Study of Quantum Genetic Algorithm Based on Mutative Scale Chaotic Optimization

Hao Teng[1,2] Baohua Zhao[2] Bingru Yang[1] Bin He[3]

[1]Knowledge Engineering Institute, University of Science and Technology Beijing, Beijing 100083, P. R. China
[2]School of Information Science and Engineering, University of Jinan, Jinan 250022, P. R. China
[3]Software faculty, Southwest Jiaotong University, Chengdu 610031, P. R. China

Abstract

Aiming at the trouble of easy getting into local minimum in quantum genetic algorithm, this paper presents a new hybrid quantum genetic algorithm. Using the method of mutative scale chaos optimization strategy, chaotic search for optimization is implemented to the colony which has been processed one time with quantum genetic algorithm, which can lead rapid evolution of the colony. The method has advantages such as high searching efficiency, good computing precision and being convenient to use, etc. The test of typical function shows the performance of this kind of method is better than quantum genetic algorithm and genetic algorithm.

Keywords: GA, QGA, Chaos optimization, Mutative scale

1. Introduction

Genetic algorithm which possesses very strong robust and adjustability is a direct search method. Though genetic algorithm has shown a lot of its tremendous advantages, it also exposes some limits such as large calculation storage, not converging to global optimization by all means, losing the best chromosome in the colony and premature convergence in the process of evolution. To resolve these troubles, Narayanan presented Quantum Genetic Algorithm (QGA), which carries out colony updating by the method of combining quantum bit coding with binary coding, and effectively settles the troubles of premature convergence and losing the best chromosome in GA [1]-[2]. A mass of literature shows that QGA is a kind of efficient parallel algorithm. But it is easy to get in local minimum, which obviously has the defect of premature convergence. Chaos movement can non-repeatedly cover all state in a certain range, according to its own rules. Thus, optimizing search with chaos variable has, without question, more advantage than random search [3]. This paper gives a new chaos quantum genetic algorithm, Mutative Scale Chaos Quantum Genetic Algorithm (MSCQGA), composed with mutative scale chaos optimization strategy and quantum genetic algorithm. Its characteristic is: to perpetually shrink optimizing variable's search space and regulation parameter, lead the colony to process a new turn evolution, and generate more advantageous optimal individual. Thereby, MSCQGA has improved the characteristic of QGA and effectively overcome the defects existed in QGA. The test of typical function shows that the performance of this kind of method is better than quantum genetic algorithm and genetic algorithm.

2. Mutative scale chaos optimization algorithm

2.1. Chaos optimization algorithm

In a way, Optimization Algorithm is the best choose to discuss the decision making problem. Through proper math modeling, the decision making can be equal to the study of seeking the global minimum and maximum in the state space. If objective function of the continuous object which needs optimization can be expressed as follows:

$$f^*=f(x_i^*)=\min f(x_i), \quad i=1, 2, \ldots, n, x_i \in [a_i, b_i] \quad (1)$$

Where, x is a decision making variable, a vector, whose dimension is equal to the number of the decision making parameters. f(x) is the math model and the objective function of the decision making problem. Chaos Optimization Algorithm is the usual method to resolve the problems like that.

In the Chaos Optimization Algorithm, logistic map are normally used to describe it:

$$x_{n+1} = \mu x_n (1 - x_n) \quad (2)$$

Where, μ is the control parameter, which can not bigger than 4, so we choose μ=4.

If $0 \le x_0 \le 1$, n=0, 1, 2,…. When μ=4, the system above is completely in chaos state. If optimization object can be expressed by formula (1) and the variable dimension is m, Chaos Optimization Algorithm's basic step usually is as follows:

Step1. Algorithm initialization: respectively endue x_0 in formula (2) with m initial numerical value which only have tiny difference between them, note to avoid choosing several especial values such as 0, 0.25, 0.5, 0.75 and 1, and get the count of m chaos variables with different track and gain the m numerical value $x_{i,n}$ by iterating for n times, where i=1, 2, …, m.

Step2. If k=0，$x_i(0)= c_i+d_ix_{i,n}$, c_i, d_i are constants, which are respectively equal to translating parameter and flexing parameter, ensuring that chaotic variable's changing range flexes the relevant optimization variable of formula (1) in its value rang, and substitute it into formula (1) to calculate the performance index $f(x(0))$ (following simply marked as $f(0)$), then order $x_i^* = x_i(0), f^*=f(0)$.

Step3. k:=k+1. With the following carrier wave method, substitutes the selected m chaos variable $x_{i,n+k}$ respectively into m numeral optimization variables in formula (1) using the formula (3), and makes it become the chaos variable $x_i(k)$:

$$x_i(k)=c_i+d_ix_{i,n+k} \tag{3}$$

Where, the meanings of c_i, d_i are equal to them in step2.

Step4. Processes iterative search with chaos variable. Substitutes $x_i(k)$ into formula (1) to calculate the performance index $f(k)$.

If $f(k)\geq f^*$ then $f^*=f(k)$, $x_i^*=x_i(k)$ Else if $f(k)<f^*$ then abandons $x_i(k)$.

If f^* are remained through some steps search, returns to step1 and repeatedly calculates to gain m numeral chaos variable $x_{i,n}$, and orders $k'=0$,then returns to step5. Or, returns to step3.

Step5. Makes use of formula (4) to process the second carrier wave.

$$x_i(k') = x_i^* + \alpha_i(x_{i,n+k} - 0.5) \tag{4}$$

Where, $x_i(k')$ is relative to the chaos variable of formula (1) in the less feasible field's ergodicity interval. Where, α_i is regulation parameter, x_i^* is the optimal solution in the coarse search stage.

Step6. Substitutes $x_i(k')$ into formula (2), continues to search by iterating, and calculates the relevant performance index $f(k')$.

If $f(k') \geq f^*$ then $f^*= f(k')$, $x_i^* = x_i(k')$

Else if $f(k') <f^*$, then abandons $x_i(k')$.

$k' := k'+1$.

Step7. If it satisfies the ending condition, stop searching, and outputs the optimal solution x_i^* , f^*. Or, returns to step5.

2.2. Mutative scale chaos optimization algorithm

The simulation result of abundant optimization examples shows that the optimizing efficiency of chaos optimization algorithm obviously overmatches that of others such as simulated annealing and genetic algorithm. However, further simulated calculation indicates that the effect of this algorithm is very prominent when the searching space is small, but not satisfied when big. So literature [4] puts forward Mutative Scale Chaos Optimization Algorithm. Its characteristics are as follows:

According to the search course, constantly shrinks the searching space of optimization variable.

According to the search course, constantly alters the regulation parameter of the second search.

The simulation calculation of several usual complicated test function shows that the algorithm overmatches that of literature [3].

Aiming at the problem existed in formula (1), Mutative Scale Chaos Optimization Algorithm's basic steps are as follows:

Step1 Algorithm initializing: if k=1, $k'=1$, $x_i(k)=x_i(0)$, $x_i^*=x_i(0)$, $f^*=f(0)$, $a_i(k')=a_i$, $b_i(k')=b_i$.

Step2 Maps the chaos variable to optimization variable's value interval, and searches according to common chaos optimization algorithm; order k=k+1, $x_i(k)=4x_i(k)(1-x_i(k))$ until f^* holds the line in a certain steps.

Step3 Alters the search scale of chaos variable. Where, regulation parameter $C\in(0, 0.5)$, mx_i^* is the current optimal solution.

Step4 Reverts the optimization variable.

$x_i^* = mx_i^* - a_i(k'+1)b_i(k'+1) - a_i(k'+1)$

Repeats the chaos search from step2 to step4 with new chaos variable $y_i(k)=(1-A)\, x_i^* + Ax_i(k)$ (A is the smaller one); order $k' = k'+1$ until f^* holds the line in a certain steps.

Step5 Ends after repeating the process of step3 and step4 for some times. Here, mx_i^* is the optimal variable and f^* is the optimal solution, obtained from the algorithm.

3. Mutative scale chaos quantum genetic algorithm (MSCQGA)

Improving Quantum Genetic Algorithm using Mutative Scale Chaos optimization strategy, there are four steps:

Step1 Initializing colony.

Generates initial colony $P(t) = \{p_1^t, p_2^t, ..., p_n^t\}$, where n is the size of the colony, and $p_j^t (j = 1, 2, ..., n)$ is one of the colony's t generation units, $p_j^t = [\alpha_1^t, \alpha_2^t, ..., \alpha_m^t, \beta_1^t, \beta_2^t, ..., \beta_m^t]$. m is the count of quantum bit, the length of quantum chromosome. When initializing, $\alpha = \beta = 1/\sqrt{2}$, which stands for linear summation in equal odds.

Step2 Completes a generic operation according to standard generic algorithm [5]-[6], and obtains more excellent colony.

S1 Constructs $R(t)=\{a^t_1,\ldots,a^t_n\}$ according to condition of the probability breadth value selected where a^t_j is a binary string whose length is m.

The generation mode is generating a random datum r in [0, 1]. If $|ai|^2 > r$, it is 1. Or it is 0.

S2 Evaluates each individual of $R(t)$, and keeps down the optimal individual. If ending conditions are satisfied, the algorithm ends, or continues.

S3 Judges if the colony needs alter. If it needs, returns to S5, or continues the algorithm.

S4 Be replaced with proper quantum gate, and gets new probability breadth.

S5 Order $t=t+1$, and returns to S1 to continue.

Step3 Picks out the unit whose value is more adaptive in Step2, and leads the colony fast evolve.

Step4 Repeats Step2 and Step3 until the ending conditions are satisfied, and that's all.

4. Example analyzing

Process optimization calculation with the several following test function using the MSCQGA given in this paper to check up its performances, which are compared with that of QGA and GA. Table 1 to Table 4 shows the result. All functions used only have one global optimal solution, but maybe have several local minimums.

$$F1=\left(4-2.1x_1^2+\frac{x_1^4}{3}\right)x_1^2+x_1x_2+\left(-4+4x_2^2\right)x_2^2,$$

$$-1\le x_1,x_2\le 1,\ g(x)=2\sin(2\pi x_2)-\sin(4\pi x_1)$$

$$F2=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6\right)^2+10\left(1-\frac{1}{8\pi}\right)\cos x_1+10,$$

$$-5\le x_1\le 10, 0\le x_2\le 15$$

$$F3=100\left(x_1^2-x_2\right)^2+\left(1-x_1\right)^2,\quad -2.048\le x_i\le 2.048,$$

i=1,2

$$F4=\left[1+\left(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2\right)\right.$$

$$\left.(x_1+x_2+1)^2\right]\cdot\left[30+(2x_1-3x_2)^2\right.$$

$$\left.(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)\right],$$

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

summation	GA		
	Optimal value	Average value	Mean-variance
50	-6.032	-6.0300	0.020
100	-6.032	-6.0300	0.019
150	-6.032	-6.0319	0.016
200	-6.032	-6.0325	0.014

(a)

summation	QGA		
	Optimal value	Average value	Mean-variance
50	-6.032	-6.0298	0.030
100	-6.032	-6.0300	0.019
150	-6.032	-6.0322	0.015
200	-6.032	-6.0326	0.012

(b)

summation	MSCQGA		
	Optimal value	Average value	Mean-variance
50	-6.032	-6.0300	0.020
100	-6.032	-6.0315	0.015
150	-6.032	-6.0323	0.013
200	-6.032	-6.0308	0.010

(c)

Table1: Test result of optimizing F1.

summation	GA		
	Optimal value	Average value	Mean-variance
50	0.3979	0.4073	0.0165
100	0.3979	0.4040	0.0108
150	0.3979	0.4030	0.0094
200	0.3979	0.4015	0.0080

(a)

summation	QGA		
	Optimal value	Average value	Mean-variance
50	0.3979	0.4070	0.0160
100	0.3979	0.4035	0.0110
150	0.3979	0.4020	0.0090
200	0.3979	0.4014	0.0082

(b)

summation	MSCQGA		
	Optimal value	Average value	Mean-variance
50	0.3979	0.4071	0.0163
100	0.3979	0.4036	0.0112
150	0.3979	0.4019	0.0091
200	0.3979	0.4014	0.0081

(c)

Table2: Test result of optimizing F2.

summation	GA		
	Optimal value	Average value	Mean-variance
50	0	0.0053	0.0024
100	0	0.0050	0.0020
150	0	0.0041	0.0018
200	0	0.0003	0.0009

(a)

summation	QGA		
	Optimal value	Average value	Mean-variance
50	0	0.0010	0.0010
100	0	0.0093	0.0009
150	0	0.0080	0.0007
200	0	0.0004	0.0003

(b)

summation	MSCQGA		
	Optimal value	Average value	Mean-variance
50	0	0.00014	0.0003
100	0	0.00009	0.0001
150	0	0.00003	0.0000
200	0	0.00003	0.0000

(c)

Table3: Test result of optimizing F3.

summation	GA		
	Optimal value	Average value	Mean-variance
50	3.0000	3.010	0.0030
100	3.0000	3.009	0.0010
150	3.0000	3.007	0.0004
200	3.0000	3.007	0.0005

(a)

summation	QGA		
	Optimal value	Average value	Mean-variance
50	3.0000	3.009	0.0011
100	3.0000	3.006	0.0009
150	3.0000	3.005	0.0009
200	3.0000	3.003	0.0004

(b)

summation	MSCQGA		
	Optimal value	Average value	Mean-variance
50	3.0000	3.000100	0.0000
100	3.0000	3.000100	0.0000
150	3.0000	3.000063	0.0000
200	3.0000	3.000048	0.0000

(c)

Table4: Test result of optimizing F4.

Mutative Scale Chaos Quantum Genetic Algorithm carries out search using its own rules of chaos variable, constantly shrinks the search space in the process of optimization, avoids local optimal solution and has very high searching efficiency. It inducts mutative scale chaos optimization algorithm to QGA not changing the searching mechanism of quantum generic algorithm, fastens the speed of the colony's evolvement, and greatly improves the performances of QGA. This algorithm has the characteristics of fast search speed, high calculation precision, simple structure, and convenience. The simulation result shows that the integrate performances of MSCQGA are better than GA and QGA. It has a very wide appliance foreground.

Acknowledgement

This work is partially supported by National Nature Science Foundation of China (Grant No. 60675030).

References

[1] A. Narayanan and M. Moore, Quantum inspired genetic algorithm, *Proc. of the International Conference on Evolutionary Computation*, pp. 61-66, 1996.

[2] G.X. Zhang, Y.J. Gu and L. Z. Hu, A novel genetic algorithm and its application to digital filter design, *Proc. of the International Conference on Intelligent Transportation Systems*, pp. 1600-1605, 2003.

[3] B. Li and W.S. Jiang, Chaos optimization method and its application, *Control Theory and Applications*, 14:613-615, 1997.

[4] T. Zhang, H.W. Wang and Z.C. Wang, Mutative Scale chaotic optimization algorithm and its application, *Control and Decision,* 14:285-288, 1999.

[5] S.Y. Yang, L.C. Jiao and F. Liu, The quantum evolutionary algorithm, *Engineering Mathematics*, 23:235-246, 2006.

[6] H.Y. Guo, Quantum genetic algorithm based on chaotic optimization, *Electronic Measurement Technology*, 29:14-18, 2006.