Optimizing Back-Propagation using PSO_Hill and PSO_A*

Priyanka Sharma[*], Asha Mishra[**]

[*] M.Tech Scholar of computer science & Engineering , BSAITM, Faridabad
[**] Department of computer science & Engineering , BSAITM, Faridabad

Abstract- Back propagation algorithm (BPA) have the complexity, local minima problem so we are using Particle Swarm optimization (PSO) algorithms to reduce and optimize BPA. In this paper, two variants of Particle Swarm Optimization (PSO) PSO_Hill and PSO_A* is used as optimization algorithm. PSO_Hill and PSO_A* algorithms are analyzed and evaluated on the basis of their advantages, applied to feed forward neural network(FNN) for back propagation algorithm(BPA) which is a gredient desent technique. where BPA is used for non_linear problems. These non_linear problems are improved by a PSO_Hill and PSO_A* algorithms.

Index Terms- BPA, PSO_hill, PSO_A*, ANN

I. Introduction

Optimization is an active area of research as many algorithms that are introduced earlier are complex and not optimize so better optimization algorithms are needed. The objective of optimization algorithm is to find a solution to satisfying a set of constraints such that objective function is maximized or minimized.

Particle swarm optimization (PSO) is an alternative population-based evolutionary computation technique. It has been shown to be capable of optimizing hard mathematical problems in continuous or binary space. The particle swarm optimization (PSO) algorithm is based on the evolutionary computation technique [11-13]. PSO optimizes an objective function by conducting population-based search. The population consists of potential solutions, called particles, similar to birds in a flock. The particles are randomly initialized and then freely fly across the multi-dimensional search space. While flying, every particle updates its velocity and position based on its own best experience and that of the entire population. The updating policy will cause the particle swarm to move toward a region with a higher object value. Eventually, all the particles will gather around the point with the highest object value.

PSO processes the search scheme using populations of particles where each particle is equivalent to a candidate solution of a problem [9]. The particle moves according to an adjusted velocity, which is based on that particle's experience and the experience of its companions. For the D-dimensional function f(.), the ith particle for the jth iteration can be represented as

$X_j = X_1, X_2 \ldots\ldots\ldots\ldots\ldots X_n$

$W_j = W_1, W_2 \ldots\ldots\ldots W_n$

Assume that the best local position of the jth particle at then nth iteration is represented as

$L\_best_j = L\_best_1, L\_best_2 \ldots\ldots\ldots\ldots L\_best_n$

The best position amongst all the particles, G_best, from the first iteration to the jth iteration, where best is defined by some function of the swarm, is

$G\_best_j = G\_best_1, G\_best2 \ldots\ldots\ldots G\_best_n$

The original particle swarm optimization algorithm can be expressed as follows:

$P_j = X_j * W_j$

$V_j^t = f(p_j)$ means $V_j^t = 1/(1+\exp(-P_j))$

$V_j^{t+1} = V_j^t + r1 * c1(L\_best_j - X_j) + r2 * c2(G\_best_j - X_j)$

$X_j^{t+1} = X_j^t + V_i^{t+1}$

r1and r_2 are random variables acts as learning signals such that $0 \le r_1, r_2 \le 1$.

where W_j is the inertia weight at the jth iteration. The weighting factors, C_1 and C_2 are used as constants. In this paper, the optimization and analysis of back propagation algorithm is done by particle swarm optimization, and two variant of particle swarm optimization PSO_Hill and PSO_A*and their algorithm, architecture are proposed.

This paper is organized as follows. In Section II, we explained original swarm techniques that are PSO and ACO. In Section III, we explained original Back-propagation network with algorithms. In section IV, we explained Related work in PSO. In Section V, we explained proposed work (PSO_Hill and PSO_A* variants with diagram and algorithm). In section VI, we explained Conclusion of paper.

II. Swarm Techniques

Swarm intelligence (SI) is the collective behavior of decentralized, self-organized systems, natural or artificial. The concept is employed in work on artificial intelligence. SI systems are typically made up of a population of simple agents interacting locally with one another and with their environment. The inspiration often comes from nature, especially biological systems. The agents follow very simple rules, and although there is no centralized control structure dictating how individual agents should behave, local, and to a certain degree random, interactions between such agents lead to the emergence of "intelligent" global behavior, unknown to the individual agents. Natural examples of SI include ant colonies, bird flocking, animal herding, bacterial growth, and fish schooling. Two well known approaches of swarm intelligence are:

Ant Colony Optimization (ACO): Ant colony optimization (ACO) is a class of optimization algorithms modeled on the actions of an ant colony [2], [13]. ACO methods are useful in problems that need to find paths to goals. Artificial 'ants'—simulation agents—locate optimal solutions by moving through a parameter space representing all possible solutions. Natural ants lay down pheromones directing each other to resources while exploring their environment. The simulated 'ants' similarly record their positions and the quality of their solutions, so that in later simulation iterations more ants locate better solutions.

Particle Swarm Optimization (PSO): Particle Swarm Optimization (PSO) is a swarm-based intelligence algorithm [6] influenced by the social behavior of animals such as a flock of birds finding a food source or a school of fish protecting themselves from a predator. A particle in PSO is analogous to a bird or fish flying through a search (problem) space. The movement of each particle is co-coordinated by a velocity which has both magnitude and direction. Each particle position at any instance of time is influenced by its best position and the position of the best particle in a problem space. The performance of a particle is measured by a fitness value, which is problem specific. Each particle will have a fitness value, which will be evaluated by a fitness function to be optimized in each generation. Each particle knows its best position L_best and the best position so far among the entire group of particles G_best. The L_best of a particle is the best result (fitness value) so far reached

The particle swarm optimization (PSO) algorithm is based on the evolutionary computation technique. PSO optimizes an objective function by conducting population-based search. The population consists of potential solutions, called particles, similar to birds in a flock. The particles are randomly initialized and then freely fly across the multi-dimensional search space. While flying, every particle updates its velocity and position based on its own best experience and that of the entire population. The updating policy will cause the particle swarm to move toward a region with a higher object value. Eventually, all the particles will gather around the point with the highest object value. PSO attempts to simulate social behavior, which differs from the natural selection schemes of genetic algorithms.

PSO processes the search scheme using populations of particles, which corresponds to the use of individuals in genetic algorithms. Each particle is equivalent to a candidate solution of a problem. The particle moves according to an adjusted velocity, which is based on that particle's experience and the experience of its companions.

A) Conventional Particle Swarm Optimization

The particle swarm optimization algorithm was introduced by Kennedy and Eberhart in 1995 [6], [13]. The algorithm consists of a swarm of particles flying through the search space. Each individual i in the swarm contains parameters for position $\mathbf{x}_i$ and velocity $\mathbf{v}_i$, where $\mathbf{x}_i \in \mathbf{R}^n$, $\mathbf{v}_i \in \mathbf{R}^n$ while n is the dimension of the search space. The position of each particle represents a potential solution to the optimization problem. The dynamics of the swarm are governed by a set of rules that modify the velocity of each particle according to the experience of the particle and by adding a velocity vector to the current position, the position of each particle is modified. As the particles move around the space, different fitness values are given to the particles at different locations according to how the current positions of particles satisfy the objective. At each iteration, each particle keeps track of its local best position, L_best and depending on the social network structure of the swarm, the global best position, G_best.

B) Application of PSO

1. Neural Network Training
2. Telecommunications
3. Data Mining
4. Design and Combinatorial optimization
5. Power systems
6. Signal processing

III. Back-Propagation Algorithm

The back- propagation algorithm is used in layered feed-forward ANNs [14]. This means that the artificial neurons are organized in layers, and send their signals "forward", and then the errors are propagated backwards. The network receives inputs by neurons in the input layer, and the output of the network is given by the neurons on an output layer. There may be one or more intermediate hidden layers. The back-propagation algorithm uses supervised learning, which means that we provide the algorithm with examples of the inputs and outputs we want the network to compute, and then the error (difference between actual and expected results) is calculated. The idea of the back propagation algorithm is to reduce this error, until the ANN learns the training data. The training begins with random weights, and the goal is to adjust them so that the error will be minimal.

The basic back propagation algorithm consists of three steps.

1) The input pattern is presented to the input layer of the network. These inputs are propagated through the network until they reach the output units. This forward pass produces the actual or predicted output pattern.
2) The actual network outputs are subtracted from the desired outputs and an error signal is produced.
3) This error signal is then the basis for the back propagation step, whereby the errors are passed back through the neural network by computing the contribution of each hidden processing unit and deriving the corresponding adjustment needed to produce the correct output. The connection weights are then adjusted and the neural network has just learned from an experience.
4) Learning parameters are used to control the training process of a back propagation network.
5) The learn rate is used to specify whether the neural network is going to make major adjustments after each

learning trial or if it is only going to make minor adjustments.

6) Momentum is used to control possible oscillations in the weights, which could be caused by alternately signed error signals.

IV. RELATED WORK

Pillai , K. G. [1] explains a novel overlapping swarm intelligence algorithm is introduced to train the weights of an artificial neural network. Training a neural network is a difficult task that requires an effective search methodology to compute the weights along the edges of a network. The back propagation algorithm, a gradient based method, is frequently used to train multilayer feed-forward networks. On the other hand, training algorithms based on evolutionary computation have been used to train multilayer feed-forward networks in an attempt to overcome the limitations of gradient based algorithms with mixed results. This paper introduces an overlapping swarm intelligence technique to train multilayer feedforward networks. The results show that OSI method performs either on par with or better than the other methods tested.

M. Conforth and Y. Meng [2] propose a swarm intelligence based reinforcement learning (SWIRL) method to train artificial neural networks (ANN). Basically, two swarm intelligence based algorithms are combined together to train the ANN models. Ant Colony Optimization (ACO) is applied to select ANN topology, while Particle Swarm Optimization (PSO) is applied to adjust ANN connection weights. To evaluate the performance of the SWIRL model, it is applied to double pole problem and robot localization through reinforcement learning. Extensive simulation results successfully demonstrate that SWIRL offers performance that is competitive with modern neuro-evolutionary techniques, as well as its viability for real-world problems.

Marco Dorigo and Mauro Birattari [3] defines Swarm intelligence as the discipline that deals with natural and artificial systems composed of many individuals that coordinate using decentralized control and self-organization. In particular, the discipline focuses on the collective behaviors that result from the local interactions of the individuals with each other and with their environment. Examples of systems studied by swarm intelligence are colonies of ants and termites, schools of fish, flocks of birds, herds of land animals. Some human artifacts also fall into the domain of swarm intelligence, notably some multi-robot systems, and also certain computer programs that are written to tackle optimization and data analysis problems.

Y. Karpat and Tugrul Ozel [4] propose a concept of particle swarm optimization, which is a recently developed evolutionary algorithm, is used to optimize machining parameters in hard turning processes where multiple conflicting objectives are present .The relationships between machining parameters and the performance measures of interest are obtained by using experimental data and swarm intelligent neural network systems (SINNS). The results showed that particle swarm optimization is an effective method for solving multi-objective optimization problems, and an integrated system of neural networks and swarm intelligence can be used in solving complex machining optimization problems.

James kennedy and Russell Eberhart [5] propose a concept for the optimization of nonlinear functions using particle swarm methodology is introduced. The evolution of several paradigms is outlined, and an implementation of one of the paradigms is discussed. Benchmark testing of the paradigm is described, and applications, including nonlinear function optimization and neural network training, are proposed. The relationships between particle swarm optimization and both artificial life and genetic algorithms are described.

V. PROPOSED WORK

5.1) *Analysis of Different Algorithm*: This is done on the basis of analysis of their advantage to introduce proposed variants for optimization. In this we analysis the limitation of BPA and to optimize BPA we use PSO approach.

5.1.1) Disadvantages in BPA:

a) Local Minima
b) Low Speed
c) Higher cost of computation
d) Error Problem
e) Gloal Maxima but less as compared to Local
f) Less accuracy

5.1.2) PSO Advantages:

a) PSO can be applied to scientific research and Engineering.
b) High computational speed
c) Learning achieved from particle own experienced
d) Learning achieved from experience of cooperation between particles

5.2) *Proposed Variants for optimization*:

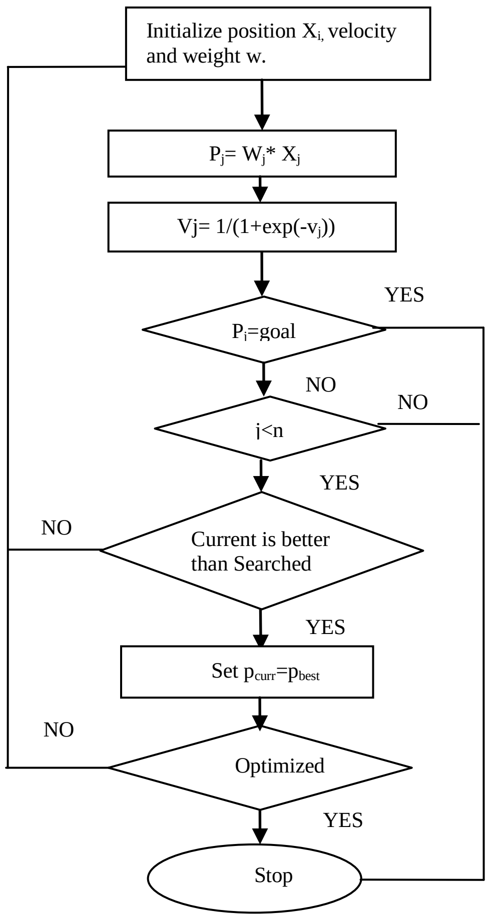
1) PSO_hill
2) PSO_A*

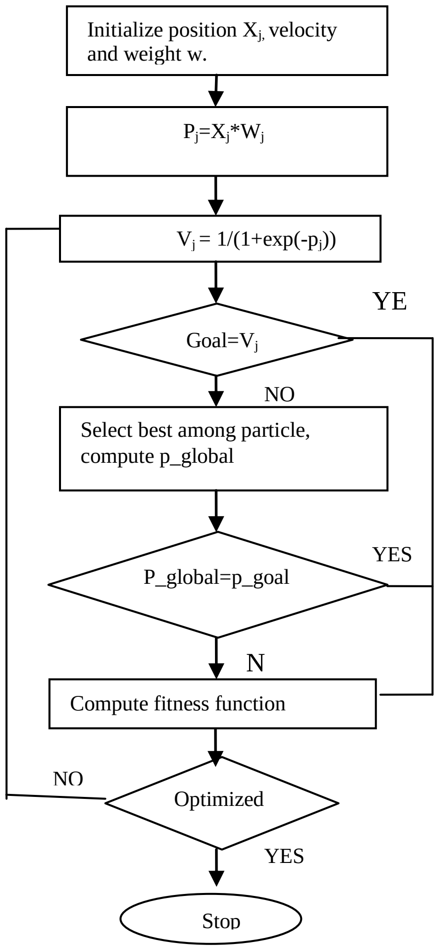
5.2.1) PSO_hill Advantge:

a) High computational speed
b) Strong ability in global search
c) Higher Accuracy
d) Learning achieved from particle own experienced
e) Learning achieved from experience of cooperation between particles.

5.2.2) PSO_Hill variables Notations:
X: Initial particle position.
W: Weight for each particle.
n: Total no. of iterations.
P_{curr}= current position of particles
$P_{best =}$ particle best position

5.2.3) PSO_Hill Diagram:





5.2.4) PSO_Hill Algorithm:

1) Start with initializing particle position X, their velocity V, Weight W, p_loc→0, p_glob→0,n→0
2) If goal=v_j then terminate it; Otherwise
3) $P_j=X_j*W_j$
4) $V_j=1/(1+exp(-p_j))$
5) If (j<n)
6) {
7) If(Evaluate a new position which is better than current position but not goal)
8) {
9) Curr_position=better_position;
10) Else
11) Keep current position & continue the search to find goal
12) }
13) Else
14) Return to step 2 to continue till it reaches to goal

5.3.1) PSO_A* Advantages:

I. Most heuristic solution
II. Optimized in terms of fitness function

5.3.2) PSO_A* Diagram:

5.2.3) PSO_A* Variables Notations:

Open_list: A list which have nodes that are generated but not expanded.
Closed_list: A list which have nodes that are expanded and its childrens are available to search program.

5.2.4) PSO_A* Algorithm:

1. Start with initial position of particles and place them on open node, p_loc→0,p_glob→0.
2. If (open_list=empty) stop and return as failure
3. Select p_loc particles n from open list that has smallest fitness function
4. {
5. if node n= goal node
6. return success
7. stop
8. }
9. Otherwise
10. Expand the successor particles of node n and as node n is explored so keep it on closed list
11. For each successor h
12. {
13. If h is not in open_list or closed_list
14. {
15. Attach a back pointer to n particle to backtrack it
16. {

17. Compute fitness of particle f*(h)
18. Else
19. Compute lowest or p_glob (g*(h))
20. }
21. Place on open list
22. Return to Step to till the goal

VI. CONCLUSION

In this paper, two variants of the particle swarm optimization scheme is presented. Two PSO Variants PSO_Hill and PSO_A* are proposed with their algorithm, architecture, advantages and disadvantages, which can be used to the optimize the BPA. In next paper, a third strategy is proposed PSO_Hill_A* on the basis of strength of two variants PSO_Hill and PSO_A* algorithm. The particle local best and global best positions help the variants to move towards the solution.

REFERENCES

[1] Pillai , K. G,"Overlapping swarm intelligence with artificial neural network" ,Swarm intelligence(SIS) ,2011 IEEE Symopsium,15 april 2011.

[2] Matthew Conforth and Yan Meng ,"Reinforcement Learning for Neural Networks using Swarm Intelligence" , 2008 IEEE Swarm Intelligence Symposium, St. Louis MO USA, September 21-23, 2008

[3] [3] Macro Dorigo and Mauro Birattari (2007), Scholarpedia, 2(9) "Swarm intelligence", http://www.scholarpedia.org/article/Swarm_intelligence. Marco Dorigo (2007) Ant colony optimization. Scholarpedia, 2(3):1461.

[4] Yiğit Karpat and Tuğrul Özel, "Swarm-Intelligent Neural Network System (SINNS) Based Multi-Objective Optimization Of Hard Turning" ,Transactions of NAMRI/SME Volume 34, 2006.

[5] James Kennedy' and Russell Eberhart2, "Particle Swarm Optimization", , Washington, DC 20212,kennedy_jim@bls.gov,http://www.cs.tufts.edu/comp/150GA/homeworks/hw3/_reading6%201995%20particle%20swarming.pdf, 1995 IEEE.

[6] .Kennedy and R.Eberhart,"Particle swarm optimization",Proceedings of. IEEE International Conference on Neural Networks, pp. 1942-1948, 1995

[7] Boonserm Kaewkamnerdpong and Peter J. Bentley,"perceptive Particle Swarm Optimization:An Investigation"

[8] Jui-Fang Chang, Shu-Chuan Chu, John F. Roddick and Jeng-Shyang Pan,"A Parallel Particle Swarm Optimization Algorithm with Communication Strategies", Journal of Information Science and Engineering 21, 809-818 (2005)

[9] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory" in Proceedings of 6th International Symposium on Micro Machine and Human Science, 1995, pp. 39-43.

[10] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Proceedings of IEEE International Conference on Neural Networks, 1995, pp. 1942-1948.

[11] P. Tarasewich and P. R. McMullen, "Swarm intelligence," Communications of the ACM, Vol. 45, 2002, pp.63-67

[12] Nitu Mathuriya and Dr. Ashish Bansal, "Applicability of back propagation neural network for recruitment data mining", International Journal of Engineering Research & Technology (IJERT) , ISSN: 2278-0181, Vol. 1 Issue 3, May - 2012

[13] V.Selvi and Dr.R.Umarani, "Comparative Analysis of Ant Colony and Particle Swarm Optimization Techniques**",** International Journal of Computer Applications (0975 – 8887), Volume 5– No.4, August 2010

[14] Carlos Gershenson, "Artificial Neural Networks for Beginners"

AUTHORS

First Author – Priyanka Sharma received B.TECH degree in Computer Science and Engineering with Hons. from Maharshi Dayanand University in 2011 and is persuing M.Tech. in Computer Engineering. Presently, She is working as Lecturer in Computer Engineering department in RITM, Palwal. Her areas of interests are Artificial Neural Network, Soft Computing, Pattern Recognition., Email: priya.sharma090@gmail.com
Second Author – Asha Mishra received B.E. degree in Computer Science & Engineering from NIT, Assam and M.Tech in Computer Science from A.I.T.M, Palwal. Presently, she is working as Senior Lecturer in Computer Engineering department in B.S.A. Institute of Technology & Management, Faridabad. Her areas of interests are DBMS, Analysis & Design of Algorithm and Network Security., Email: asha1.mishra@gmail.com