Hybrid Real-Coded Genetic Algorithm with Quasi-Simplex Technique

Guoli Zhang[†] *,and Haiyan Lu*[††]

[†] *North China Electric Power University, China*
[††] *University of Technology, Sydney, Australia*

Summary
This paper proposes a new real-value mutation operator and a hybrid real-coded genetic algorithm with quasi-simplex technique using this new mutation operator (RCGAQS). Compared with the classical GA (CGA), RCGAQS has the following distinguish features: (1) A new real-value mutation mechanism was used to increase the capability of global search (exploration); (2) The modified simplex technique, so called the quasi-simplex technique, was employed to generate prospective offspring to increase the capability of local search (exploitation); and (3) The dynamic subpopulation strategy, in which the entire generation is subdivided into a number of subgroups in each evolution step, was adopted to enhance the abilities in both exploration and exploitation. RCGAQS algorithm has been implemented and tested on typical benchmark functions along with CGA. The experimental study has shown that RCGAQS is impressive in finding the near global optimal solutions cross all the selected benchmark functions and is substantially robust.

Key words:
Genetic algorithm, Real-coded, Elitist strategy, Quasi- simplex technique

1. Introduction

Conventional optimization methods suffer from the local optimality problem and some of them usually requires the function to have good characteristics, such as differentiability, continuity etc.. To a certain extend, this limits the application of these traditional methods to a small range of real word problems. In recent years, stochastic optimization techniques, such as the simulated annealing (SA), the genetic algorithm (GA), and the evolutionary algorithm (EA), have drawn many researchers' attention due to their capability of finding the near global optimal solutions without putting restrictions on the characteristics of the target functions although they require significant computing power and generally take a fairly long time to reach the solution. A great amount of efforts have been devoted in improving these methods and some of the improved methods have been successfully used in variety of real world problems [1-7].

GA was initially introduced by John Holland in seventies as a special technique for function optimization [8]. Hereafter, we refer to it as the classical GA (CGA). A typical CGA has three phases, i.e., initialization, evaluation and genetic operations, which are consisted of reproduction, crossover and mutation. The performance of CGA precedes the traditional optimization methods in aspect of global search and robustness in handling an arbitrary non-linear function. However, it suffers premature convergence problem and usually consumes enormous computing times.

In the CGA, ability of local search mainly relies on the reproduction and crossover operations, which can be referred to as the exploitation operations, while the capability of global search is assured by the mutation operation, which can be regarded as the exploration operation. Generally speaking, the velocity of local search increases when the probability of crossover increases. Similarly, the level of capability of global search will increase when the probability of mutation increases. Since the sum of probabilities of all the generic operations must be unity, in order to assure a reasonable level of capability of local search, the mutation probability has to be reduced to increase the crossover probability. This contributes to the fact that the probability of mutation in CGA is very low which usually comes in a range of 0.1-5%. On the other hand, to achieve a satisfactory level of capability of global search, the probabilities of reproduction and crossover have to be decreased to increase the mutation probability. This will weaken capability of local search dramatically, slow down the convergence rate and makes the global search ability unachievable eventually. In the process of balancing exploration and exploitation based on reproduction/crossover and mutation operations for a fixed population, it is hardly possible to achieve the win situation for both sides simultaneously. Therefore, how to balance between exploration and exploitation in GA-type algorithms has long been a challenge task and retained its attractiveness to many researchers [9-17]. Majority of the improvements proposed were centralized in three aspects, i.e., decreasing the computing time, reducing premature convergence and enhancing global search capability.

Manuscript received October 5, 2006.
Manuscript revised October 25, 2006.

To reduce computing time, some researchers proposed to use real-coded values instead of binary bit sequences [4-5] [9-11]. Taking this approach, the computation time spent on encoding/decoding has been eliminated, the diversity of mutation, however, is significantly limited. Two typical approaches are commonly used: one is to generate a new random value to replace the existing one and the other is to add up one extra part, which is usually chosen randomly, to the existing one [4-5],[9-11]. None of them preserves the meaning of natural mutation in a sense that the new value has nothing to do with the existing one.

To enhance the capability of global search, a lot of works have been done in increasing the diversity of the population [9-17]. Typical approaches are the sub-grouping approach and the fitness sharing approach. The sub-grouping approach makes use of local information to search optimal and exchanges the information among the sub-groups when the evolution process goes along. From the fitness sharing approach, several methods, such as the fitness sharing, the crowding methods and the clearing methods have been proposed [14-15]. In a typical method of fitness sharing, the similarity levels amongst individuals are measured by the distances and the new fitness of individuals will be re-assigned according to the similarity levels. The higher the similarity level is, the lower the fitness [15-17]. Although there are a number of fitness sharing methods proposed, the main difference is the way in which the distances are calculated. The crowding method is similar to the fitness sharing method. According to Bruno [15], crowding methods can be classified into two groups, one is the standard crowding method and the other is the deterministic crowding method. In the standard crowding scheme, an offspring replaces the most similar individual taken from a randomly drawn subpopulation of size CF (crowding factor) from the global population. The deterministic crowding method introduces competition between children and parents of identical niches. After crossover and eventually mutation, each child replaces the nearest parent if it has a higher fitness. The clearing method is also very similar to the fitness sharing method except that it is based on the concept of limited resources of the environment. Instead of sharing the resources amongst all individuals of a single subpopulation as in fitness sharing, clearing method preserves the fitness of the k best individuals of the niche and resets the fitness of the others that belong to the same subpopulation. All these methods aim at maintaining the diversity of a population by modifying the search landscape by reducing the payoff in densely populated regions to eventually enhance global search capability [17].

Our approach is to enhance the capability of global search by increasing the probability of mutation operation while assuring satisfactory level of capability of local search by employing the idea of simplex method, so called the quasi-simplex technique. From this approach, a new hybrid real-coded genetic algorithm with quasi-simplex technique (RCGAQS) is proposed. RCGAQS aims at the following three points: (1) assuring the capability of global search by increasing the probability of mutation, (2) mutation is implemented by using an effective real-value mutation operator instead of traditional binary mutation; and (3) enhance the capability of local search by introducing the so-called quasi-simplex techniques into the CGA since the capability of local search will be significantly weaken when the probability of reproduction/crossover decrease as a result of increasing the probability of mutation. In each iteration, RCGAQS firstly divides the population into a number of subpopulations and each subpopulation is treated as a classical simplex, then for every simplex, RCGAQS applies fours operations in parallel to produce offspring. The first operation is the quasi-simplex evolution in which two prospective individuals will be chosen to be offspring. The other three operations are the reproduction, crossover and mutation, respectively, which are very similar to the traditional genetic operation except that the probability of mutation is fairly high. All these four operations together will produce a new sub-population with the same size as the corresponding parent sub-group. The new generation is the collection of all the newly produced sub-groups. In brief, RCGAQS maintain the diversity of a population to eventually enhance global search capability because a higher diversity of population leads to a higher level of capability to explore the search space while the local search are mainly implemented by quasi-simplex technique, reproduction including the elitist strategy and crossover operations.

The rest of this paper is organized as follows. Section 2 describes briefly the global minimization problem described by Xin Yao and Yong Liu [18] and the idea of simple algorithm. Section 3 presents a new hybrid genetic algorithm with quasi-simplex technique, the new real-coded mutation operator and dynamic subpopulation in detail. In section 4, the experimental study and verification of the proposed new algorithm are focused. The comparison of the experimental results of the CGA and the proposed new method is also depicted across over a set of popular benchmark functions. Finally, the conclusions are drawn in Section 5.

2. Function Optimization and quasi-simplex technique

Consider the global minimization problem described by Xin Yao and Yong Liu [18] for the purpose of development of new search algorithm. According to Yao and Liu, the

problem can be formalized as a pair of real valued vectors (s, f), where $S \subseteq R^n$ is a bounded set on R^n and $f : S \to R$ is an n-dimensional real-valued function. f needs not to be continuous but it must be bounded. The problem is to find a point such that $f(x_{\min})$ is a global minimum on S. More specially, it is required to find an $x_{\min} \in S$ such that

$$\forall x \in S, f(x_{\min}) \leq f(x) \tag{1}$$

On solving above optimization problem by genetic algorithm, one effective method, which can speed up local convergence rate, is to combine the CGA with conventional optimization methods. Since it has been highly recognized that GA has no special request on the characteristics of the objective functions as one of its merits, the conventional optimization methods that can go with GA should not require that the objective functions having special characteristics neither. In this light, Simplex method is promising because it demands less on the function characteristics. Therefore, we choose to combine the conventional GA with simplex technique to form a hybrid generic algorithm in which a real-value scheme and dynamic sub-grouping are used. This algorithm is referred to as the hybrid real-coded genetic algorithm with quasi-simplex technique (RCGAQS).

For the ease of understanding of RCGAQS algorithm, let us recall briefly the basic idea of simplex technique. Simplex is a kind of direct search method, which is a widely accepted search technique. A simplex in an n-dimensional space is defined by a convex polyhedron consisting of $n+1$ vertices, which are not in the same hyper-plane. Assume that there are n+1 individuals, denoted by x^i, and their function values are denoted as f_i, i = 1,2,…, n+1. The worst and the best points in terms of function values are denoted by x^H and x^B, respectively, and can be determined by

$$f(x^H) = f_H = \max_i f_i \quad \text{i = 1,2,…, n+1} \tag{2}$$

and

$$f(x^B) = f_B = \min_i f_i \quad \text{i = 1,2,…, n+1} \tag{3}$$

where f_H and f_B denotes the worst and the best function values, respectively.

To determine a new better point than the worst point x^H, the centroid x^C of the polyhedron whose vertexes are all the points but the worst one needs to be calculated by

$$x^C = \left(\left(\sum_{i=1}^{n+1} x^i \right) - x^H \right) / n \tag{4}$$

The better point predicted by simplex techniques lies on the line starting from the worst point and towards the centroid, which can be referred to as the worst-opposite direction, and the actual location can be determined by the following formula:

$$x = x^C + \alpha(x^C - x^H) \tag{5}$$

where α is a constant and can be different values for different points lying on the worst-opposite direction, such as the reflection point, expansion points, or the compression points. The actual value ranges of α for different points are shown in table 1.

Conventional simplex techniques are mainly consisted of four operations, i.e., the reflection, the expansion, the compression, and the contraction operations. The simplex algorithm produces a new simplex either by replacing the worst point by a better point produced using the simplex techniques or contract the current simplex towards the best point in each iteration step. The process will be continuing until the termination criterion is satisfied. The crucial idea of the classical simplex techniques is to approach the local optimal following the worst-opposite direction of each simplex, which can be regarded as a kind of guidance in the search landscape. Therefore, simplex algorithm has a higher level of ability of local search.

Table 1: Points obtained using the simplex techniques with different value of α

$x = x^c + \alpha(x^c - x^H)$	$\alpha = 1$	reflection point	Reflection point of x^H respect to x^C
	$\alpha > 1$	Expansion point	A point farther than the reflection point from x^C
	$1 > \alpha > 0$	Compression point	Points between x^C and reflection point
	$0 > \alpha > -1$	Compression point	Points between x^H and x^C

3. Hybrid Real-Coded GA with Quasi-Simplex Technique

3.1 Simulation Hybrid Real-Coded GA

In the CGA, local exploitation relies heavily on reproduction and crossover operation while global exploration mainly makes use of mutation operation. Since the summation of probabilities of reproduction, crossover and mutation operations should be unity, it is hardly possible to achieve satisfactory results in both local exploitation and global exploration in CGA. The major problems are low convergence rate and premature. It has been a key issue in GA-type algorithms how to balance exploration and exploitation.

The hybrid real-coded GA (RCGAQS) circumvents the difficulties in CGA by combining a technique evolved from the traditional simplex technique, which is referred to as quasi-simplex technique, with the CGA. In doing so, RCGAQS can achieve substantially high level of global exploration by increasing the probability of mutation while its capability of local exploitation can also be reasonable high by using both reproduction/crossover and quasi-simplex techniques.

The process of RCGAQS works can be described as follows: Firstly, RCGAQS initializes a random-generated population with μ individuals (real-coded chromosomes) and each individual has n components. Then the population starts to evolve. At the beginning of each iteration, the generation is divided into a number of subpopulations (or subgroups) with each subgroup having n+1 individuals. Each subgroup will then evolve into a new subpopulation of the same size by four operations in parallel, which are the quasi-simplex operation, reproduction, crossover and mutation. The quasi-simplex operation (QS) will generate two new individuals, the reproduction will retain the best individual by applying the elitist strategy and also produce some individuals based on the probability of reproduction (R), the crossover operation will also generate a number of pairs of individuals according to the probability of crossover (C) and the left-over individuals will be produced by mutation (M). At the end of each iteration, all the new individuals from the subpopulations will merge together and evolution enters into a new generation. If the termination criterion is not met, evolution starts a new iteration. This process continues until the termination criterion satisfied. The population in the final generation will be the solution.

3.2 Highlights of RCGAQS

RCGAQS has a number of outstanding features which equip it with capability of both local exploitation and global exploration. Firstly, RCGAQS adopts the dynamic sub-grouping idea to ensure each simplex consists of reasonably correlated individuals in the entire evolution process to enhance the convergence rate. RCGAQS implements population partition differently from the proposed strategies proposed in literature by two aspects: One is to take into account the dimension of individuals in deciding the number of subgroups. RCGAQS divides a population into a number of subgroups with each subgroup consisting of n+1 individuals to ensure the validity of search and efficiency in terms of computing times. The detailed discussion about the size of a subpopulation and how many subpopulations should be used will be presented in another paper. And the other is to do partition for each iteration. Although the computation time for each iteration may increase due to the partition process, the enhancement in convergence rate could decrease the number of iteration needed

Secondly, RCGAQS employs the quasi-simplex technique with ancillary reproduction and crossover operation to assure the local exploitation. The quasi-simplex technique absorbs the idea of classical simplex techniques to perform a guided search. It produces four prospective individuals using the reflection, expansion and compression operations along with the worst-opposite direction. The quasi-simplex technique also expands the conventional simplex technique by looking at the prospective individuals lying on a line starting from the centroid and towards the best point of the simplex. We refer to this direction as the best-forward direction in contrast with the worst-opposite direction. Three prospective individuals x^e, x^m and x^n will be produced along the best-forward direction by the expansion and compression operations using the following formula.

$$x = x^B + \beta(x^B - x^D) \tag{6}$$

where x^D denotes the centroid of the remaining points except for the best point x^B and can be calculated by

$$x^D = \left(\left(\sum_{i=1}^{n+1} x^i\right) - x^B\right) \Big/ n \tag{7}$$

The points x^e, x^m and x^n can be determined by the value of β in (6) and corresponding the range of β are shown in the table 2.

In order to avoid a situation in which too many individuals are similar so that the diversity of the population decreases dramatically, RCGAQS selects the best one from the two prospective individual groups along the worst-opposite and the best-forward directions to produce two new individuals as a part of offspring.

Table 2: Range of β in (6) for x^e, x^m and x^n

Formula	Range of β	Calculated point
$x^D = \left(\left(\sum_{i=1}^{n+1} x^i\right) - x^B\right) \Big/ n$	$\beta > 1$	x^e
	$\beta = 1$	x^m
	$0 < \beta < 1$	x^n

Thirdly, RCGAQS uses a new real-coded mutation operator. Generally speaking, mutation operation is mainly used for enhancing the diversity of a population. In CGA, each individual is represented by a chromosome, which is a sequence of genes in genetics and usually represented by a string of 0s and 1s in computation. Mutation operation converts 0 to 1 or 1 to 0 for a specific bit if that bit is meant to mutate. As mentioned in Section 1, encoding and decoding individuals may consume significant computing time, therefore, real-coded mutation scheme in which an individual is represented by a decimal number is highly desirable and preferable. The common practice in real-coded mutation is to replace a gene (a decimal digit) by a new random number or add a random number to the existing gene. Although it works under certain circumstances, this practice lacks of mechanics of natural genetics. RCGAQS uses a new mutation operation which is proposed based on the idea of mutation in natural genetics. This mutation operation converts a big number (>= 5) to a small number (< 5) or a small number into a big number.

Let x_{ij}, $i = 1, 2, \cdots, \mu$, $j = 1, 2, \cdots, n$, denotes the j-th component of the i-th individual in the population, where μ is the size of a population and n is the dimension of each individual. Under the real-coded scheme, x_{ij} can be represented by a sequence of decimal numbers including the decimal point as

$$x_{ij} = d_{ij}^{w_1} d_{ij}^{w_2} \ldots d_{ij}^{w_p} \square d_{ij}^{f_1} d_{ij}^{f_2} \ldots d_{ij}^{f_q} \tag{8}$$

where superscript w and f denote the whole number part and the fractional part respectively, and p and q are constants representing the number of digits in the whole number part and the fractional part for a given x_{ij}, respectively. In practice, q is determined by the precision required by the problems with maximum value being determined by the hardware used in computing. During the mutation operation, firstly, a mask will be generated for each x_{ij}, $i = 1, 2, \cdots, \mu$, $j = 1, 2, \cdots, n$. The mask is a sequence of binary bits with length being the same as x_{ij}'s excluding the decimal point. Let $m = b^{w_1} b^{w_2} \cdots b^{w_p} b^{f_1} b^{f_2} \cdots b^{f_q}$ denotes the mask. If $m(b^r)=0$, the r-th number will not change and if $m(b^r)=1$, the r-th number will mutate. After the mutation, the result $\overline{x}_{ij}$ can be obtained using the new mutation operation as

$$\overline{x}_{ij} = \overline{d}_{ij}^{w_1} \overline{d}_{ij}^{w_2} \cdots \overline{d}_{ij}^{w_p} \square \overline{d}_{ij}^{f_1} \overline{d}_{ij}^{f_2} \cdots \overline{d}_{ij}^{f_q} \tag{9}$$

where

$$\overline{d}_{ij}^{r} = \begin{cases} d_{ij}^{r} & m(b^r) = 0 \\ 9 - d_{ij}^{r} & m(b^r) = 1 \end{cases} \tag{10}$$

where $r = w_1, w_2, \cdots, w_p, f_1, f_2 2, \cdots f_q$.

3.3 RCGAQS algorithm procedure

The RCGAQS algorithm procedure can be outlined in the following steps:

Step 1 Initialize a random population of size μ=K(n+1), X.

Step 2 Divide the population into K subpopulations with each subgroup consists of n+1 individuals.

Step 2.1 Select the best individual of the population x.

Step 2.2 Select n individuals which are most close to x in terms of their Euclid distances

Step 2.3 Combine the individuals obtained from steps 2.1 and 2.2 to form a subpopulation S;

Step 2.4 Remove S from the original population.

Step 2.5 Repeat Steps 2.1 – 2.4 for other subpopulations, until no individuals are left.

Step 3 Each subpopulation evolves into a group of new individuals

Step 3.1 Produce two new individuals using quasi-simplex technique.

Step 3.2 Implement elitist strategy, i.e., reserve the best one in the subpopulation to be a part of offspring

Step 3.3 Produce new individuals by reproduction using linear ranking. The probability of the i-th individual, x^i, in the targeting subgroup (sorted into an order of descending in fitness) calculated by following formula

$$P_i = \frac{1}{n+1}\left(\eta - 2(\eta-1)\cdot\frac{rank(x_i)-1}{n}\right) \quad (11)$$

where $\eta > 1$, which can be determined by the desired probability of the best individual

Step 3.4 Crossover operation is processed as follows:

Select $[(n-2)P_C/2]$ pair of parents randomly, where $[\square]$ is an operator producing the maximum integer which is less than the operand; for every pair of parent selected,

$$x^i = (x_1^i, x_2^i, ..., x_{m1}^i, ..., x_{m2}^i, ..., x_n^i) \quad (12)$$

$$x^j = (x_1^j, x_2^j, ..., x_{m1}^j, ..., x_{m2}^j, ..., x_n^j) \quad (12')$$

where subscripts i and j denotes the i-th and the j-th individual in the population, respectively and the subscript m1 and m2 are two random numbers between 1 and n. The two new individuals will be

$$x_{new}^i = (x_1^i, x_2^i, ..., x_{m1}^j, ..., x_{m2}^j, ..., x_n^i) \quad (13)$$

$$x_{new}^j = (x_1^j, x_2^j, ..., x_{m1}^i, ..., x_{m2}^i, ..., x_n^j) \quad (13')$$

Step 3.5 Mutation operation. The remained individuals will participate the mutation operation. For each individual, a mask will be generated and a new individual will be produced by using (10).

In summary, this algorithm assures the capability of local search by combining the traditional genetic operations (reproduction and crossover) with the quasi-simplex techniques and the capability of global search by using high probability mutation operation.

4. Experimental Verification

4. 1 Test Functions

In order to test the effectiveness of the proposed new algorithm in terms of the global search capability and the convergence rate, six typical benchmark functions were chosen from the set of 23 benchmark functions listed in [18] and re-numbered as f_1 to f_6. The definitions of these six functions are depicted in Table 3. The coefficients a_i and c_i in the functions f_5 and f_6 with imax = 5 in f_5 and imax = 10 in f_6 are shown in Table 4.

Table 3: Definitions of test benchmark functions

No	Test function	n	Domain	Minimum value fmin
1	$f_1 = \sum_{i=1}^{n} x_i^2$	30	[-100, 100]30	0.0
2	$f_2 = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[-100, 100]30	0.0
3	$f_3 = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i) + 20 + e$	30	[-32, 32]30	0.0
4	$f_4 = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$	30	[-600, 600]30	0.0
5	$f_5 = -\sum_{i=1}^{5}[(x-a_i)(x-a_i)^T + c_i]^{-1}$	4	[0,10]4	-10.0*
6	$f_6 = -\sum_{i=1}^{10}[(x-a_i)(x-a_i)^T + c_i]^{-1}$	4	[0,10]4	-10.0*

Note: * given in [18], but we found they may not be the actual minimum values for the given functions.

Table 4 coefficients of functions f_5 and f_6

i	ai=(ai1,ai2,ai3,ai4)				ci
	j=1	j=2	j=3	j=4	
1	4	4	4	4	0.1
2	1	1	1	1	0.2
3	8	8	8	8	0.2
4	6	6	6	6	0.4
5	3	7	3	7	0.4
6	2	9	2	9	0.6

7	5	5	3	3	0.3
8	8	1	8	1	0.7
9	6	2	6	2	0.5
10	7	3.6	7	3.6	0.5

Among the test functions, function f_1 and f_2 are typical unimodal functions, f_3 and f_4 are multimodal functions with many local minima, and f_5 and f_6 are multimodal functions with only a few local minima. As claimed by Xin Yao in [18], these functions are challenging to every search algorithm. Figs.1-6 illustrate the plots of these functions with dimension n = 2. For the functions f_3 and f_4 as shown in Figs. 3(a) and 4(a), the enlarged plots are also shown in Figs. 3(b) and 4(b) to depict the local minima. For the functions f_5 and f_6, as shown in Figs.5(a) and 6(a), the plots of two other functions - f_5 and - f_6 are also plotted in Figs. 5(b) and 6(b) to show the local optimal points in f_5 and f_6 clearly.

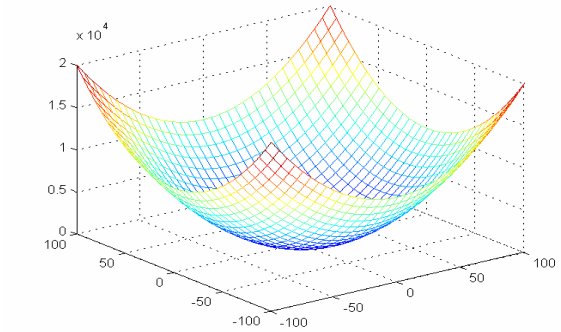


Fig.1 Benchmark function f_1

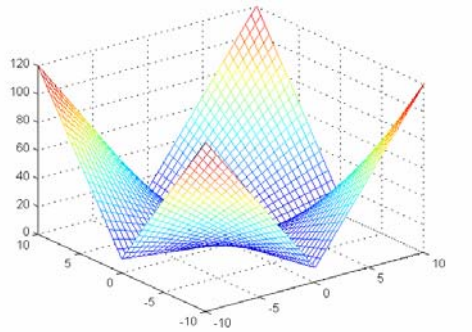
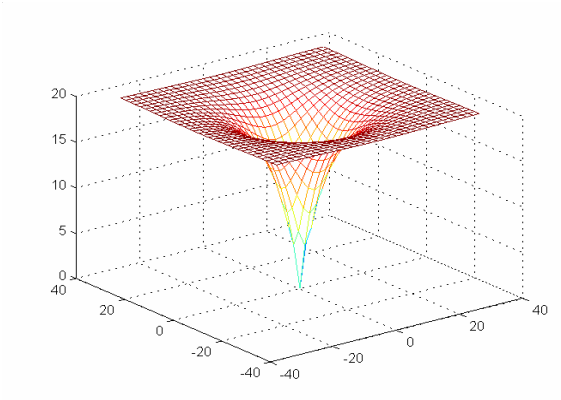
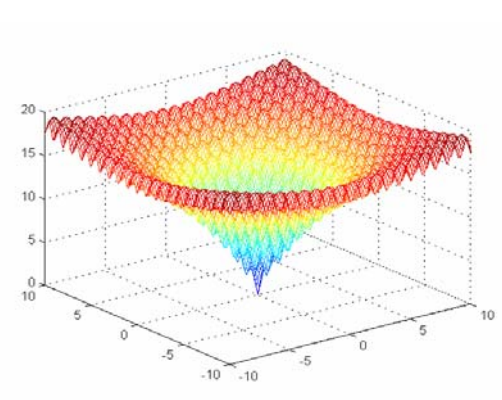


Fig.2 Benchmark function f_2

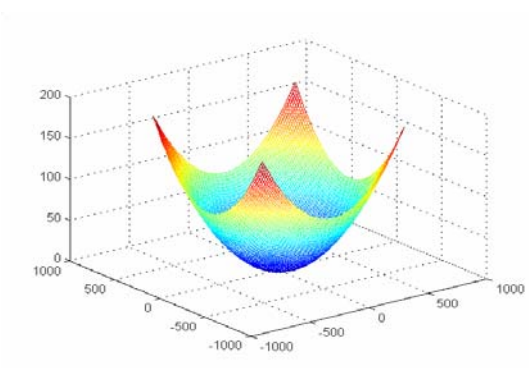


(a) With the given domain range

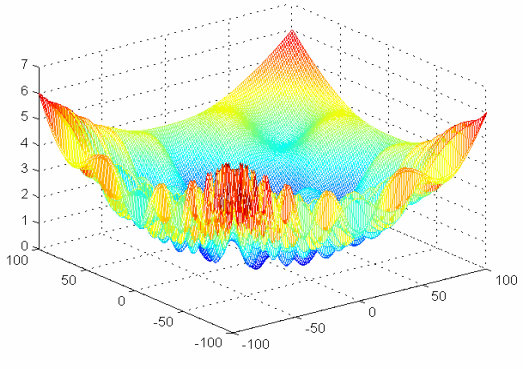


(b) enlarged with smaller scales

Fig.3 Benchmark function

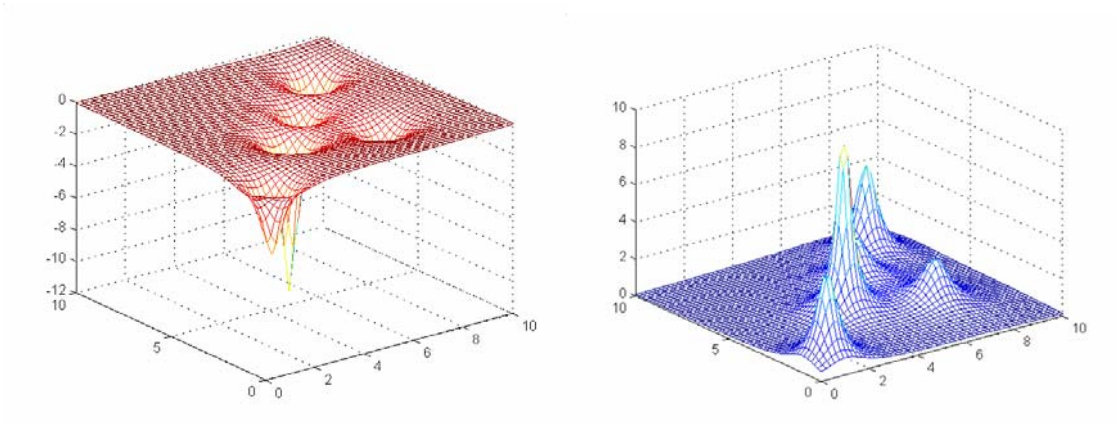


(a) With the given domain range



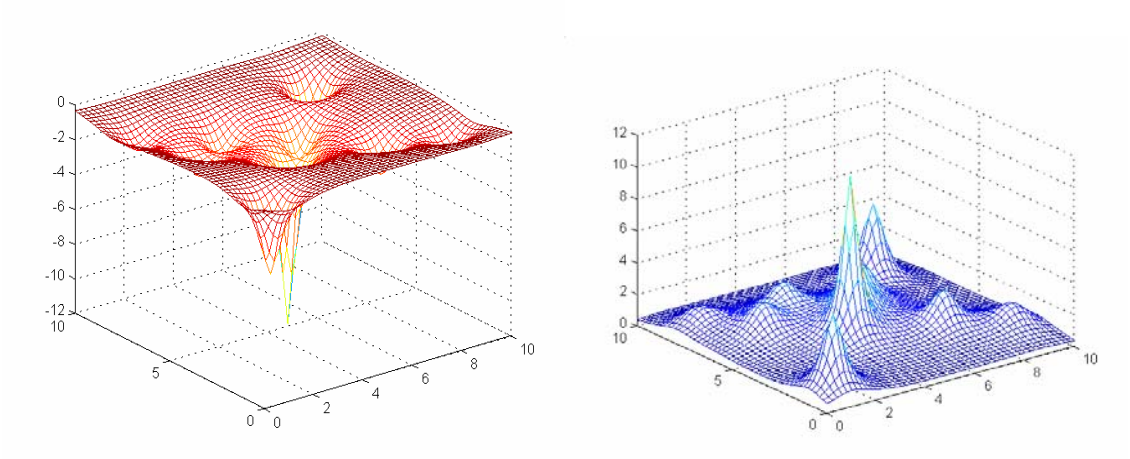
(b) enlarged with smaller scales

Fig.4 Benchmark function f_4



(a) f_5 (b) $-f_5$

Fig. 5 Benchmark function f_5



(a) f_6 (b) $-f_6$

Fig.6 Benchmark function f_6

4.2 Test Results

The two algorithms use same parameters. The size of population is $\mu = K(n+1)$. Table 3 tabulates the results obtained by running 10 times of the program. The experimental results show the proposed algorithm RCGAQS is more effective than the CGA cross over all the test benchmark functions in terms of both the mean function value and the standard derivation with the same inputs.

Another finding is that the last two functions, which have a few local minima, have smaller function values than the ones given in [18]. The averaged minimal values for the functions f_5 and f_6 of 10 runs are –10.13 and –10.40 using RCGAQS, respectively.

Table3：Comparison between RCGAQS and CGA on functions $f_1 - f_6$ listed in Table 2

Function	K	n	Number of Generation	RCGAQS		CGA	
				Mean Best	Std Dev	Mean Best	Std Dev
f1	10	10	3000	3.2030e-29	1.0128e-28	9.6979e+0	3.8623e-1
f2	10	10	3000	3.2377e-6	6.8730e-6	1.0508e+1	6.8703e-1

f3	10	10	3000	3.9697e-1	6.5006e-1	3.6244e+0	1.9395e-3
f4	15	10	3000	3.4927e-2	1.7985e-2	7.8300e-1	6.3444e-2
f5	10	4	3000	-1.0131e+1	1.2506e-2	-7.7584e-3	1.9784e-4
f6	10	4	3000	-1.0395e+1	5.4299e-3	-2.1031e-2	1.7364e-3

Note:
- All results have been averaged over 10 runs, where "Mean Best" indicates the mean best function values and "Std Dev" stands for the standard deviation.

5. Conclusions and Discussion

A new search algorithm RCGAQS algorithm for non-linear function optimization has been proposed based on the classical genetic algorithm and quasi-simplex techniques with a new mutation operator. RCGAQS uses real-value for the representation of individuals (chromosomes) to eliminate the encoding/decoding phases and the difficulty in determining the maximum length of binary sequences. RCGAQS partitions each population into subpopulations in each iteration to increase the diversity of populations. Each subpopulation consists of n+1 individuals following the rule of nearest, i.e., amongst the individuals, one is the best individual locally and the others are closer to the best individual than other individuals out this subpopulation. The introduction of quasi-simplex techniques into genetic algorithm makes it possible to assign a higher probability for mutation to enhance the global exploration because the quasi-simplex can sufficiently make use of local information of the subpopulation to speed up local search velocity. RCGAQS makes a good balance between local exploitation and global exploration by allowing more individuals to mutate into new individuals scattered in the landscape globally and once a better individual is found, it will quickly search around the individual to find the local optimal using quasi-simplex techniques on top of reproduction and crossover. Also it could take all related factors, such as simplicity, accuracy, computing load, convergence rate and global search capability, into account. The experimental results show that RCGAQS could find global optimal across all the benchmark functions selected to be representatives of singlemodel, multimodel with many optimal and multimodel with few optimal. More importantly, the standard derivations show that RCGAQS is independent of the initial population. Finally, RCGAQS found the smaller function values for the benchmark functions f_5 and f_6 than the ones claimed in the literature [18].

Acknowledgments

This work is supported by the Research Foundation of North China Electric Power University. The first author would like to thank the Faculty of Information Technology (FIT), University of Technology, Sydney (UTS), Australia, for supporting him to work in Sydney on this study.

References

[1] C. H. Im, H. K. Jung and Y. J. Kim, "Hybrid genetic algorithm for electromagnetic topology optimization," IEEE Trans. Magn., Vol. 39, pp. 2163-2169, 2003.
[2] E. Gil, J. Bustos and H. Rudnick, "Short-term hydrothermal generation scheduling model using a genetic algorithm," IEEE Trans. Power Syst., Vol. 18, pp.1256 – 1264, 2003.
[3] J. M. Arroyo and A.J. Conejo, "A parallel repair genetic algorithm to solve the unit commitment problem," IEEE Trans. Power Syst., Vol. 17, pp. 1216-1224, 2002.
[4] T. Yalcinoz and H. Altun, "Power economic dispatch using a hybrid genetic algorithm," IEEE Power Eng. Rev.,.Vol. 21, pp. 59-60, 2001.
[5] R. M. Ramos, R. R. Saldanha, R. H. C. Takahashi and F. J. S. Moreira, "The real-biased multiobjective genetic algorithm and its application to the design of wire antennas," IEEE Trans. Magn., Vol. 39, p. 1329-1332, 2003.
[6] W. Ziomek, M. Reformat and E. Kuffel, "Application of genetic algorithms to pattern recognition of defects in GIS," IEEE Trans. Dielect. Elect. Insulation, Vol. 7, pp.161-168, 2000.
[7] S. Chakravarty, and R.Mittra, "Application of the micro-genetic algorithm to the design of spatial filters with frequency-selective surfaces embedded in dielectric media," IEEE Trans. Eletromagn. Compat., Vol. 44, pp338-346, 2002.
[8] J. H. Holland, "Adaptation in Natural and Artificial Systems". Ann Arbor, MI: Univ. Michigan Press, 1975.
[9] F. Herrera and M. Lozano, "Gradual distributed real-coded genetic algorithms," IEEE Trans. Evol. Comput., Vol. 4, pp. 43-63, 2000.
[10] F. Herrera and M. Lozano, "Heterogeneous distributed genetic algorithms based on the crossover operator," in Proc. GALESIA, 1997, pp. 203 – 208.
[11] E. Alba, F. Luna and A. J. Nebro, "Parallel heterogeneous genetic algorithms for continuous optimization," in Proc. Parallel and Distributed Processing Symposium, 2003, pp. 7.

[12] S. Choi and C. Wu, "Partitioning and allocation of objects in heterogeneous distributed environments using the niched Pareto genetic-algorithm," in Proc. Asia Pacific Software Engineering Conference, 1998, pp. 322 – 329.
[13] T. Kuo and S. Y. Hwang, "A genetic algorithm with disruptive selection," IEEE Trans. Syst., Man, Cybern. B, Vol. 26, pp. 299-307, 1996.
[14] M. Cioffi, A. Formisano and R. Martone, "Distributed niching concept for electromagnetic shape optimization by genetic algorithm," in Proc. PARELEC, 2000, pp. 186 – 190.
[15] B. Sareni and L. Krahenbuhl, "Fitness sharing and niching methods revisited," IEEE Trans. Evol. Comput., Vol. 2, pp. 97-106, 1998.
[16] P. Darwen and X. Yao, "A dilemma for fitness sharing with a scaling function," in Proc. IEEE Evolutionary Computation, Vol. 1, 29, 1995, pp.166.
[17] A. Petrowski, "A clearing procedure as a niching method for genetic algorithms," in Proc. IEEE Evolutionary Computation, 1996, pp. 798 – 803.
[18] X. Yao and Y. Liu, "Evolutionary Programming Made Fast," IEEE Trans. Evol. Comput., Vol. 3, pp. 82-102, 1999.

Guoli Zhang received the B.S. degree in science from Jilin University in 1982, M.S. degrees in science from Hebei University in 1986, and Dr. degree in engineering from North China Electric Power University in 2006. During 2003-2004, he stayed in University of Technology, Sydney in Australia as visiting scholar. He has been a professor at North China Electric Power University since 1999. His research interest includes soft computation and applications, electric Power Market.