Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, 26-29 August 2004

HYBRID PARTICLE SWARM OPTIMIZATION WITH SIMULATED ANNEALING

XI-HUAI WANG, JUN-JUN LI

Department of Electrical and Automation, Shanghai Maritime University, Shanghai 200135, China
E-MAIL: wxh@shmtu.edu.cn

Abstract:

Particle swarm optimization is a recently invented intelligent optimizer with several highly desirable attributes. In this paper, a hybrid particle swarm optimization is proposed. This method integrates the particle swarm optimization with the simulated annealing. The method is applied to six test functions' optimization and the simulation shows that the performance of this algorithm is better than that of the adaptive particle swarm optimization and the genetic chaos optimization.

Keywords:

Particle swarm optimization; simulated annealing; function optimization

1. Introduction

Particle Swarm Optimization (PSO) is a new method of evolutionary computation. It was first introduced by Doctor Eberhart and Doctor Kennedy[1][2]. PSO is a kind of simulation of the movement and flocking of birds. And we can also say it is a kind of simulation of social psychology. PSO has a strong ability to find the most optimistic result. Meanwhile it has a disadvantage of local minimum. After modulating parameters, the rate of convergence can be speeded up and the ability to find the global optimistic result can be enhanced.

Simulated annealing (SA) is a kind of global optimization technique based on annealing of metal[3]. It can find the global minimum using stochastic searching technology from the means of probability. Simulated annealing algorithm has a strong ability to find the local optimistic result. And it can avoid the problem of local minimum, but its ability of finding the global optimistic result is weak.

This paper introduces an adaptive particle swarm optimization algorithm (APSO) firstly[4][5]. Then the SA-based particle swarm optimization (SAPSO) is proposed after combining the PSO with the SA. These two algorithms are applied to function optimization problem and simulation shows that the SAPSO is more effective than APSO.

2. Adaptive particle swarm optimization

PSO is a kind of algorithm searching the best answer by simulating the movement and flocking of birds. The algorithm initialized the flock of birds randomly over the searching space, every bird is called as a "particle". These "particles" move with a certain law and find the global best result after some iteration. At each iteration, each particle adjusts its velocity vector, based on its momentum and the influence of its best solution (P_{best}) and the best solution of its neighbors (G_{best}), then computes a new point to examine. The original PSO formulae are:

$$v[] = w \cdot v[] + c_1 \cdot rand() \cdot (G_{best}[] - present[]) + c_2 \cdot rand()(P_{best} - present[]) \quad (1a)$$

$$present[] = present[] + v[] \quad (1b)$$

where $v[]$ is the velocity vector, $n \times d$ matrix; n is the number of the particles; d is the number of channels (variables); c_1, c_2 are the acceleration constants and positive constants; $rand$ is a random number between 0 and 1; $present[]$ is the location vector; w is the inertia weight.

In addition to the c_1, and c_2 parameters, implementation of the original algorithm also requires placing limits on the velocities (V_{max}). After adjusting the parameters w and V_{max}, PSO can obtain the best ability of searching.

APSO is based on basic PSO algorithm. it adjusts the parameter w and makes w reduce gradually as the generation increasing. In the searching process of the PSO algorithm, the searching space will reduce gradually as the generation increasing. So the APSO algorithm is more effective. Because the searching space reduces step by step, not linearly, so the parameter w here also reduce s step by step.

0-7803-8403-2/04/$20.00 ©2004 IEEE

3. SA-based particle swarm optimization

The SAPSO is an optimization algorithm combining the PSO with the SA. PSO has a strong ability finding the most optimistic result. Meanwhile it has a disadvantage of local minimum. SA has a strong ability finding the local optimistic result. And it can avoid the problem of local minimum. But it' ability finding the global optimistic result is weak. Combining PSO and SA, learning from other's strong points to offset one's weaknesses each other, this is the basic idea of the SAPSO.

Similar with the APSO algorithm, the SAPSO algorithm's searching process is also started from initializing a group of random particles. First, each particle is simulated annealed independently and a group of new individuals are generated. Then, particles of the new generation are obtained after transforming each particle's velocity and position according to the equation (1). This process repeats time after time until the terminating condition is satisfied.

The parameter w in the SAPSO algorithm also reduces gradually as the iterative generation increasing just as the APSO algorithm. In the process of simulated annealing, the new individuals are given randomly around the original individuals. Here we set the changing range of original particles as a parameter $r1$, to each particle:

$$present = present + r1 - r1 \cdot 2 \cdot rand(1) \tag{2}$$

where rand(1) is a random number between 0 and 1, the parameter $r1$ here also reduces step by step as the generation increasing just like w.

To the continuous global minimum optimizing problem

$$\begin{aligned} &\min f(x_1, x_2, \cdots, x_n) \\ &x_i \in [a_i, b_i], \qquad i = 1,2,\cdots,n \end{aligned} \tag{3}$$

The steps of the simulated annealing particle swarm optimization is:

(1) initialize a group of particles (the scale is m), including random position and velocity;

(2) evaluate each particle's fitness;

(3) each particle is simulated annealed independently (where the parameter $r1$ reduces gradually as the generation increasing) and a group of new individuals are obtained;

(4) for each particle, compares its fitness and its personal best position P_{best}, if its fitness is better, replace P_{best} with its fitness;

(5) for each particle, compares its fitness and the global best position G_{best}, if its fitness is better, replace G_{best} with its fitness;

(6) transform each particle's velocity and position according to the expressions (1) (where the parameter w reduces gradually as the generation increasing).

4. Optimization example

APSO, SAPSO, and GCO (genetic chaos optimization combination algorithm[6]) are applied to six testing functions optimization problem and then the results are compared.

$$F_1 = 100(x_1^2 - x_2^2) + (1 - x_1)^2$$
$$-2.048 \le x_i \le 2.048$$

$$F_2 = 4 + 4.5x_1 - 4x_2 + x_1^2 + 2x_2^2 - 2x_1x_2 + x_1^4 - 2x_1^2x_2$$
$$-8 \le x_i \le 8$$

$$F_3 = (x_1^2 + x_2^2)^{0.25}[\sin^2(50(x_1^2 + x_2^2)^{0.1}) + 1.0]$$
$$-100 < x_i < 100$$

$$F_4 = (4 - 2.1x_1^2 + \frac{x_1^4}{3})x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$$
$$-100 < x_i < 100$$

$$F_5 = 0.5 - \frac{\sin^2\sqrt{x_1^2 + x_2^2} - 0.5}{(1 + 0.001(x_1^2 + x_2^2))^4}$$
$$-100 < x_i < 100$$

$$F_6 = \sum_{i=1}^{5} \mathrm{int}\,eger(x_i)$$
$$-5.12 \le x_i \le 5.12$$

Where F_5 has the global maximum, others have the global minimum.

These three algorithms are applied to these six testing functions optimization problem in MATLAB when they have same generations. The simulation results are showed in table 1.

We can see in table 1, the SAPSO' constringency is much better than the GCO algorithm. The SAPSO algorithm can obtain much better result when they have same generations.

Take function F_1 as an example, the changing charts of the best result and the best particle with increasing of the generations are plotted respectively with SAPSO and APSO in figures 1, 2.

Table 1. Results compared SAPSO with GCO

No.	Global value	GCO			SAPSO		
		Worst value	Best value (%)	Average value	Worst value	Best value (%)	Average value
F_1	0	2.05×10^{-21}	99	1.27×10^{-23}	4.93×10^{-24}	100	6.15×10^{-25}
F_2	-0.5134	-0.5134	100	-0.5134	-0.5134	100	-0.5134
F_3	0	2.05×10^{-19}	98	5.22×10^{-22}	1.62×10^{-26}	100	3.49×10^{-28}
F_4	-1.316	-0.9210	95	-1.295	-1.316	99	-1.316
F_5	1.00	0.9903	95	1	1.00	99	1.00
F_6	-30	-30	100	-30	-30	100	-30

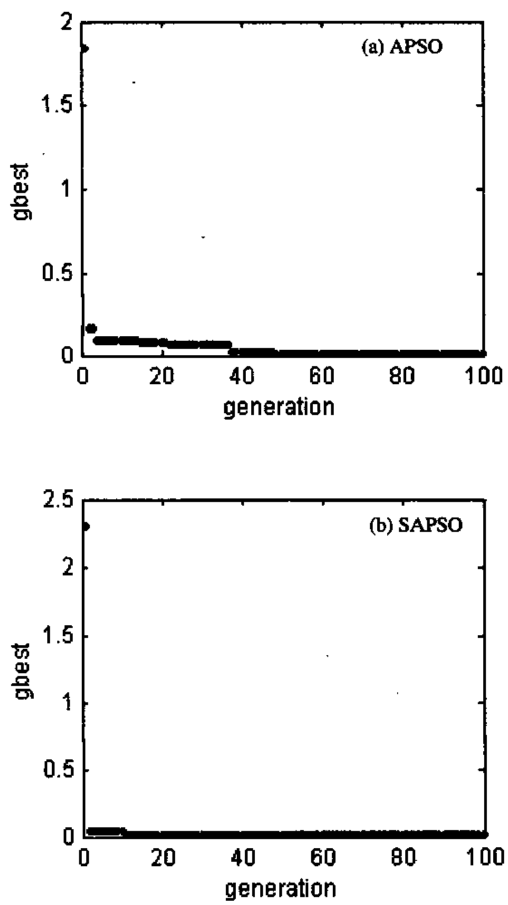

Figure 1. The best result of F_1 with generations

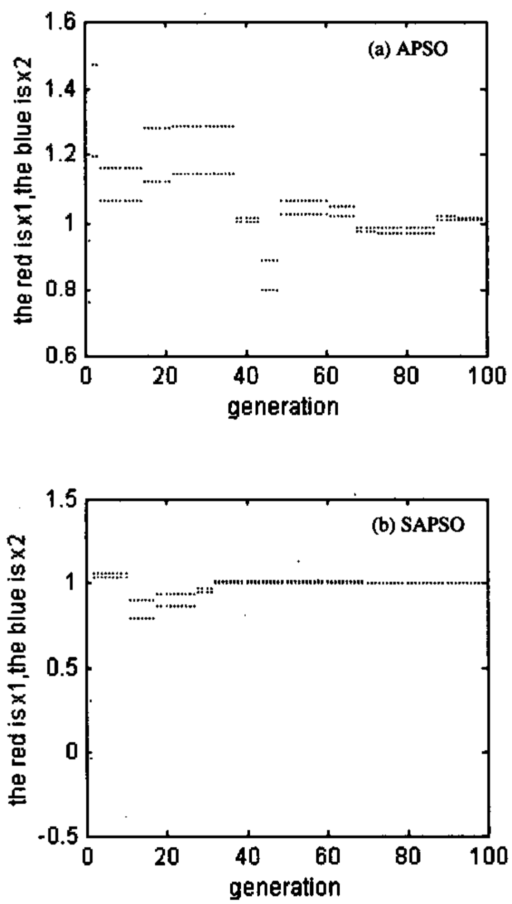

Figure 2. The best particle of F_1 with generations

We can see in the figures 1 and 2, the SAPSO constringency is faster and better. Especially in earlier stage, the constringency of SAPSO is more evident.

5. Conclusions

Combining PSO and SA, learning from other's strong points to offset one's weaknesses each other, the Simulated Annealing Particle Swarm Optimization can narrow the field of search and speed up the rate of convergence continually in the optimizing process. It has higher searching efficiency. It can also escape from the local minimums. These two algorithms are applied to several test functions optimization problem and simulation shows that the SAPSO algorithm is much better. The SAPSO algorithm is easy to carry out. It is a potential optimization method.

Acknowledgements

This work was supported by Science Foundation of Shanghai Education (02IK01) and Key Subject Construction Foundation (No. 4) of Shanghai Municipal Education Commission.

References

[1] Eberhart R.C., and Kennedy J., "A new optimizer using particles swarm theory", Proceeding of Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, pp. 39-43, 1995.

[2] Eberhart R.C., and Kennedy J., "Particles swarm optimization", Proceeding of IEEE International Conference on Neural Network, Perth, Australia, pp. 1942-1948, 1995.

[3] Kirkpatrick S., Gelatt C.D. and Vecchi M.P., "Optimization by simulated annealing", Science, Vol. 220, pp. 671-680, 1983.

[4] Shi Y., Eberhart R.C., "A modified particle swarm optimizer", Proceeding of IEEE World Congress on Computation Intelligence, pp. 69-73, 1998.

[5] Michalewiicz Z., and Schoenauer M., "Evolutionary algorithms for constrained parameter optimization problems", Evolutionary Computation, Vol. 1, No. 4, pp. 1-32, 1996.

[6] Li Y.D., Li S.Y., "A new genetic chaos optimization combination method", Control Theory and Application, Vol. 19, No. 1, pp. 143-145, 2002.