ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 5, Issue 4, October 2015

Solution of Problem of Unconditional Optimization by Means of Genetic Algorithm

Natela Ananiashvili
Faculty of Exact and Natural Sciences, I.Javakhishvili Tbilisi State University, Tbilisi, Georgia

Abstract: Genetic algorithm for solution of problem of unconditional optimization is offered. It is based on the main principles of a classical genetic algorithm. Certain techniques of determination of function, crossover and realization of mutation are described. Algorithm permanently gives good results for test problems.

***Keywords:* Unconditional optimization, genetic algorithm, crossover, mutation.**

I. INTRODUCTION

Genetic algorithms were developed as a result of observations on living organisms, particularly by trying to copy the processes of evolution and selection. The idea was introduced in 60-70s of the past century and belongs to Holland [1]. The main complexity of solution of optimization problems is related to non-linearity of functions and scales of problems. Searching of optimal values with classical techniques often finishes without results. That's why genetic algorithms were developed. Genetic algorithm is one of the types of random search. Majority of classical algorithms of optimization and their modifications begin to operate in search space from one initial point and then deterministically calculate gradient of target function, or derivative of higher order [2-3]. Afterwards, solution is gradually improved. Disadvantage of such technique is that we can get to local minimum. Genetic algorithm simultaneously searches in several directions [4-5]. Transition from one population to another prevents from getting to local minimum. Genetic algorithms have two main advantages over techniques of classical optimization: 1) genetic algorithms do not have significant requirements towards the target function. Simplification of model is not necessary to artificially create conditions for usage of available mathematical techniques. Besides, we may get completely different target functions and restrictions, such as linear or non-linear, discrete, determined for continuous or mixed universal sets; 2) When we use classical techniques, global optimum can be determined only if the condition of convexity exists. Evolutionary processes of genetic algorithms can effectively find global optimum with or without this condition.

II. THE GENERAL SCHEME OF GENETIC ALGORITHM AND ITS MODIFIED REALIZATION

A. The General Scheme of Genetic Algorithm

The article considers the problem of unconditional minimization: $f(x) \to \min, x \in R_n$. The algorithm offered for solution of this problem implies the general principles of genetic algorithm. This algorithm can be described non-formally in the following way:

Step 1. Selection of initial population;

Step 2. Determination of a fitness function and estimation of fitness of individuals in the given population;

Step 3. Selection of parents according to fitness. Usage of genetic operators, i.e. crossover and mutation for selected individuals and getting offspring;

Step 4. Estimation of fitness of offspring individuals. Replacement of one of the parents with the best offspring in initial population;

Step 5. If condition for finishing the algorithm is met: if quantity of iterations is exhausted, then finish, otherwise jump to Step 3.

Now let us describe the proposed algorithm step by step.

B. Selection of Initial Population

Selection of initial population implies selection of a random set of initial solution. When genetic algorithms are realized, it is very important to select a scheme of encoding. In the proposed work binary scheme is selected. First of all, we must determine quantity of bits that is necessary to write variables of a problem. Length of a binary row depends on a precision of encoding. Let us assume that $x_j \in [a_j, b_j]$ and necessary precision is 7 positions after the separator. In this case quantity k_j of bits necessary for encoding is determined from the following inequality:

$$2^{k_j-1} < (b_j - a_j)\cdot 10^7 \le 2^{k_j} - 1 \quad (1)$$

Let us assume that $k_j = 21$ from (1) inequality. In this case we need 21 binary digits for each chromosome. We can write 0s and 1s in the following way:

$$q_1 = (010000110001000111010)$$

We select size of an initial population, for instance $P = 25$ and create the initial population. Each element of the population is called chromosome. Therefore, we get randomly selected P chromosomes: $q_1, q_2, \dots, q_P$.

We can make reverse transformation from bitwise row and get real value of x_j by means of the following equation:

$$x_j = a_j + D\cdot\frac{b_j - a_j}{2^{k_j} - 1} \quad (2)$$

Here D is a decimal value that is encoded in a binary row q_j.

In our example decimal value of binary number q_1 is $D = 549434$. Value of x_1 can be calculated by means of (2) equation:

$$x_j = 1.1 + 549434 \cdot \frac{2.9 - 1.1}{2^{21} - 1} = 1.571583$$

We can calculate real decimal values $x_1, x_2, \dots, x_P$ of each q_j, $j = 1,2, \dots P$ in the same way.

C. Determination of a fitness function and estimation of fitness of individuals in the given population.

We must calculate values of corresponding target function and their probabilities for estimation of fitness of individuals in a given population. We can calculate values $f(x_j)$, $j = 1,2, \dots, P$ of corresponding target function for each chromosome after the selection of initial population.

Let us denote:

$fmin = \min\{f(x_1), f(x_2), \dots, f(x_P)\}$ (3)

Calculate general function of correspondence of population:

$$F = \sum_{i=1}^{P} (f(x_i) - fmin) \quad (4)$$

Calculate probability for each chromosome:

$$p_j = \frac{f(x_j) - fmin}{F}, \; j = 1,2, \dots, P \quad (5)$$

Calculate probabilities for each $f(x_j)$, $j = 1,2, \dots, P$:

$$\tau_j = \sum_{i=1}^{j} p_i, \; j = 1,2, \dots, P \quad (6)$$

It is necessary to select the individuals (with certain rule) from previous population and then crossover them to get the following population from the previous one. Afterwards, selected parent individuals are replaced with offspring individuals that are born from crossover. For selection so-called technique of "roulette" is used. By means of this technique, individuals of population change order. Selection of a new order begins from rotation of "roulette". Quantity of rotations is P. Every rotation implies selection of normalized numbers from [0,1] range. Besides, one chromosome is selected every time by means of the following algorithm:

1. Select random number from range [0, 1]. Let us assume this number is r;
2. If $r \le p_1$, then we select the first chromosome x1. Otherwise, we select chromosome x1, for which $p_{i-1} < r \le p_i$, $2 \le i \le P$.

Chromosome that is selected with the described technique takes the first position in the population. Then we find the second chromosome, Pth chromosome, etc.

D. Crossover

One-point crossover technique is used for crossover. Let us take random numbers $\varphi_1, \varphi_2, \dots, \varphi_P$ from range [0, 1] and select two of them with minimal values. Let us assume such numbers are φ_4 and φ_6, i.e. crossover chromosomes are fourth and sixth. They are called parents. Because we need bitwise operations for crossover and their realization is convenient with bitwise operators of language C++, it is better to represent them with software fragments instead of verbal description. Bitwise operations are known and their perception is not hard. Let us use Grey's encoding for binary chromosomes x4 and x6, because it ensures that change of individual (chromosome) isn't very large. Particularly, our chromosomes are encoded into Unsigned Long type array. Each element of this array can be converted into Grey's code by means of the following instruction:

```
gf=xch[numb]^(xch[numb]>>1)
```

Where gf is unsigned long type variable and xch is unsigned long type array of chromosomes. numb is number of selected chromosome for encoding; in this case it can be 4 or 6. Afterwards, gf is decoded and values of its bits are written into any array vf. This operation can be realized by means of the following software fragment:

```
for(int i=0; i<bit; i++)
    vf[i]=0;
  unsigned long c0, c; int c1;
    for (int j=0; j<bit; j++){
      c0=1;
      c=c0<<(bit-j-1);
      c1=gf & c;
      if (c1!=0) vf[j]=1;       }
```

Where bit is a quantity of bits that is necessary to encode chromosome. In our case bit = 21. It is worthwhile to note that if we need only 21 positions for encoding, as a result of equation (1), if we wish to leave xjs in the range $[a_j, b_j]$ then from 21 to 32 positions we must write 0-s.

In this way x4 and x6 chromosomes are converted into Grey's code and then decoded. Let us assume that arrays vf and vm corresponding to x4 and x6 are:

```
vf:   0000 0000 0001 0011 0111 1011 0100 1000
vm:  0000 0000 0000 1100 0110 0101 1101 0000
```

Let us take random number $\vartheta \in [1, 21]$. It will be the position towards which the fourth and the sixth chromosomes will be crossover. Assume $\vartheta = 11$, then vg and vm will change in the following way:

```
vf:   0000 0000 0001 0011 0111 1101 1101 0000
vm:  0000 0000 0000 1100 0110 0011 0100 1000
```

For reverse decoding let us use the following software fragment:

```
gf=0;
for(int p=1; p<=bit; p++){
          if (vf[p-1]==1) c=1;
          else c=0;
          c=c<<(bit-p); gf=gf|c;        }
```

to convert to Grey's Code:

```
for (xch[numbf] = 0; gf; gf >>= 1)
   xch[numbf] ^= gf;
```

ISSN: 2277-3754
ISO 9001:2008 Certified
International Journal of Engineering and Innovative Technology (IJEIT)
Volume 5, Issue 4, October 2015

E. Mutation

Mutation changes one or several genes. Quantity of mutated genes can be calculated in the following way: quantity of bits of population is multiplied by size of population (21* P =21*25=525) and one percent of this number, i.e. 5 genes are prone to mutation. Let us take 5 random numbers from range [1, 525] and assume these numbers are r_1, r_2, r_3, r_4, r_5. Genotypes with these numbers are prone to mutation. For example, if r1=65 and the second (65-3*21=2) bit of the fourth chromosome (65/21=3) is 0, then it must be replaced with 1 and vice versa. Before mutation, chromosome is once again converted into Grey's code, as it was in the case of crossover. The first iteration of our algorithm finishes with mutation. These steps (determination of fitness function, crossover, mutation) are repeated until exhaustion of iterations. After each iteration, the best value of fitness function is selected, this value is compared to best values of previous and new iterations and the better value is remembered. The last value will be the best solution of our problem after the exhaustion of iterations.

III. THE RESULTS OF TESTING

TABLE 1. The results of calculations for test functions

Test function	Function value	Compu-tation time	x	y
Eason's function	0,98655	0,055	3.07515	3.07164
Rsatrigin's function	0.11925	3.829	0.01203	0.02139
Six-hump camel black function	-1.0306	3.781	-0.09898	0.72261

On Figure 1 Deep dark point shows the point of global minimum.

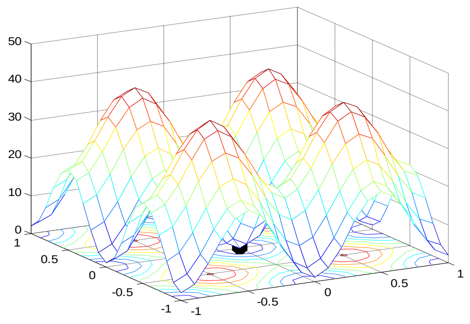

Fig 1. The Graph of Rastrigin's Functions

The algorithm was tested on computer with standard specifications: Intel(R) Pentium (R) Dual CPU E2220, 2.40GHz, 2.00 GB of RAM.

IV. CONCLUSIONS

Genetic algorithms are based on natural processes, such as natural selection and heredity. These algorithms operate with less restrictions and rapidly get optimal or near optimal solutions. They can be used for solution of engineering problems when working on artificial intelligence, optimization, neural network and fuzzy systems.

REFERENCES

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

[2] Boris T. Polyak . Introduction to Optimization. Optimization Software, 1987.

[3] K. Gelashvili, L. Alkhazishvili, I. Khutsishvili, N. Ananiashvili. "On the modification of heavy ball method", A. Razmadze Mathematical Institute, Ivane Javakhishvili Tbilisi State University, Tbilisi, Journal, ISSN 1512-0007, Vol. 161, 2013.

[4] Rutkowska D., PiliĚski M., Rutkowsli L., Sieci neuronowe, algorytmy Genetyczne i systemy rozmyte, Wydawnictwo Naukowe PWN, Warszawa-àódĨ, 1999.

[5] Randy L. Haupt, Sue Ellen Haupt, "Practical genetic algorithms"-2nd ed. Published by John Wiley & Sons, Inc., Hoboken, New Jersey. 2004.

AUTHOR'S PROFILE

Natela Ananiashvili is a doctoral student of faculty of exact and natural sciences at I. Javakhishvili Tbilisi State University, Tbilisi, Georgia. She is currently a lecturer in the department of computer science at Tbilisi State University and has published papers in both local and international journals.