TELKOMNIKA, Vol.10, No.4, August 2012, pp. 858~863
ISSN: 2087-278X
Indexed by Ei Compendex (ELSEVIER)

Research of Function Optimization Algorithm

Qinghua Wu*[1,2], Hanmin Liu[3], Yuxin Sun[1,2], Fang Xie[1,2], Jin Zhang[1,2], Xuesong Yan[4*]
[1] Hubei Provincial Key Laboratory of Intelligent Robot, Wuhan Institute of Technology, Wuhan, China
[2]School of Computer Science and Engineering, Wu-Han Institute of Technology, Wuhan, China
[3] Wuhan Institute of Ship Building Technology, Wuhan, China
[4]School of Computer Science, China University of Geosciences, Wuhan, China
e-mail: yanxs1999@126.com*

Abstrak

Algoritma evolusioner tradisional terjebak dalam minimum lokal dengan mudah. Oleh karena itu, didasarkan pada algoritma evolusioner sederhana dan menggabungkan ideologi dasar uji ortogonal kemudian diaplikasikan pada inisialisasi populasi, operator crossover, sebagai pengenalan inver-over operator untuk mencegah konvergensi lokal membentuk algoritma evolusioner baru. Melalui serangkaian eksperimen numerik, algoritma baru telah terbukti efisien untuk optimasi fungsi.

Kata kunci: *algoritma evolusioner, inver-over, optimasi fungsi*

Abstract

Traditional evolutionary algorithm trapped into the local minimum easily. Therefore, based on a simple evolutionary algorithm and combine the base ideology of orthogonal test then applied it to the population initialization, crossover operator, as well as the introduction of inver-over operator to prevent local convergence to form a new evolutionary algorithm. Through the series of numerical experiments, the new algorithm has been proved is efficiency for function optimization.

Keywords*: evolutionary algorithm, function optimization, inver-over*

1. Introduction

An important area in current research is the development and application of search techniques based upon the principles of natural evolution. Evolution can be viewed as s change in the genetic composition of a population of individuals over time. In a simplified form, evolution is result of the successive processes of reproduction and genetic variation followed by natural selection, which allows the fittest individuals to survive and reproduce, thus propagating their genetic material to future generations. We shall use this as a starting point in introducing evolutionary computation. The theory of natural selection proposes that the plants and animals that exist today are the result of millions of years of adaptation to the demands of the environment. At any given time, a number of different organisms may co-exist and compete for the same resources in an ecosystem. The organisms that are most capable of acquiring resources and successfully procreating are the ones whose descendants will tend to be numerous in the future. Organisms that are less capable, for whatever reason, will tend to have few or no descendants in the future. The former are said to be more fit than the latter, and the distinguishing characteristics that caused the former to be more fitness are said to be selected for over the characteristics of the latter. Over time, the entire population of the ecosystem is said to evolve to contain organisms that, on average, are more fit than those of previous generations of the population because they exhibit more of those characteristics that tend to promote survival.

Evolutionary computation techniques abstract these evolutionary principles into algorithms that may be used to search for optimal solutions to a problem. In a search algorithm, a number of possible solutions to a problem are available and the task is to find the best solution possible in a fixed amount of time. For a search space with only a small number of possible solutions, all the solutions can be examined in a reasonable amount of time and the optimal one found. This exhaustive search, however, quickly becomes impractical as the search space grows in size. Traditional search algorithms randomly sample or heuristically sample the

Received May 9, 2012; Revised July 22, 2012; Accepted July 27, 2012

search space one solution at a time in the hopes of finding the optimal solution. The key aspect distinguishing an evolutionary search algorithm from such traditional algorithms is that it is population-based. Through the adaptation of successive generations of a large number of individuals, an evolutionary algorithm performs an efficient directed search. Evolutionary search is generally better than random search and is not susceptible to the hill-climbing behaviors of gradient based search [1-3].

By mimicking the process of natural evolution, researchers developed the evolutionary algorithms (EA), which are based on the collective adaptability within a population of individuals, each of which represents a search point in the space of potential solutions to a given problem. In order to an evolutionary algorithms to work, a population of candidate solution is initialized, and it evolves towards increasingly better regions of the search space by means of selection, reproduction and genetic variation mechanisms. The environment in which the population evolves is defined by the aim of the search, and delivers the information, termed fitness, that quantifies how good an individual is. The selection process favors the reproduction of individuals of higher fitness, and a recombination mechanism allows the mixing of parental information while passing it to their descendants. Finally, mutation introduces novelties in the population.

An Evolutionary Algorithm is an iterative and stochastic process that operates on a set of individuals (population). Each individual represents a potential solution to the problem being solved. This solution is obtained by means of a encoding/decoding mechanism. Initially, the population is randomly generated (perhaps with the help of a construction heuristic). Every individual in the population is assigned, by means of a fitness function, a measure of its goodness with respect to the problem under consideration. This value is the quantitative information the algorithm uses to guide the search. The whole process is sketched in following.

```
Generate [P(0)]
t ←0
WHILE NOT Termination_Criterion [P(t)] DO
    Evaluate [P(t)]
P' (t) ←Select [P(t)]
P"(t) ←Apply_Reproduction_Operators [P'(t)]
P(t+1) ←Replace [P(t), P"(t)]
t ←t + 1
    END
RETURN Best Solution
```

It can be seen that the algorithm comprises three major stages: selection, reproduction and replacement. During the selection stage, a temporary population is created in which the fittest individuals (those corresponding to the best solutions contained in the population) have a higher number of instances than those less fit (natural selection). The reproductive operators are applied to the individuals in this population yielding a new population. Finally, individuals of the original population are substituted by the new created individuals. This replacement usually tries to keep the best individuals deleting the worst ones. The whole process is repeated until a certain termination criterion is achieved (usually after a given number of iterations).

Notice that the algorithm establishes a trade-off between the exploitation of good solutions (selection stage) and the exploration of new zones of the search space (reproduction stage), based on the fact that the replacement policy allows the acceptation of new solutions that not necessarily improve the existing ones.

EA's are heuristics and thus they do not ensure an optimal solution. The behavior of these algorithms is stochastic so they may potentially present different solutions in different runs of the same algorithm. That's why it is very common to need averaged results when studying some problem and why probabilities of success (or failure), percentages of search extension and so on, are normally used for describing their properties and work.

There are three main types of evolutionary algorithms: 1) evolutionary strategies; 2) genetic algorithms, and 3) evolutionary programming. Note that some authors consider genetic programming and classifier systems as other branches of the evolutionary algorithms; a discussion that will not be pursued here.

From a practical perspective, evolutionary algorithms are aimed at performing a search to identify an approximation of an (global) optimum of an objective function. The search is performed by evolving a population of individuals represented, in most cases, according to the application domain and type of evolutionary algorithms. The individuals of the population correspond to chromosomes that during the evolutionary process are allowed to suffer genetic variation and/or exchange genetic material.

Evolutionary algorithms are a kind of ways, which simulate the evolutionary processes of the nature, especially the evolutionary processes of creatures by computers for solving complex problems. The EA has such characteristics as self-adaptation, self-organization, self-learning, self-optimization and intrinsic parallelism. Using it can get rid of the needs of knowledge bases and formal logic deduction in some sense and may realize automatic programming [4-6].

2. Function Optimization Algorithm

In our algorithm, the code representation we use the real coding, the real coding method has advantages compared with the binary coding in the function optimization problems, because the real coding can solve the "Hamming cliff" problem which the binary coding has no idea to solve it, and then the encoding, the decoding and the calculation of the fitness function is more convenient when we use the real coding.

Initialize population:

The traditional method of genetic algorithm is randomly initialized population, that is, generate a series of random numbers in the solution space of the question. In this paper, the new algorithm using the orthogonal initialization [7] in the initialization phase. For the general condition, before seeking out the optimal solution the location of the global optimal solution is impossible to know. For some high-dimensional and multi-mode functions to optimize, the function itself has a lot of poles, and the global optimum location of the function is unknown. If the initial population of chromosomes can be evenly distributed in the feasible solution space, the algorithm can evenly search in the solution space for the global optimum. Orthogonal initialization is to use the orthogonal table has the dispersion and uniformity comparable; the individual will be initialized uniformly dispersed into the search space, so the orthogonal design method can be used to generate uniformly distributed initial population.

Inver-Over operator:

In this paper, we used the Inver-Over operator in our evolutionary algorithm. The Inver-over operator has proved to be a high efficient operator in evolutionary algorithm [8]. The creativity of this operator is that it adopts the operation of inversion in genetic operators, which can effectively broaden the variety of population and prevent from local minimum and lead to find the best solutions quickly and accurately. Our algorithm is be perceived as a set of parallel hill-climbing procedures.

Framework of algorithm:

Step 1: Orthogonal initialize population;
Step 2: Using selection method select the excellent individual;
Step 3: Using select operator and strategy select the two parents individuals;
Step 4: Using the orthogonal crossover operation
Step 5: Executive the adaptive local search strategy;
Step 6: Executive the adaptive orthogonal mutation strategy for the repeat individuals in the population;
Step 7: Repeat Step 3 to Step 7 until end of the algorithm.

3. Experiment Results

In order to verify the validity of the new algorithm, we using four benchmarks function to verify the effectiveness of improvement.

F1: Schaffer function

$$\min f(x_i) = 0.5 - \frac{(\sin^2 \sqrt{x_1^2 + x_2^2} - 0.5)}{[1 + 0.001(x_1^2 + x_2^2)]^2}, -100 \le x_i \le 100 \quad (1)$$

In this function the biggest point is that the overall situation (0,0), the largest in the overall points for the center, to 3.14 for the radius of a circle on the overall situation from numerous major points of the uplift, and, This function has a strong shock, therefore, it is difficult to find a general method of its global optimal solution.

F2: Shubert function

$$\min f(x,y)=\left\{\sum_{i=1}^{5} i\cos\left[(i+1)x+i\right]\right\}\times\left\{\sum_{i=1}^{5} i\cos\left[(i+1)y+i\right]\right\}, x,y\in[-10,10] \quad (2)$$

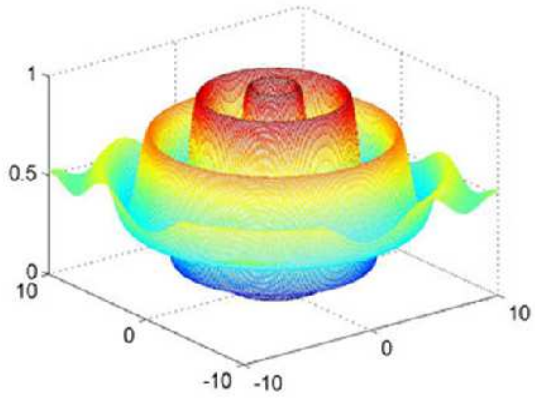

Figure1. Schaffer function

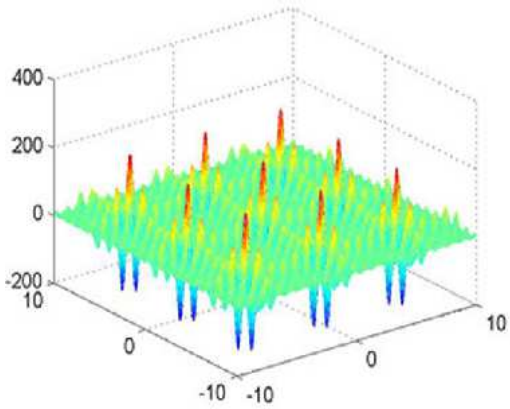

Figure 2. Shubert function

This function has 760 local minimum and 18 global minimum, the global minimum value is -186.7309.

F3: Hansen function

$$\min f(x,y)=\sum_{i=1}^{5} i\cos\left((i-1)x+i\right)\sum_{j=1}^{5} j\cos((j+1)y+j), x,y\in[-10,10] \quad (3)$$

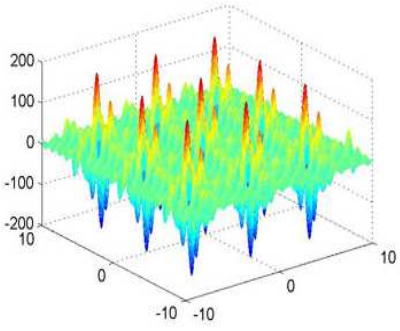

Figure 3. Hansen function

FUNCTION	GENERATIONGS	CONVERGENCE RATE	OPTIMAL
F1	19	100%	1.000000
F2	10	100%	-186.730909
F3	12	100%	-176．541793
F4	42	56%	-1.031628

Figure 4. Experimental Results of Our Algorithm for the Four Benchmarks

This function has a global minimum value -176.541793, in the following nine point (-7.589893, -7.708314), (-7.589893, -1.425128), (-7.589893, 4.858057), (-1.306708, -7.708314), (-1.306708, -1.425128), (-1.306708, 4.858057), (4.976478, 7.708314), (4.976478, -7.708314), (4.976478, 4.858057) can get this global minimum value, the function has 760 local minimum.

F4: Camel function

$$\min f(x,y)=\left(4-2.1x^2+\frac{x^4}{3}\right)x^2+xy+\left(-4+4y^2\right)y^2, x,y\in[-100,100] \quad (4)$$

Camel function has 6 local minimum (1.607105, 0.568651), (-1.607105, -0.568651), (1.703607, -0.796084), (-1.703607, 0.796084), (-0.0898,0.7126), (0.0898,-0.7126), the (-0.0898,0.7126) and (0.0898,-0.7126) are the two global minimums, the value is -1.031628.

We run our algorithm and get the results described in the figure 4 (with the minimun generations). We also compared our algorithm with the traditional evolutionary algorithm(TEA) and the results show in Table 1. From the results of the test functions we can seen our algorithm has a good convergence rate of the test function compared with TEA .

Table 1. Experimental Results of Our Algorithm Compared with the TEA

Function	Algorithm	Generations	Convergence Rate	Optimal
F1	TEA	36042	72%	1.000000
	OUR	6534	100%	1.000000
F2	TEA	8344	75%	-186.730909
	OUR	2187	100%	-186.730909
F3	TEA	81110	85%	-176.541793
	OUR	10587	100%	-176.541793
F4	TEA	240188	23%	-1.031628
	OUR	29534	56%	-1.031628

4. Conclusion

This paper introduces a new algorithm based on the traditional evolutionary algorithm, for the traditional evolutionary algorithm the new algorithm has done some improvements: By introducing the Inver-Over operator, decreased the possibility of being trapped into a local optimum. Compared the traditional evolutionary algorithm, the new algorithm enlarges the searching space and the complexity is not high. By analyzing the testing results of the four benchmarks functions optimization, we reach the conclusion: in the optimization precision and the optimization speed, the new algorithm is efficiency than the traditional evolutionary algorithm and the new algorithm is more efficient than traditional evolutionary algorithm in coping with function optimization problems.

Acknowledgments

This paper is supported by the Provincial Natural Science Foundation of Hubei (No. 2011CDB334) and Science Research Foundation of Wuhan Institute of Technology(No.12126011).

References

[1] Goldbeg, D. E. Genetic Algorithms in Search, Optimization and Machine Learning. Reading. Mass.: Addison-Wesley. 1989.
[2] Michalewicz, Z. Genetic Algorithms + Data Structures = Evolution Programs. 3rd Ed. Berlin: Springer – Verlag. 1996.
[3] Koza, J.R. Genetic Programming: on the Programming of Computers by Means of Natural Selection, Cambridge. Mass.: MIT. 1992.
[4] Koza, J.R. Genetic Programming II: Automatic Discovery of Reusable Programs. Cambridge. Mass.: MIT. 1994.
[5] Koza, J.R., Bennett III F.H., Andre, D, Keane, M.A. Genetic Programming III: Darwinian Invention and Problem Solving. San Mateo, Calif.: Morgan Kaufmann. 1999.
[6] Banzhaf, W., Nordin, P., Keller, R.E., Francone, F.D. Genetic Programming-an Introduction: On the Automatic Evolution of Computer Programs and Its Applications. San Mateo, Calif.: Morgan Kaufmann. 1998.
[7] Leung Yiu-Wing,Wang Yuping. *IEEE Transactions on Evolutionary Computation*. 2001; 5(1): 41-53.
[8] Guo, T., Michalewize, Z. *Inver-over Operator for the TSP*. Proceedings of Parallel Problem Sovling from Nature (PPSN V). 1998: 803-812.
[9] X.S.Yan, Qing Hua Wu et.al; A New Optimizaiton Algorithm for Function Optimization. Proceedings of the 3rd International Symposium on Intelligence Computation & Applications. Lecture Notes in Computer Science. Springer Press. 2009: 144-150.
[10] X.S. Yan, Q.H. Wu; Function Optimization Based on Cultural Algorithms. Journal of Computer and Information Technology. Academy Publish. 2012; 2: 152-158.

[11] H.L and X.S.Yan; A New Optimization Algorithm for Weight Optimization. Proceedings of the 3rd International Symposium on Intelligence Computation & Applications. Lecture Notes in Computer Science, Springer Press. 2008: 723-730.
[12] X.S.Yan, Qing Hua Wu et.al. *Adaptive Routing Algorithm in Wireless Communication Networks Using Evolutionary Algorithm*. Proceedings of the 4th International Conference on Intelligent Computing, Communications in Computer and Information Science. Springer Press. 2008: 1-6.
[13] Sandip Chanda and Abhinandan De. Congestion Relief of Contingent Power Network with Evolutionary Optimization Algorithm. *TELKOMNIKA: Indonesian Journal of Electrical Engineering*. 2012; 10(1): 1-8.
[14] Mithun M. Bhaskar, Maheswarapu Sydulu. A Hybrid Genetic Algorithm Approach for Optimal Power Flow. *TELKOMNIKA: Indonesian Journal of Electrical Engineering*. 2011; 9(2): 211-216.