Center Particle Swarm Optimization Algorithm

Yang Xiaojing, Jiao Qingju, Liu Xinke

College of Computer and information engineering, Anyang Normal University, Anyang 455000

yangxj84@163.com, 2875085969@qq.com

Abstract—**The linear decreasing weight particle swarm optimization algorithm (LDWPSO) is mentioned in the concept of a center particle, and then puts forward center particle swarm optimization algorithm (PSO). The linear decreasing weight particle swarm optimization algorithm, unlike other general center particle, particle velocity center is not clear, and is always placed in the center of the particle swarm. In addition, the neural network training algorithm compared to particle swarm optimization algorithm and the linear decreasing weight particle swarm optimization algorithm, results show that: the performance is better than the linear optimization center particle swarm decreasing weight PSO algorithm. algorithm.**

Keywords: particle swarm optimization algorithm; neural network; evolutionary computation

I. Introduction

PSO algorithm is a kind of method like social behavior and imitate birds and fish evolved from computing technology[1-3], PSO algorithm has good convergence and good performance in nonlinear function optimization. So it is getting more and more attention. Many researchers are committed to improving his performance with a variety of different methods and advanced interesting variables. These methods are broadly classified into the following categories.

The improved method is to add a new coefficient to the velocity and position equation of the particle swarm optimization algorithm[4]. Finally, the coefficient should be selected reasonably. Angeline points out that the local search capability of the elementary particle swarm optimization algorithm is very low. To overcome this shortcoming, Shi and Eberhart proposed a LDWPSO algorithm[5], which introduced linear decrement inertial factor into the velocity update equation of the basic PSO algorithm.

Because inertial factors effectively balance the global and local searching ability of particle swarm. The performance of particle swarm optimization algorithm is improved and its efficiency is greatly improved[6]. A key feature of particle swarm optimization is the sharing of social information among particles in a region. Therefore, various information sharing methods are proposed to improve the performance of the algorithm[7]. Kennedy studies various topological structures that affect the performance of the algorithm and points out that the von Neumann topology has better performance. Suganthan proposed a variable neighborhood, which in the initial period of optimization was the neighborhood of a single particle itself. As this particle algebra grew, the neighborhood gradually extended to all particles. Mohai puts forward a dynamic adaptive neighborhood. This dynamic neighborhood adaptation can be generated at random, and directly construct the topological structure of the original value population. On the edge of the population structure in the process of running a population is freely from one point to another point. It is Lovbjerg and others who combine the basic particle swarm algorithm with the idea of group feeding. Zhang and xie introduced the differential evolution operator to the elementary particle swarm optimization algorithm. Krink and Lovbjerg combine the basic particle swarm optimization algorithm with the genetic algorithm.

II. Central particle swarm optimization algorithm

A. Linear decreasing weight particle swarm optimization algorithm

A population consists of N particles moving in the d-dimensional search space. The position of the ith particle in the t thiteration is denoted as : $X_i^{(t)} = (x_{i1}, x_{i2}, \Lambda\ x_{iD})$,Speed is expressed as $V_i^{(t)} = (v_{i1}, v_{i2}, \Lambda\ v_{iD})$,The individual extreme point reached by individual particles is denoted as: $P_i^{(t)} = (p_{i1}, p_{i2}, \Lambda\ p_{iD})$ .The global extremum reached by all the particles in the group is denoted as: $G_i^{(t)} = (g_{i1}, g_{i2}, \Lambda\ g_{iD})$ .The position of the ith example in the next iteration will be calculated by the following equation:

$$V_{id}^{(t+1)} = w \cdot V_{id}^{t} | + c_1 \cdot rand() \cdot (p_{id} - X_{id}^{t}) + c2 \cdot rand() \cdot (p_{gd} - X_{id}^{t}) \qquad (1)$$
$$X_{id}^{t+1} = X_{id}^{t} + X_{id}^{t+1}$$

The c_1 and c_2 are normal numbers, respectively representing individual cognitive ability and social cognitive ability. Rand () is a random function in [0, 1] and w is an inertial factor. It decreases linearly from 0.9 to 0.4 during the search. In addition, particle velocity is limited to the range of [Vmin, Vmax]D. If an element of velocity exceeds the threshold of Vmin or Vmax, the corresponding particle exceeds the threshold.

B. Central particle swarm optimization algorithm

The development of cationic algorithms is driven by the observation of population behavior.

From equation (1), the velocity of particles is determined by the previous velocity and the individual cognitive ability and social cognitive ability. Because social cognition (part three of the equation) means that all particles are attracted to the global optimum and move towards it. The other two parts, the former speed and the individual's cognitive ability are equivalent to self-control, allowing particles to retain their own information. Therefore, during the search, all particles move towards the global optimal position, their positions are often different but very close to the global optimal position.

In this paper, it is clearly proposed that the central particle is located at the center of the population in each iteration. Another N-1 example in the particle swarm uses the basic particle swarm algorithm in each iteration to actually update their positions, changing with the number of iterations. The central particle updates its position using the following equation.

$$X_{cd}^{(t+1)} = \frac{1}{N-1} \sum_{i=1}^{N-1} X_{id}^{(t+1)} \qquad (2)$$

It is different with other particles, and center of the particle velocity, it besides a particle calculation, like all other example participate in all process, such as adaptive value calculation, the competition best particle, etc. This algorithm is called the central particle swarm optimization algorithm.The pseudo-code of the central PSO algorithm is defined as follows:

```
    Start the particle swarm optimization
    Initialization ();
    T goes from 1 to the maximum number of iterations
$(\mathrm{Adaptive\ value})_i^{(t)} = \mathrm{Adaptive\ value\ assessment}(X_c^t)$ (3)
    If needed, update the individual extreme value () and
global extreme value ();
    I goes from 1 to N – 1
$(Adaptive\ value)_i^{(t)} = Adaptive\ value\ assessment(X_i^t)$
Update speed ($V_I^{(t+1)}$)
    Limited speed ($V_I^{(t+1)}$);
    According to the equation (update position);
    If needed, update the individual extreme value ($P_i$) and
global extreme value (Pg);
    The end;
    To renew the position of the centralion;
    If the global extremum () satisfies the requirements of
the problem, is terminated.
    The end;
    The end of the particle swarm optimization.
```

As mentioned above, central particles have potential efficient solutions. If the result of a central particle is to get it to find a good position, then it won't have much impact on the algorithm. One particle is too small to compare with other ordinary particles in the population. More importantly, the central particle is more likely to be the social extreme of the population. So the central particle can guide the future region of the entire population and accelerate convergence.

III. Experiments

A. Function optimization

We're going to use three benchmark functions in this experiment.

The first function is the Rosenbrock function:

$$f_1(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2) \quad (-100 \le x_i \le 100) \quad (4)$$

The second function is the generalized Rosenbrock function:

$$f_2(x) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10) \quad (-10 \le x_i \le 10)$$

The third function is the generalized Griewank function:

$$f_3(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}} + 1) \quad (-600 \le x_i \le 600) \quad (5)$$

In our experiment, the linear decreasing weight particle swarm optimization algorithm and the central particle removal algorithm are compared. The same parameters are set for the two algorithms: inertia weight decreases linearly from 0.9 to 0.4; The learning factor c_1=c_2=2; Vmin is equal to Xmin; Vmax is equal to Vmax; Each equation is measured on a scale of 10, 20, 30; The corresponding maximum number of iterations is set to 1000, 1500 and 2000 respectively. In order to detect the measurability of the algorithm, four populations (size N= 20, 40, 80, 160) are needed to detect the different dimensions of each function. The central particle swarm algorithm consists of a central particle and n-1 ordinary particle. In each experiment, each algorithm was executed 100 times.

B. Neural network training

Neural network training is a complex optimization problem. Usually, the goal is the mean square error of all training modes. Variables include weight and weight. Suppose a standard network structure uses D to represent the entry unit, M to represent the hidden unit, C to represent the output unit, and the number of variable factors: $W = (D + 1) \times M + (M + 1) \times C$. In conclusion, neural network training is a multispace optimization problem with multiple local minimum values.

Neural network training uses n random partition technology to divide data into n mutually exclusive data of equal size. Select one set of data as the data group and the other as the test group.

We set n to 10 in our experiment. Three - layer feed forward neural networks of s-type transfer function are used for character classification. The neural network is set to 14 input units, 5 hidden units, and 2 output units, so the dimension of each particle is 87. For diabetes, the neural network is set to eight input units, five hidden units, and two output units, so the dimension of each particle is 57. In the function optimization of the above experiment, all the parameters of the central particle swarm optimization and the LDWPSO are the same, except Vmin= -2 and Vmax =2.The maximum number of iterations is set to 1000 and the population size is set to 20.

IV. Conclusions

In this paper, we proposed the central particle swarm optimization algorithm, and the concept of central particle was introduced from the linear decreasing weight particle swarm optimization algorithm. The population consists of ordinary particles, and the position of the central particle changes with the change of the population center during each iteration. In typical particle activity, all particles oscillate around the center of the population and gradually approach the center of the population. Particles in the center of the operation process, usually can get a better position and become the global optimal particle, so it's only a center of the particle. It have more chance to guide population search and greatly influence the performance of the algorithm. The experimental results show that the performance of the central particle swarm optimization algorithm is better than that of the LDWPSO algorithm.

This work was supported by the National Language Committee scientific research projects of China under Grant No. YB135-50, the National Natural Science Foundation of China under Grant No. 61806007 and Science and Technology Development Plan Project of Henan under Grant No. 182102310920.

REFERENCES

[1] P.J. Angeline, Evolutionary optimization versus particle swarm optimization: philosophy And Performance differences, Lecture Notesin Computer Science,vol.1447, Springer, Berlin,1998, pp,601-610.

[2] P.J. Angeline,Using selection to improve particle swarm optimization, Proceedings of the IEEE,1998,pp. 84–89.

[3] R. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, Proceedings of the IEEE Conference on Evolutionary Computation,2000, pp.84-88.

[4] M. Clerc, J. Kennedy, The particle swarm-explosion, stability and convergenceina multidi-Mensional complex space, IEEE Trans.Evol.Comput.6 (2002) 58– 73.

[5] S. Baskar, P. Suganthan, A novel concurrent particle swarmoptimization, Proceedings of the Congress on Evolutionary Computation, 2004, 792–796.

[6] C. Blake,C.J. Merz, UCI repository of machine learning databases,1998

[7] S.C. Esquivel, C. A. Coello Coello, On the use of particle swarm optimization with multimodal functions, proceedings of the Congressn on Evolutionary Computation, 2003, pp.1130–1136.