IJESRT
INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

NATURAL CLUSTERING ANALYSIS

Anupama Vishwas Gajbhiye*, Trimbak Ramchandra Sontakke
* Department of Computer Science JJT University,Jhunjhunu,Rajasthan.

ABSTRACT

Particle Swarm Optimization (PSO) is a biologically inspired computational search and optimization method based on the social behavior of birds flocking or fish schooling. A number of basic variations have been developed due to improved speed of convergence and quality of solution found by the PSO. In this paper the overview of PSO Technique is provided and the PSO System is compared with Evolutionary techniques with respect to genetic algorithm and artificial neural network (GA and ANN systems).

KEYWORDS: Swarm Optimization (PSO), bird flocking, Evolutionary techniques, GA, ANN.

INTRODUCTION

Particle swarm optimization (PSO) is a population based stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995, inspired by social behavior of bird flocking of fish schooling. [1][2] PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles.

Compared to GA, the advantages of PSO are that PSO is easy to implement and there are few parameters to adjust. The ANN Back propagation algorithm works as follows.

[1] Initially build a Network with the chosen number of input, hidden, and output units.
[2] Initialize all the weights to low random values.
[3] Choose a single training pair at random.
[4] Copy the input Patten to the input layer.
[5] Cycle the network so that the activities from the inputs generate the activations in the hidden and output layers.
[6] Calculate the error derivative between the output activation and the target output.
[7] Use Back-propagation to find the summed products of the weights and errors in the output layer in order to calculate the error in the hidden units.
[8] Update the weights attached to each unit according to the error in that unit, the output from the unit below it, and the learning parameters, until the error is sufficiently low or the network settles.

In the present paper PSO algorithm is used to replace ANN for performing clustering.

PSO has been successfully applied in many areas: function optimization, artificial neural network training, fuzzy system control, and other areas where can be applied.

In this type of biological system- social system, more specifically, the collective social behaviour of simple individuals interacting with their environment and each other is considered. This technique is called swarm intelligence. All of the simulations utilize local processes, such as those modeled by cellular automata, and might underlie the unpredictable group dynamics of social behavior.

There are two popular swarm inspired methods in computational intelligence areas: Ant colony optimization (ACO) and particle swarm optimization (PSO). ACO was inspired by the behaviors of ants and has many successful applications in discrete optimization problems.

The particle swarm concept originated as a simulation of simplified social system. The original intent was to graphically simulate the choreography of a bird block or fish school. However, it was found that particle swarm model can be used as an optimizer.

THE CLUSTER ALGORITHM

PSO simulates the behaviors of bird flocking. In this simulation a group of birds are randomly searching food in an area. There is only one piece of food in the area being searched. All the birds do not know where the food is and they know how far the food is placed from the source in each subsequent iterative step. So what's the best strategy to find the food? The effective one is to follow the bird which is nearest to the food.

PSO learned from the scenario and used it to solve the optimization problems. In PSO, each single solution is a "bird" in the search space. This is called "particle". All the particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. The particles fly through the problem space by following the current optimum particles.

PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. With each iterative step, each particle is updated by following two “best" values. The first one is the best solution (fitness) it has achieved so far. (The fitness value is also stored.) This value is called pbest. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best and called gbest. When a particle takes part of the population as its topological neighbors, the best value is a local best and is called lbest. [2][3]

After finding the two best values, the particle updates its velocity and positions with following Equations (a) and (b)**.**

v[] = v[] + c l * rand() * (pbest[] - present[]) + c2 * rand() * (gbest[] - present[]) (a)
present[] = present[] + v[] (b)

v[] is the particle velocity, present[] is the current particle (solution). pbest[] and gbest[] are defined as stated before, rand 0 is a random number between (0,1). c1, c2 are learning factors, usually c 1 = c2 = 2.

The pseudo code of the procedure is as follows:

```
For each particle
Initialize particle
End

Do
For each particle
Calculate fitness value
If the fitness value is better than the best fitness value (pBest) in history (Memory)
Set current value as the new pBest
End

Choose the particle with the best fitness value of all the particles as the gBest
For each particle
        Calculate particle velocity according to equation (a)
        Update particle position according to equation (b)
End
While maximum iterations or minimum error criteria is not attained
```

Particles' velocities on each dimension are clamped to a maximum velocity Vmax. If the sum of accelerations causes the velocity on that dimension to exceed Vmax(whose parameter is specified by the user) then the velocity on that dimension is limited to Vmax.

COMPARISONS BETWEEN GENETIC ALGORITHM AND PSO

Most of evolutionary techniques have the following procedure:

1. Random generation of an initial population
2. Reckoning of a fitness value for each subject. It will directly depend on the distance to the optimum.
3. Reproduction of the population based on fitness values.
4. If requirements are met, then stop. Otherwise go back to 2.

From the above procedure, we find that PSO shares many common points with GA. Both algorithms start with a group of a randomly generated population, both have fitness values to evaluate the population. Both update the population and search for the optimum with random techniques.

However, PSO does not have genetic operators like crossover and mutation. Particles update themselves with the internal velocity. They also have memory, which plays a significant role in the algorithm. [3] [4] [5]

Compared with genetic algorithms (GAs), the information sharing mechanism in PSO is significantly different. In GAs, chromosomes share information with each other. So the whole population moves like one group towards an optimal area. In PSO, only gBest (or lBest) gives out the information to others. It is a one-way information sharing mechanism. The evolution only looks for the best cluster solution. Compared with GA, all the particles tend to converge to the best solution quickly even in the local version in most cases.

ARTIFICIAL NEURAL NETWORK AND PSO

An artificial neural network (ANN) is an analysis paradigm that is a simple model of the brain and the back-propagation algorithm is the one of the most popular method to train the artificial neural network.

Evolutionary computation (EC) methodologies have been applied to three main attributes of neural networks: network connection weights, network architecture (network topology, transfer function), and network learning algorithms.

Most of the work involving the evolution of ANN has focused on the network weights and topological structure. Usually the weights and/or topological structure are encoded as a chromosome in GA. The selection of fitness function depends on the research goals. For a classification problem, the rate of non-classified patterns can be viewed as the fitness value.

The advantage of the EC is that EC can be used in cases with non-differentiable transfer functions and no gradient information is available.
The disadvantages are 1. The performance is not competitive in some problems. 2. Representation of the weights is difficult and the genetic operators have to be carefully selected or developed.

There are several papers reported using PSO to replace the back-propagation learning algorithm in ANN in the past several years. It showed PSO is a promising method to train ANN. It is faster and gets better results in most cases.
A simple example of evolving ANN with PSO is presented here. The problem is a benchmark function of Classification problem: iris data set. Measurements of four attributes of iris flowers are provided in each data set record: sepal length, sepal width, petal length, and petal width. Fifty sets of measurements are present for each of three varieties of iris flowers, for a total of 150 records or patterns.

A 3-layer ANN is used to do the classification. There are 4 inputs and 3 outputs. So the input layer has 4 neurons and the output layer has 3 neurons. One can evolve the number of hidden neurons. Here it is assumed that the hidden layer has 6 neurons. We can evolve other parameters in the feed-forward network. Here only the network weights are considered. So the particle will be a group of weights, there are 4*6+6*3 = 42 weights, so the particle consists of 42 real numbers. The range of weights can be set to [-100, 100] (In real cases, one might try different ranges). After encoding the particles, the fitness function is determined. For the classification problem, feed all the patterns to the network whose weights is determined by the particle, get the outputs and compare it the standard outputs. Then record the number of misclassified patterns as the fitness value of that particle. PSO can be applied to train the ANN to get lower number of misclassified patterns as possible. There are not many parameters in PSO which need to be adjusted. We only need to adjust the number of hidden layers and the range of the weights to get better results in different trials.

ISSN: 2277-9655
(I2OR), Publication Impact Factor: 3.785

PSO PARAMETER CONTROL

There are two key steps when applying PSO to optimization problems: the representation of the solution and the fitness function. One of the advantages of PSO is that PSO takes real numbers as particles. It is not like GA, which needs to change to binary encoding, or special genetic operators have to be used. For example, we try to find the solution for f(x) = xl^2 + x2^2+x3^2, the particle can be set as (xl, x2, x3), and fitness function is f(x). Then we can use the standard procedure to find the optimum.

The searching is a repeat process, and the stop criteria are that the maximum iteration number is reached or the minimum error condition is satisfied. [10]

There are not many parameter need to be tuned in PSO. Here is a list of the parameters and their typical values.

The number of particles: the typical range is 20 - 40. Actually for most of the problems 10 particles is large enough to get good results. For some difficult or special problems, one can try 100 or 200 particles as well.

Dimension of particles: It is determined by the problem to be optimized.

Range of particles: It is also determined by the problem to be optimized, the application can specify different ranges for different dimension of particles.

Vmax: it determines the maximum change one particle can take duration of one iterative step. Usually we set the range of the particle as the Vmax for example, the particle (xl, x2, x3)
Xl belongs [-10, 10], then Vmax = 20

Learning factors: cl and c2 usually equal to 2. However, other settings may also be used. But usually c 1 equals to c2 and ranges from [0, 4]

CONCLUSION

The two versions of PSO were tested. The global and local versions were also tested. The global version is faster but might converge to local optimum for some problems and the local version is a bit slower but not easy to be trapped into local optimum. One can use global version to get quick result and use local version to refine the search.

REFERENCES

[1] http://www.psotoolbox.sourceforge.net
[2] http://icdweb.cc.purdue.edu/- hux/PSO.shtml
[3] http://www.researchindex.com
[4] http://www.engr.iupui.edu/~eberhart/
[5] http://users.erols.com/cathyk/jimk.html
[6] chern.ie.nthu.edu.tw/swarm.htm.
[7] http://www.aridolan.com
[8] www.red3d.com/cwr/boids/
[9] www.inf.ed.ac.uk/teaching/courses /net/slides/lect13h.pdf
[10] http://www.swarmintelligence.org