Applied Soft Computing 11 (2011) 2017–2034

Contents lists available at ScienceDirect

Applied Soft Computing

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

Genetic Algorithm with adaptive elitist-population strategies for multimodal function optimization

Yong Liang[a,*], Kwong-Sak Leung[b]

[a] Macau University of Science and Technology, Macau, China
[b] The Chinese University of Hong Kong, Hong Kong, China

ARTICLE INFO

Article history:
Received 10 July 2009
Received in revised form 2 March 2010
Accepted 19 June 2010
Available online 27 July 2010

Keywords:
Genetic algorithms
Multimodal optimization
Elitist strategy

ABSTRACT

This paper introduces a new technique called adaptive elitist-population search method. This technique allows unimodal function optimization methods to be extended to efficiently explore multiple optima of multimodal problems. It is based on the concept of adaptively adjusting the population size according to the individuals' dissimilarity and a novel direction dependent elitist genetic operators. Incorporation of the new multimodal technique in any known evolutionary algorithm leads to a multimodal version of the algorithm. As a case study, we have integrated the new technique into Genetic Algorithms (GAs), yielding an Adaptive Elitist-population based Genetic Algorithm (AEGA). AEGA has been shown to be very efficient and effective in finding multiple solutions of complicated benchmark and real-world multimodal optimization problems. We demonstrate this by applying it to a set of test problems, including rough and stepwise multimodal functions. Empirical results are also compared with other multimodal evolutionary algorithms from the literature, showing that AEGA generally outperforms existing approaches.

1. Introduction

Genetic algorithms (GAs) have proven useful in solving a variety of search and optimization problems [2,8–10,22,24,39]. Many real-world problems require an optimization algorithm that is able to explore multiple optima in their search space. In this respect, GAs have demonstrated the best potential for finding the optimal solutions because they are population-based search approaches and have strong global optimization capabilities. However, in the standard GA for maximization problems, all individuals, which may be located on different peaks at the beginning of the search process, eventually converge to a single peak. Thus, it usually ends up with only one solution. If this solution is a local optimum, we call it premature convergence in GAs. This phenomenon is even more serious in GAs with elitist strategy, which is a widely adopted method to improve GAs' convergence [16].

Over the years, various population diversity enhancement mechanisms have been proposed, which enable GAs to maintain a diverse population of individuals throughout their search, to avoid convergence of the population to a single peak and to allow GAs to identify multiple optima in a multimodal function landscape. However, various current population diversity enhancement mechanisms have not demonstrated themselves to be very efficient as expected. The efficiency problems, in essence, are related to some fundamental dilemmas in GAs implementation. We believe any attempt to improve the efficiency of GAs has to compromise between these two dilemmas:

- *The elitist search versus diversity maintenance dilemma*:
  GAs are expected to be global optimizers with global search capability to encourage exploration of the global optimal solutions. So the elitist strategy is widely adopted in the GAs' search processes to improve the chance of finding the global optimal solution. Unfortunately, the elitist strategy concentrates on some "super" individuals, but reduces the diversity of the population, and in turn leads to premature convergence. Contrarily, GAs need to maintain the diversity of the population in their search processes to find the multiple optimal solutions. How to balance both the elitist search and the diversity maintenance is important for constructing an efficient multimodal GA. Some researchers have attempted to handle the dilemma, e.g., Mahfoud's Deterministic Crowding methods [34], Petrowski's Clearing Procedure [41] and Li's Species Conserving Genetic Algorithm (SCGA) [32].
- *The algorithm effectiveness versus population redundancy dilemma*:
  For many GAs, we can use a large population size to improve the chance to obtain the global and multiple optima for optimization problems. However, the large population size will notably increase the computational complexity of the algorithms and

* Corresponding author.
E-mail addresses: yliang@must.edu.mo (Y. Liang), ksleung@cse.cuhk.edu.hk (K.-S. Leung).

1568-4946/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2010.06.017

generate a lot of redundant individuals in the population, thereby decrease the efficiency of GAs.

Our idea in this study is to strike a tactical balance between the two contradictory dilemmas. We propose a new adaptive elitist-population search technique to identify and search for multiple optima efficiently in the multimodal function landscape. The technique is based on an elitist population with a dynamically adapting size and the adoption of a series of new GA mechanisms: a specific definition of an elitist individual for the multimodal function landscape, a new principle for the individual's dissimilarity, and a new set of direction dependent elitist genetic operators. Combining this technique with GA, we propose a novel multimodal GA—adaptive elitist-population based genetic algorithm (AEGA). AEGA was first proposed by the authors [31]. This paper describes an improved version of AEGA. Using multiple test functions, we demonstrate empirically that our proposed approach generally outperforms the existing multimodal evolutionary algorithms reported in the literature.

To illustrate our technique, we will use unconstrained optimization problems of real-valued functions, defined over an array of real numbers. Where no confusion could occur we denote the objective function by $f(x)$. AEGA in this paper makes no distinction between genotypes and phenotypes. Thus, genetic operators will be applied directly to individuals represented by arrays of real numbers. Note that none of the above restrictions are required for our technique to be applicable. The only reason for imposing them is for simplicity of presentation.

The remainder of this paper is organized as follows. The next section describes related work relevant to our proposed technique. Section 3 introduces the adaptive elitist-population search technique and describes the implementation of the algorithm. Section 4 presents the results from a series of experiments on a set of test functions, and the comparison of our results with other multimodal evolutionary algorithms. Sections 5 and 6 present the analyses of the parameter choice in AEGA. Section 7 presents the conclusion and some future directions of research.

2. Related work

When applying GAs to multimodal optimization problems, it is very important to maintain two apparently contradictory requirements, which are to preserve promising individuals from one generation to the next and maintain the diversity of the population [32]. This section briefly reviews the existing methods developed to address the related issues: elitism, niche formation methods and other parallel subpopulations search methods.

2.1. Elitism

Many elitist methods in the literature on GAs preserve the best solution in different ways. For instance, Whitley [49] proposed a GENITOR approach that generates just one child for each cycle, which then replaces the worst individual of the population. Eshelman [18] introduced the CHC Adaptive Search algorithm to select the best M (the population size) individuals from the population, which merges all parents and offspring together. For multimodal optimization, Mahfoud's Deterministic Crowding methods [34] only replace a parent if the competing offspring is better. Petrowski's Clearing Procedure [41] preserves the fitness of the dominant individual, while it resets the fitness of all the other individuals of the same subpopulation to zero. Li's SCGA [32] copies the dominating individual of each of the species into the next generation as the species' seeds.

The elitism strategies have been also widely used in implementations of other evolutionary algorithms (EAs). For instance, Costa and Oliveira [12] proposed evolution strategy (ES) with elitist method for multiobjective optimization. Cortés et al. [11] proposed a novel viral systems (VS) with elitist strategy to deal with combinatorial problems. Zhang et al. [51] proposed an efficient population-based incremental learning (PBIL) algorithm with elitist strategy. PBIL is one of the simplest estimation of distribution algorithms (EDAs). However, it is pointed out that "elitist strategies tend to make the search more exploitative rather than explorative and may not work for problems in which one is required to find multiple optimal solutions" [43].

2.2. Evolving parallel subpopulations by niching

Niching methods extend GAs to problems that require to locate and maintain multiple optima through parallel subpopulations' search.

Cavicchio [6] proposed a preselection scheme in which a child replaces the worse parent if the child's fitness is higher than that of the worse parent. De Jong [15] generalized the preselection technique and suggested a crowding scheme. The approach works by reproducing and killing off a fixed percentage of the population each generation. Each newly generated member must replace an existing one, preferably the most similar one. Subsequently, two further variants of crowding, Deterministic Crowding [34] and Probabilistic Crowding [35], were proposed. Both of them use the tournament selection and hold tournaments between similar children and parents. The main difference between them is that the former uses a deterministic acceptance rule, while the latter uses probabilistic tournaments. Cadeño and Vemuri [7] proposed the Multi-Niche Crowding GA (MNC GA) for dynamic landscapes. The algorithm introduces the concept of crowding selection to promote mating among members with similar traits while allowing many members of the population to participate in mating. Then the MNC GA uses the worst among most similar replacement (WAMS) policy to promote competition among members with similar traits while allowing competition among members of different niches.

In another way, fitness sharing is frequently employed to induce niching behavior in GAs. Goldberg and Richardson [23] proposed a sharing scheme in which the idea is to force the individuals of the populations to share available resources by dividing the populations into different subpopulations on the basis of the similarity of the chromosomes. To implement the sharing scheme, a simple linear function called sharing function $Sh(d_{i,j})$ is adopted as a function of $d_{i,j}$, which is the distance between two individuals x_i and x_j. The sharing function is evaluated for each pair of N individuals in the population, and then the sum $Sh_i = \sum_{j=1}^{N} Sh(d_{i,j})$ is computed for each individual x_i. Finally, the fitness of this individual is adjusted by dividing by Sh_i. This sharing scheme was shown [23] to be better able to preserve diversity than the crowding scheme and was successfully applied to solve a variety of multimodal functions. To make the notion of species clearer, Yin and Germay [50] introduced a Clustering methodology to the sharing scheme in which each population is divided into clusters directly, but it does not increase any more diversity than the classical sharing scheme does.

The main drawbacks to use sharing are: (i) setting dissimilarity threshold σ_{share} requires a priori knowledge of how far apart the optima are [13,34]; and (ii) the computational complexity of niche counts is $O(N^2)$ per generation [37,50]. To reduce the computational complexity of sharing scheme, Beasley at al. [3] proposed the Sequential Fitness Sharing. It works by iterating a traditional GA, and maintaining the best solutions of each run off-line. To avoid converging to the same area of the search space multiple times, whenever sequential fitness sharing locates a solution, it depresses the fitness landscape at all points within some dissimilarity threshold σ_{share} of that solution.

Some research works aimed at speeding up sharing may have inadvertently attacked the problem of nonuniform niches. Miller and Shaw [37] proposed the Dynamic Niche Sharing methods. In the study, they defined a fixed number of dynamic niches with fixed radii and niche centers determined by a full population sort. For those individuals not in a niche, regular fixed sharing is used. The authors claimed certain advantages over other nichers based on limited testing, but the primary weakness of the scheme is the use of fixed sharing outside the dynamic niches. Goldberg and Wang [27] proposed another adaptive niching scheme called Coevolutionary Shared Niching (CSN), the scheme takes its inspiration from the model of *monopolistic competition* in economics and utilizes two populations, a population of businessmen and a population of customers, where the locations of the businessmen correspond to niche locations and the locations of the customers correspond to solutions. The scheme overcomes the limitations of fixed sharing schemes by permitting the distribution of solutions and the radii of the niches to adapt to complex function landscapes having many multiple (local or global) optima that are close to each other.

Spears [44] proposed the simple subpopulation schemes using tag bits to identify different species—individuals with the same tag bit value belong to the same species. The tag bits are used to restrict mating and to perform fitness sharing, and can be changed, i.e., the species type of a solution can be changed through mutation. Li et al. [32] developed a new technique called Species Conservation for evolving parallel subpopulations. The technique is based on the concept of dividing the population into several species according to their similarity. Each of these species is built around a dominating individual called the species seed. Species seeds found in the current generation are saved (conserved) by moving them into the next generation. Incorporating the idea of species into Particle Swam Optimization (PSO), Parrott et al. [40] proposed a species-based PSO to solve multimodal optimization problems and track multiple optima in a dynamic environment.

Some researchers attempted to design more competent subpopulations approaches without sharing. Elo [17] proposed a parallel GA for multimodal optimization. The general idea of the parallel GA is to dynamically divide the population into an increasing number of subpopulations to allow specialization on the different maxima discovered during the search process. Harik [28] developed a form of Restricted Tournament Selection for solving of multimodal problems using GAs. It is based on the concept of direct local competition. The parents are crossed-over and mutated to form their offspring, then the offspring competes with their close (similar) individuals in the population, but not their parents. Tsutsui and Fujimoto [47] proposed a forking GA, which uses one parent population that explores one subspace; and one or more child populations exploiting the other forking subspace. Ursem [48] developed a multinational evolutionary algorithm, which tries to use the topology to group the individuals into sub-populations each covering a part of the fitness landscape. Gan and Warwick [21] proposed a variable radius niching technique called dynamic niche clustering (DNC). DNC employs a separate population of overlapping fuzzy niches with independent radii which operate in the decoded parameter space and are maintained alongside the GA population.

Other researchers used an Artificial Immune System (AIS) to solve the multimodal optimization problems. Castro and Zuben [4] proposed a Clonal Selection principle to solve the multimodal optimization problem. This strategy suggests that the algorithm performs a greedy search, where single members will be optimized locally (exploitation of the surrounding space) and the newcomers yield a broader exploration of the search space. The population of the clonal selection includes two parts. The first part is the clonal part. Each individual will generate some clonal points and select the best one to replace its parent. The second part is the newcomer part, the function of which is to find new peaks. The clonal selection algorithm also incurs expensive computational complexity to get better results. Castro and Timmis [5] proposed an Artificial Immune Network that presents the adaptation of an immune network model to perform information compression and data clustering. The main feature of the algorithm is the dynamic search for an optimal population size based upon the network suppression threshold and a well-defined stopping criterion.

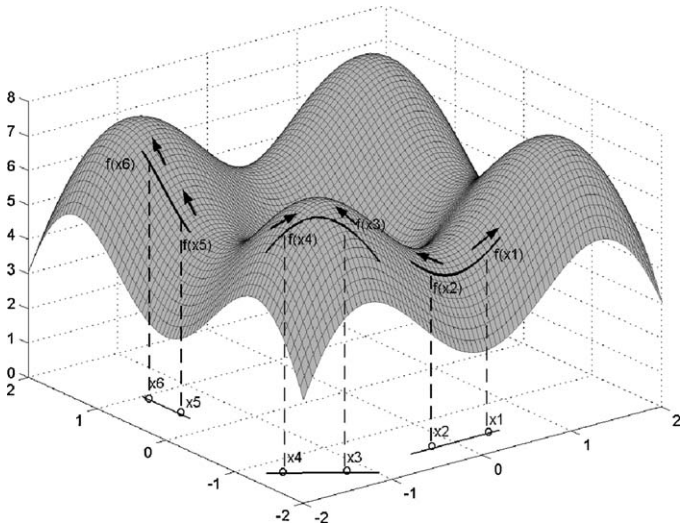

Fig. 1. The relative ascending directions of each pair of points (x_1 and x_2, x_3 and x_4, x_5 and x_6).

Kumar and Rockett [29,30] solved multimodal optimizations reformulated into multiobjective problems using a multiobjective GA, which preserves diversity without niching. Their algorithm without any explicit diversity-preserving operator is found to produce diverse sampling of the Pareto-front with lower computational effort.

Many multimodal techniques found in the literature try to give global and good local optimal solutions an equal or similar opportunity to survive. Sometimes, however, survival of relative low fitness but very different individuals may be as, if not more, important than that of some highly fit ones. The purpose of this paper is to present a new technique that addresses this problem. We show that using our proposed technique, the individuals in GAs will converge to multiple optimal solutions of multimodal optimization problems.

3. A new adaptive elitist-population search technique

The new technique for multimodal function maximization presented in this paper achieves adaptive elitist-population searching by exploring the notion of the relative ascending directions of two individuals (and for a minimization problem it is called the relative descending directions).

For a high dimensional maximization problem, every individual generally can have many possible ascending directions. But along the line, which is uniquely defined by each pair of individuals (e.g., x_1 and x_2, x_3 and x_4, and x_5 and x_6 in Fig. 1 in a two dimensional case), generally each individual only has one relative ascending direction with respect to the other one. Now we introduce how to determine this relative ascending direction. For simplicity, we consider the multimodal optimization problem with n dimensional domains specified by

$$\Omega = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_n, b_n], \quad (1)$$

where a_i, $b_i \in R$ and $a_i < b_i$, $(i = 1, 2, \cdots, n)$. Let x_i and x_j be the points in Ω, we then generate two reference points x_i' and x_j' by

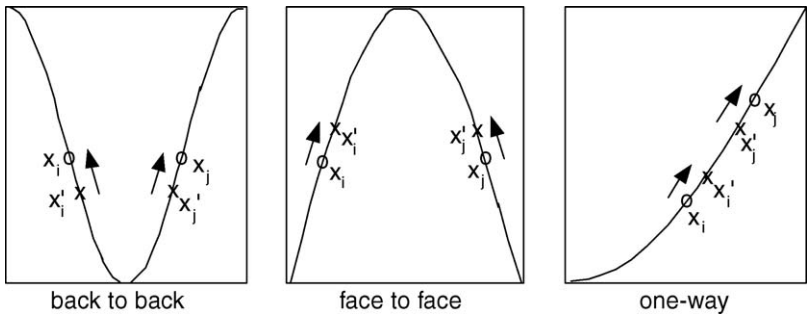

Fig. 2. The relative ascending directions of both individuals being considered: back to back, face to face and one-way.

$x'_i = x_i + \delta(x_j - x_i)$ and $x'_j = x_j - \delta(x_j - x_i)$, where δ is an absolute small positive value.

We compare the function values of x_i and x'_i to determine the relative ascending direction of x_i with respect to x_j. If $f(x'_i) - f(x_i) > 0$, we define the relative ascending direction of x_i to be moving towards x_j; if $f(x'_i) - f(x_i) = 0$, we define the relative ascending direction of x_i to be flat; otherwise, if $f(x'_i) - f(x_i) < 0$, the ascending direction of x_i is moving away from x_j. Similarly, we can determine the relative ascending direction of x_j with respect to x_i through comparing the function values of x_j and x'_j. To estimate the relative ascending directions for any two individuals x_i and x_j on the landscape of the high dimensional maximization problem, we add two reference points x'_i and x'_j on the line, which is uniquely defined by joining the two points x_i and x_j, and compare the fitness of x_i, x'_i, and x_j, x'_j respectively. Thereby, to determine the relative ascending direction of any two points x_i and x_j, only 4 function evaluations are needed. However, the direction is very important information to identify whether x_i and x_j are located on the same peak or not.

The points located on different peaks are considered as dissimilar individuals in the population of GA for multimodal maximization problems. The relative ascending directions of two points mainly have four possibilities (cases): back to back, face to face, one-way and flat. Fig. 2 shows the first three cases while Fig. 3 illustrates the three possible combinations for the flat (4th) case. We can measure the dissimilarity of the individuals according to the combination of their relative ascending directions and their distance. For the flat case, when the part of the objective function landscape under consideration is flat partially or completely, we will only use the individuals' distance to identify their dissimilarity, since the relative ascending direction of two points cannot be uniquely defined.

The distance between two individuals $x_i = (x_{i1}, x_{i2}, \cdots, x_{in})$ and $x_j = (x_{j1}, x_{j2}, \cdots, x_{jn})$ is defined by:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^{n} (x_{ik} - x_{jk})^2} \tag{2}$$

Note that this is not the only way in which the distance, and hence the dissimilarity, between two individuals represented by vectors of real numbers could be defined. In this paper, we use the above definition of distance, but the method we describe will work for other distance definitions as well. On the other hand, in many applications, the variable values have different orders of magnitude. We can normalize each variable into the domain [0,1] to calculate the distance between two individuals.

3.1. The principle of the individuals' dissimilarity

Our proposed principle of the individuals' dissimilarity, as well as the operation of AEGA, depends on the relative ascending directions and the distance between two individuals. We define the distance threshold as σ_s. The principle to measure the individuals' dissimilarity is described as follows:

- If the relative ascending directions of both individuals are back to back, or the distance between them equal to or larger than the distance threshold σ_s, these two individuals are considered to be dissimilar and located on different peaks;
- If the relative ascending directions of both individuals are face to face, one-way or flat, and the distance between them is smaller than σ_s, these two individuals are considered to be similar and located on the same peak.

As we know, in many niching approaches, the distance between two individuals is the only measurement to determine whether these two individuals are located on the same peak, but this is often not accurate. Suppose, for example, that our problem is to maximize the function shown in Fig. 4. O_1 and O_2 are two maxima and assume that, in a particular generation, the population of GA consists of the points shown. The individuals a and b are located on the same peak, and the individual c is on another peak. According to the distance between two individuals only, the individuals b and c will be put into the same subpopulation, which is denoted by "×", and the individual a into another subpopulation, which is denoted by "○" (as shown in Fig. 4(a)). Since the fitness of c is smaller than that

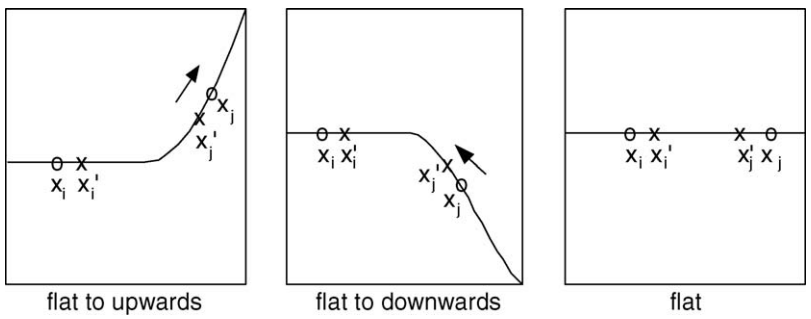

Fig. 3. The three possible combinations of the relative ascending directions of both individuals for the flat cases.

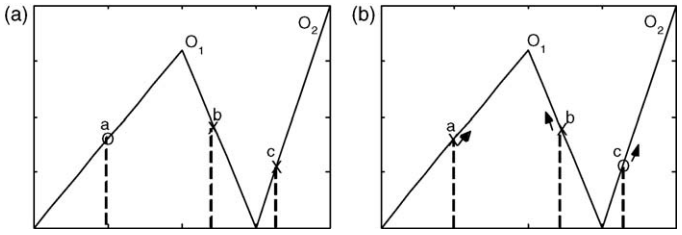

Fig. 4. Determining subpopulations by other niching methods (a) and the relative ascending directions of the individuals (b).

of *b*, the probability of *c* surviving to the next generation is low. This is true even for GAs using fitness sharing, unless a sharing function is specifically designed for this problem. However, the individual *c* is very important to the search, if the global optimum O_2 is to be found. Applying our proposed principle of the individuals' dissimilarity, the relative ascending directions of both individuals *b* and *c* are back to back, they will be considered to be located on different peaks. The individuals *b* and *c* are denoted by "×" and "○" respectively (as shown in Fig. 4(b)). Identifying and preserving the "good quality" of individual *c* is advantageous to maintain the diversity of the population in the GAs' search process.

The combination of the individuals' relative ascending directions and their distances is more powerful than only using the later alone to identify the individuals' dissimilarity. Unfortunately, it is still not always correct because the objective function landscapes can be very tricky sometimes. For instance, in some high dimensional case, if two individuals are located on two different ridges that lead to the same peak, our proposed principle may identify that they are located on two different peaks because their relative ascending directions are back to back. On the other hand, if the relative ascending directions of two individuals on different peaks are face to face or one-way but their distance is smaller than σ_s, our proposed principle will also wrongly identify that they are located on the same peak. Therefore, we propose a novel and tolerant adaptive elitist-population for our AEGA to reduce the effects of all the possible incorrect identifications through adjusting the population size dynamically and conserving the possibly useful individuals. The details are given below.

3.2. Adaptive elitist-population for multimodal maximization problem

As we know, introducing elitism into GA will often lead to one highly fit individual gradually replacing all competing rivals. Thus, in addition to introducing the possibility of premature convergence, elitism also prevents GA from finding multiple optima of multimodal optimization problems. The question is: "How can the idea of elitism be transferred to GA using some form of genetic operations in such a way that the possibly useful individuals are conserved to the next generation?" [32].

The answer to this question is not trivial. Some researchers have attempted to tackle this problem, e.g., Mahfoud's Deterministic Crowding methods [34], Petrowski's Clearing Procedure [41] and Li's SCGA [32]. However, these approaches use the best individuals in the population as the elitists but do not appropriately define elitist individuals for the multimodal optimization problems. In our work, we first define an elitist individual in the population so that maximum diversity and a minimum population size can be maintained for the multimodal optimization problems.

Definition 1. For a multimodal optimization problem, an elitist individual is a point with the best fitness on its located peak in the multimodal function landscape.

Input: p_i and $p_j \longleftarrow$ randomly selected parents from the current population $Pop(t)$ at generation t;
begin
μ_1 and $\mu_2 \longleftarrow$ random number over [-0.5,0.5];
generate two offspring c_i and c_j:
$c_i \longleftarrow p_i + \mu_1 \times (p_j - p_i)$;
$c_j \longleftarrow p_j + \mu_2 \times (p_i - p_j)$;
if $max\ \{d(x_i, x_j)|x_i, x_j \in \{p_i, p_j, c_i, c_j\}\} < \sigma_s$
if all the relative directions of p_i, p_j, c_i and c_j are face to face, one-way or flat; **then**
if $f(p_i) = f(p_j) = f(c_i) = f(c_j)$ **then**
$p_i \leftarrow$ random select from $\{p_i, p_j, c_i, c_j\}$;
else
$p_i \leftarrow$ best$\{p_i, p_j, c_i, c_j\}$;
end if
remove p_j from $Pop(t)$;
else if $max\ \{d(x_i, x_j)|x_i, x_j \in \{p_i, p_j, c_i, c_j\}\} \geq \sigma_s$
or the relative directions of p_i, p_j are back to back;
if $f(c_i) > f(p_i)$ **then** $p_i \leftarrow c_i$;
if $f(c_j) > f(p_j)$ **then** $p_j \leftarrow c_j$;
if $f(p_i) = f(p_j) = f(c_i) = f(c_j)$
then $p_i, p_j \leftarrow$ two individuals with the largest distance among $\{p_i, p_j, c_i, c_j\}$;
end if
end

Fig. 5. Pseudocode for the elitist crossover operator.

As shown in Fig. 4(a) and (b), *b* and *c* are the elitist individuals for the multiple optima O_1 and O_2 respectively. Based on Definition 1, the population can maintain multiple elitist individuals, which are located on different peaks. The goal of the adaptive elitist-population search method is to adaptively adjust the population size according to the features of the multimodal problem in order to achieve:

- a dynamic number of individuals in the population so that only a minimum number of elitist individuals is used to search for each peak; and
- all the individuals in the population search for and occupy different peaks in parallel.

Accordingly, we design the direction dependent elitist genetic operators, which can gradually produce and evolve the elitist individuals on their own peaks in parallel, to explore multiple optima of the multimodal optimization problems. In the following four sections, we present the two elitist genetic operators, the population control constraints and the algorithm for the elitist search respectively.

3.3. Elitist crossover operator

First, we construct an elitist crossover operator to search for the elitist individuals corresponding to the multiple optima. The procedure of the elitist crossover operator is shown in Fig. 5. Let two individuals p_i and p_j be selected from the population at random. c_i and c_j are their offspring generated through any version of a crossover operator. Any conventional crossover operator can be chosen to be the first part of our proposed elitist crossover operator. Please note that the conventional crossover operator would not by itself contribute significantly to the maintenance of diversity of the population. Here we have chosen the random uniformly distributed variable to perform crossover (with probability P_c), which

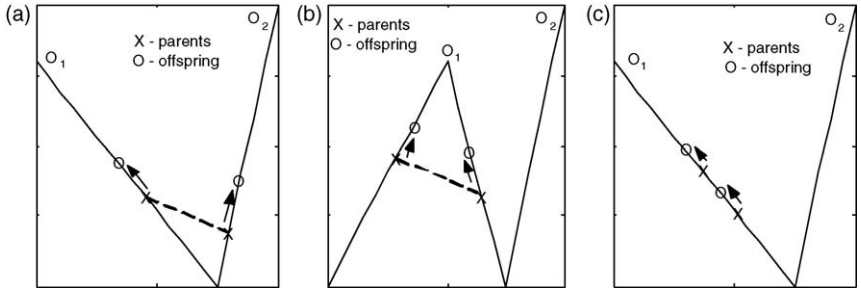

Fig. 6. A schematic illustration of the elitist crossover operation.

is similar to the BLX-0.5 crossover operator [19]. The offspring c_i and c_j of randomly chosen parents p_i and p_j are:

$$c_i = p_i + \mu_1 \times (p_j - p_i)$$
$$c_j = p_j + \mu_2 \times (p_i - p_j) \quad (3)$$

where μ_1, μ_2 are uniformly distributed random numbers over $[-0.5, 0.5]$. Then the elitist crossover operator applies our principle of the individuals' dissimilarity on the parents p_i and p_j. If p_i and p_j satisfy back to back (Fig. 6(a)) or $\max\{d(x_i, x_j) \mid x_i, x_j \in \{p_i, p_j, c_i, c_j\}\} \geq \sigma_s$, then the elitist crossover operator will select the better from each of the pairs (p_i, c_i) and (p_j, c_j) to the next generation. Thus if the parents p_i and p_j are located on different peaks, the two selected individuals may still remain on different peaks. If $\max\{d(x_i, x_j) \mid x_i, x_j \in \{p_i, p_j, c_i, c_j\}\} \geq \sigma_s$ and $f(p_i) = f(p_j) = f(c_i) = f(c_j)$, then two points with the largest distance among $\{p_i, p_j, c_i, c_j\}$ are conserved to the next generation.

When the parents p_i and p_j are on the same peak and satisfy: (*i*) face to face and $max\{d(x_i, x_j) \mid x_i, x_j \in \{p_i, p_j, c_i, c_j\}\} < \sigma_s$ (Fig. 6(b)), or (ii) one-way and $\max\{d(x_i, x_j) \mid x_i, x_j \in \{p_i, p_j, c_i, c_j\}\} < \sigma_s$ (Fig. 6(c)), the elitist crossover operator will identify the relative ascending directions and the distances of each pair of the parents p_i, p_j and their offspring c_i, c_j. If all the relative ascending directions are face to face, one-way or flat and all the distances are smaller than σ_s, we consider all the four points p_i, p_j, c_i and c_j are located on the same peak with a high probability. For such a case, the elitist crossover operator will only select the best one of them to the next generation to reduce the redundancy of the population. As shown in Fig. 5, the elitist crossover operator selects the best one of p_i, p_j, c_i, c_j to replace p_i ($p_i \leftarrow$ best $\{p_i, p_j, c_i, c_j\}$) and deletes p_j from the current population $Pop(t)$. If $\max\{d(x_i, x_j) \mid x_i, x_j \in \{p_i, p_j, c_i, c_j\}\} < \sigma_s$ and $f(p_i) = f(p_j) = f(c_i) = f(c_j)$, then randomly select one from $\{p_i, p_j, c_i, c_j\}$ to the next generation.

Why do we need to consider the relative ascending directions of each pair of the parents and their offspring, as well as the relative distances between them? It is because we can only guess whether these individuals are on the same peak based on both the relative ascending directions and the appropriately set pre-determined distance threshold σ_s. For example, as shown in Fig. 7(a) and (b), the relative directions of p_i and p_j are face to face and one-way respectively, but they are located on different peaks. Thus, if the distance threshold σ_s is too large and only the ascending directions of p_i and p_j are used, the elitist crossover operator will identify that p_i and p_j are located on the same peak. However, in Fig. 7(a) and (b), the relative directions of each pair of the parents p_i, p_j and their offspring c_i, c_j do reflect the back to back case with a high probability (c_i, c_j in Fig. 7(a) and c_i, p_j in Fig. 7(b)), and accurately identify that p_i and p_j are located on different peaks. Therefore, considering the relative ascending directions of each pair of the parents and their offspring can significantly reduce the error of the identification of individuals' dissimilarity based only on the distance threshold σ_s.

The aim of conserving the best one to replace all the parents and offspring, is to reduce the computational complexity and population's redundancy of GA in the case that many individuals are close to each other and located on the same peak.

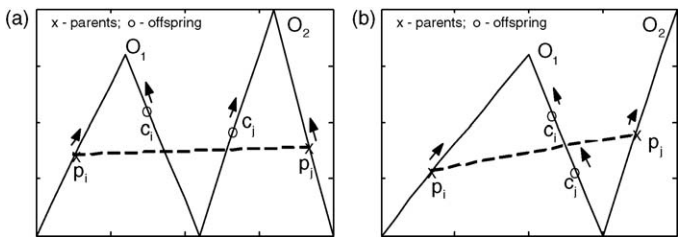

Fig. 7. A schematic illustration of the elitist crossover operation when the parents p_i and p_j are located on different peaks.

3.4. Elitist mutation operator

The main function of the mutation operator is to find unexplored peaks. However, many mutation operators in existing niching methods [4,23,34,41] cannot efficiently satisfy this requirement well. For example, in deterministic crowding [34], even the offspring is located on an unexplored peak, but since its fitness is not better than that of its parent, it is difficult to be retained, and hence this unexplored peak cannot be located by this mutation operation (Fig. 9(a)). Some researchers tried to improve it. For example, in Probabilistic Crowding [35], the individuals with low fitness can be conserved to the next generation probabilistically.

To ensure the diversity of the GA's population, we construct an elitist mutation operator that can more efficiently produce and conserve the elitist individuals that are located on the unexplored peaks. The procedure of the elitist mutation operator is shown in Fig. 8.

Let p_i and c_i be the parent and the offspring involved in the mutation respectively. Here we use Mühlenbein's mutation operator (with probability P_m) [38]:

$$c_i = p_i \pm \lambda \times r_m \quad (4)$$

where λ is a uniformly distributed random number over $[-1, 1]$, r_m defines the mutation range and it is normally set to $0.5 \times (b_k - a_k)$, a_k and b_k are defined by Eq. (1), and the '+' and '−' signs are chosen with a probability of 0.5 each.

According to our proposed principle of individuals' dissimilarity, if the relative ascending directions of two individuals are face to face, one-way or flat, and their distance is smaller than σ_s, we hypothesize that the two individuals are often located on the same peak. Thereby, the elitist mutation operator, like the conventional mutation operator, will select the best one of them to the next generation. If $f(p_i) = f(c_i)$, p_i will be conserved to the next generation.

If the relative ascending directions of p_i and c_i are back to back or $d(p_i, c_i) \geq \sigma_s$, they are considered to be on different peaks. For such a case, p_i is directly conserved to the next generation and c_i

```
Input: p_i ⟵ randomly selected parent from the
       current population Pop(t);

begin
 c_i ⟵ p_i ± λ × γ_m;
 determine the relative directions of both p_i and c_i;
 if (face to face, one-way or flat) and d(c_i, p_i) < σ_s,
then
     if f(c_i) > f(p_i), then p_i ⟵ c_j;
 else if (back to back) or d(c_i, p_i) ≥ σ_s, then
     set "c_i is a new elitist individual"⟵ TRUE;
     for all p_j ∈ Pop(t) do
       if d(c_i, p_j) < σ_s and f(c_i) < f(p_j), then
        "c_i is the new elitist individual" ⟵ FALSE;
         break;
       end if
     end for
 end if
 if "c_i is the new elitist individual" ⟵ TRUE, then
     let Pop(t) ⟵ Pop(t) ∪ {c_i};
 end if
end
```

Fig. 8. Pseudocode for the elitist mutation operator.

is considered as a new individual candidate. Then the elitist mutation operator compares the fitness of c_i with the other individual p_j, which satisfies $d(c_i, p_j) < \sigma_s$. If there are no such individual p_j satisfying $d(c_i, p_j) < \sigma_s$ in the current population, or there exists such p_j, but its fitness is not better than that of c_i, the new individual candidate c_i is confirmed to be on an unexplored or at least a different peak. The elitist mutation operator will conserve such c_i to the next generation. As shown in Fig. 9(a), the offspring will be conserved to the next generation because it is the elitist individual on its own located peak, and in Fig. 9(b), the offspring will be deleted because there is another individual better than it on their located peak. Therefore, the elitist mutation operator conserves both parent and its offspring to the next generation only if they are uniquely occupying a peak each to maintain or even improve the diversity of the GA's population.

3.5. Population control constraints

Sometimes the multimodal optimization problems have a lot of local optima and very rugged landscapes. Hence the goal of multimodal optimizers is to find as many global optima and possibly good local optima as possible. Our proposed elitist genetic operators can add new individuals into the population if they are located on unexplored peaks according to their relative ascending directions and distances. This criterion of occupying unexplored peaks is more important than the relative fitness. When the multimodal functions have very rugged landscapes, our proposed approach will keep many low fitness individuals instead of high fitness ones;

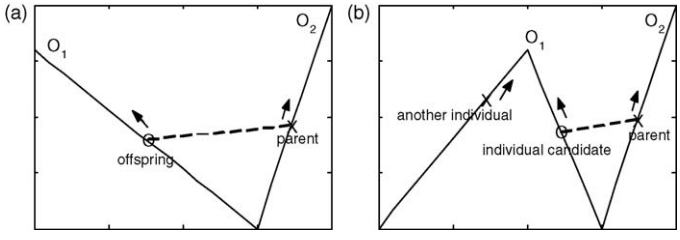

Fig. 9. A schematic illustration of the elitist mutation operation.

```
begin
 t ⟵ 0;
 Initialize Pop(t);
 Evaluate Pop(t);
 while (not termination condition) do
  Elitist crossover operation Pop(t + 1);
  Elitist mutation operation Pop(t + 1);
  Evaluate Pop(t + 1);
  Adjust Pop(t + 1) through the population control
  constraints;
  t ⟵ t + 1;
 end while
end
```

Fig. 10. Pseudocode for AEGA algorithm.

hence its efficiency may be significantly reduced. To avoid the elitist genetic operators wasting computational power on low fitness optima, we design two population control constraints as follows:

i. If a newly generated individual has evolved for α generations and its fitness is still lower than a fitness threshold, $\beta\times$ the best-so-far solution (e.g., 0.5 $\times$ the best-so-far solution), it will be deleted from the current population because it may be located on a peak with low fitness;
ii. If the initial population size is N and it increases to $\lambda \times N$ ($\lambda > 1$) at a certain generation in the search process, the adaptive elitist-population approach will delete the lower fitness individuals from the current population to reduce its size to $\mu \times N$ ($0 < \mu < 1$) as the population size in the next generation.

The two population control constraints can delete individuals which are located on the peaks below the threshold set, allowing the population to focus on the high fitness optima.

3.6. AEGA algorithm

In this section, we will present the implementation outline of the adaptive elitist-population based genetic algorithm (AEGA) for solving multimodal optimization problems. Because our proposed elitist genetic operators and population control constraints can adaptively adjust the population size and conserve the possibly useful individuals, thereby they effectively simulate the "survival for the fittest" principle without any special selection operator. On the other hand, since the population of AEGA includes most of the elitist individuals, a classical selection operator could copy some individuals to the next generation and delete others from the population. Thus the selection operator will decrease the diversity of the population, increase the redundancy of the population, and reduce the efficiency of the algorithm. Hence, we design AEGA without any selection operator. The pseudocode for AEGA is shown in Fig. 10.

4. Evaluation of AEGA

In this section, the performance of AEGA is study. In Section 4.1, the experiment methodology is described. In Sections 4.2, 4.3 and 4.4, the experimental results on comparing AEGA with other multimodal evolutionary algorithms for different kinds of optimization problems are reported respectively.

4.1. Experiment methodology

The evaluation of any optimization algorithm is an extensive process. The true test of our proposed AEGA will, of course, be on real-world design problems for which the number, distribution, and quality of optima are unknown. However, the algorithm's

Table 1
The test suite of multimodal functions used in our experiments for finding all multiple optima.

Deb's function (5 optima):
$f_1(x) = \sin^6(5\pi x), x \in [0, 1]$;

Deb's decreasing function (5 optima):

$f_2(x) = 2^{-2((x-0.1)/0.9)^2} \sin^6(5\pi x), x \in [0, 1]$;

Roots function (6 optima):

$f_3(x) = \frac{1}{1+|x^6-1|}$, where $x \in C, x = x_1 + ix_2 \in [-2, 2]$;

Two dimensional multimodal function (100 optima):
$f_4(x) = x_1 \sin(4\pi x_1) - x_2 \sin(4\pi x_2 + \pi) + 1; x_1, x_2 \in [-2, 2]$,

Ten dimensional multimodal function (1024 optima):

$f_5(x_1, x_2, \ldots, x_{10}) = \frac{\sum_{i=1}^{10} |\sin(2\pi(1-x_i)^{3/5})|}{10}$; where $x_i \in [0, 1], i = 1, 2, \ldots, 10$.

performance must first be investigated on suitable test problems. When testing the algorithm on well understood problems, there are two measures of performance:

- The consistency of locating all known optima; and
- The average number of objective function evaluations required to find these optima (or the running time under the same condition).

The test suite used in our experiments includes the multimodal maximization problems listed in Tables 1, 3 and a multimodal stepwise function. The suite mainly contains some representative, complicated, and multimodal functions with many local optima. These types of functions are normally regarded as difficult to be optimized, and they are particularly challenging to the applicability and efficiency of the multimodal evolutionary algorithms. Our experiments of multimodal optimization problems were divided into three groups with different purposes. We report the results of each group below.

In our experiments, we compare the performance of AEGA with Deterministic Crowding [34], Probabilistic Crowding [35], Sequential Fitness Sharing [3], Clearing Procedure [41], Clustering Based Niching (CBN) [50], Clonal Selection [4], and Species Conserving Genetic Algorithm (SCGA) [32]. Since we were solving real-valued multimodal functions, we used consistent real coding variable representation, uniform crossover (Eq. (3)) and mutation (Eq. (4)) operators for each algorithm for fairness of comparison. The crossover probability $P_c = 0.8$ and the mutation probability $P_m = 0.1$ were used. We use the standard tournament selection operator with tournament size = 2 in our implementation of Sequential Fitness Sharing, Clearing Procedure, CBN, Clonal Selection, and SCGA.

The performance of the algorithms was measured by the number of optima each algorithm found, averaged over 30 runs. An optimum o_j was considered found if $\exists x_i \in Pop(t = T) \mid d(x_i, o_j) \leq \epsilon = 0.005$, where $Pop(t = T)$ is the complete population at the end of each run and x_i an individual in $Pop(t = T)$.

All algorithms were implemented in the C++ language and compiled using the same Microsoft Visual C++ 6.0 compiler. All these experiments were conducted on the same Dell Optiplex GX260

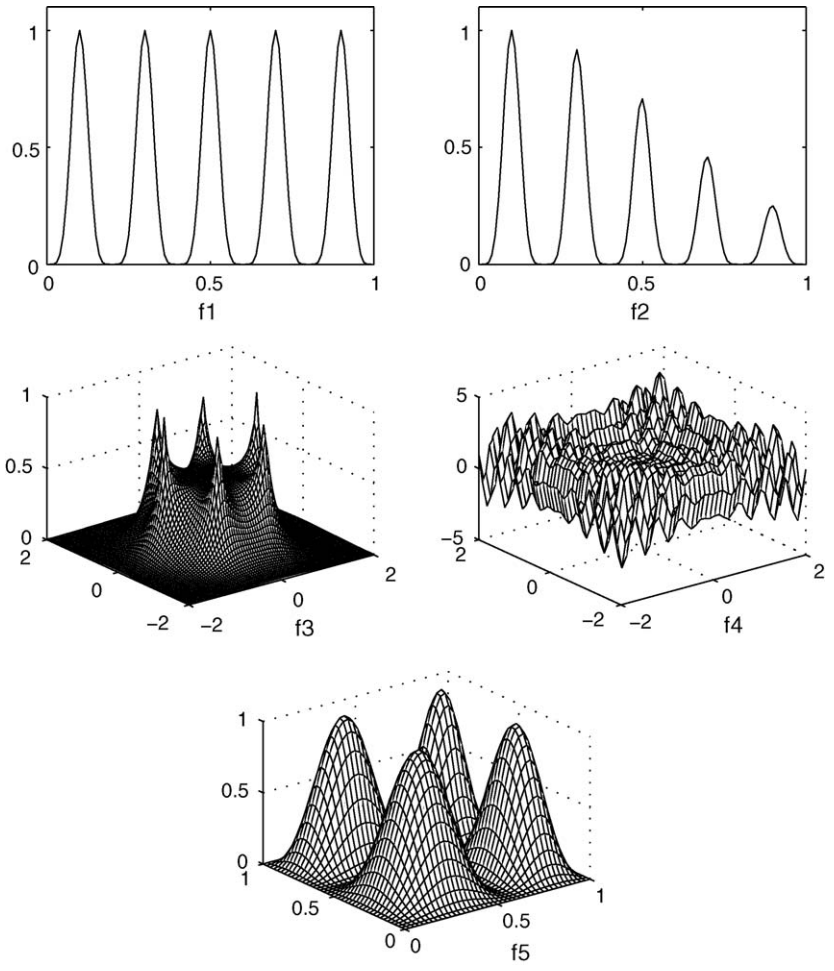

Fig. 11. The test suite of the multimodal function f_1–f_5.

Table 2
Performance comparison among the multimodal optimization algorithms for the test functions f_1–f_5.

Test function		Optima (ANO)	Distance (ADC)	Function evaluations (ANF)	Time (AET)
	Deterministic crowding	**5** (**0**)	1.52×10^{-4} (1.38×10^{-4})	7,153 (358)	0.091 (0.013)
	Probabilistic crowding	**5** (**0**)	3.63×10^{-4} (6.45×10^{-5})	10,304 (487)	0.163 (0.011)
	Sequential fitness sharing	**5** (**0**)	4.76×10^{-4} (6.82×10^{-5})	9,927 (691)	0.166 (0.028)
f_1	Clearing procedure	**5** (**0**)	1.27×10^{-4} (2.13×10^{-5})	5,860 (623)	0.128 (0.021)
	CBN	**5** (**0**)	2.94×10^{-4} (4.21×10^{-5})	10,781 (527)	0.237 (0.019)
	Clonal selection	**5** (**0**)	1.99×10^{-4} (8.25×10^{-5})	15,803 (381)	0.359 (0.015)
	SCGA	**5** (**0**)	1.16×10^{-4} (3.11×10^{-5})	6,792 (352)	0.131 (0.009)
	AEGA	**5** (**0**)	**4.60 \times 10^{-5}** (1.35×10^{-5})	**2,591** (278)	**0.039** (0.007)
	Deterministic crowding	3.53 (0.73)	3.61×10^{-3} (6.88×10^{-4})	6,026 (832)	0.271 (0.06)
	Probabilistic crowding	4.73 (0.64)	2.82×10^{-3} (8.52×10^{-4})	10,940 (951)	0.392 (0.07)
	Sequential fitness sharing	4.77 (0.57)	2.33×10^{-3} (4.36×10^{-4})	12,796 (1,430)	0.473 (0.11)
f_2	Clearing procedure	4.73 (0.58)	4.21×10^{-3} (1.24×10^{-3})	8,465 (773)	0.326 (0.05)
	CBN	4.70 (0.53)	2.19×10^{-3} (4.53×10^{-4})	14,120 (2,187)	0.581 (0.14)
	Clonal selection	**5** (**0**)	1.37×10^{-3} (6.87×10^{-4})	21,922 (746)	0.728 (0.06)
	SCGA	4.83 (0.38)	3.15×10^{-3} (4.71×10^{-4})	10,548 (1,382)	0.374 (0.09)
	AEGA	**5** (**0**)	**1.38 \times 10^{-4}** (2.32×10^{-5})	**3,605** (426)	**0.102** (0.04)
	Deterministic crowding	4.23 (1.17)	7.79×10^{-4} (4.76×10^{-4})	11,009 (1,137)	1.07 (0.13)
	Probabilistic crowding	4.97 (0.64)	3.35×10^{-3} (7.14×10^{-4})	16,391 (1,204)	1.72 (0.12)
	Sequential fitness sharing	4.87 (0.97)	2.56×10^{-3} (2.58×10^{-3})	14,424 (2,045)	1.84 (0.26)
f_3	Clearing procedure	**6** (**0**)	**7.43 \times 10^{-5}** (4.07×10^{-5})	12,684 (1,729)	1.59 (0.19)
	CBN	4.73 (1.14)	1.85×10^{-3} (5.42×10^{-4})	18,755 (2,404)	2.03 (0.31)
	Clonal selection	5.50 (0.51)	4.95×10^{-3} (1.39×10^{-3})	25,953 (2,918)	2.55 (0.33)
	SCGA	**6** (**0**)	3.27×10^{-4} (7.46×10^{-5})	13,814 (2,116)	1.75 (0.21)
	AEGA	**6** (**0**)	1.21×10^{-4} (8.63×10^{-5})	**6,218** (935)	**0.53** (0.07)
	Deterministic crowding	76.3 (11.4)	4.52×10^{-3} (4.17×10^{-3})	1,861,707 (329,254)	21.63 (2.01)
	Probabilistic crowding	92.8 (3.46)	3.46×10^{-3} (9.75×10^{-4})	2,638,581 (597,658)	31.24 (5.32)
	Sequential fitness sharing	89.9 (5.19)	2.75×10^{-3} (6.98×10^{-4})	2,498,257 (374,804)	28.47 (3.51)
f_4	Clearing procedure	89.5 (5.61)	3.83×10^{-3} (9.22×10^{-4})	2,257,964 (742,569)	25.31 (6.24)
	CBN	90.8 (6.50)	4.26×10^{-3} (1.14×10^{-3})	2,978,385 (872,050)	35.27 (8.41)
	Clonal selection	92.1 (4.63)	4.08×10^{-3} (8.25×10^{-3})	3,752,136 (191,849)	45.95 (1.56)
	SCGA	91.4 (3.04)	3.73×10^{-3} (2.29×10^{-3})	2,845,789 (432,117)	32.15 (4.85)
	AEGA	**99.6** (1.61)	**1.44 \times 10^{-4}** (2.82×10^{-5})	**426,599** (63,898)	**5.17** (0.93)
	Deterministic crowding	494.8 (50.4)	3.54×10^{-3} (2.63×10^{-3})	22,488,696 (4,108,125)	534.1 (137.5)
	Probabilistic crowding	811.3 (30.6)	4.85×10^{-3} (3.97×10^{-3})	35,963,701 (3,845,957)	807.8 (112.7)
	Sequential fitness sharing	753.9 (40.7)	7.96×10^{-4} (1.37×10^{-3})	32,052,038 (2,071,319)	739.6 (75.3)
f_5	Clearing procedure	952.7 (35.7)	6.24×10^{-4} (6.12×10^{-4})	26,239,172 (1,198,050)	549.2 (51.9)
	CBN	859.4 (58.3)	4.38×10^{-3} (3.24×10^{-3})	37,194,360 (5,352,638)	928.1 (184.3)
	Clonal selection	893.2 (47.1)	2.87×10^{-3} (1.63×10^{-3})	45,739,015 (1,084,927)	1,372.4 (49.6)
	SCGA	977.5 (23.4)	8.42×10^{-4} (1.97×10^{-4})	28,340,692 (4,619,737)	605.8 (114.9)
	AEGA	**1022.6** (3.86)	**4.58 \times 10^{-4}** (1.68×10^{-4})	**8,627,084** (435,968)	**93.21** (10.3)

The standard unit of the column Time (AET) is in seconds. (Numbers in parentheses are the standard deviations.)

computer with a Pentium-4 2.66G-HZ processor and 1G memory running Windows XP operating system. In the following sections, we shall present our experimental results.

4.2. Comparing AEGA with other algorithms for finding all multiple optima of the problems

In this section, we compare the performance of different algorithms on the multimodal problems f_1–f_5 listed in Table 1. The surfaces of f_1–f_4 and a two dimensional slice of f_5 are shown in Fig. 11 respectively. Our main objective in these experiments was to determine whether AEGA is more efficient and effective than other existing algorithms for finding all multiple optima of f_1–f_5. In the two population control constraints of AEGA, we set the parameters $\alpha = \beta = 0$ and $\lambda = \mu = 1$. This means that the population will not be adaptively adjusted by the population control constraints. Therefore, AEGA will focus on searching all the optima of the problems. For fairness of comparison, we implement all algorithms with the same parameter setting. For f_1–f_3, f_4 and f_5, the initial population sizes of the all algorithms were set to 100, 1000, and 2000 respectively. For Sequential Fitness Sharing, Clearing Procedure, CBN, Clonal Selection, SCGA, and AEGA, we set the distance threshold σ_s to 0.1. The algorithms' stopping criterion is such that when the number of optima found cannot be further increased in 10 successive generations after the first 50 generations, the execution of the algorithm will be stopped. We evaluated the performance of all the algorithms using four measures:

- The average number of optima found in the final population (ANO);
- The average distance between the multiple optima the algorithm found and their closest individuals in the final population (ADC);
- The average number of function evaluations (ANF); and
- The average execution time in seconds (AET).

Table 2 provides a summary of the performance comparison among different algorithms. From the ANO measure, AEGA could always find better or equally optimal solutions for the multimodal problems f_1–f_5. We can see that each algorithm can find all optima of f_1. For function f_2, only Clonal Selection and AEGA can find all optima each time. For function f_3, Clearing Procedure, SCGA and AEGA can get all optima each run. For functions f_4 and f_5, Deterministic Crowding leads to premature convergence and all other algorithms cannot get any better results, but AEGA can find all multiple optima 28 and 26 times respectively for 30 runs and its average successful rate of each run is higher than 99%. In addition, for functions f_3 and f_5, the performance of the algorithms with the elitist strategy (Clearing Procedure, SCGA and AEGA) are better than those without elitist strategy (Probabilistic Crowding, Sequential Fitness Sharing, CBN and Clonal Selection)

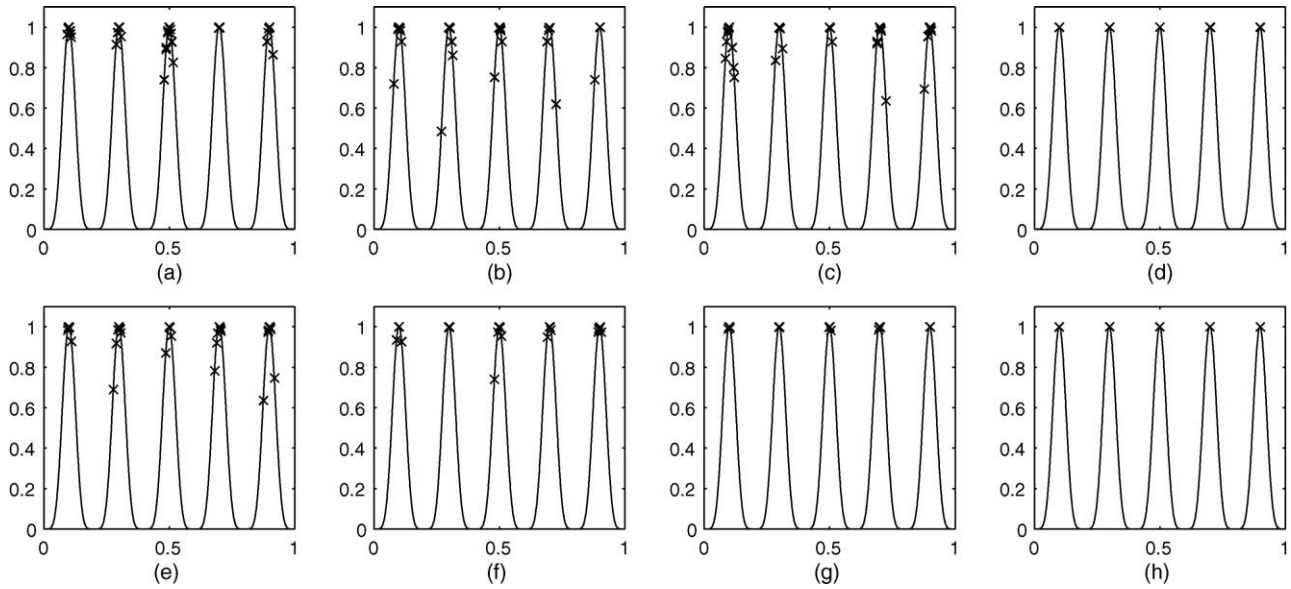

Fig. 12. A schematic illustration that the results of the algorithms for f_1, (a) Deterministic Crowding, (b) Probabilistic Crowding, (c) Sequential Fitness Sharing, (d) Clearing Procedure, (e) CBN, (f) Clonal Selection, (g) SCGA, (h) AEGA. (Keys: "×"—final population of the multimodal algorithms.)

because these functions have multiple global optima of equal value. If we compare the ADC measure listed in Table 2, it is found that AEGA obtained the best score for all the multimodal problems except f_3. For f_3, the solution precision of AEGA is only worse than that of Clearing Procedure. On the other hand, AEGA has smaller standard deviations in the ANO and ADC measures than the other algorithms, and hence its solution quality is more stable.

From the ANF measure in Table 2, we can clearly observe that AEGA makes orders of magnitude fewer function evaluations than other algorithms under the same termination criterion. Recall that all algorithms use the same conventional crossover and mutation operators, we can easily deduce that the adaptive elitist-population search technique in AEGA is able to produce better population (low redundancy and high diversity) more efficiently and effectively than the other multimodal search strategies.

To validate that AEGA improves over the other algorithms because AEGA produces more better populations in the iterations, Figs. 12 and 13 show the comparison results of AEGA and other multimodal algorithms for f_1 and f_2, respectively. The initial populations of all algorithms have 100 identical individuals. However, in the final population of AEGA, the 100 individuals decrease to 5 individuals corresponding to the 5 multiple optima, while, on the contrary, the final population of other seven algorithms still have 100 individuals, which are located on 5 peaks. Fig. 14(a) and (b) show the AEGA's changing population size in the optimization processes for f_1 and f_2 respectively. These clearly show why AEGA is more efficient than the other algorithms.

When comparing the execution time (AET) in Table 2, AEGA uses significantly less time to finish than other algorithms. The situation could be accounted for by the reduction of the redundancy in the population due to the elitist crossover operator. All these compar-

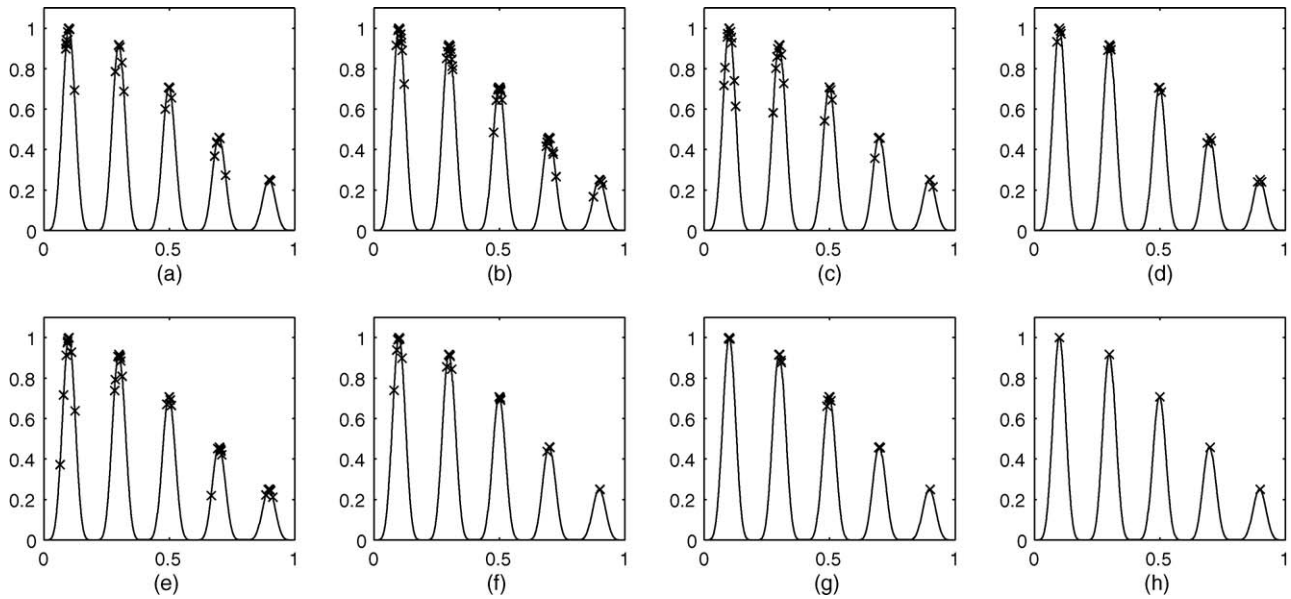

Fig. 13. A schematic illustration that the results of the algorithms for f_2, (a) Deterministic Crowding, (b) Probabilistic Crowding, (c) Sequential Fitness Sharing, (d) Clearing Procedure, (e) CBN, (f) Clonal Selection, (g) SCGA, (h) AEGA. (Keys: "×"—final population of the multimodal algorithms.)

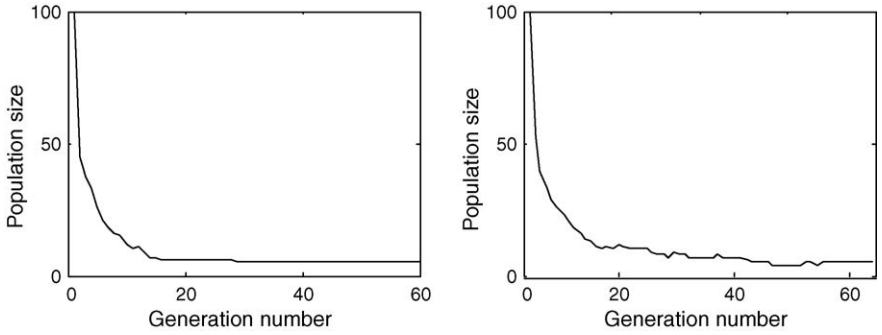

Fig. 14. The change processes of the AEGA's population sizes for f_1 (left) and f_2 (right) respectively.

Table 3
The test suite of multimodal functions used in our experiments for finding the multiple high fitness optima.

Rastrigin's function:
$f_6(x_1, x_2) = -(20 + x_1^2 + x_2^2 - 10(\cos 2\pi x_1 + \cos 2\pi x_2)), x_1, x_2 \in [-10, 10]$;
Griewank's function:
$f_7(x_1, x_2) = -(\frac{(x_1)^2+(x_2)^2}{4000} - \cos(\frac{x_1}{\sqrt{2}}) \times \cos(\frac{x_2}{\sqrt{2}}) + 1), x_1, x_2 \in [-600, 600]$;
Modified Griewank's function:
$f_8(x_1, x_2) = \frac{\cos(0.5x_1)+\cos(0.5x_2)}{10} + \cos(10x_1) \times \cos(10x_2) - 1.2, x_1, x_2 \in [0, 120]$;
n dimensional negative Shubert function:
$f_{9-n}(x_i) = -\prod_{i=1}^{n} \sum_{j=1}^{5} j \cos\{(j+1)x_i + j\}$, where $x_i \in [-10, 10]$;

isons show that AEGA generally outperforms the other multimodal algorithms in efficacy and efficiency.

4.3. Comparing AEGA with other algorithms for finding the multiple high fitness optima of the problems

In this section, we compare the performance of different algorithms on the multimodal optimization problems listed in Table 3. The surface of Rastrigin's function f_6, the one dimensional slice of Griewank's function f_7 for $[-200, 200]$, the one dimensional slice of Modified Griewank's function f_8 for [0, 120] and Two dimensional negative Shubert function f_{9-2} are shown in Fig. 15. These problems have a lot of local optima and very rugged landscapes. The goal of the multimodal optimizers is to find as many as possible global optima, and possibly good local optima. Rastrigin's function f_6 and Griewank's function f_7 only have one global optimum, so we want to test whether the multimodal algorithms can find their global optimum and 100 higher fitness local optima to validate the algorithms' performance. Modified Griewank's function f_8 has 100 global optima and the n dimensional negative Shubert functions f_{9-n} have $n \times 3^n$ global maxima, and the multimodal algorithms need to find all their global maxima. Recall that if the landscapes of the objective functions are very rugged, our proposed population control constraints of AEGA can avoid wasting computational power to obtain low fitness optima. Here, we set the parameters $\alpha = 3$, $\beta = 0.5$, $\lambda = 1.5$ and $\mu = 0.5$ in the two population constraints of AEGA. This means that if a newly generated individual has evolved for 3 generations and its fitness is still lower than half of the value of the best-so-far solution, it will be deleted from the current population because it may have

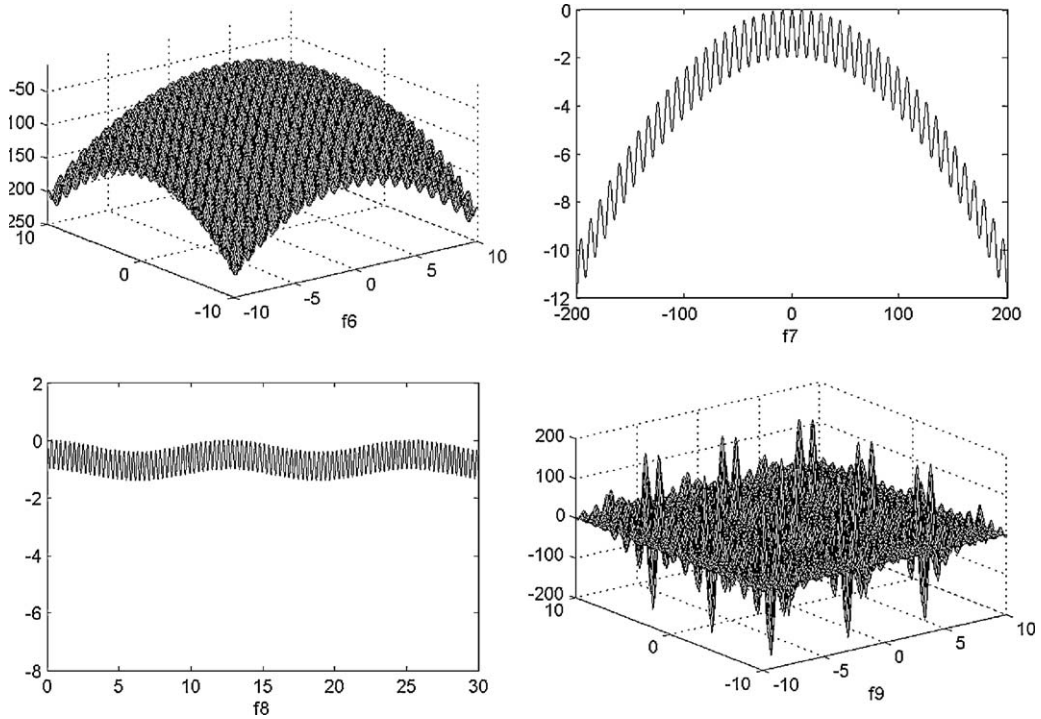

Fig. 15. The test suite of the multimodal function f_6–f_{9-2}.

located on a peak with a low fitness optimum. On the other hand, if the initial population size is N and it increases to $1.5 \times N$ at a certain generation in the search process, AEGA will delete the lower fitness individuals from the current population to reduce its size to $0.5 \times N$ as the initial population in the next generation.

Our main objective in these experiments was to determine whether AEGA is more efficient and effective than other existing algorithms for finding the multiple high fitness optima of functions f_6–f_{9-n}. In the experiments, the initial population sizes of all algorithms were set to 1000 and 2500 for f_6–f_{9-4} and f_{9-5} respectively. For Sequential Fitness Sharing, Clearing Procedure, CBN, Clonal Selection, SCGA, and AEGA, we set the distance threshold σ_s to 1, 5, 5, 5 for f_6, f_7, f_8, f_{9-n} respectively. The algorithms' stopping criterion is such that when the number of optima found cannot be further increased in 50 successive generations after the first 500 generations, the execution of the algorithm will be stopped. We still evaluate the performance of the all algorithms using the above four measures ANO, ADC, ANF, and AET. However, ANO and ADC only focus on a portion of the higher fitness optima found by the algorithm, i.e.:

- The average number of higher fitness optima found in the final population (ANO); and
- The average distance between the higher fitness multiple optima the algorithm found and their closest individuals in the final population (ADC);

Table 4 provides a summary of the performance comparison among different algorithms. From the ANO measure, we observe that AEGA could always find more optimal solutions for the multimodal problems f_6–f_{9-n}. For Rastrigin's function f_6, only AEGA can find all multiple high fitness optima 27 times for 30 runs and its average successful rate of each run is higher than 99%. On the contrary, the other algorithms cannot find all multiple higher fitness optima for any run. For Griewank's function f_7, only AEGA can get all multiple higher fitness optima for each run. Modified Griewank's function f_8 has numerous local optima whose value is close to the value of the global optima, AEGA still can find all global optima with higher than 97%. For Two and Three dimensional negative Shubert functions f_{9-2} and f_{9-3}, 3 algorithms (Clearing Procedure, SCGA and AEGA) and 2 algorithms (SCGA and AEGA) can get all multiple higher fitness maxima for each run respectively. For Four and Five dimensional negative Shubert functions f_{9-4} and f_{9-5}, no algorithm can find all global maxima for each run, but AEGA still gets better results than other algorithms. In addition, for finding multiple higher fitness optima, the performance of the algorithms with the elitist strategy (Clearing Procedure, SCGA and AEGA) are better than those without elitist strategy (Probabilistic Crowding, Sequential Fitness Sharing, CBN and Clonal Selection) and Deterministic Crowding leads to premature convergence. If we compare the ADC measure listed in Table 4, it is found that AEGA could always obtain the highest accurate solutions for all the multimodal problems f_6–f_{9-n}. Moreover, AEGA has smaller standard deviations in the ANO and ADC measures than other algorithms, and hence its solution quality is more stable.

From the ANF and AET measures in Table 4, we can clearly observe that AEGA uses significantly fewer function evaluations and less running time than other algorithms under the same termination criterion. Deterministic Crowding leads to premature convergence. Comparing with other algorithms except Deterministic Crowding, AEGA is at least 4.1, 4.5, 3.2, 4.6, 4.9, 6.1 and 6.3 times faster than them for functions f_6–f_{9-5} respectively. All these comparisons show that AEGA generally outperforms the other multimodal algorithms in efficacy and efficiency.

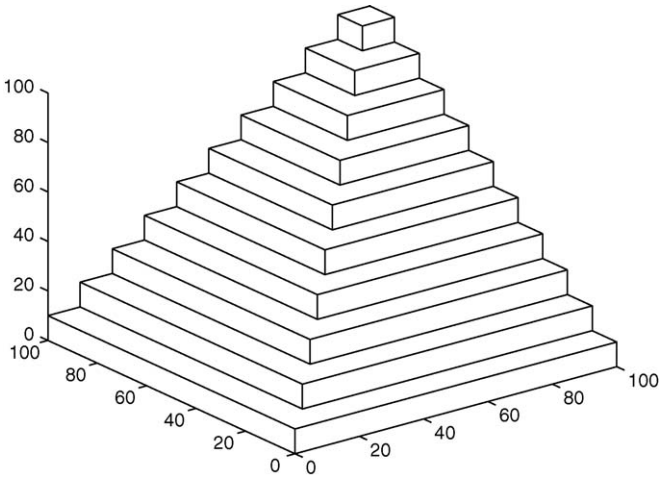

Fig. 16. The stepwise function f_{10} from the test suite.

4.4. Comparing AEGA with other algorithms for stepwise function

In this section, we compare the performance of different algorithms on the two dimensional stepwise function f_{10} (Fig. 16), which possesses ten absolutely flat regions of the step fitness from 10 to 100 respectively.

We use the percentage of the area of each step fitness θ in the total feasible solution space as the targeted solution distribution (% of the total number of solutions) for each θ. Our main objective in these experiments was to determine whether AEGA is more efficient and effective than other existing algorithms in distributing individuals regularly to all steps. For regular distribution of the individuals on each step surface, it should have a percentage (%) of individuals close to the targeted distribution of that step. In the experiments, the initial population size was set to 1000 for all algorithms. For Sequential Fitness Sharing, Clearing Procedure, CBN, Clonal Selection, SCGA, and AEGA, we set the distance threshold σ_s to 10. The algorithms' stopping criterion is such that when the average fitness of the population cannot be further increased in 50 successive generations after the first 500 generations, the execution of the algorithm will be stopped. We evaluate the performance of all the algorithms using the above two measures ANF, AET, and a new one called PIP(θ), which is defined as follows:

- The percentage (%) of individuals with the step fitness θ in the final population (PIP(θ)).

In the two population control constraints of AEGA, we set the parameters $\alpha = \beta = 0$ and $\lambda = \mu = 1$. This means that the population is not adaptively adjusted by the population control constrains. Therefore, AEGA will focus on searching all the optima of the stepwise problem.

Table 5 provides a summary of the performance comparison among different algorithms and the last row lists the targeted distribution of the flat steps of f_{10}. From the PIP(θ) measures, we can clearly observe that AEGA could always find better solutions for the stepwise problems f_{10} except PIP(40) and PIP(50). In Table 6, based on the targeted distribution of flat solutions, the Sum of Squared Error (SSE) of AEGA' solution of 5.08 is much smaller than

Table 4
Performance comparison among the multimodal optimization algorithms for the test functions f_6–f_{9-5}.

Test function		Optima (ANO)	Distance (ADC)	Function evaluations (ANF)	Time (AET)
	Deterministic crowding	62.4 (14.3)	4.72×10^{-3} (4.59×10^{-3})	1,760,199 (254,341)	14.62 (2.83)
	Probabilistic crowding	84.7 (5.48)	1.50×10^{-3} (9.38×10^{-4})	2,631,627 (443,522)	34.39 (5.20)
	Sequential fitness sharing	76.3 (7.08)	3.51×10^{-3} (1.66×10^{-3})	2,726,394 (562,723)	36.55 (7.13)
f_6	Clearing procedure	93.6 (2.31)	2.78×10^{-3} (1.20×10^{-3})	2,107,962 (462,662)	28.61 (6.47)
(101)	CBN	87.9 (7.78)	4.33×10^{-3} (2.82×10^{-3})	2,835,119 (683,195)	37.05 (8.23)
	Clonal selection	90.6 (9.95)	3.15×10^{-3} (1.47×10^{-3})	5,075,208 (194,376)	58.02 (2.19)
	SCGA	97.4 (4.80)	1.34×10^{-3} (8.72×10^{-4})	2,518,301 (643,129)	30.27 (7.04)
	AEGA	**100.4** (1.39)	**6.77 \times 10^{-4}** (3.18×10^{-4})	**688,731** (71,813)	**7.08** (0.51)
	Deterministic crowding	52.6 (8.86)	3.71×10^{-3} (1.54×10^{-3})	2,386,960 (221,982)	19.10 (2.26)
	Probabilistic crowding	79.2 (4.94)	3.48×10^{-3} (3.79×10^{-3})	3,861,904 (457,862)	43.53 (4.38)
	Sequential fitness sharing	63.0 (5.49)	4.76×10^{-3} (3.55×10^{-3})	3,619,057 (565,392)	42.98 (6.35)
f_7	Clearing procedure	79.4 (4.31)	2.95×10^{-3} (1.64×10^{-3})	3,746,325 (594,758)	45.42 (7.64)
(101)	CBN	71.3 (9.26)	3.29×10^{-3} (4.11×10^{-3})	4,155,209 (465,613)	48.23 (5.42)
	Clonal selection	89.2 (5.44)	3.02×10^{-3} (1.63×10^{-3})	5,423,739 (231,005)	69.39 (2.63)
	SCGA	94.9 (8.18)	2.63×10^{-3} (1.81×10^{-3})	3,629,461 (373,382)	47.84 (6.09)
	AEGA	**101** (0)	**1.32 \times 10^{-3}** (8.76×10^{-4})	**1,067,792** (13,241)	**9.64** (1.12)
	Deterministic crowding	44.2 (7.93)	4.45×10^{-3} (3.63×10^{-3})	2,843,452 (353,529)	23.14 (3.85)
	Probabilistic crowding	70.1 (8.36)	2.52×10^{-3} (1.47×10^{-3})	4,325,469 (574,368)	49.51 (6.72)
	Sequential fitness sharing	58.2 (9.48)	4.14×10^{-3} (3.31×10^{-3})	4,416,150 (642,415)	54.43 (12.6)
f_8	Clearing procedure	67.5 (10.11)	2.31×10^{-3} (1.43×10^{-3})	4,172,462 (413,537)	52.39 (7.21)
(100)	CBN	53.1 (7.58)	4.36×10^{-3} (3.57×10^{-3})	4,711,925 (584,396)	61.07 (8.14)
	Clonal selection	74.4 (12.75)	3.52×10^{-3} (2.19×10^{-3})	5,835,452 (498,033)	74.26 (5.47)
	SCGA	87.3 (9.61)	3.15×10^{-3} (2.07×10^{-3})	3,964,491 (412,538)	53.87 (8.46)
	AEGA	**98.6** (1.65)	**1.54 \times 10^{-3}** (9.55×10^{-4})	**1,725,183** (73,497)	**15.42** (2.30)
	Deterministic crowding	9.37 (1.91)	3.26×10^{-3} (5.34×10^{-4})	648,602 (75,413)	4.58 (0.57)
	Probabilistic crowding	15.17 (2.43)	2.87×10^{-3} (5.98×10^{-4})	1,823,774 (265,387)	12.92 (2.01)
	Sequential fitness sharing	12.29 (2.14)	1.42×10^{-3} (5.29×10^{-4})	1,767,562 (528,317)	14.12 (3.51)
f_{9-2}	Clearing procedure	**18** (0)	1.19×10^{-3} (6.05×10^{-4})	1,875,729 (265,173)	11.20 (2.69)
(18)	CBN	14.84 (2.70)	4.39×10^{-3} (2.86×10^{-3})	2,049,225 465,098)	18.26 (4.41)
	Clonal selection	14.61 (2.33)	3.42×10^{-3} (1.58×10^{-3})	4,989,856 (618,759)	33.85 (5.36)
	SCGA	**18** (0)	1.58×10^{-3} (4.12×10^{-4})	2,261,469 (315,727)	13.71 (1.84)
	AEGA	**18** (0)	**3.34 \times 10^{-4}** (1.27×10^{-4})	**450,823** (84,012)	**2.39** (0.43)
	Deterministic crowding	39.1 (7.24)	3.52×10^{-3} (1.87×10^{-3})	4,963,832 (536,959)	41.34 (4.72)
	Probabilistic crowding	56.6 (6.35)	4.09×10^{-3} (2.47×10^{-3})	6,625,478 (947,516)	60.34 (8.77)
	Sequential fitness sharing	48.5 (9.04)	3.95×10^{-3} (1.98×10^{-3})	5,611,468 (803,659)	54.76 (7.58)
f_{9-3}	Clearing procedure	68.2 (5.71)	1.48×10^{-3} (8.37×10^{-4})	6,973,616 (783,614)	56.85 (6.35)
(81)	CBN	56.9 (5.97)	3.91×10^{-3} (2.15×10^{-3})	8,061,334 (1,139,233)	86.16 (12.5)
	Clonal selection	60.5 (5.11)	3.71×10^{-3} (1.96×10^{-3})	13,761,253 (1,482,397)	112.71 (13.2)
	SCGA	**81** (0)	2.39×10^{-3} (9.31×10^{-4})	6,561,413 (627,383)	53.07 (5.20)
	AEGA	**81** (0)	**8.59 \times 10^{-4}** (2.75×10^{-4})	**1,394,557** (162,932)	**10.62** (1.44)
	Deterministic crowding	141.5 (17.4)	4.41×10^{-3} (2.59×10^{-3})	15,751,612 (2,692,755)	183.49 (28.4)
	Probabilistic crowding	170.9 (15.3)	3.65×10^{-3} (2.09×10^{-3})	25,270,263 (5,936,534)	293.35 (61.5)
	Sequential fitness sharing	154.6 (14.3)	3.85×10^{-3} (1.03×10^{-3})	28,757,064 (5,602,523)	369.79 (68.1)
f_{9-4}	Clearing procedure	226.8 (13.7)	3.42×10^{-3} (1.98×10^{-3})	21,881,324 (3,304,173)	243.91 (37.8)
(324)	CBN	193.4 (18.6)	3.95×10^{-3} (1.18×10^{-3})	42,898,602 (5,267,027)	644.95 (73.7)
	Clonal selection	227.8 (16.1)	3.52×10^{-3} (2.16×10^{-3})	51,817,693 (5,829,516)	627.17 (70.9)
	SCGA	244.2 (15.4)	3.13×10^{-3} (1.18×10^{-3})	26,802,791 (4,121,909)	329.74 (53.3)
	AEGA	**318.3** (4.18)	**1.13 \times 10^{-3}** (4.42×10^{-4})	**3,725,621** (319,536)	**39.64** (3.46)
	Deterministic crowding	638.2 (62.7)	4.69×10^{-3} (2.19×10^{-3})	103,564,891 (12,934,724)	2087.33 (241.2)
	Probabilistic crowding	744.6 (45.1)	3.71×10^{-3} (1.67×10^{-3})	150,294,243 (15,714,937)	2992.61 (298.1)
	Sequential fitness sharing	675.6 (55.8)	3.96×10^{-3} (2.84×10^{-3})	136,205,221 (17,349,408)	2819.46 (362.5)
f_{9-5}	Clearing procedure	883.1 (41.2)	3.44×10^{-3} (1.33×10^{-3})	113,749,762 (16,150,218)	2241.06 (319.3)
(1215)	CBN	726.8 (43.9)	4.11×10^{-3} (3.05×10^{-3})	175,874,166 (19,362,825)	3906.14 (434.7)
	Clonal selection	863.4 (83.1)	3.45×10^{-3} (2.32×10^{-3})	214,211,935 (18,595,896)	4270.63 (391.2)
	SCGA	924.6 (51.8)	2.95×10^{-3} (1.51×10^{-3})	121,270,156 (11,662,490)	2594.17 (217.6)
	AEGA	**1194.8** (16.1)	**2.14 \times 10^{-3}** (9.27×10^{-4})	**18,447,699** (1,678,397)	**354.88** (32.71)

(The numbers of multiple optima considered are shown in the brackets.) The standard unit of the column Time (AET) is in seconds.

the SSE of 79.35 of the second best algorithm Sequential Fitness Sharing. This means that AEGA can distribute the individuals more regularly to all flat solutions. In addition, for finding multiple flat solutions, the performance of the algorithms without elitist strategy (Probabilistic Crowding, Sequential Fitness Sharing, CBN and Clonal Selection) are better than those of the algorithms with the elitist strategy (Deterministic Crowding, Clearing Procedure and SCGA) except AEGA.

From the ANF and AET measures listed in Table 5, we can clearly observe that AEGA uses fewer function evaluations and less running time than the other algorithms under the same termination criterion. For the stepwise function f_{10}, AEGA is at least 3 times faster than the other algorithms because the average of its final population sizes is only 159.27 for 30 runs as compared to 1000 for all other algorithms. All these comparisons show the strong performance of AEGA in efficacy and efficiency.

Table 5
Performance comparison among the multimodal optimization algorithms for the multiple stepwise function f_{10}.

	PIP(10)	PIP(20)	PIP(30)	PIP(40)	PIP(50)	PIP(60)
Deterministic crowding	0.83 (2.36)	1.66 (2.13)	2.23 (5.19)	3.69 (4.45)	4.23 (2.53)	7.49 (9.40)
Probabilistic crowding	6.92 (4.27)	10.44 (9.42)	9.48 (5.33)	**12.95** (4.59)	**11.13** (7.58)	10.87 (6.41)
Sequential fitness sharing	13.75 (12.06)	12.58 (6.19)	12.73 (19.17)	12.86 (10.29)	13.34 (9.72)	12.10 (9.96)
Clearing procedure	5.83 (8.11)	5.71 (6.23)	6.09 (6,35)	8.12 (8.22)	8.52 (7.19)	7.95 (7.38)
CBN	13.77 (14.31)	12.82 (5.35)	13.47 (13.91)	9.26 (8.79)	9.86 (8.82)	8.09 (2.67)
Clonal selection	7.71 (9.65)	6.37 (8.03)	7.96 (5.39)	9.34 (4.66)	11.57 (8.98)	14.86 (7.61)
SCGA	1.64 (2.47)	3.52 (4.62)	5.41 (1.29)	6.36 (4.69)	8.75 (3.76)	9.71 (11.16)
AEGA	**17.89** (3.28)	**16.15** (2.95)	**15.43** (3.27)	12.61 (2.71)	10.39 (3.82)	**9.35** (2.26)
Targeted distribution of flat optima	**19**	**17**	**15**	**13**	**11**	**9**
	PIP(70)	PIP(80)	PIP(90)	PIP(100)	Function evaluations (ANF)	Time (AET)
Deterministic crowding	9.57 (4.52)	15.84 (7.94)	19.25 (9.87)	25.21 (8.58)	1,683,082 (182,532)	14.29 (1.17)
Probabilistic crowding	9.86 (5.52)	11.52 (8.73)	9.07 (12.64)	7.76 (8.02)	3,494,483 (343,778)	32.41 (3.46)
Sequential fitness sharing	9.38 (8.70)	6.13 (7.25)	4.29 (5.14)	2.84 (2.31)	3,197,843 (437,906)	30.54 (4.38)
Clearing procedure	10.91 (8.77)	11.20 (10.14)	14.41 (15.38)	21.26 (3.69)	2,764,416 (262,851)	26.31 (2.23)
CBN	7.85 (6.17)	9.21 (8.63)	9.73 (7.16)	5.94 (7.11)	2,619,020 (472,955)	23.94 (5.03)
Clonal selection	12.09 (3.63)	12.44 (9.83)	11.18 (6.41)	6.48 (2.15)	4,020,947 (194,152)	46.76 (0.91)
SCGA	10.46 (7.97)	16.28 (12.46)	18.75 (9.86)	19.12 (6.51)	2,997,643 (332,374)	27.96 (3.16)
AEGA	**7.58** (5.47)	**4.86** (1.98)	**3.41** (1.11)	**2.33** (1.93)	**594,874** (82,257)	**4.12** (0.73)
Targeted distribution of flat optima	**7**	**5**	**3**	**1**	-	-

PIP(θ) describes the percentage (%) of individuals with fitness θ in the final population, where $\theta = 10, 20, \ldots, 100$. The standard unit of the column Time (AET) is in seconds.

Table 6
Comparison of Sum of Squared Error (SSE) among the multimodal evolutionary algorithms for the multiple stepwise function f_{10}.

	Deterministic Crowding	Probabilistic Crowding	Sequential fitness Sharing	Clearing Procedure	CBN	Clonal Selection	SCGA	AEGA
SSE	1837.624	356.18	79.35	1005.75	151.42	516.29	1340.31	**5.08**

Two dimensional stepwise function:

$$f_{10}(x_1, x_2) = \begin{cases} 10, & \text{if} 0 \le x_1 < 10 \text{and} 0 \le x_2 \le 100, \text{or} 0 \le x_1 \le 100 \text{and} 0 \le x_2 < 10; \\ 20, & \text{if} 10 \le x_1 < 20 \text{and} 10 \le x_2 \le 100, \text{or} 10 \le x_1 \le 100 \text{and} 10 \le x_2 < 20; \\ 30, & \text{if} 20 \le x_1 < 30 \text{and} 20 \le x_2 \le 100, \text{or} 20 \le x_1 \le 100 \text{and} 20 \le x_2 < 30; \\ 40, & \text{if} 30 \le x_1 < 40 \text{and} 30 \le x_2 \le 100, \text{or} 30 \le x_1 \le 100 \text{and} 30 \le x_2 < 40; \\ 50, & \text{if} 40 \le x_1 < 50 \text{and} 40 \le x_2 \le 100, \text{or} 40 \le x_1 \le 100 \text{and} 40 \le x_2 < 50; \\ 60, & \text{if} 50 \le x_1 < 60 \text{and} 50 \le x_2 \le 100, \text{or} 50 \le x_1 \le 100 \text{and} 50 \le x_2 < 60; \\ 70, & \text{if} 60 \le x_1 < 70 \text{and} 60 \le x_2 \le 100, \text{or} 60 \le x_1 \le 100 \text{and} 60 \le x_2 < 70; \\ 80, & \text{if} 70 \le x_1 < 80 \text{and} 70 \le x_2 \le 100, \text{or} 70 \le x_1 \le 100 \text{and} 70 \le x_2 < 80; \\ 90, & \text{if} 80 \le x_1 < 90 \text{and} 80 \le x_2 \le 100, \text{or} 80 \le x_1 \le 100 \text{and} 80 \le x_2 < 90; \\ 100, & \text{if} 90 \le x_1 < 100 \text{and} 90 \le x_2 \le 100, \text{or} 90 \le x_1 \le 100 \text{and} 90 \le x_2 < 100. \end{cases}$$

In the above experiments, we set the parameters $\alpha = \beta = 0$, $\lambda = \mu = 1$ and do not use the population control constraints to delete the individuals with low fitness from the population. When we set $\alpha = 5$ and $\lambda = \mu = 1$, if β increases to 0.6, 0.7, 0.8 and 0.9 respectively, the populations of AEGA converged to the highest step with fitness 100 as shown in Table 7.

4.5. Comparison with the simple version of AEGA

In [31], we proposed the simple version of AEGA. Comparing with [31], here we have proposed new strategies to extend AEGA to solve the complicated multimodal optimization problems, which include multimodal functions with very rugged landscapes and stepwise functions. We have analyzed the relative ascending directions with the flat case, and renewed the elitist genetic operators based on it. We have also proposed the two population control constraints to construct the improved version of AEGA. In this section, we have added 4 multimodal optimization algorithms (totally 8 algorithms) and 9 complicated problems (totally 13 problems) to validate the performance of AEGA. The parameter settings for these algorithms are also different from [31]. We have also added some parameters' discussions in the following sections.

Table 7
Comparison of PIP(100) among AEGA with different parameters' setting for the multiple stepwise functions f_{10}.

	$\beta = 0.6$	$\beta = 0.7$	$\beta = 0.8$	$\beta = 0.9$
PIP(100)	8.61	19.07	35.92	79.83

5. The effect of the distance parameter

In this section, we study how the distance parameter σ_s affects the performance of AEGA. In Sections 5.1 and 5.2, the experimental results on comparing AEGA with different σ_s for Roots function f_3 and Four dimensional negative Shubert function f_{9-4} are reported respectively.

5.1. Effect of different σ_s for f_3

In many niching methods, the distance threshold σ_s plays a crucial role. AEGA also utilizes this parameter, but it is not as important as in the other niching methods, because the relative

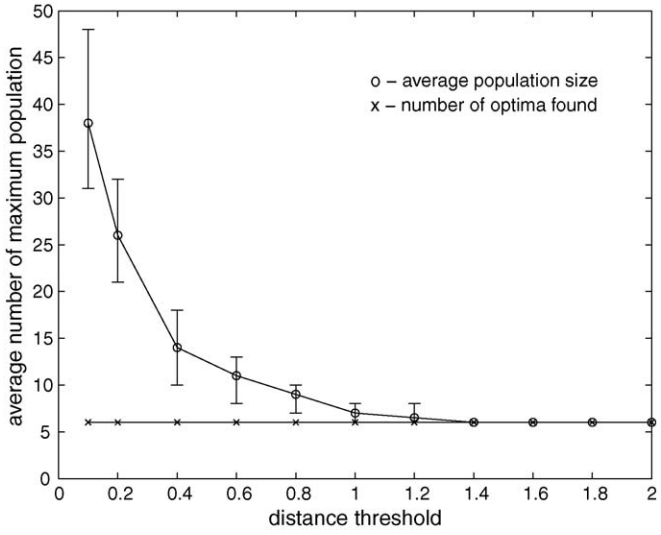

Fig. 17. Variation in the average number of maxima population size and optima found with the distance threshold for Roots function f_3, with one standard deviation error bars (Keys: "○" – average population size; "×"—number of optima found).

ascending directions between individuals are more important to identify the individuals' dissimilarity. To examine the influence of the distance threshold, we conducted a series of experiments, again using Roots function f_3, in which we varied the value of σ_s, and recorded the number of multiple optima found and the maximum population size in the AEGA search process. For these runs we used the initial population size of 2, the crossover probability $P_c = 1$, the mutation probability $P_m = 1$, the number of generation $T = 200$, and set $\alpha = \beta = 0$, $\lambda = \mu = 1$ in the population control constraints. We averaged the results over 30 AEGA runs for each value of σ_s. The experimental results obtained are shown in Fig. 17.

Fig. 17 shows that, as expected, as the distance threshold is increased, the number of multiple optima found does not decrease. For values of σ_s between 0.1 and 2, AEGA can find all multiple optima (note: 2 is the half value of the domain width of all the dimensions). The difference is that when σ_s is small, more individuals are generated in the beginning generations. Then the population size gradually decreases through the elitist genetic operators eliminating other individuals belonging to the same peak in the evolutionary search process, until the population size and the number of peaks equalize. When σ_s is large, generating new individuals mainly depends on determining the relative ascending directions of individuals. New individuals are generated parsimoniously and the population size gradually increases to the number of multiple optima. This demonstrates AEGA is stable with respect to various distance thresholds.

There are three possibilities:

i. For the case that the relative ascending directions between two individuals are back to back, we need not consider their distance and can consider them located on different peaks.
ii. For the case that the relative ascending directions between two individuals are face to face or one-way, and the distance between them is smaller than the distance threshold, the two individuals can be considered to be on the same peak. If they are in fact located on different peaks, through several AEGA's generations, eventually an offspring on the new peak will have back to back directions with one of the other individuals. Using these directions AEGA can find a new peak even with the width smaller than the distance threshold.
iii. For the case that the relative ascending directions between two individuals are face to face or one-way, and the distance between them equal to or larger than the distance threshold, the two individuals can be considered to be on different peaks. But if they are in fact located on the same peak, through several AEGA's generations, when they get closer to the optimum, their distance decreases and becomes smaller than the distance threshold eventually. Then the elitist operators will only select the elitist one to next generation to reduce the algorithm's redundancy. From the experiment, we can observe that the average population size varied with the distance threshold σ_s, but eventually converge to a constant which is equal to the number of optima found.

In other words, our proposed algorithms based on the distance threshold is relatively robust.

5.2. Effect of different σ_s for f_{9-4}

In this section, we compare the performance of AEGA with different σ_s values on Four dimensional negative Shubert function f_{9-4}. As described in Section IV-C, this problem has 324 global maxima and a lot of local maxima. The goal of multimodal optimizers was to find all the global maxima from the rough surface.

In this experiment, we investigate the effect of varying the values of σ_s. The initial population size of AEGA was set to 1000, the crossover probability $P_c = 0.8$, the mutation probability $P_m = 0.1$ and $\alpha = 3$, $\beta = 0.5$, $\lambda = 1.5$, $\mu = 0.5$ in the population control constraints. The algorithms' stopping criterion is such that when the number of optima found cannot be further increased in 50 successive generations after the first 500 generations, the execution of the algorithm will be stopped. We performed the experiments using σ_s values of 2, 5, 10, 12, 15 respectively. The results are presented in Table 8.

From the table, we notice that the best final results (see the ANO and ADC measures) are obtained for $\sigma_s = 10$. However, it is not significantly different from those with other values of σ_s. For different values of σ_s, the successful rates of AEGA are all higher than 97%. The longest and smallest average execution time (AET) values are obtained for $\sigma_s = 2$ and $\sigma_s = 15$ respectively. The ANF and AET measures decrease with increasing values of σ_s. However, the differences are small compared with those for the other values of σ_s. Therefore, we can conclude that the efficacy and efficiency of AEGA with the two population constraints are stable for different σ_s values in a relative wide range. This is important for using AEGA to solve hard multimodal engineering design problems.

Table 8
Performance comparison among AEGA with different σ_s value for the test function f_{9-4} (324 global optima).

Test function		Optima (ANO)	Distance (ADC)	Function evaluations (ANF)	Time (AET)
f_{9-4}	$\sigma_s = 2$	317.7 (18.12)	1.57×10^{-3} (3.81×10^{-4})	3,749,726 (366,827)	40.08 (3.84)
	$\sigma_s = 5$	318.3 (14.18)	1.13×10^{-3} (4.42×10^{-4})	3,725,621 (349,536)	39.64 (3.16)
	$\sigma_s = 10$	320.1 (13.49)	2.31×10^{-3} (2.97×10^{-4})	3,704,583 (412,368)	39.43 (4.20)
	$\sigma_s = 12$	317.8 (13.63)	2.63×10^{-3} (3.04×10^{-4})	3,625,245 (380,401)	38.74 (3.96)
	$\sigma_s = 15$	315.4 (16.82)	1.71×10^{-3} (4.69×10^{-4})	3,618,106 (321,398)	38.59 (3.28)

The standard unit of the column Time (AET) is in seconds.

Table 9
Different parameter settings in the two population control constraints.

Cases	α	β	λ	μ	Individual survival	Population size fluctuation allowed
setting 1	10	0.5	1.5	0.5	long	large
setting 2	10	0.5	1.2	0.8	long	small
setting 3	3	0.75	1.5	0.5	short	large
setting 4	3	0.75	1.2	0.8	short	small

6. Effect of different parameters α, β, λ, μ in the population control constraints

In our proposed AEGA, we have added four parameters α, β, λ, μ into the two population control constraints. These constraints can avoid AEGA wasting computational power to obtain low fitness optima. In this section, we study how these parameters affect the performance of AEGA.

In the first population control constraint of AEGA, if a newly generated individual has evolved for α generations and its fitness is still lower than a fitness threshold, $\beta\times$ the best-so-far solution (e.g., $0.5\times$ the best-so-far solution), it will be deleted from the current population because it may be located on a peak with low fitness. When we set the parameter α relative large and the parameter β relative small, the new generated individual can survive in the population for relative longer (more generations) and vice versa. Here, we set $\alpha = 3$, $\beta = 0.75$ (short survival for individuals) and $\alpha = 10$, $\beta = 0.5$ (long survival for individuals) to evaluate these two situations.

In the second population control constraint of AEGA, if the initial population size is N and it increases to $\lambda \times N$ ($\lambda > 1$) at a certain generation in the search process, the adaptive elitist-population approach will delete the lower fitness individuals from the current population to reduce its size to $\mu \times N$ (where $0 < \mu < 1$) as the population size in the next generation. When we set the parameter λ relative large and the parameter μ relative small, the population size can change in a relative larger range, and vice versa. Here, we set $\lambda = 1.5$, $\mu = 0.5$ (large fluctuation) and $\lambda = 1.2$, $\mu = 0.8$ (small fluctuation) to evaluate these two situations.

Consequently, we get the four combinations of parameter settings as listed in Table 9. To examine the influence of the different parameter settings, we conduct a series of experiments, again using the negative Shubert function f_{9-4}, and recorded the number of multiple optima found and the population size in the AEGA search processes. For these runs, we used the initial population size of 1000, the crossover probability $P_c = 0.8$, the mutation probability $P_m = 0.1$, the distance threshold $\sigma_s = 10$. The algorithms' stopping criterion is the same as the previous experiments. We average the results over 30 AEGA runs for each setting. The experimental results obtained are shown in Figs. 18, 19 and Table 10 respectively.

Fig. 18 shows the average population size curves of AEGA with each parameter setting when applied to f_{9-4}. It is observed that the AEGA's population sizes fluctuate between the upper and lower bounds and their trendlines smoothen out towards the end of the search processes. The initial patch of relatively high frequency of fluctuations in each of the four experiments is due to the vigorous exploration in the search space by the genetic operators. It should be noted that the elitist crossover and the first population con-

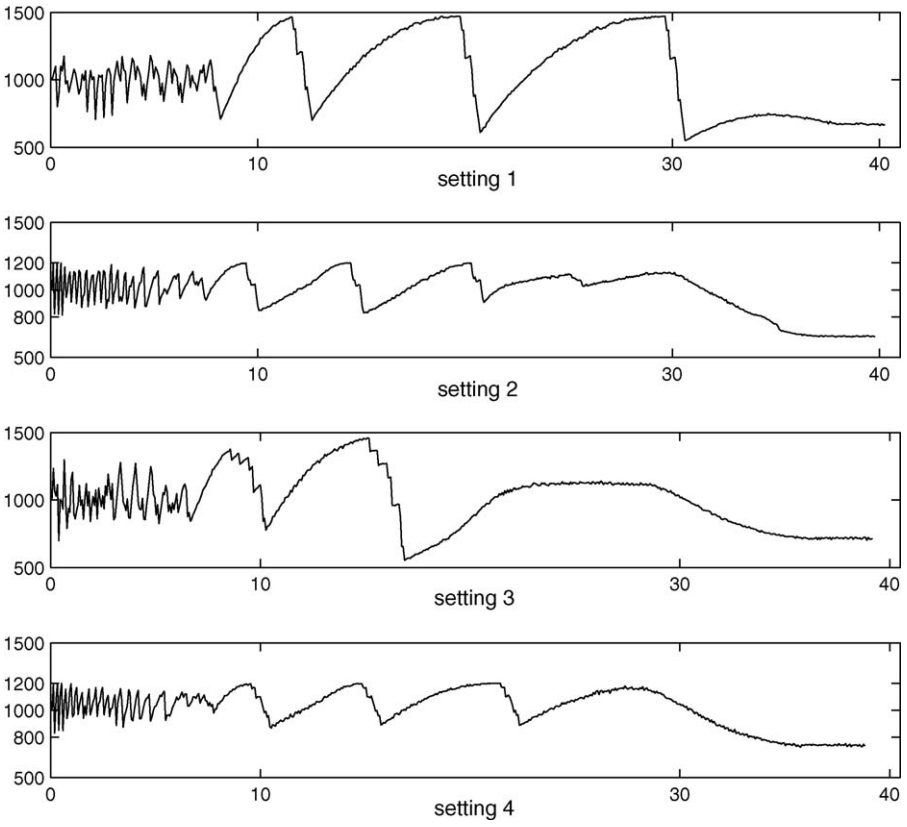

Fig. 18. Variation in the average numbers of population sizes against the execution time with the settings 1–4 for the test function f_{9-4}. (Keys: x-axis—execution time; y-axis—average population size.)

Table 10
Performance comparison among AEGA with different α, β, λ, μ settings for the test function f_{9-4} (324 global optima).

Test function		Optima (ANO)	Distance (ADC)	Function evaluations (ANF)	Time (AET)
f_{9-4}	setting 1	320.5 (10.31)	1.85×10^{-3} (4.12×10^{-4})	3,764,478 (290,849)	40.27 (3.18)
	setting 2	319.2 (14.59)	2.65×10^{-3} (2.92×10^{-4})	3,738,407 (371,953)	39.79 (3.96)
	setting 3	318.6 (12.71)	3.15×10^{-3} (3.24×10^{-4})	3,675,478 (347,432)	39.18 (3.29)
	setting 4	317.4 (13.65)	2.28×10^{-3} (4.18×10^{-4})	3,628,831 (328,631)	38.82 (3.64)

The standard unit of the column Time (AET) is in seconds.

straint tend to reduce the population size while the elitist mutation tends to increase it. Before the convergence towards the end, the effect of the elitist mutation dominates over the other two operators and increases the population size. When the population size increases to the upper bound $\lambda \times N$, it will be decreased to the lower bound $\mu \times N$ by the second population control constraint. After the initial violent fluctuations, the relatively smooth fluctuations continue until the search algorithm converges and has allocated the elitist individuals for the desired optima towards the end of the search process. For the different parameter settings 1–4, if the individuals are allowed to stay in the population for relative long time (settings 1 and 2), the population size tends to increase faster and fluctuates up and down more times between the lower and upper bounds. On the other hand, if the range of the population size allowed is set relatively narrower (settings 2 and 4), the population size also fluctuates more times before convergence (compared to settings 1 and 3 respectively).

Fig. 19 shows the curves of the average numbers of multiple optima found against the execution time by AEGA with each parameter setting when applied to f_{9-4}. In comparison, if the individuals can only stay in the population for relative few generations (settings 3 and 4), the AEGA runs appear to do well in the early generations, then they are overtaken by AEGA with settings 1 and 2 respectively. Finally, all of the AEGA runs finish with similar mean values. On the other hand, if the population size is allowed to fluctuate between relative large ranges (settings 1 and 3), the AEGA runs appear to do well in the early generations, but all of the AEGA runs finish with similar mean values as mentioned above.

The comparison results of AEGA with the four different parameter settings 1–4 are presented in Table 10. From the table, we notice that the best final results (see the ANO and ADC measures) are obtained for AEGA with parameter setting 1. However, the average number of function evaluations (ANF) and the average execution time (AET) of the AEGA with setting 1 is also slightly higher than the others. For the four different parameter settings 1–4 covering the possible ranges, the differences in the ANO and AET measures are smaller than 1.0% and 4.5% respectively. Therefore, we can conclude that the efficacy and efficiency of AEGA with the two population constraints are stable with respect to various parameter settings of α, β, λ and μ. Based on our experiment results, AEGA can efficiently solve multimodal optimization problems if $\alpha \in [3, 10]$, $\beta \in [0.5, 0.75]$, $\lambda \in [1.2, 1.5]$ and $\mu \in [0.5, 0.8]$ respectively.

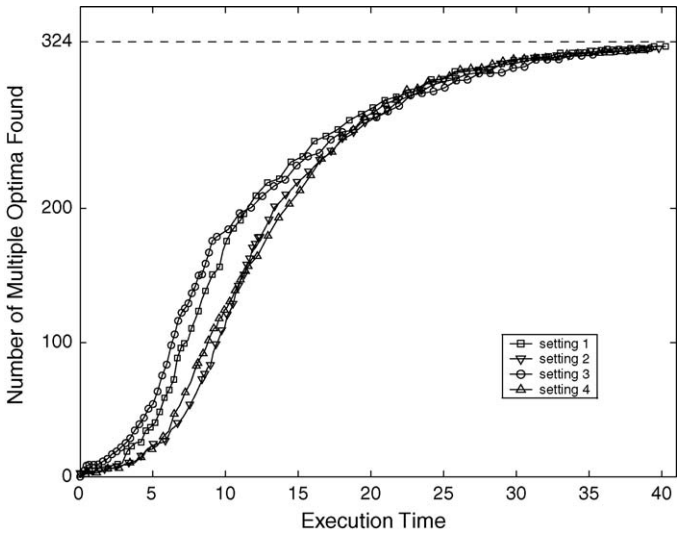

Fig. 19. Variation in the average number of multiple optima found and the execution time with the settings 1–4 for the test function f_{9-4}.

7. Conclusion

In this paper we have presented the adaptive elitist-population search method, a new technique for evolving parallel elitist individuals for multimodal function optimization. The technique is based on the concept of adaptively adjusting the population size according to the individuals' dissimilarity using direction dependent elitist genetic operators.

The adaptive elitist-population search technique can be implemented with any combinations of standard genetic operators. To use it, we just need to introduce one additional control parameter, the distance threshold, and the population size is adaptively adjusted according to the number of multiple optima. As an example, we have integrated the new technique with genetic algorithms, yielding an Adaptive Elitist-population based Genetic Algorithm (AEGA).

AEGA has been experimentally tested with a difficult test suite consisting of 10 benchmark multimodal function optimization examples. The performance of AEGA has been compared against the existing algorithms, Deterministic Crowding, Probabilistic Crowding, Sequential Fitness Sharing, Clearing Procedure, Clustering Based Niching (CBN), Clonal Selection, and Species Conserving Genetic Algorithm (SCGA). All experiments have demonstrated that AEGA generally outperforms the other multimodal evolutionary algorithms in efficiency and solution quality, typically with significant efficiency speedups. The improvements of AEGA is due to two visible features: (*i*) the adaptive elitist-population size is dynamically adjusted to reduce the redundancy of computation complexity; and (*ii*) the diversity of the population is maintained and even improved in term of multimodal optimization. All these support the good performance and power of the technique developed in the present work.

In [33], we have used AEGA to solve an optimal control model of drug scheduling in cancer chemotherapy. To explore multiple efficient drug scheduling policies, we have proposed a novel variable representation—a cycle-wise representation and modified the elitist genetic search operators in AEGA. The simulation results obtained by AEGA match well with the clinical treatment experiences, and can provide multiple efficient alternative solutions for the oncologists to consider. It has been shown that the evolutionary drug scheduling approach is simple and capable of solving complex cancer chemotherapy problems requiring multimodal solutions by adapting AEGA.

We plan to apply our technique to hard multimodal engineering design problems with the expectation of discovering novel solu-

tions. We will also need to investigate the behavior of AEGA on the more theoretical side.

Acknowledgments

This research was partially supported by RGC Research Grant GRF 414708 of Hong Kong SAR and Macau Science and Technology Develop Fund (Grant No. 021/2008/A) of Macau SAR. The authors would like to thank the anonymous reviewers for their constructive comments and suggestions which have significantly improved this paper.

References

[2] T. Bäck, Evolutionary Algorithms in Theory and Practice, Oxford University Press, New York, 1996.
[3] D. Beasley, D.R. Bull, R.R. Matin, A sequential niche technique for multimodal function optimization, Evolutionary Computation 1 (2) (1993) 101–125.
[4] L.N. Castro, F.J. Zuben, Learning and optimization using the clonal selection principle, IEEE Transactions on Evolutionary Computation 6 (2002) 239–251.
[5] L.N. Castro, J. Timmis, An artificial immune network for multimodal function optimization, in: Proc. of the 2002 IEEE Int. Conf. on Evolutionary Computation, IEEE Press, New York, NY, Honolulu, Hawaii, 2002, pp. 699–704.
[6] D.J. Cavicchio, Adaptive search using simulated evolution, Ph.D. thesis, University of Michigan, Ann. Arbor, Michigan, 1970.
[7] W. Cedeño, V. Vemuri, On the use of niching for dynamic landscapes, in: Proc. of the 1997 IEEE Int. Conf. on Evolutionary Computation, IEEE Press, New York, NY, Indianapolis, IN, 1997, pp. 361–366.
[8] P. Cortés, J. Larrañeta, L. Onieva, J.M. García, M.S. Caraballo, Genetic algorithm for planning cable telecommunication networks, Applied Soft Computing 1 (2001) 21–33.
[9] P. Cortés, J. Larrañeta, L. Onieva, Genetic algorithm for controllers in elevator groups: analysis and simulation during lunchpeak traffic, Applied Soft Computing 4 (2004) 159–174.
[10] P. Cortés, J. Muñuzuri, L. Onieva, J. Larrañeta, J.M. Vozmediano, J.C. Alarcón, Andalucía assesses the investment needed to deploy a fiber-optic network, Interfaces 36 (2) (2006) 105–117.
[11] P. Cortés, J.M. García, J. Larrañeta, L. Onieva, Viral systems: a new bio-inspired optimisation approach, Computers & Operations Research 35 (9) (2008) 2840–2860.
[12] L. Costa, P. Oliveira, An adaptive sharing elitist evolution strategy for multiobjective optimization, Evolutionary Computation 11 (4) (2003) 417–438.
[13] K. Deb, D.E. Goldberg, An investigation of niche and species formation in generic function optimization, in: J.D. Schaffer (Ed.), Proc. of the 3rd Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989, pp. 42–50.
[15] K.A. De Jong, An analysis of the behavior of a class of genetic adaptive systems, Doctoral dissertation, University of Michigan, 1975.
[16] D. Dumitrescu, B. Lazzerini, L.G. Jain, A. Dumitrescu, Evolutionary Computation, CRC Press LLC, 2000.
[17] S. Elo, A parallel genetic algorithm on the CM-2 for multi-modal optimization, in: Proc. of the 1994 IEEE Int. Conf. on Evolutionary Computation, IEEE Press, New York, NY, Orlando, FL, 1994, pp. 818–822.
[18] L.J. Eshelman, The CHC adaptive search algorithm: how to have safe search when engaging in nontraditional genetic recombination, in: G.J.E. Rawlins (Eds.), Foundation of Genetic Algorithms-I, 1991, pp. 265–283.
[19] L.J. Eshelman, J.D. Schaffer, Real-coded genetic algorithms and interval-schamata, in: L.D. Whitley (Ed.), Foundation of Genetic Algorithms-2, 1993, pp. 187–202.
[21] J. Gan, K. Warwick, Dynamic Niche Clustering: a fuzzy variable radius niching techniquefor multimodal optimisation, in: Proc. of the 2001 IEEE Int. Conf. on Evolutionary Computation, IEEE Press, 2001, pp. 215–222.
[22] M. Gen, R. Cheng, Genetic Algorithm and Engineering Design, Wiley J. and Sons editors, New York, 1997.
[23] D.E. Goldberg, J. Richardson, et al., Genetic algorithms with sharing for multimodal function optimization, in: J.R. Koza (Ed.), Proc. of the 2nd Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Francisco, CA, 1987, pp. 41–49.
[24] D.E. Goldberg, Genetic Algorithm in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, MA, 1989.
[27] D.E. Goldberg, L. Wang, Adaptive niching via coevolutionary sharing, ILLiGAL Technical Report No. 97007, Illinois Genetic Algorithms Laboratory, University of Illinoise, Urbana, Illinois, 1997.
[28] G.R. Harik, Finding multimodal solutions using restricted tournament selection, in: L. Eshelman (Ed.), Proc. of the 6th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Francisco, CA, 1995, pp. 24–31.
[29] R. Kumar, P. Rockett, Evolutionary multimodal optimization revisited, in: Proc. of Genetic and Evolutionary Computing Conference (GECCO-03), LNCS 2723, 2003, pp. 1592–1593.
[30] R. Kumar, P. Rockett, Effective volutionary multimodal optimization by multiobjective reformulation without explicit niching/sharing, in: Proc. of Asian Applied Computing Conference (AACC), LNCS 3285, 2004, pp. 1–8.
[31] K.S. Leung, Y. Liang, Adaptive elitist-population based genetic algorithm for multimodal function optimization, in: Proc. of Genetic and Evolutionary Computing Conference (GECCO-03), LNCS 2723, 2003, pp. 1160–1171.
[32] J.P. Li, M.E. Balazs, G.T. Parks, P.J. Glarkson, A species conserving genetic algorithms for multimodal function optimization, Evolutionary Computation 10 (3) (2002) 207–234.
[33] Y. Liang, K.S. Leung, S.K. Mok, A novel evolutionary drug scheduling model in cancer chemotherapy, IEEE Transactions on Information Technology in Biomedicine 10 (2) (2006) 237–245.
[34] S.W. Mahfoud, Crowding and Preselection Revisited, IlliGAL Report 92004, University of Illinois at Urbana Champaign, Illinois Genetic Algorithm Laboratory, 1992.
[35] O.J. Mengshoel, D.E. Goldberg, et al., Probability crowding: deterministic crowding with probabilistic replacement, in: W. Banzhaf (Ed.), Proc. of the Int. Conf. GECCO-1999, Orlando, FL, 1999, pp. 409–416.
[37] B.L. Miller, M.J. Shaw, Genetic algorithms with dynamic niche sharing for multimodal function optimization, in: Proc. of the 1996 IEEE Int. Conf. on Evolutionary Computation, IEEE Press, New York, NY, Nagoya, Japan, 1996, pp. 786–791.
[38] H. Mühlenbein, D. Schlierkamp-Voosen, Predictive models for the breeder genetic algorithm I: continuous parametter Optimization, Evolutionary Computation 1 (1993) 25–49.
[39] I.C. Parmee, A review of evolutionary-adaptive search in engineering design, Evolutionary Computation 1 (1) (1999) 13–39.
[40] D. Parrott, X.D. Li, Locating and tracking multiple dynamic optima by a particle swarm model using speciation, IEEE Transactions on Evolutionary Computation 10 (4) (2006) 440–458.
[41] A. Petrowski, A Clearing procedure as a niching method for genetic algorithms, in: Proc. of the 1996 IEEE Int. Conf. on Evolutionary Computation, IEEE Press, New York, NY, Nagoya, Japan, 1996, pp. 798–803.
[43] J. Sarma, J. De, Generation gap methods, Handbook of Evolutionary Computation C2 (7) (1997) 1–5.
[44] W.M. Spears, Simple subpopulation schemes, in: A.V. Sebald, L.J. Fogle (Eds.), Proc. of the 3rd Ann. Conf. on Evolutionary Programming, Singapore, 1994, pp. 157–163.
[47] S. Tsutsui, Y. Fujimoto, Forking genetic algorithms: GAs with search space division schemes, Evolutionary Computation, MIT Press 5 (1) (1997) 61–80.
[48] R.K. Ursem, Multinational evolutionary algorithms, in: Proc. of the 1999 IEEE Int. Conf. on Evolutionary Computation, IEEE Press, 1999, pp. 1633–1640.
[49] D. Whitley, The GENITOR algorithm and selection pressure. why rank-based allocation of reproductive trials is best, in: J.D. Schaffer (Ed.), Proc. of the 3rd Int. Conf. on Genetic Algorithms, Morgan Kaufmann, San Mateo, CA, 1989, pp. 116–121.
[50] X. Yin, N. Germay, A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization, in: Proc. of the 1993 Int. Conf. on Artificial Neural Networks and Genetic Algorithms, 1993, pp. 450–457.
[51] Q.B. Zhang, T.H. Wu, B. Liu, A population-based incremental learning algorithm with elitist strategy, Proc. of the 3rd Int. Conf. on Natural Computation 3 (2007) 583–587.