A Hybrid Evolution Algorithm with Application Based on Chaos Genetic Algorithm and Particle Swarm Optimization

Fu Yu
School of Computer & Information technology, Northeast Petroleum University, Daqing, 163318 China

Wang Bing*
School of Computer & Information technology, Northeast Petroleum University, Daqing, 163318 China

Xu Shao-Hua
School of Computer & Information technology, Northeast Petroleum University, Daqing, 163318 China

***Abstract*—Aiming at the complex function extreme value and non-linear system model parameters adjust, a hybrid optimization algorithm based on chaos GA combined with PSO is proposed in the paper. With application of applying experience of PSO, sharing information of GA, and traversing pathway of chaos, the adaptive switching of two algorithms are implemented through estimating the fitness and optimization efficiency, which may quickly obtain the global optimal solution. The proposed algorithm is applied to the function extreme optimizing and the parameter adjusting of fuzzy controller, and the experimental results show that the optimization ability of proposed algorithm is obviously superior to the single one, and that the integration of some intelligent optimization algorithms is a potential research direction.**

Key Words-chaotic genetic algorithm, particle swarm optimization, hybrid evolution algorithm, algorithm design, continuous optimization

I. INTRODUCTION

In recent years, the optimization techniques based on evolutionary computation are widely used to solve the optimal solutions of various engineering problems, which play an important role [1-5] in the fields of multi-objective optimization, experimental design and analysis, system identification and control. Since the principles and mechanisms of single evolutionary algorithm is generally proposed by a certain characteristic in the evolution of the simulated biological communities, and different algorithms consider the issue from different angles, single algorithm shows the performance advantages in a certain aspect and the disadvantages of principles and strategies in other certain aspects [6, 7]. For examples, be easy to fall into local optimum, cannot search in the global solution space, fail to meet the requirements of multi-objective constraints, etc.

Chaos genetic algorithm (CGA) is an optimization algorithm that combines evolutionary mechanism of genetic algorithms with chaos search strategy, and has properties of groups search and track traversing, but there are defects that the degree of global information sharing is low [8,9]. Particle swarm optimization (PSO) is an evolutionary computation technique based on swarm intelligence theory, to set up the simulation of biological predation phenomenon in nature based on of the evolution of computing technology which simulates biological predation phenomenon in nature. It solves optimization problems through individual cooperation and competition in the population, which can remember the best particle position and share information between the particles, but the rate of convergence is slow [10]. Therefore, If can find a hybrid algorithm combined CGA with PSO, the ability of complex problem solving and adaptability can be improved.

Aiming at the problems of complex function calculation and parameter optimization of nonlinear systems, a Hybrid Evolution Algorithm based on Chaos Genetic Algorithm (CGA) and Particle Swarm Optimization (PSO) is proposed in the paper. The algorithm combines chaotic variables with parameters to be optimized and sets the traversing range of chaotic motion in the feasible solution space of optimization variables, which builds the adaptive adjustment mechanism based on fitness assessment and efficiency analysis to switch both algorithms of CGA and PSO to solve Global optimization solution in the feasible solution space of problems. The hybrid optimization strategy and detailed algorithm steps of CGA-PSO are given in the paper. The examples of function extremum solving and parameters adjustment of fuzzy controllers are used as experiments, and the results show that the method of CGA-PSO is better than that of GA or PSO.

II. CHAOTIC GENETIC ALGORITHM AND PARTICLE SWARM OPTIMIZATION

A Chaotic Search

Chaos is a nonlinear phenomenon common in nature, which has properties of intrinsic random, orbit ergodicity and implicit rules. Chaotic search is that chaotic states are introduced to the optimization variables, which can traverse all states repeatedly according to the laws of the system itself within a certain range. The method has good adaptability in mechanism to determine the global optimal solution in the feasible solution space of problems.

Consider a chaotic search strategy based on the insect population model, of which logistic map is a chaotic sequence generator [11], and chaotic state is introduced to the optimization variables. The Iterative equation is as follows:

$$\delta^{j+1} = u\delta^{j}(1-\delta^{j}) \quad (1)$$

where u is a chaotic attractor. When $u = 4$, the system becomes into a chaotic state, a chaos variable of δ^{j} is emerged, which changes in the interval of $[0,1]$.

B Chaotic Genetic Algorithm

Chaos Genetic Algorithm based on orbit ergodicity of chaotic variables movement evolutionary mechanisms of Genetic Algorithms, combines chaotic search properties with parameters to be optimized, and encode the chaotic variables, which are represented as chromosomes and are placed in the environment of the problem to select, copy, chaotic cross and chaotic mutation according to the principle of survival of the fittest. According to the evolving of evolutionary iterations repeatedly, the optimal solution is obtained, which is converged to the individual on the most suitable environment finally.

C The basic PSO algorithm

The basic PSO algorithm [12] can be described as follows: Set that there is a population composed of *m* particles in n-dimensional space, the location and the speed of the *i* th particle is X_i and V_i, the local optimal locations of P_i are $X_i = (x_{i1}, x_{i2}, \cdots, x_{in})$ and $V_i = (v_{i1}, v_{i2}, \cdots, v_{in})$, and the global optimum positions of P_g are $P_i = (p_{i1}, p_{i2}, \cdots, p_{in})$ and $P_g = (p_{g1}, p_{g2}, \cdots, p_{gn})$. The update strategies of particle states are as follows:

$$V_i(t+1) = wV_i(t) + c_1 r_1 (P_i - X_i(t)) + c_2 r_2 (P_g - X_i(t)) \quad (2)$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (3)$$

Where $i = 1,2,\cdots,m$, w is the inertia factor, c_1 and c_2 are constants, r_1 and r_2 are random numbers in the interval of $[0,1]$. Loop iterate each particle of the population into Eq. (2) and Eq. (3), the whole population can approach to the global optimal solution gradually.

III. THE HYBRID ALGORITHM OF CGA AND PSO

The hybrid optimization strategy of Chaos genetic and particle swarm is to combine the advantages of PSO with those of CGA, which sets fitness assessment and analysis rule of algorithm efficiency to realize adaptive alternating iteration of PSO and CGA in control guidelines and achieve the purpose of Natures complement each other and global optimization.

A The rules of hybrid optimization algorithm switching

Combined the changes of objective function values with operating efficiency of algorithms, two self-adaptive switching rules are proposed as follows:

Set the switching threshold is θ, the algebraic interval is ΔG, the increment value of objective function after iterations of ΔG times is ΔF. If $\frac{|\Delta F|}{\Delta G} < \theta$, switch the algorithm. According to the above rules, the algorithm will not be switched until satisfying the condition of meeting the optimization precision or the maximum evolution times.

B The implementation steps of Hybrid optimization algorithm

Firstly, the algorithm realizes evolution iteration of PSO or CGA with a certain population size and checks the fitness and the efficiency of the algorithm at the same time. When the termination condition is satisfied, choose the particles whose fitness is better as the objects of next round and choose other particles randomly as supplement populations. When reaching the given size, the other algorithm is chosen to iterate continually. The implementation steps of Hybrid optimization algorithm are described as follows:

Step1 Determine the population size N .Generate the initial population G^0 randomly, and encode the chromosomes with decimal numbers, on which the number of genes is the number of variables to be optimized. Set the switching threshold θ, the algebraic interval ΔG.

Step 2 Build the fitness function. For the function minimum optimization problem of F, the fitness function is chosen as $fit = e^{-F}$.

Step 3 Implement of the PSO algorithm

(1) PSO initialization.

(2) Iterate Eq. (2) and Eq. (3) ΔG times, and calculate the changes ΔF of the objective function.

(3) If $\frac{|\Delta F|}{\Delta G} < \theta$, turn to Eq. (4). Otherwise, turn to Eq. (2).

(4) If satisfies the termination condition, save and stop. Otherwise, choose the particles whose fitness is better than others and other particles as supplement populations until reaching the given size of N, a new population of G^1 is generated. Turn to step 4 and switch to CGA.

Step 4 Implement CGA searching

(1) Iterate ΔG times as the following steps, and calculate the changes ΔF of the objective function.

selection operations：Choose chromosomes by runner rules, of which the selection probability is proportional to its fitness.

Chaotic cross: Two chromosomes are combined as follows: $W_1' = \vec{\lambda} W_1 + (\vec{1} - \vec{\lambda}) W_2$,

$W_2' = \vec{\lambda} W_2 + (\vec{1} - \vec{\lambda}) W_1$, where $\lambda_i \in (-1, 1)$ are chaotic

variables. At first, define a cross-amplitude of λ_k, then determine λ_i according to Eq. (4). In order to show the bidirectionality, the chaotic variable of δ_i^{j+1} are determined according to Eq. (5).

$$\lambda_i = \lambda_k \delta_i^{j+1} \tag{4}$$

$$\delta_i^{j+1} = 8\delta_i^j(1-\delta_i^j)-1 \tag{5}$$

Mutation operations: Set the mutated gene w_i, the following as $w_i^{'} = w_i + \beta(w_i^U - w_i)$ or

$w_i^{'} = w_i + \beta(w_i - w_i^L)$, where w_i^U is the upper bound, w_i^L is the lower bound, and β is the chaotic variable that changes in the interval of $(-1,1)$. Mutation operations need to define variation amplitude of $\tilde{\lambda}_k$ at first, and then chaos variables are introduced. For m individuals elected to be variated, sort in ascending order of fitness. For the *k* th individual, variation amplitude of $\tilde{\lambda}_k$ can be chosen as Eq. (6), where $\tilde{\lambda}_0$ is the parameter that controls the disturbance size, and β searches in the interval of $[-\tilde{\lambda}_k, \tilde{\lambda}_k]$.

$$\tilde{\lambda}_k = \tilde{\lambda}_0 \exp((m-k)/m) \tag{6}$$

If satisfies the termination condition, save the optimal solution and stop. Otherwise, select the particles whose adaptation degree is before 50%, supply the others to the population in the feasible solution space randomly, make the population reach the given size of *N*, and generate a new population G^1. Go to Step 3.

(2) If $\frac{|\Delta F|}{\Delta G} < \theta$, turn to Eq. (3). Otherwise, turn to Eq. (1).

(3) If satisfies the termination condition, save and stop. Otherwise, choose the particles whose fitness is better than others and other particles as supplement populations until reaching the given size of N, a new population of G^2 is generated. Turn to step 3, and switch to PSO.

With application of sharing information and traversing pathway of chaos from the hybrid algorithm of CGA and PSO, the advantages of both algorithms are shown and the optimization efficiency is improved.

IV. Analysis of Experiment Results

A *Function global optimization*

Use the following four standard test functions and verify the performance of the algorithm proposed in the paper.

(1) Rosenbrock function

$$f_1(x) = \sum_{i=1}^{n-1}[100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2], x_i \in [-30, 30] \tag{7}$$

(2) Rastrigin function

$$f_2(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i)), x_i \in [-5.12,\ 5.12] \tag{8}$$

(3) Griewank function

$$f_3(x) = \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1, x_i \in [-600,\ 600] \tag{9}$$

(4) Ackley function

$$f_4(x) = 20 + e - 20e^{-\frac{1}{5}\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}} - e^{-\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}, x_i \in [-32, 32] \tag{10}$$

An average of the global minimum of the above four functions is 0. When the dimension of n is 10, population size of GA and PSO are taken 20, and the total optimization times are 1000. When the dimension of n is 20, population size of GA and PSO are taken 40, and the total optimization times are 1500.

Set the algebraic interval $\Delta G = 100$ and the switching threshold $\theta = 10^{-3}$. In order to reflect the efficiency of CGA-PSO, contrast with the GA and PSO in the paper, of which parameters are the same as those of CGA-PSO. Each algorithm runs 10 times, and the optimization results are shown in Table 1. The average switching times are shown in Table 2.

TAB.1 Comparison of Optimization Results among CGA-PSO, GA and PSO

functions	Dimension	CGA-PSO		GA		PSO	
		average results	average time	average results	average time	average results	average time
Rosenbrock	10	9.1371	0.0870 s	38.1913	0.1195	13.1932	0.0865
	20	19.9836	0.5359 s	35.3267	0.5850	28.8323	0.5149
Rastrigin	10	4.5131	0.0891 s	5.5673	0.1167	4.7863	0.0514
	20	11.7323	0.5036 s	13.3206	0.5236	21.3615	0.4639
Griewank	10	0.0785	0.0806 s	0.0753	0.0905	0.0908	0.0346
	20	0.0176	0.5386 s	0.0203	0.5699	0.0216	0.4964
Ackley	10	5.3633×10^{-13}	0.0894 s	0.3132	0.1261	0.0115	0.0710
	20	3.2721×10^{-13}	0.5217 s	0.1791	0.5405	0.0323	0.4907

TAB.2 CGA-PSO ALGORITHM SWITCHING TIMES

Rosenbrock		Rastrigin		Griewank		Ackley	
10	20	10	20	10	20	10	20
2.1	2.7	1.7	2.5	2.2	2.8	2.5	3.2

We can see form the optimization results in table 1 that CGA-PSO is better than GA or PSO. We can see form the average time in table 2 that CGA-PSO is better than GA but worse than PSO. The above results can be explained as follows: for optimization results, since CGA-PSO in the optimization process achieved the adaptive switching of both algorithms to complement advantages of each other when an algorithm is slightly stagnant immediately switch to the other algorithm. For average time, since the number of iterations of the three models is the same, the run-time of CGA-PSO is approximate to the weighted average of that of each algorithm, and the calculation of chaos search of CGA-PSO is small, the average time of CGA-PSO is between that of each algorithm. In fact, CGA-PSO is to increase the running time of PSO for the price, to exchange for the optimization of the performance improvement.

B parameters optimization of Fuzzy controller

In the design of fuzzy controller, the control action of u is determined by the error and the change of the error. In order to satisfy requirements of different controlled objects, an adjustment factor of α is introduced. We can get the fuzzy control rules described by the adjustment factors as follows:

$$u = -<\alpha E + (1-\alpha)EC> \quad \alpha \in (0,1) \qquad (11)$$

By adjusting the size of α, we can change the weighted degree of the error and the change of the error. In the design of fuzzy controller, since relying on a fixed weighted factor is usually difficult to meet the requirements, we consider that introducing different weighted factors in different error levels to adaptive the adjustment of fuzzy control rules. Set a second-order system of $\frac{20}{(2s+1)(4s+1)}$ for the controlled object, the input as step signals, the error, the change of the error, the domain of controller as $\{E\}=\{EC\} \ =\{u\}=\{-3,-2,-1,0,1,2,3\}$,and the control rules as $u=-<\alpha E+(1-\alpha)EC>$, where $\alpha \in (0,1)$.

Consider that the fuzzy control system in different states has different requirements in the control rules of α rules, α are divided into three levels

$$u = \begin{cases} -<\alpha_1 E + (1-\alpha_1)EC> & E = 0, \pm 1 \\ -<\alpha_2 E + (1-\alpha_2)EC> & E = \pm 2 \\ -<\alpha_3 E + (1-\alpha_3)EC> & E = \pm 3 \end{cases} \qquad (12)$$

Therefore, the six fuzzy controller parameters that need optimizing at the same time are ke、kc、ku、α_1、α_2、α_3, where ke and kc are the quantization factor of the error and the change of the error, ku is the scale factor of the output of the controller.

Use the ITAE integral performance to design evaluation function of $\frac{1}{a+J(\text{ITAE})}$, where $J(\text{ITAE}) = \int_0^\infty t\left|e(t)\right|dt$. In order to make the denominator is not zero, set a as a small positive number. Obviously, when the tracking error is smaller, the value of evaluation function is greater. Based on the experience, the range of the six controller parameters to be optimized is determined as follows: ke、kc、$ku \in [0,10]$; $\alpha_1 \in [0,\ 0.4]$; $\alpha_2 \in [0.4,\ 0.8]$; $\alpha_3 \in [0.8, 1.0]$. This experiment will also contrast CGA-PSO with GA and PSO.

Set the population size of CGA-PSO, GA and PSO are taken 20, the total optimization times of CGA-PSO are 100, the total optimization times of GA and PSO are 500, the algebraic interval $\Delta G = 20$ and the switching threshold $\theta = 0.05$. Table 2 is the optimization results comparison of the three algorithms.

The results comparison of the corresponding step response curve is shown in Figure 1, the evolution curve comparison of the evaluation function value shown in Figure 2.

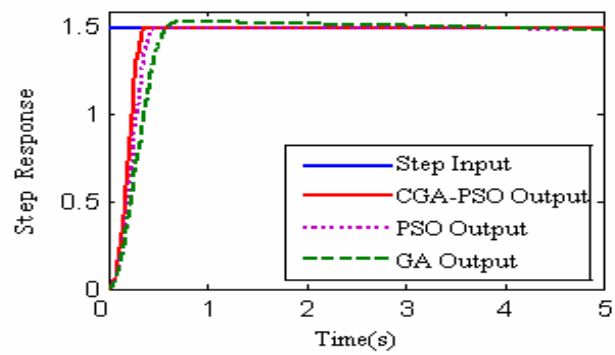

Fig. 1Tracking Response Comparison Curves of Optimization Results

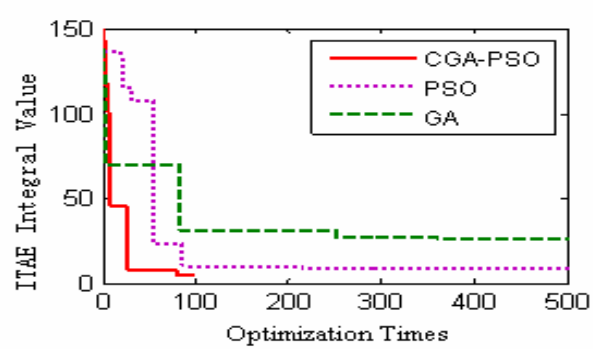

Fig. 2 Index Comparison Curves in Optimization Process

TAB.3 PARAMETER OPTIMIZATION COMPARISON OF FUZZY CONTROLLERS AMONG CGA-PSO, GA AND PSO

Algorithms	α_1	α_2	α_3	ke	kc	ku	J (ITAE)
CGA-PSO	0.0389	0.4813	0.8193	6.3156	5.2938	4.4867	5.6318
PSO	0.0401	0.5133	0.8259	6.2628	4.9899	3.9983	9.0636
GA	0.3870	0.5775	0.8164	2.1517	0.2720	5.6736	26.2406

Figure 1 is shown that there is little difference with the performance of three controllers. After CGA-PSO optimization, the speed of the controller is faster and the overshoot is smaller. However, consider the optimization times of GA and PSO is five times of those of CGA-PSO, CGA-PSO has stronger searching capability and the convergence speed and optimization performance of CGA-PSO is significantly better than that of GA and PSO, which shows that there is greater value of CGA-PSO to promote. Figure 2 is shown that: For CGA-PSO, after CGA optimization ΔG times, we can get $\Delta ITAE / \Delta G = 8.0293 > \theta$. Continue to use CGA, after CGA optimization ΔG times, we can get $\Delta ITAE / \Delta G = 1.8668 > \theta$. Continue to use CGA, after CGA optimization ΔG times, we can get $\Delta ITAE / \Delta G = 0.0025 < \theta$. Switch to PSO, use PSO, after PSO optimization ΔG times, we can get $\Delta ITAE / \Delta G = 0.1390 > \theta$. Switch to PSO, Continue to use PSO, after PSO optimization ΔG times, stop the optimization.

We can see form the above results: Early in the algorithm run, it is relatively easy to optimize, when the ITAE index value drops significantly, which results in the algorithms can not switch in the first two rounds, enter into the irregular surface area to increase the difficulty of searching optimization, stop in the third round, and switch the algorithm at last.

V. CONCLUSIONS

A hybrid optimization algorithm based on chaos GA combined with PSO is proposed in the paper. The experiment results show that the optimization ability of proposed algorithm is obviously superior to the single one. However, since the algorithm combines the chaotic mechanism, the algorithm traversing capabilities are improved, but it depends on the initial values at the same time. How to improve the robustness of the algorithm, is the next research we need to study.

REFERENCES

[1] Ma Ruixin, Liu Yu, QinZheng, Wang Xiao. Momentum particle swarm optimizer for constrained optimization [J]. Journal of System Simulation, 2010, 22 (11):2485-2488.

[2] Feng Zhenping,Li Jun,Shen Zyda. Application of Genetic Algorithm To Design For Turbine machinery [J].Gas Turbine Technology, 1997,11(2):13-22

[3] Li Jun, Feng Zhenping. Aerodynamic Optimum Design of Transonic Turbine Cascades Using Genetic Algorithms [J]. Journal of Thermal Science，1997,6(2):364-368.

[4] Tong Tong, Feng Zhen ping, Li Jun. Application of Genetic Algorithm To Multi-objective Optimization Design For Turbine Cascades [J]. Proceedings of t he CSEE，1999,19 (6):74-76

[5] Miao Mingfei, Zhang Yongliang, Ma Jiming. Multi-objective Evolutionary Optimization of Large Differential Surge Chamber [J]. Journal of hydroelectric engineering, 2010, 29 (1) :57-61.

[6] Gong Maoguo, Jiao Licheng, Yang Dongdong, Ma WenPing. Research on Evolutionary Multi-Objective Optimization Algorithms [J]. Journal of Software, 2009, 20 (2) :271-289.

[7] KONG Wei-jian, DING Jin-liang, CHAI Tian-you. Survey on large-dimensional multi-objective evolutionary algorithms [J]. Control and Decision，2010,25 (3) :321-325.

[8] Chen Bingrui, Yang Chengxiang. Self-Adapting Chaos-Genetic Hybrid Algorithm and Sensitivity Analysis of Its Parameters [J]. Journal of Northeastern University, 2006,27 (6) :689-692.

[9] Li Bing, Jiang Weisun. Chaos Optimization Method and Its Application [J]. CONTROL THEORY AND APPLICATIONS, 1997, 14(4): 613-615

[10] SU Shou-bao, WANG Ji-wen, FANG Jie. Overview Applications and Research on Particle Swarm Optimization Algorithm [J]. COMPUTER TECHNOLOGY AND DEVELOPMENT , 2007,17(5):249-253.

[11] WU Tie-bin,CHENG Yun, ZHOU Tao-yun,YUE Zhou. Optim ization Control of PID Based on ChaosGenetic Algorithm [J]. Computer Simulation, 2009,26(5):202-205

[12] Cai X.J.,Cui Z.H.,Zeng J.C., et al. Particle Swarm Optimization with Self-adjusting Cognitive Selection Strategy [J]. International Journal of Innovative Computing, Information and Control, 2008, 14(4): 943-952.