DOI: 10.21276/sjet.2016.4.6.4

Scholars Journal of Engineering and Technology (SJET)
Sch. J. Eng. Tech., 2016; 4(6):280-283
©Scholars Academic and Scientific Publisher
(An International Publisher for Academic and Scientific Resources)
www.saspublisher.com

ISSN 2321-435X (Online)
ISSN 2347-9523 (Print)

Research Article

A Genetic Algorithm for Solving Multimodal Functions Based on Neighborhood Penalty Function

Nengfa Hu
Department of Computer Science and Engineering, Hanshan Normal University, Chaozhou 521041, China

***Corresponding author**
Nengfa Hu
Email: hunengfa326@sina.com

Abstract: By utilizing the neighborhood penalty function and mutation method, the research puts forward a novel genetic algorithm (GA) by combining global search and local search. Based on the strategy of multiple evolutions, the algorithm constructs a neighborhood with the result of each evolution as the centre, and then sets a penalty function to punish individuals in the neighborhood. The experiment proves that the algorithm converges rapidly, shows favorable global superiority, and is not likely to get trapped in a local optimum. Endowed with these advantages, the algorithm presents preferable global performance and therefore is universally applicable to multimodal functions with multiple solutions.
Keywords: genetic algorithm, multimodal function, optimization.

INTRODUCTION

Genetic algorithm (GA) is a random global optimization search algorithm that simulates the natural selection and genetic mechanism in the biological evolutionary process [1]. In recent years, the algorithm has been widely applied in various fields including neural network, function optimization, image processing, system identification, and expert system due to its high efficiency and practicability [2-8].

With regard to function optimization, there is a kind of complex functions which not only show high dimensions but also have numerous local optimal solutions as well as multiple global optimal solutions. Moreover, these solutions are distributed in a completely unknown pattern. All these characteristics make it difficult to search the global optimal solutions from these local ones. To solve such function optimization, scholars have proposed lots effective methods such as niche algorithm and penalty algorithm. In the study, to find out all the global optimal solutions so as to avoid the premature convergence and the loss of some global optimal solutions, the evolution method for adjusting population diversity is put forth. Meanwhile, the multi-parent GA based on real coding is designed to obtain all the global optimal solutions.

GA FOR SOLVING FUNCTION OPTIMIZATION

GA put forward by John Holland in Michigan University of the United States is an effective probability search algorithm based on the natural selection and genetic mechanism according to biological evolutionism. It is used to perform search and optimization by simulating biological evolution. It is able to automatically obtain and accumulate knowledge of the searching space in the searching process, which is adaptively controlled to solve the optimal solution of a problem[9-14]. The steps of adopting GA to solve a practical optimization problem are as follows:

1) Determining an evaluation function, namely, fitness function, to assess the quality of a solution;
2) Designing the coding scheme for solving a problem;
3) Generating the initial population in which each individual represents a feasible solution;
4) Determining the genetic operation scheme for the problem to be solved;
5) Determining all the parameters in the GA.

While searching the solutions of a problem, relevant parameters have to be adjusted constantly along with repeated experiments and result analysis, to finally obtain the optimal solution for the problem.

BASIC IDEA OF THE NOVEL ALGORITHM

As the traditional genetic algorithm (TGA) fails to solve the multiple solutions for multimodal functions effectively, the novel GA adopts multiple evolutions based on the optimization idea of TGA. To avoid repeated search in different evolutions, the novel GA stores the results of each evolution in a global array (denoted as Xa). Then, in the next evolution, the fitness values of the points in the neighborhood ($O(Xa[k], \delta)$, where k is a subscript) that employs the element of Xa as the centre and δ as the radius are punished and mutated, followed by random crossover in the whole population. In this way, all the individuals in different evolutions share a same fitness function. For the test function used in the research, the fitness function is defined as:

$$f\{x\} = \begin{cases} C & \text{if } x \in \bigcup_{k=1}^{L} O(Xa[k], \delta) \\ C & \text{if } x \in\not\in \{(x_1, x_2, \cdots, x_n) \mid\ |x_i| \le 10\} \\ \prod_{i=1}^{n} \sum_{j=1}^{5} j\cos[(j+1)x_i + j] & \text{else} \end{cases}$$

Where C is a constant and variable L represents the number of obtained optimal solutions. When the objective function is to acquire the maximum value, C is set to be small; otherwise, C is large. In the research, C is set as 9999.

ALGORITHM DESCRIPTION

The major variables include Maxcoun, kbest, kworst, and fbest which represent the maximum evolution times, optimal individuals, worst individuals, and optimal solutions.

BEGIN
L: = 0;
for count: = 1 to Maxcount do
begin
Randomly generating the initial population P={p[1], p[2], …, p[M]} in the size of M
Calculating the fitness values f[k] (k=1, 2, …, M) of all the individuals in the population
Finding out optimal and worst individuals from the current population and store their fitness values in variables fbest and fworst, respectively. The corresponding subscripts are preserved in kbest and kworst.
repeat
for k: = 1 to M do
if f[k] = C then
f[k] mutates;
k: = random(M)+1;
The optimal individual kbest crosses with a random individual k to generate a new individual X;
if the new individual X is superior to the original worst individual kworst, then
the new individual replaces the original worst individual kworst;
Search the optimal and worst individuals from the current population repeatedly and store their fitness values in variables fbest and fworst, separately. The corresponding subscripts are stored in kbest and kworst.
until fbest ≤ worst;
if fbest is a optimal solution, then
add 1 to L and store the current evolution results in the array Xa;
end;
output the evolution results;
END.

The above algorithm shows that each evolution utilizes the knowledge accumulated in previous evolution. As mutation is conducted to all the individuals in the neighborhood of the obtained optimal solution, the repeated and blind search is avoided. Optimal solutions are easily to be obtained in the first evolution owing to there is no disturbance. So, to judge whether fbest is an optimal solution or not, the difference between the fbest obtained in the first evolution and that acquired in current evolution can be regarded as the standard. Once the absolute value of the difference is less than a given positive number ε, the fbest in current evolution is considered as an optimal solution.

The mathematical form of the crossover operator in the algorithm is

$$X = \lambda_1 p[kbest] + \lambda_2 p[k], (\lambda_1 + \lambda_2 = 1, -0.5 \le \lambda_i \le 1.5, i = 1,2)$$

The mathematical form of the mutation operator is

$$p[k] = p[k] + 2\delta(2rand - 1)$$

Where p[k] is the kth individual in a population p, rand is a random number in the range of (0, 1), and δ represents the radius of the neighborhood.

EXPERIMENTAL RESULTS

The test function is

$$\text{Min } f(x) = \prod_{i=1}^{2} \sum_{j=1}^{5} j\cos[(j+1)x_i + j]$$

Where $-10 \le x_i \le 10, i = 1,2$

the corresponding graphs is as shown in Fig.1 - Fig.2.

Through multiple evolutions, the obtained optimal solutions are illustrated in Table 1. In the operation, the population is in the size of M=100 and the radius of the neighborhood is δ=0.1. Under such condition, it takes a short time to calculate 18 optimal solutions: all the optimal solutions are computed in one second or so generally. Therefore, the value of Maxcount can be a bit larger in the operation. By doing so, all the solutions can be calculated without slowing down the operation speed.

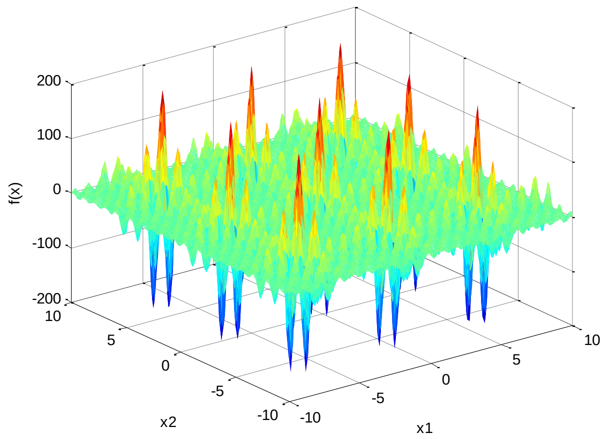

Fig-1: The test function graphs

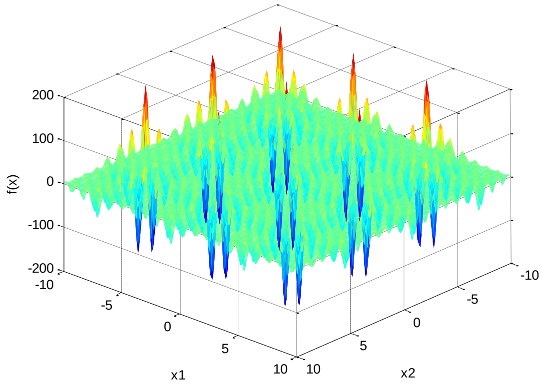

Fig-2: The test function graphs

Table 1: Experimental results (Intel(R) Core(TM) i3-4160 CPU @ 3.60GHz 3.60GHz)

No.	X_1	X_2	fbest
1.	4.85805687906020928	5.48286420696880384	-186.730908831023936
2.	-1.42512842833996592	5.48286420684839616	-186.730908831023904
3.	-7.70831373479955200	-0.80032109972328320	-186.730908831023936
4.	-7.08350640722188416	-1.42512842801855968	-186.730908831023936
5.	5.48286420796178048	-7.70831373374785536	-186.730908831023936
6.	-7.08350640847843712	-7.70831373469204608	-186.730908831023936
7.	-7.08350640737659776	4.85805687891071808	-186.730908831023936
8.	-7.70831373626399360	5.48286420436218624	-186.730908831023936
9.	5.48286420837339840	-1.42512842832009264)	-186.730908831023904
10.	4.85805687943886464	-0.80032110092372768	-186.730908831023936
11.	-1.42512842902836704	-0.80032109934927392	-186.730908831023904
12.	-0.80032109933748592	-1.42512842893701072	-186.730908831023904
13.	5.48286420785347968	4.85805687761748736	-186.730908831023936
14.	4.85805687757381760	-7.08350640621181824	-186.730908831023936
15.	-1.42512842652967424	-7.08350640898138240	-186.730908831023904
16.	-0.80032110022062400	-7.70831373560382208	-186.730908831023936
17.	-0.80032110068112528	4.85805687899588352	-186.730908831023936
18.	-7.70831373421708416	-7.08350640750272000	-186.730908831023936

CONCLUSIONS

While solving multimodal functions, TGA is very likely to get trapped in a local optimum and fails to find out all the solutions in a short time for multi-solution problems. The experiment verifies that the proposed novel GA converges faster, shows better global superiority, and is not likely to run into a local optimum compared with TGA and other algorithms.

REFERENCES

1. Holland JH; Adaptation in natural and artificial systems. Ann Arbor, MI: University of Michigan Press,1975.
2. Nengfa Hu; A Design Model for the General Evolutionary Algorithm. Scholars Journal of Engineering and Technology, 2016; 4(5):249-253.
3. Qing-dao-er-ji R, Wang YP; A new hybrid genetic algorithm for job shop scheduling problem. Computers & Operations Research, 2012; 39:2291-2299.
4. Koza JR; Genetic Programming I: Automatic Discovery of reusable programs. MIT Press, Cambridge, 1994.
5. Darwin C, Burrow JW; The origin of species by means of natural selection: The preservation of favoured races in the struggle for life. USA: Penguin Classics, 2009.
6. Darwin C; The descent of man, and selection in relation to sex. London: John Murray,1994.
7. Moore AJ; The evolution of sexual dimorphism by sexual selection: The separate effects of intrasexual selection and intersexual selection. Evolution, 1990; 34(2): 315-331.
8. Fisher RA; The evolution of sexual preference. Eugenics Review, 1995; 7(3): 184-192.
9. Agata C, Anna N, Mikolaj R; Sensitivity Analysis of Mathematical Models of Signaling Pathways. Journal of Biotechnology, Computational Biology and Bionanotechnology, 2012,93(3), 291-308.
10. Subhrajit R, Minhazul SK, Swagatam D; Multimodal Optimization by Localized Group Search Optimizers. Applied Soft Computing, 2013; 13(1): 27-46.
11. Antanas Z; On Strong Homogeneity of Two Global Optimization Algorithms Based on Statistical Models of Multimodal Objective Functions. Applied Mathematics & Computation, 2012; 218(16): 8131-8136.
12. Shima K; Using a Self-adaptive Neighborhood Scheme with Crowding Replacement Memory in Genetic Algorithm for Multimodal Optimization. S warm and Evolutionary Computation, 2013; 12:1-17.
13. Minqiang L, Dan L, Jisong K; A Hybrid Niching PSO Enhanced with Recombination replacement Crowding Strategy for Multimodal Function Optimization. Applied Soft Computing, 2012; 12(3): 975-987.
14. Jovani LF, Luiz FLR, Paulo LC; Comparison of Methods for Multivariate Moment Inversion Introducing the Independent Component Analysis. Computers & Chemical Engineering, 2014; 60:41-56.