METHOD RM* WAS INTENDED TO PROVIDE A PERFORMANCE UPPER BOUND FOR TOPOLOGY-BASED SPECIATION

	HTS	DIS-	DIS	DIS+	HV1	HV3	HV5	RM	RM*
F1	7.73E−2	5.94E−2	5.70E−2	3.97E−2	5.93E−2	6.44E−2	6.77E−2	7.03E−2	3.85E−2
	2.12E-2	*1.55E-2*	*2.13E-2*	*1.58E-2*	*1.63E-2*	*1.65E-2*	*1.48E-2*	*1.45E-2*	*1.64E-2*
F2	1.33E−1	3.91E−2	8.45E−2	8.99E−2	1.00E−1	6.10E−2	5.91E−2	5.41E−2	2.52E−2
	1.54E-2	*1.32E-2*	*3.55E-2*	*2.93E-2*	*2.12E-2*	*2.34E-2*	*2.39E-2*	*1.42E-2*	*1.99E-2*
F3	1.91E−2	2.39E−2	1.93E−2	2.14E−2	3.27E−2	4.22E−2	4.45E−2	5.56E−2	1.88E−2
	6.84E-3	*9.35E-3*	*6.67E-3*	*6.58E-3*	*1.12E-2*	*1.62E-2*	*1.32E-2*	*1.80E-2*	*7.26E-3*
F4	5.81E−3	8.29E−3	7.48E−3	1.00E−2	2.24E−2	3.28E−2	3.92E−2	5.32E−2	4.77E−3
	2.93E-3	*5.23E-3*	*4.40E-3*	*5.11E-3*	*1.01E-2*	*1.39E-2*	*1.70E-2*	*2.22E-2*	*2.16E-3*
F5	1.25E−2	1.54E−2	1.22E−2	1.56E−2	2.53E−2	3.28E−2	3.95E−2	4.97E−2	1.27E−2
	6.18E-3	*7.36E-3*	*8.96E-3*	*7.08E-3*	*1.12E-2*	*1.52E-2*	*1.71E-2*	*1.87E-2*	*5.90E-3*
F6	3.10E−2	6.45E−2	2.51E−2	3.56E−2	5.40E−2	7.29E−2	7.87E−2	9.51E−2	2.67E−2
	2.08E-2	*2.86E-2*	*8.48E-3*	*1.13E-2*	*1.28E-2*	*1.96E-2*	*2.26E-2*	*1.94E-2*	*7.83E-3*
F7	5.44E−2	1.39E−1	4.30E−2	5.92E−2	1.01E−1	1.21E−1	1.34E−1	1.34E−1	3.64E−2
	4.07E-3	*7.17E-3*	*3.64E-3*	*4.54E-3*	*6.31E-3*	*9.00E-3*	*6.24E-3*	*6.15E-3*	*3.03E-3*
F8	1.76E−1	3.79E−1	1.46E−1	1.76E−1	2.75E−1	3.29E−1	3.49E−1	3.90E−1	8.87E−2
	1.08E-2	*1.49E-2*	*1.12E-2*	*1.18E-2*	*1.50E-2*	*1.26E-2*	*1.39E-2*	*1.54E-2*	*9.35E-3*
F9	3.22E−1	6.68E−1	3.18E−1	3.22E−1	4.68E−1	5.84E−1	6.35E−1	7.52E−1	1.51E−1
	1.50E-2	*2.19E-2*	*1.62E-2*	*1.50E-2*	*2.37E-2*	*2.83E-2*	*2.82E-2*	*2.69E-2*	*1.43E-2*
F10	5.11E−1	1.03E+0	5.09E−1	5.11E−1	7.42E−1	9.41E−1	1.02E+0	1.17E+0	2.51E−1
	2.18E-2	*3.12E-2*	*2.24E-2*	*2.18E-2*	*3.05E-2*	*3.77E-2*	*3.20E-2*	*3.29E-2*	*2.30E-2*
F11	1.37E+0	1.51E+0	1.52E+0	1.37E+0	1.53E+0	1.51E+0	1.52E+0	1.59E+0	1.45E+0
	3.10E-2	*1.63E-2*	*6.22E-2*	*3.10E-2*	*2.60E-2*	*2.17E-2*	*1.74E-2*	*1.62E-2*	*3.76E-2*
W/D/L	–	9/0/2	2/5/4	4/5/2	9/0/2	9/0/2	9/0/2	9/0/2	1/3/7

TABLE IV

p-VALUES OBTAINED FOR HTS VERSUS EVERY OTHER METHOD USING TWO-SIDED t-TEST WITHOUT EQUAL VARIANCES ASSUMPTION

	DIS-	DIS	DIS+	HV1	HV3	HV5	RM	RM*
F1	2.42E−008	2.73E−008	6.10E−024	3.56E−008	5.06E−005	1.61E−003	1.95E−002	1.45E−024
F2	9.18E−080	1.95E−018	6.74E−020	1.77E−019	0.91E−045	7.93E−046	8.85E−069	1.04E−073
F3	4.10E−004	4.68E−002	4.02E−002	3.56E−015	1.08E−019	7.06E−028	1.44E−029	7.79E−001
F4	5.05E−004	7.20E−003	7.46E−009	2.87E−023	1.79E−027	1.06E−027	1.05E−029	1.47E−002
F5	9.43E−003	7.61E−001	4.43E−003	2.94E−014	3.67E−018	1.91E−022	8.48E−029	8.23E−001
F6	1.60E−013	2.63E−002	9.32E−002	3.29E−013	2.29E−025	2.03E−027	1.18E−042	9.96E−002
F7	2.65E−109	1.99E−039	4.44E−010	2.46E−089	6.03E−081	1.84E−118	1.27E−119	2.11E−063
F8	1.97E−125	1.71E−035	9.33E−001	1.05E−084	2.06E−121	1.10E−121	1.16E−125	5.90E−097
F9	1.60E−132	1.18E−001	1.00E+000	2.95E−079	3.63E−095	5.74E−104	5.95E−124	1.97E−116
F10	1.61E−135	6.79E−001	1.00E+000	4.08E−092	1.81E−008	1.04E−132	2.82E−144	8.24E−116
F11	5.16E−062	7.51E−036	1.00E+000	2.76E−070	9.23E−065	9.58E−066	3.88E−081	2.26E−028

instance for each of MMP(4), MMP(8), and MMP(16). They all possess 48 optima. They were adopted in the experiments and are numbered as F7–F9 in Table I. We additionally defined an instance of MMP(32) numbered as F10, which also has 48 optima. MMP functions were originally designed to be minimized. To make maximization problems, they were all inversed by appending a minus sign to the front.

MMP functions are separable [51]. In order to assess the performance in a more general setting, we have created and used a 32-D nonseparable benchmark function numbered as F11. T he function was synthesized by combining 50 shifted and rotated ellipsoidal subfunctions

$$f_{11}(\boldsymbol{x}) = \max_{i=1..50} -\boldsymbol{c_i}^{\mathrm{T}}(\boldsymbol{z_i} \circ \boldsymbol{z_i}),\ \boldsymbol{z_i} = \boldsymbol{M_i}(\boldsymbol{x} - \boldsymbol{o_i}) \tag{22}$$

where $\boldsymbol{c_i}$, $\boldsymbol{M_i}$, and $\boldsymbol{o_i}$ are the coefficient (column) vector, rotation matrix, and shift vector for the ith subfunction, respectively. Each entry in $\boldsymbol{c_i}$ and $\boldsymbol{o_i}$ was generated uniformly in the unit interval $[0, 1]$, while $\boldsymbol{M_i}$ was generated uniformly in the rotation matrix space [52]. F11 possesses 50 optima in total, each of which corresponds to the center of an involved ellipsoidal subfunction.

C. Performance Measure

The EA was terminated when the given FE budget was expended. The distance error (also called distance accuracy) [28] of the final generation was then used to measure the performance. The distance error of a population P is computed in the following way. Denote the number of optima by N_o. For each optimum $\boldsymbol{o}$ possessed by the function, the distance

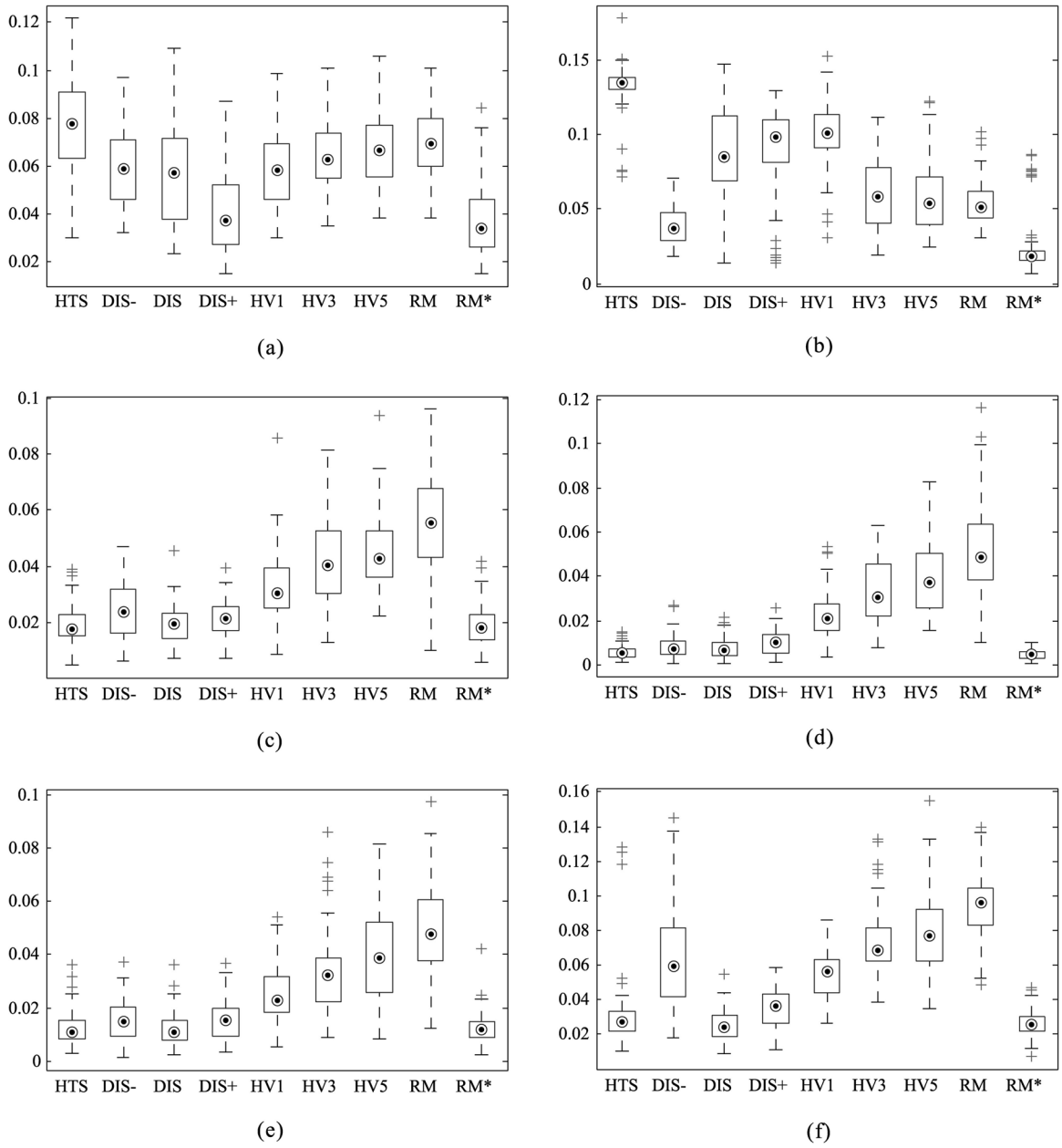


Fig. 13. Box plots of the distance error obtained on the 2-D functions. On each box, the median is shown by a dot inside a circle, the lower and upper edges of the box mark the 25th and 75th percentiles respectively, the whiskers extend to the most extreme data points not considered outliers, while outliers are plotted individually with the + sign. The fictional method RM* was employed to provide a performance upper bound for topology-based speciation. (a) F1. (b) F2. (c) F3. (d) F4. (e) F5. (f) F6.

to its nearest individual in P is computed. The distance error is defined as the average of this minimum distance taken over all optima. Mathematically, that is

$$error(P) = \frac{1}{N_o} \sum_{\boldsymbol{o}} \min_{\boldsymbol{p} \in P} dis(\boldsymbol{o}, \boldsymbol{p}). \tag{23}$$

The smaller the value, the better the performance is. We chose this performance measure because it is parameter-free and is expected to produce an objective measuring result.

To get an actual feeling of the time overhead of HTS, the physical execution time (wall-clock time) of each call to HTS to determine whether two individuals are of the same species was recorded. The experimental environment was as follows. The program was written in C++, single-threaded, and compiled in 64-bit using Visual C++ 2010. Double-precision floating-point representation was used for real numbers. The operating system was Windows 7 64-bit. The computer was a personal ultrabook equipped with an Intel Core i5-2537M CPU working at 1.40GHz and 4GB RAM.

D. Parameter Settings

FE budgets are limited in expensive optimization problems [39], [53]. In the experiments, the FE budget was set to 500 for F1–F6, and to 2500, 5000, 10 000, 20 000, and 20 000, respectively, for F7–F11. With respect to the EA framework, the population size was set to 50 for F1–F6 and to 250 for

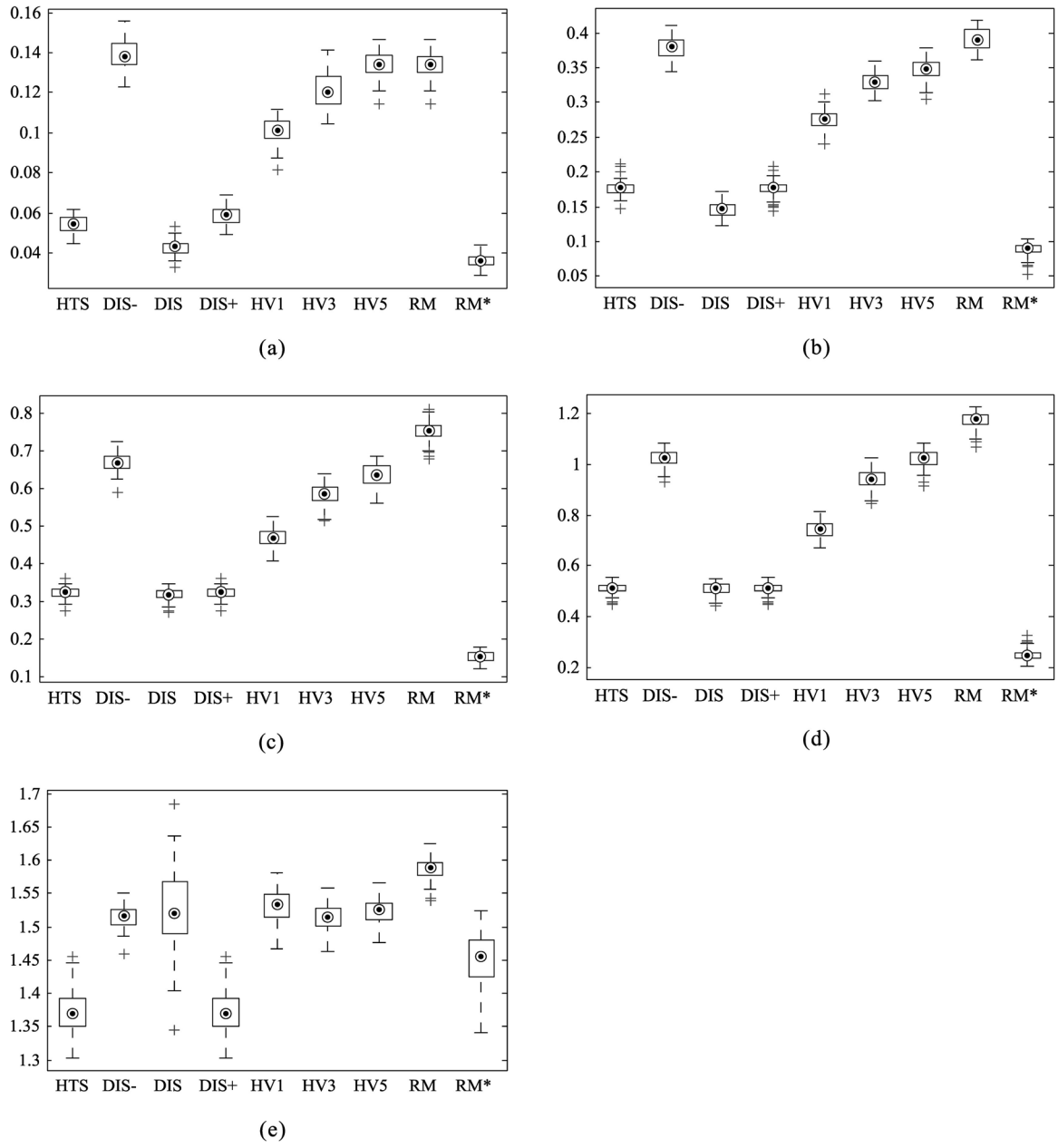


Fig. 14. Box plots of the distance error obtained on F7–F11. On each box, the median is shown by a dot inside a circle, the lower and upper edges of the box mark the 25th and 75th percentiles respectively, the whiskers extend to the most extreme data points not considered outliers, while outliers are plotted individually with the + sign. The fictional method RM* was employed to provide a performance upper bound for topology-based speciation. (a) F7. (b) F8. (c) F9. (d) F10. (e) F11.

F7–F11. When the FE budget is limited, quick convergence should typically be preferred to more reliable but slower convergence. Therefore, relatively small populations were used. The scaling factor F and crossover probability CR used by DE/rand/1/bin were set to 0.9 and 0.1, respectively [54].

The speciation methods were configured as follows. HTS is parameter-free. With respect to the distance-based method, three configurations were tested. In the first configuration denoted by DIS, the niche radius is computed by Deb and Goldberg's formula using the exact number of optima possessed by a benchmark function. In the other two configurations, denoted by DIS- and DIS+ respectively, the niche radius is set to 1/5 and 5 times the one used in DIS, respectively. Hill-Valley requires the sample size K to be set, and three configurations HV1, HV3, and HV5 were tested, where the sample size is set to 1, 3, and 5, respectively. The distance threshold ϵ for RM was set to 0.01 for F1–F6, and to 0.02, 0.04, 0.08, 0.16, and 0.16, respectively, for F7–F11. To obtain meaningful statistics, all experiments were conducted for 75 independent runs.

VI. Experimental Results

The means and standard deviations of the distance error obtained are shown in Table III. For a more comprehensive graphical view of the results, box plots are provided in Figs. 13 and 14. To assess the statistical significance of the performance

TABLE V

p-VALUES OBTAINED FOR HTS VERSUS EVERY OTHER METHOD USING TWO-SIDED WILCOXON RANK-SUM TEST

	DIS-	DIS	DIS+	HV1	HV3	HV5	RM	RM*
F1	1.16E−007	2.32E−007	1.74E−018	1.82E−007	6.56E−005	1.81E−003	2.25E−002	5.55E−019
F2	4.13E−026	1.34E−017	7.45E−022	1.38E−018	8.33E−025	4.82E−025	8.81E−026	8.14E−026
F3	2.08E−003	6.93E−001	6.21E−003	6.82E−016	5.19E−019	2.95E−023	8.12E−024	6.99E−001
F4	1.92E−003	3.56E−002	1.00E−007	3.99E−023	1.60E−025	4.13E−026	5.68E−026	3.77E−002
F5	6.00E−003	8.01E−001	3.37E−003	6.45E−014	3.69E−019	6.97E−021	1.68E−024	5.43E−001
F6	1.63E−017	6.12E−002	4.32E−005	8.05E−019	4.63E−022	4.98E−022	1.14E−022	5.28E−001
F7	4.13E−026	2.04E−024	9.28E−009	4.13E−026	4.13E−026	4.13E−026	4.13E−026	4.13E−026
F8	4.13E−026	1.14E−023	6.93E−001	4.13E−026	4.13E−026	4.13E−026	4.13E−026	4.13E−026
F9	4.13E−026	1.64E−001	1.00E+000	4.13E−026	4.13E−026	4.13E−026	4.13E−026	4.13E−026
F10	4.13E−026	7.13E−001	1.00E+000	4.13E−026	4.13E−026	4.13E−026	4.13E−026	4.13E−026
F11	4.13E−026	3.36E−024	1.00E+000	4.13E−026	4.13E−026	4.13E−026	4.13E−026	1.37E−020

TABLE VI

MEANS AND STANDARD DEVIATIONS (STD) OF THE RECORDED PHYSICAL EXECUTION TIME OF A SINGLE CALL TO HTS TO DETERMINE WHETHER TWO INDIVIDUALS ARE OF THE SAME SPECIES

	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11
Mean (ms)	7.33E−3	1.14E−2	1.03E−2	9.21E−3	9.00E−3	1.07E−2	2.75E−2	7.37E−2	1.69E−1	3.82E−1	3.76E−1
STD	8.53E−2	1.06E−1	1.01E−1	9.97E−2	9.44E−2	1.03E−1	6.08E−1	1.07E+0	1.36E+0	2.41E+0	5.72E−1

difference between HTS and every other method, both *t*-test and Wilcoxon rank-sum test [55] were employed. The *p* values obtained using the two tests are reported in Tables IV and V, respectively. We consider a difference to be statistically significant only if it was asserted so by both tests at the significance level of 0.05. As can be seen, the results of the two tests are highly consistent. Different assertions were made on only 3 out of all 88 comparisons. Two of them concern the comparisons with DIS on F3 and F6 respectively, while the third concerns the comparison with DIS+ on F6. The last row in Table III summarizes the number of wins/draws/losses of HTS versus every other method on all 11 benchmark functions. A draw is counted when no statistically significant difference is observed.

First of all, it can be clearly observed that HTS outperformed the other two topology-based methods in general. In particular, it achieved better results than all three configurations of Hill-Valley (HV1, HV3, HV5) and RM on all functions except F1 and F2. The superiority of HTS over existing topology-based methods was expected. Being FE-free, HTS saved all precious FEs for the EA to converge, and thereby effectively improved the overall performance.

In comparison with DIS, the performance of HTS was not as promising as expected. It won on F4 and F11 only, while lost on F1, F2, F7, and F8. No statistically significant difference is observed on the other functions. However, it should be noted that the niche radius setting of DIS is kind of optimal as it is computed using the exact number of optima, which is typically not known *a priori* in practice. For suboptimal niche radius settings, the performance of the distance-based method may degrade dramatically. This can be seen by referring to the results of DIS- and DIS+. Comparing with DIS-, HTS won on 9 out of all 11 functions. Comparing with DIS+, HTS won on F3–F5 and F7, while lost on F1 and F2 only. No statistically significant difference is observed on the remaining functions. To summarize, the mixed results suggest that neither of HTS nor the distance-based method is absolutely better than the other. Nevertheless, since HTS is parameter-free, it still provides a competitive approach to speciation.

RM* is a fictional method made up to obtain some kind of performance upper bound for topology-based speciation. It is therefore unsurprising that RM* achieved the best overall performance among all methods. This demonstrates the great potential of topology-based speciation for finding multiple optima. Loosely speaking, HTS can be seen as an effort to approximate RM*. Despite the great difficulties, no statistically significant difference is observed between them on F3, F5, and F6. Somewhat surprisingly, HTS even managed to outperform RM* on the probably most difficult F11.

The means and standard deviations of the recorded physical execution time of HTS to determine whether two individuals are of the same species are shown in Table VI. The means were taken over all calls in a run and over all 75 independent runs. It can be observed that the average time was less than a millisecond on all functions. In this case, when a single FE took seconds or more, the time overhead incurred by HTS was practically negligible. Therefore, HTS is particularly suitable for expensive optimization problems, where a single FE may take hours or even days to complete.

VII. CONCLUSION

In this paper, a new speciation method named HTS is proposed, which can be integrated into a variety of niching techniques for solving multimodal problems. The crucial task in HTS is to approximate a continuous point sequence along a line segment using evaluated history points. The task is formulated as a non-trivial constrained combinatorial optimization problem. A specialized divide-and-conquer algorithm is then designed to solve this problem efficiently.

HTS has a number of advantages over existing speciation methods. On the one hand, being topology-based, it makes decisions by referring to the fitness landscape. Hence, it does not suffer from the difficulty of fine tuning the niche radius parameter, which is a common drawback of many distance-based methods. On the other hand, it relies exclusively on search history to capture the landscape topography. Therefore, it does not have the FE overhead associated with existing topology-based methods. Last but not least, HTS is parameter-free and thus readily available in practice. Empirical study has demonstrated that HTS clearly outperforms existing topology-based methods when the FE budget is limited.

The computational cost of HTS is generally acceptable for modern personal computers. In particular, in the experiments, the time overhead was practically negligible if a single FE took seconds. Therefore, HTS is particularly suitable for, albeit not restricted to, expensive optimization problems.

As to future paper, four issues are worth further consideration. 1) By comparing HTS with the fictional RM*, we see that the approximation accuracy of the current method seems to get worse in higher dimensions. How to improve the approximation accuracy, especially in high dimensional spaces, is of major concern. 2) The time complexity of HTS may be reduced to make it more useful and competitive on relatively cheap problems. Currently, the search history is being stored in a trivial sequential list. It is possible that more efficient implementations be realized by utilizing advanced data structures. 3) In evolutionary multimodal optimization, the population size should suit the number of optima, which is usually not known *a priori* in practice. In speciation, the number of identified species (seeds) can be used as an indicator of the number of optima [28]. Hence, HTS may be used as a method to estimate the number of optima during optimization and lead to new schemes for population size adaptation. 4) The current method is shift-invariant, but not rotation-invariant. In particular, the definition of an R region is relative to the coordinate axes. Designing a rotation-invariant method is thus of interest.

REFERENCES

[1] C. Hocaoglu and A. Sanderson, "Planning multiple paths with evolutionary speciation," *IEEE Trans. Evol. Comput.*, vol. 5, no. 3, pp. 169–191, Jun. 2001.

[2] J. Yao, N. Kharma, and P. Grogono, "A multi-population genetic algorithm for robust and fast ellipse detection," *Pattern Anal. Applicat.*, vol. 8, no. 1, pp. 149–162, Sep. 2005.

[3] K. Wong, C. Wu, R. Mok, C. Peng, and Z. Zhang, "Evolutionary multimodal optimization using the principle of locality," *Inf. Sci.*, vol. 194, pp. 138–170, Jul. 2012.

[4] K. De Jong, "Evolutionary computation: A unified approach," in *Proc. GECCO*, 2012, pp. 737–750.

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

[6] G. Dick and P. Whigham, "The behaviour of genetic drift in a spatially-structured evolutionary algorithm," in *Proc. CEC*, 2005, pp. 1855–1860.

[7] M. Preuss, L. Schönemann, and M. Emmerich, "Counteracting genetic drift and disruptive recombination in (μ+/,λ)-EA on multimodal fitness landscapes," in *Proc. GECCO*, 2005, pp. 865–872.

[8] J. Masel, "Genetic drift," *Current Biol.*, vol. 21, no. 20, pp. R837–R838, 2011.

[9] S. Mahfoud, "Niching methods for genetic algorithms," Ph.D. dissertation, Dept. General Eng., Univ. Illinois Urbana-Champaign, Champaign, IL, USA, Rep. 95001, 1995.

[10] E. Mayr, *Systematics and the Origin of Species, from the Viewpoint of a Zoologist*. Cambridge, MA, USA: Harvard Univ. Press, 1999.

[11] D. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proc. Int. Conf. Genetic Algorithms Their Applicat.*, 1987, pp. 41–49.

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

[13] S. Mahfoud, "Crowding and preselection revisited," in *Proc. PPSN*, 1992, pp. 27–36.

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

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

[16] S. Galán and O. Mengshoel, "Generalized crowding for genetic algorithms," in *Proc. GECCO*, 2010, pp. 775–782.

[17] J. Gan and K. Warwick, "Dynamic niche clustering: A fuzzy variable radius niching technique for multimodal optimization in GAs," in *Proc. CEC*, 2001, pp. 215–222.

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

[19] J. Li and A. Wood, "Random search with species conservation for multimodal functions," in *Proc. CEC*, 2009, pp. 3164–3171.

[20] M. Balizs, J. Li, G. Parks, and P. Clarkson, "The effect of distance measure in a GA with species conservation," in *Proc. CEC*, 2001, pp. 67–74.

[21] J. Li, X. Li, and A. Wood, "Species based evolutionary algorithms for multimodal optimization: A brief review," in *Proc. CEC*, 2010, pp. 1–8.

[22] R. Ursem, "Multinational evolutionary algorithms," in *Proc. CEC*, 1999, pp. 1633–1640.

[23] X. Li, "Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization," in *Proc. GECCO*, 2004, pp. 105–116.

[24] X. Li, "Efficient differential evolution using speciation for multimodal function optimization," in *Proc. GECCO*, 2005, pp. 873–880.

[25] X. Yin and N. Germay, "A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization," in *Artificial Neural Nets and Genetic Algorithms*. Berlin, Germany: Springer, 1993, pp. 450–457.

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

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

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

[29] D. Parrott and X. Li, "Locating and tracking multiple dynamic optima by a particle swarm model using speciation," *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 440–458, Aug. 2006.

[30] S. Bird and X. Li, "Enhancing the robustness of a speciation-based PSO," in *Proc. CEC*, 2006, pp. 843–850.

[31] R. Ursem, "Multinational GAs: Multimodal optimization techniques in dynamic environments," in *Proc. GECCO*, 2000, pp. 19–26.

[32] J. Yao, N. Kharma, and Y. Zhu, "On clustering in evolutionary computation," in *Proc. CEC*, 2006, pp. 1752–1759.

[33] A. Della Cioppa, C. De Stefano, and A. Marcelli, "On the role of population size and niche radius in fitness sharing," *IEEE Trans. Evol. Comput.*, vol. 8, no. 6, pp. 580–592, Dec. 2004.

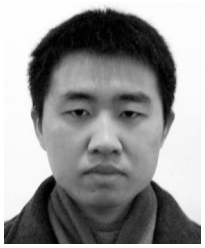
[34] X. Li, "Niching without niching parameters: Particle swarm optimization using a ring topology," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 150–169, Feb. 2010.

[35] K. Deb and D. Goldberg, "An investigation of niche and species formation in genetic function optimization," in *Proc. Int. Conf. Genetic Algorithms*, 1989, pp. 42–50.

[36] O. Shir and T. Bäck, "Niche radius adaptation in the CMA-ES niching algorithm," in *Proc. PPSN*, 2006, pp. 142–151.

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

[38] C. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu, "Disburdening the species conservation evolutionary algorithm of arguing with radii," in *Proc. GECCO*, 2007, pp. 1420–1427.
[39] Y. Jin, "Surrogate-assisted evolutionary computation: Recent advances and future challenges," *Swarm Evol. Comput.*, vol. 1, no. 2, pp. 61–70, 2011.
[40] B. Miller and M. Shaw, "Genetic algorithms with dynamic niche sharing for multimodal function optimization," in *Proc. Int. Conf. Evol. Comput.*, 1996, pp. 786–791.
[41] Q. Ling, G. Wu, Z. Yang, and Q. Wang, "Crowding clustering genetic algorithm for multimodal function optimization," *Appl. Soft Comput.*, vol. 8, no. 1, pp. 88–95, Jan. 2008.
[42] J. Yao, N. Kharma, and P. Grogono, "BMPGA: A bi-objective multi-population genetic algorithm for multi-modal function optimization," in *Proc. CEC*, 2005, pp. 816–823.
[43] J. Branke, "Creating robust solutions by means of evolutionary algorithms," in *Proc. PPSN*, 1998, pp. 119–128.
[44] S. Yuen and C. Chow, "A genetic algorithm that adaptively mutates and never revisits," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 454–472, Apr. 2009.
[45] C. Chow and S. Yuen, "An evolutionary algorithm that makes decision based on the entire previous search history," *IEEE Trans. Evol. Comput.*, vol. 15, no. 6, pp. 741–769, Dec. 2011.
[46] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evol. Comput.*, vol. 9, no. 2, pp. 159–195, Jun. 2001.
[47] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. Cambridge, MA, USA/New York, NY, USA: MIT Press and McGraw-Hill, 2001, pp. 73–90.
[48] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," in *Proc. CEC*, 2004, pp. 1382–1389.
[49] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.
[50] J. Kruisselbrink, "Evolution strategies for robust optimization," Ph.D. dissertation, Leiden Institute of Advanced Computer Science (LIACS), Faculty of Science, Leiden Univ., EZ Leiden, Netherlands, 2012, pp. 51–53.
[51] R. Salomon, "Re-evaluating genetic algorithm performance under coordinate rotation of benchmark functions. A survey of some theoretical and practical aspects of genetic algorithms," *Bio Syst.*, vol. 39, no. 3, pp. 263–278, 1996.
[52] P. Diaconis and M. Shahshahani, "The subgroup algorithm for generating uniform random variables," *Probability Eng. Inform. Sci.*, vol. 1, no. 1, pp. 15–32, 1987.
[53] Y. Ong, P. Nair, and K. Lum, "Max-min surrogated-assisted evolutionary algorithm for robust design," *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 392–404, Aug. 2006.
[54] B. Qu, P. Suganthan, and J. Liang, "Differential evolution with neighborhood mutation for multimodal optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 5, pp. 601–614, Oct. 2012.
[55] M. Hollander and D. Wolfe, *Nonparametric Statistical Methods*, 2nd ed. New York, NY, USA: Wiley, 1999.



Lingxi Li received the bachelor's degree in software engineering from Southwest University of Science and Technology, Mianyang, Sichuan, China, in 2011 and is currently working toward the master's degree in computer science with the University of Science and Technology of China–Birmingham Joint Research Institute in Intelligent Computation and its Applications, Hefei, Anhui, China.

His research concerns evolutionary computation.

Ke Tang (SM'14) received the B.Eng. degree from Huazhong University of Science and Technology, Wuhan, China, in 2002 and the Ph.D. degree from Nanyang Technological University, Singapore, in 2007.

Since 2011, he has been a Professor with the University of Science and Technology of China (USTC)–Birmingham Joint Research Institute in Intelligent Computation and its Applications, School of Computer Science and Technology, USTC. He has authored or co-authored over 80 refereed publications. His research interests include evolutionary computation, machine learning, data mining, large scale global optimization, dynamic and robust optimization, and real-world applications.

Dr. Tang is an Associate Editor of *IEEE Computational Intelligence Magazine, Computational Optimization* and *Applications Journal* and *Frontiers of Computer Science Journal*.