Applied Mathematics and Computation 203 (2008) 598–607

Contents lists available at ScienceDirect

Applied Mathematics and Computation

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

Modifications of real code genetic algorithm for global optimization

Ioannis G. Tsoulos

Department of Computer Science, University of Ioannina, Ioannina 45110, Greece

ARTICLE INFO

Keywords:
Global optimization
Genetic algorithms
Stochastic procedure
Stopping rules
Mutation

ABSTRACT

A series of modifications to the real coded genetic algorithm for the task of locating the global minimum of multidimensional functions are presented here. These modifications include a new stopping rule, a novel mutation mechanism and a periodically application of a local search procedure. The proposed modifications are tested on a series of optimization problems and the results are reported.

1. Introduction

The problem of locating the global minimum of a multidimensional function $f(x) : \Omega \subset R^n \to R$ with $x_i \in [l_i, r_i], i = 1, \ldots, n$ finds application in many scientific fields such as physics [1,2], astronomy [3,4], chemistry [5,6], economics [7,8] etc. During the past years many methods have been proposed to tackle this problem. These methods can be divided into two main categories: deterministic and stochastic. The methods of the first category are more difficult to be implemented and they depend on a priori information about the objective function. On the other side, the methods of the second category are more general and they are implemented more easily. Some examples of stochastic methods are: Adaptive Random Search [9], Completive Evolution [10], Controlled Random Search [11], Simulated Annealing [12–15], Genetic Algorithms [16,17], Differential Evolution [18], Particle Swarm Optimization [19], etc.

The genetic algorithm is a biologically inspired global optimization technique which is based on natural selection, reproduction and mutation. This technique works by creation of a population of candidate solutions (chromosomes) that are evolved through the so called genetic operations of selection, crossover and mutation until some stopping criteria are met. The technique is general enough and it has been used with success in many scientific and practical fields such as combinatorial problems [20], neural network training [21,22], electromagnetics [23], design of water distribution networks [24], etc. Also, it can be parallelized very easily and many methods that utilize parallel genetic algorithms have been proposed in the relevant literature [25–27]. Although, genetic algorithms suffer from some disadvantages such as premature or slow convergence to the global minimum. In order to overcome these disefficiencies many variations have been proposed such as intelligent initialization procedures to ensure diversity in the initial population [28,29], adaptation of control parameters [30,31], improved genetic operations [32,30,31,33–35,39], new stopping rules [36], hybrid schemes in conjunction with other stochastic methods [37,38], etc. This article focuses on the enhancement of genetic algorithms by proposing three modifications namely: (a) a new stopping rule, (b) a new mutation scheme and (c) a periodical application of a local search procedure.

The rest of this article is organized as follows: in Section 2 the basic real coded genetic algorithm is explained in detail, in Section 3 the proposed modifications are described, in Section 4 the test problems are listed accompanied with the experimental results and in Section 5 some conclusions are derived.

E-mail address: itsoulos@cs.uoi.gr

0096-3003/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.amc.2008.05.005

2. The basic genetic algorithm

The basic genetic algorithm used in this paper is a genetic algorithm due to Kaelo and Ali [39] and more specific the algorithm GA(c_{r1}, l), which is described in Algorithm 1. In the following the main parts of the algorithm GA(c_{r1}, l) are outlined.

2.1. Termination check

The algorithm terminates when

$$|f_h - f_l| \leqslant e \text{ OR iter} > \text{ITERMAX}, \tag{1}$$

where f_l denotes the function value of the best chromosome in the population, f_h stands for the function value of the worst chromosome in the population, iter is the current number of generations and ITERMAX denotes the maximum number of generations allowed.

2.2. Selection and crossover

The selection of two parents $x = (x_1, x_2, \ldots, x_n), y = (y_1, y_2, \ldots, y_n)$ for crossover is performed using the well known technique of tournament selection. After the selection of the parents, the offsprings $\tilde{x}$ and $\tilde{y}$ are created using the following scheme:

$$\begin{aligned} \tilde{x}_i &= a_i x_i + (1 - a_i) y_i, \\ \tilde{y}_i &= a_i y_i + (1 - a_i) x_i, \end{aligned} \tag{2}$$

where a_i are random numbers in $[-0.5, 1.5]$ [17].

Algorithm 1. The simple genetic algorithm proposed by Caelo and Ali

```
• Step 1 (initialization):
  – Generate $N$ uniformly distributed random points (chromosomes) in $\Omega$ and store them to the set $S$.
  – Set iter = 0
• Step 2 (evaluation): Evaluate the function value of each chromosome.
• Step 3 (termination check): If termination criteria are hold terminate.
• Step 4 (genetic operations):
  – Selection: Select $m \leqslant N$ parents from $S$.
  – Crossover: Create $m$ new points (offsprings) from the previously selected parents.
  – Mutation: Mutate the offsprings produced in the crossover step with probability $p_m$.
• Step 5 (replacement): Replace the $m$ worst chromosomes in the population with the previously generated offsprings.
• Step 6 (local technique): Create using the local technique procedure a trial point $\tilde{x}$. If $f(\tilde{x}) \leqslant f(x_h)$ where $x_h$ is the current
  worst point in $S$, then replace $x_h$ by $\tilde{x}$.
• Step 7:
  – Set iter = iter + 1
  – goto step 2
```

2.3. Mutation procedure

The mutation scheme of the Algorithm 1 is the following: let $x = (x_1, x_2, \ldots, x_n)$ be the chromosome to be mutated where x_i is the element to be changed during the mutation procedure. The resulting element x_i' is calculated by

$$x_i' = \begin{cases} x_i + \Delta(\text{iter}, r_i - x_i), & t = 0, \\ x_i - \Delta(\text{iter}, x_i - l_i), & t = 1, \end{cases} \tag{3}$$

where t is a random number that takes either the values 0 or 1 and $\Delta(\text{iter}, y)$ is given by

$$\Delta(\text{iter}, y) = y\left(1 - r^{\left(1 - \frac{\text{iter}}{\text{ITERMAX}}\right)}\right), \tag{4}$$

where r is a random number in $[0, 1]$ and b is a user defined parameter that controls the magnituted of change for element x_i.

2.4. Local technique

The proposed local technique creates trial points, in order to replace the worst point in the population. This technique helps to concentrate the points in S around the global minimum. The steps of the local technique are the following:

1. Select a random point y from S.
2. Construct a trial point $\tilde{x}$ using the formula
$$\tilde{x}_i = (1+\gamma_i)x_{l,i} - \gamma_i y_i, \quad i = 1,\ldots,n, \tag{5}$$
where γ_i is a random number in $[-0.5, 1.5]$ and $x_{l,i}$ is the ith component of the best chromosome x_l.
3. Replace the worst point x_h in S with $\tilde{x}$, if $f(\tilde{x}) \leqslant f(x_h)$.

3. The proposed modifications

The proposed modifications are consisted of a new stopping rule, a novel mutation mechanism and a periodically application of a local optimization procedure. The modifications are explained in the following sections.

3.1. Proposed stopping rule

The stopping rule of Kaelo and Ali focuses on the difference between the best and the worst chromosome in the population in order to decide for termination. The algorithm terminates when $|f_h - f_l| \leqslant e$, but this decision can be postponed in many cases, even though the algorithm has managed already to discover the global minimum. This paper proposes an additional termination check that is based on the observation of the variance of the best discovered value f_l. At every generation denoted by iter, the variance $\sigma^{(\text{iter})}$ of f_l is recorded. If there is not any improvement for a number of generations, it is highly possible that the global minimum is already found and hence the algorithm should terminate. The new stopping rule is used in conjunction with the stopping rule of Eq. (1) and hence the algorithm terminates when

$$|f_h - f_l| \leqslant e \text{ OR } \sigma^{(\text{iter})} \leqslant \frac{\sigma^{(\text{last})}}{2} \text{OR iter} > \text{ITERMAX}. \tag{6}$$

The quantity last denotes the generation where the current best value f_l was discovered for the first time.

3.2. The proposed mutation mechanism

The proposed mutation mechanism is based on the update of the velocity in Particle Swarm Optimization methods. The mutation mechanism is implemented as follows: suppose that the element x_i of the chromosome $x = (x_1, x_2, \ldots, x_n)$ is to be mutated. The new element x_i' is calculated using the following equation:

$$x_i' = c_1 r_1 \left(x_i^b - x_i\right) + c_2 r_2 \left(x_l^b - x_i\right), \tag{7}$$

where the parameters c_1 and c_2 are two positive constants (acceleration coefficients), r_1 and r_2 are random numbers in the range $[0, 1]$ and the vector x^b is a copy of the best so far position of chromosome x (i.e. the position with the lowest function value).

3.3. Application of local search

A local search procedure is applied to the best located chromosome x_l every K_{ls} generations, where K_{ls} is a user defined constant that denotes how frequent the local search procedure has to be applied. The purpose of this application is to improve the function value of x_l and to speed up the convergence of the algorithm.

4. Experiments

4.1. Test problems

The proposed method was tested against the genetic algorithm of Caelo and Ali on a series of test problems proposed in [40] and [41]. The description of these test problems is given below.

Ap function
The function Alluffi–Pentiny is given by

$$f(x) = \frac{1}{4}x_1^4 - \frac{1}{2}x_1^2 + \frac{1}{10}x_1 + \frac{1}{2}x_2^2$$

with $x \in [-10, 10]^2$. The value of global minimum is -0.352386.

Bf1 function

The function Bohachevsky 1 is given by the equation

$$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1) - \frac{4}{10}\cos(4\pi x_2) + \frac{7}{10}$$

with $x \in [-100, 100]^2$. The value of global minimum is 0.0.

Bf2 function

The function Bohachevsky 2 is given by the equation

$$f(x) = x_1^2 + 2x_2^2 - \frac{3}{10}\cos(3\pi x_1)\cos(4\pi x_2) + \frac{3}{10}$$

with $x \in [-50, 50]^2$. The value of the global minimum is 0.0.

BL function

The Becker and Lago function is given by the equation

$$f(x) = (|x_1| - 5)^2 + (|x_2| - 5)^2$$

with $x \in [-10, 10]^2$. The value of the global minimum is 0.0.

Branin function

The function is defined by $f(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1) + 10$ with $-5 \leqslant x_1 \leqslant 10,\ 0 \leqslant x_2 \leqslant 15$. The value of global minimum is 0.397887.

Camel function

The function is given by

$$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4, \quad x \in [-5, 5]^2.$$

The global minimum has the value of $f(x^*) = -1.0316$.

Cb3 function

The three Hump function is given by the equation

$$f(x) = 2x_1^2 - 1.05x_1^4 + \frac{x_1^6}{6} + x_1x_2 + x_2^2$$

with $x \in [-5, 5]^2$. The value of the global minimum is 0.0.

Cosine mixture function (CM)

The function is given by the equation

$$f(x) = \sum_{i=1}^{n} x_i^2 - \frac{1}{10}\sum_{i=1}^{n}\cos(5\pi x_i)$$

with $x \in [-1, 1]^n$. The value of the global minimum is -0.4 and in our experiments we have used $n = 4$.

DeJoung function

This function is given by the equation

$$f(x) = x_1^2 + x_2^2 + x_3^2$$

with $x \in [-5.12, 5.12]^3$. The value of the global minimum is 0.0.

Easom function

The function is given by the equation

$$f(x) = -\cos(x_1)\cos(x_2)\exp\left((x_2 - \pi)^2 - (x_1 - \pi)^2\right)$$

with $x \in [-100, 100]^2$. The value of the global minimum is -1.0.

Exponential function

The function is given by

$$f(x) = -\exp\left(-0.5\sum_{i=1}^{n} x_i^2\right), \quad -1 \leqslant x_i \leqslant 1.$$

The global minimum is located at $x^* = (0, 0, \ldots, 0)$ with value -1. In our experiments we used this function with $n = 2, 4, 8, 16, 32, 64$ and the corresponding functions are denoted by the labels EXP2, EXP4, EXP8, EXP16, EXP32 and EXP64.

Gkls function

$f(x) = \mathrm{Gkls}(x, n, w)$, is a function with w local minima, described in [42] with $x \in [-1, 1]^n$ and n a positive integer between 2 and 100. The value of the global minimum is -1 and in our experiments we have used $n = 2, 3$ and $w = 50$. The corresponding functions are denoted by the labels GKLS250 and GKLS350.

Goldstein and price function

The function is given by the equation

$$\begin{aligned} f(x) = {} & \left[1 + (x_1 + x_2 + 1)^2 (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \\ & \times [30 + (2x_1 - 3x_2)^2 (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)] \end{aligned}$$

with $x \in [-2, 2]^2$. The global minimum is located at $x^* = (0, -1)$ with value 3.0

Griewank2 function

The function is given by

$$(x) = 1 + \frac{1}{200}\sum_{i=1}^{2} x_i^2 - \prod_{i=1}^{2} \frac{\cos(x_i)}{\sqrt{(i)}}, \quad x \in [-100, 100]^2.$$

The global minimum is located at the $x^* = (0, 0, \ldots, 0)$ with value 0.

Hansen function

$f(x) = \sum_{i=1}^{5} i\cos[(i-1)x_1 + i]\sum_{j=1}^{5} j\cos[(j+1)x_2 + j]$, $x \in [-10, 10]^2$. The global minimum of the function is -176.541793.

Hartman 3 function

The function is given by

$$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right)$$

with $x \in [0, 1]^3$ and $a = \begin{pmatrix} 3 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3 & 10 & 30 \\ 0.1 & 10 & 35 \end{pmatrix}$, $c = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 33.2 \end{pmatrix}$ and

$$p = \begin{pmatrix} 0.3689 & 0.117 & 0.2673 \\ 0.4699 & 0.4387 & 0.747 \\ 0.1091 & 0.8732 & 0.5547 \\ 0.03815 & 0.5743 & 0.8828 \end{pmatrix}.$$

The value of global minimum is -3.862782.

Hartman 6 function

$$f(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right)$$

with $x \in [0, 1]^6$ and $a = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}$, $c = \begin{pmatrix} 1 \\ 1.2 \\ 3 \\ 3.2 \end{pmatrix}$ and

$$p = \begin{pmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5886 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{pmatrix}.$$

The value of global minimum is -3.322368.

Rastrigin function
The function is given by

$$f(x) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2), \quad x \in [-1,1]^2.$$

The global minimum is located at $x^* = (0,0)$ with value -2.0.

Rosenbrock function
This function is given by

$$f(x) = \sum_{i=1}^{n-1} \left(100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right), \quad -30 \leqslant x_i \leqslant 30.$$

The global minimum is located at the $x^* = (0,0,\ldots,0)$ with $f(x^*) = 0$. In our experiments we used this function with $n = 2$.

Shekel 5

$$f(x) = -\sum_{i=1}^{5} \frac{1}{(x - a_i)(x - a_i)^T + c_i}$$

with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \end{pmatrix}, c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \end{pmatrix}$. The value of global minimum is -10.107749.

Shekel 7

$$f(x) = -\sum_{i=1}^{7} \frac{1}{(x - a_i)(x - a_i)^T + c_i}$$

with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 3 & 5 & 3 \end{pmatrix}, c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \end{pmatrix}$. The value of global minimum is -10.342378.

Shekel 10

$$f(x) = -\sum_{i=1}^{10} \frac{1}{(x - a_i)(x - a_i)^T + c_i}$$

with $x \in [0,10]^4$ and $a = \begin{pmatrix} 4 & 4 & 4 & 4 \\ 1 & 1 & 1 & 1 \\ 8 & 8 & 8 & 8 \\ 6 & 6 & 6 & 6 \\ 3 & 7 & 3 & 7 \\ 2 & 9 & 2 & 9 \\ 5 & 5 & 3 & 3 \\ 8 & 1 & 8 & 1 \\ 6 & 2 & 6 & 2 \\ 7 & 3.6 & 7 & 3.6 \end{pmatrix}, c = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.2 \\ 0.4 \\ 0.4 \\ 0.6 \\ 0.3 \\ 0.7 \\ 0.5 \\ 0.6 \end{pmatrix}$. The value of global minimum is -10.536410.

Shubert function
The function is given by $f(x) = -\sum_{i=1}^{2}\sum_{j=1}^{5} j\{\sin((j+1)x_i) + 1\}$, $x \in [-10,10]^2$. The value of global minimum is -24.06249.

Sinusoidal function

The function is given by

$$f(x) = -\left(2.5prod_{i=1}^{n} \sin(x_i - z) + \prod_{i=1}^{n} \sin(5(x_i - z))\right), \quad 0 \leqslant x_i \leqslant \pi.$$

The global minimum is located at $x^* = (2.09435, 2.09435, \ldots, 2.09435)$ with $f(x^*) = -3.5$. In our experiments we used $n = 2, 4, 8, 16, 32$ and $z = \frac{\pi}{6}$ and the corresponding functions are denoted by the labels SINU2, SINU4, SINU8, SINU16 and SINU32, respectively.

Test2N function

This function is given by the equation

$$f(x) = \frac{1}{2}\sum_{i=1}^{n} x_i^4 - 16x_i^2 + 5x_i, \quad x_i \in [-5, 5].$$

The function has 2^n in the specified range and in our experiments we used $n = 4, 5, 6, 7$. The corresponding values of global minimum is -156.664663 for $n = 4$, -195.830829 for $n = 5$, -234.996994 for $n = 6$ and -274.163160 for $n = 7$.

Table 1
Results of all genetic algorithms for the proposed test functions

Function	GEN	GEN_S	GEN_S_M	GEN_S_M_LS
AP	1360(0.99)	1360	1277	1253
BF1	3992	3356	1640	1615
BF2	20234	3373	1676	1636
BL	19596	2412	2439	1463
BRANIN	1442	1418	1404	1257
CAMEL	1358	1358	1336	1300
CB3	9771	2045	1163	1118
CM	2105	2105	1743	1539
DEJOUNG	9900	3040	1462	1281
EASOM	1318	1061	1097	1057
EXP2	938	936	817	807
EXP4	1668	1668	1279	1169
EXP8	3237	3237	2054	1496
EXP16	8061	8061	3251	1945
EXP32	9934	9932	5113	2241
EXP64	9940	9758	7817	2512
GKLS250	1286	1286	1284	1218
GKLS350	1831(0.94)	1831(0.94)	1757(0.96)	1541(0.96)
GOLDSTEIN	1478	1478	1408	1325
GRIEWANK2	18838(0.91)	3111(0.91)	1764	1652(0.99)
HANSEN	1887(0.96)	1727(0.96)	1746(0.97)	1624(0.97)
HARTMAN3	1350	1350	1332	1274
HARTMAN6	2562(0.54)	2562(0.54)	2530(0.67)	1865(0.68)
RASTRIGIN	1533(0.97)	1523(0.97)	1392	1381
ROSENBROCK2	9380	3739	1675	1462
SHEKEL5	2527(0.61)	2507(0.61)	2509(0.69)	2049(0.67)
SHEKEL7	2567(0.72)	2500(0.72)	2511(0.74)	2032(0.75)
SHEKEL10	2641(0.71)	2598(0.71)	2567(0.77)	2141(0.76)
SHUBERT	1873	1808	1744	1631
SINU2	1145	1145	1145	1115
SINU4	2061	2047	2017	1741
SINU8	3952	3952	3914	3057
SINU16	9676	9668	9496	6305
SINU32	9457(0.95)	9366(0.94)	9641(0.97)	8431(0.91)
TEST2N4	1896	1896	1894	1625
TEST2N5	2352	2352	2345(0.99)	1928(0.98)
TEST2N6	2758	2758	2735(0.98)	2229(0.98)
TEST2N7	3265	3265	3172(0.96)	2586(0.95)
TEST30N3	9656	1783	1877	1402
TEST30N4	7172	2291	2855	1484
POTENTIAL3	4995	3208	3814	2097
POTENTIAL5	9276	6684	6642	4301
Total	**221768(0.96)**	**133555(0.96)**	**111334(0.97)**	**83185(0.97)**

Test30N function

This function is given by

$$f(x) = \frac{1}{10}\sin^2(3\pi x_1)\sum_{i=2}^{n-1}\left((x_i-1)^2\left(1+\sin^2(3\pi x_{i+1})\right)\right)+(x_n-1)^2\left(1+\sin^2(2\pi x_n)\right)$$

with $x \in [-10, 10]$. The function has 30^n local minima in the specified range and we used $n = 3, 4$ in our experiments. The value of global minimum for this function is 0.0.

Potential function

The molecular conformation corresponding to the global minimum of the energy of N atoms interacting via the Lennard-Jones potential is determined for the case of $N = 3$ and $N = 5$ atoms (denoted by POTENTIAL3 and POTENTIAL5). The value of global minimum for POTENTIAL3 is -3.0 and -9.103852 for POTENTIAL5.

4.2. Results

The results from the application of all genetic algorithms are listed in Table 1. The number is cells denote the average number of function evaluations from 100 independent runs for every objective function in the column Function. The numbers in parentheses denote the fraction of runs that located the global minimum and were not trapped in one of the local minima. Absence of parentheses denotes that the global minimum has been recovered in every single run (100% success). The columns in the table have the following meaning:

1. The column Function denotes the name of the objective function.
2. The column GEN denotes the simple genetic algorithm listed in Algorithm 1.
3. The column GEN_S denotes the simple genetic algorithm using the additional stopping rule presented in 3.1.
4. The column GEN_S_M denotes the simple genetic algorithm with the use of the previous stopping rule and the mutation mechanism introduced in 3.2.
5. The column GEN_S_M_LS is the simple genetic algorithm using the proposed stopping rule in conjunction with the proposed mutation mechanism and a repeated application of a local search procedure as described in 3.3 with $K_{\text{ls}} = 5$.

In all algorithms the number of chromosomes (parameter N) used was set to 100 and the maximum number of allowed generations (parameter ITERMAX) was set to 200. The suggested by Caelo and Ali value $e = 10^{-4}$ was used for the termination criteria. The mutation was performed with probability 5% and the parameter b of Eq. (4) was set to $b = 5$. Tournament selection with tournament size 4 was used for the selection procedure. Also, the local search procedure was applied after the termination on the best located chromosome, in order to ensure that the located point is a true local minimum. The local search procedure used was a BFGS variant due to Powell [43]. All experiments were performed 100 times on every test problem, using different seed for the random number generator. The last row denoted by Total is the total number of function calls for the test problems.

As we can see from the experimental results, all the algorithms have managed to locate the global minimum in most cases and the all variations do not show notable differentiations. Although, it is clear that the proposed stopping rule enhances the performance of the simple genetic algorithm by preventing the algorithm from the execution of non - required generations. The speed gain from the application of the additional stopping rule is about 40%. Also, the proposed mutation mechanism found to be superior than the original mutation scheme and the average gain in function evaluations found to be almost 50%. Finally, the periodic application of the local search procedure to the best located chromosome significantly improves the speed and the efficiency of the original algorithm by 63%.

Also, the effect of the parameter K_{ls} is shown in Table 2, where all the proposed modifications (stopping rule, new mutation mechanism and periodical application of the local search procedure) are applied on a series of objective functions for different value of K_{ls}. As we can see, as the value of K_{ls} is increased, the average number of function evaluations is also increased without significant improvement to the efficiency of the proposed method.

Table 2
Experimental results for different values of the parameter K_{ls}

Problem	$K_{\text{ls}} = 5$	$K_{\text{ls}} = 10$	$K_{\text{ls}} = 15$	$K_{\text{ls}} = 20$
SHUBERT	1631	2207	2288	2348
HARTMAN6	1865(0.68)	2755(0.67)	3253(0.67)	3616(0.67)
SHEKEL7	2032(0.75)	2799(0.74)	3038(0.74)	3260(0.74)
SHEKEL10	2141(0.76)	2863(0.78)	3068(0.77)	3370(0.77)
TEST2N7	2586(0.95)	3354(0.96)	3753(0.96)	3960(0.96)

5. Conclusions

In this manuscript three modifications of the standard genetic algorithm have been proposed: (a) a new stopping rule based on asymptotic considerations, (b) a new mutation scheme based on the Particle Swarm Optimization method and (c) a periodically application of a local search procedure. These modifications are general enough and they can be applied in every real coded genetic algorithm. The experimental results have clearly shown that these modifications significantly improve the speed of the original algorithm. Future research will include improved selection and crossover operators in order to improve the speed of the algorithm even further.

Acknowledgements

All the experiments of this paper were conducted at the Research Center of Scientific Simulations of the University of Ioannina, which is composed of 200 computing nodes with dual cpus (AMD OPTERON 2.2 GHz 64 bit) running Redhat Enterprise Linux.

References

[1] Q. Duan, S. Sorooshian, V. Gupta, Effective and efficient global optimization for conceptual rainfall-runoff models, Water Resources Research 28 (1992) 1015–1031.
[2] D.J. Wales, H.A. Scheraga, Global optimization of clusters, crystals, and biomolecules, Science 285 (1999) 1368–1372.
[3] P. Charbonneau, Genetic algorithms in astronomy and astrophysics, Astrophysical Journal Supplement 101 (1995) 309.
[4] T.S. Metcalfe, P. Charbonneau, Stellar structure modeling using a parallel genetic algorithm for objective global optimization, Journal of Computational Physics 185 (2003) 176–193.
[5] A. Liwo, J. Lee, D.R. Ripoll, J. Pillardy, H.A. Scheraga, Protein structure prediction by global optimization of a potential energy function, Biophysics 96 (1999) 5482–5485.
[6] P.M. Pardalos, D. Shalloway, G. Xue, Optimization methods for computing global minima of nonconvex potential energy functions, Journal of Global Optimization 4 (1994) 117–133.
[7] Zwe-Lee Gaing, Particle swarm optimization to solving the economic dispatch considering the generator constraints, IEEE Transactions on Power Systems 18 (2003) 1187–1195.
[8] C.D. Maranas, I.P. Androulakis, C.A. Floudas, A.J. Berger, J.M. Mulvey, Solving long-term financial planning problems via global optimization, Journal of Economic Dynamics and Control 21 (1997) 1405–1425.
[9] H.A. Bremermann, A method for unconstrained global optimization, Mathematical Biosciences 9 (1970) 1–15.
[10] R.A. Jarvis, Adaptive global search by the process of competitive evolution, IEEE Transactions on Systems, Man and Cybergenetics 75 (1975) 297–311.
[11] W.L. Price, Global optimization by controlled random search, Computer Journal 20 (1977) 367–370.
[12] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671–680.
[13] P.J.M. van Laarhoven, E.H.L. Aarts, Simulated Annealing: Theory and Applications, D. Riedel, Boston, 1987.
[14] A. Corana, M. Marchesi, C. Martini, S. Ridella, Minimizing multimodal functions of continuous variables with the simulated annealing algorithm, ACM Transactions on Mathematical Software 13 (1987) 262–280.
[15] W.L. Goffe, G.D. Ferrier, J. Rogers, Global optimization of statistical functions with simulated annealing, Journal of Econometrics 60 (1994) 65–100.
[16] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Publishing Company, Reading, Massachussets, 1989.
[17] Z. Michaelewicz, Genetic Algorithms + Data Structures = Evolution Programs, Springer-Verlag, Berlin, 1996.
[18] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11 (1997) 341–359.
[19] J. Kennedy, R.C. Everhart, Particle swarm optimization, Proceedings of the 1995 IEEE International Conference on Neural Networks, vol. 4, IEEE Press, 1995, pp. 1942–1948.
[20] J.J. Grefenstette, R. Gopal, B.J. Rosmaita, D. Van Gucht, Genetic algorithms for the traveling salesman problem, in: Proceedings of the 1st International Conference on Genetic Algorithms, Lawrence Erlbaum Associates, 1985, pp. 160–168.
[21] G.F. Miller, P.M. Todd, S.U. Hegde, Designing neural networks using genetic algorithms, in: Proceedings of the Third International Conference on Genetic Algorithms, George Mason University, United States, Morgan Kaufmann, 1989, pp. 379–384.
[22] D. Whitley, T. Starkweather, C. Bogart, Genetic algorithms and neural networks: optimizing connections and connectivity, Parallel Computing 14 (1990) 347–361.
[23] R.L. Haupt, An introduction to genetic algorithms for electromagnetics, Antennas and Propagation Magazine 37 (1995) 7–15.
[24] D.A. Savic, G.A. Walters, Genetic algorithms for least-cost design of water distribution networks, Journal of Water Resources Planning and Management 123 (1997) 67–77.
[25] B. Manderick, P. Spiessens, Fine-grained parallel genetic algorithms, in: Proceedings of the Third international Conference on Genetic Algorithms, George Mason University, United States, Morgan Kaufmann, 1989, pp. 428–433.
[26] C.B. Pettey, M.R. Leuze, J.J. Grefenstette, A parallel genetic algorithm, in: Proceedings of the Second International Conference on Genetic Algorithms and their Application, Cambridge, Massachusetts, United States, Lawrence Erlbaum, 1987, pp. 155–161.
[27] P. Jog, J.Y. Suh, D. Van Gucht, Parallel genetic algorithms applied to the traveling salesman problem, SIAM Journal on Optimization 1 (1991) 515–529.
[28] Y. Yun, M. Gen, S. Seo, Various hybrid methods based on genetic algorithm with fuzzy logic controller, Journal of Intelligent Manufacturing 14 (2003) 401–419.
[29] H. Maaranen, K. Miettinen, M.M. Mäkelä, Quasi-random initial population for genetic algorithms, Computers and Mathematics with Applications 47 (2004) 1885–1895.
[30] M. Srinivas, L.M. Patnaik, Adaptive probabilities of crossover and mutation in genetic algorithms, IEEE Transactions on Systems, Man and Cybernetics 24 (1994) 656–667.
[31] J.J. Grefenstette, Optimization of control parameters for genetic algorithms, IEEE Transactions on Systems, Man and Cybernetics 16 (1986) 122–128.
[32] J.E. Baker, Adaptive selection methods for genetic algorithms, in: Proceedings of the 1st International Conference on Genetic Algorithms, Lawrence Erlbaum, 1985, pp. 101–111.
[33] S. Tsutsui, D.E. Goldberg, Search space boundary extension method in real-coded genetic algorithms, Information Sciences 133 (2001) 229–247.
[34] K. Deep, M. Thakur, A new mutation operator for real coded genetic algorithms, Applied Mathematics and Computation 193 (2007) 229–247.
[35] K. Deep, M. Thakur, A new crossover operator for real coded genetic algorithms, Applied Mathematics and Computation 188 (2007) 895–911.
[36] H. Aytug, G.J. Koehler, New stopping criterion for genetic algorithms, European Journal of Operational Research 26 (2000) 662–674.
[37] Y.T. Kao, E. Zahara, A hybrid genetic algorithm and particle swarm optimization for multimodal functions, Applied Soft Computing 8 (2008) 849–857.

[38] Q. Yuan, Z. He, H. Leng, A hybrid genetic algorithm for a class of global optimization problems with box constraints, Applied Mathematics and Computation 197 (2008) 924–929.
[39] P. Kaelo, M.M. Ali, Integrated crossover rules in real coded genetic algorithms, European Journal of Operational Research 176 (2007) 60–76.
[40] M.M. Ali, C. Khompatraporn, Z.B. Zabinsky, A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems, Journal of Global Optimization 31 (2005) 635–672.
[41] C.A. Floudas, P.M. Pardalos, C. Adjiman, W. Esposoto, Z. Gümüs, S. Harding, J. Klepeis, C. Meyer, C. Schweiger, Handbook of Test Problems in Local and Global Optimization, Kluwer Academic Publishers, Dordrecht, 1999.
[42] M. Gaviano, D.E. Ksasov, D. Lera, Y.D. Sergeyev, Software for generation of classes of test functions with known local and global minima for global optimization, ACM Transactions on Mathematical Software 29 (2003) 469–480.
[43] M.J.D. Powell, A tolerant algorithm for linearly constrained optimization calculations, Mathematical Programming 45 (1989) 547–566.