Journal of Engineering and Development, Vol. 16, No.1, March 2012 ISSN 1813- 7822

Comparison Study on Using of BP and Genetic NN for Digital Logic Circuit Application

Dr. Raaed Khalid Ibrahem Al-Azzawi
Raid_khalid2000@yahoo.com

Dr. Anas Ali Hussien
anasali78@yahoo.com

Abstract

Neural networks are facing many problems when they employ a back propagation algorithm. There problems are characterized by long training tome and trapping the network into local minima. For these reasons the trend, in the recent years, started toward the application of the genetic algorithm because of its ability to discover wide and complex search spaces. In the present work a number of comparisons between BP and GA have been carried out. The results regarding training speed and performance, show that GA is more suitable than BP for training neural networks (ANN).with respect to results obtained, a novel approach for designing a multiplayer artificial neural network system has been introduced and implemented. The new system uses GA for updating and modification of the architecture and weight coefficients of the neural network.

دراسة مقارنة حول استخدام الانتشار العكسي والشبكة العصبية الجينية لتطبيق الدوائر الرقمية

الخلاصة

تواجه الشبكات العصبية العديد من المشاكل عند استخدامها خوارزمية الانتشار الراجع لغرض عملية التدريب. تتمثل هذه المشاكل بطول زمن التدريب والوقوع في مصيدة الحلول الموضعية ولهذه الاسباب بدء التوجه في الفترة الاخيرة يتزايد نحو استخدام الخوارزمية الجينية بسبب مقدرتها الجيدة على استكشاف فضاءات البحث المعقدة الكبيرة في البحث الحال تم تنفيذ عدد من المقارنات بين نظامي خوارزمية الانتشار الراجع والخوارزمية الجينية GA واظهرت النتائج ان الخوارزمية الجينية اكثر ملائمة من خوارزمية الانتشار الراجع في عملية تدريب الشبكات العصبية واستنادا الى النتائج المستحصلة تم تصميم وتجربة نظام جديد لبناء الشبكات العصبية الاصطناعية متعددة الطبقات.

1. Introduction to Neural Networks

Neural networks are algorithms for optimization and learning based loosely on concepts inspired by research in to the nature of the brain. They generally consist of five components:

1. A directed graph known as the network topology whose arcs refer to as links.

2. A state variable associated with each node.
3. A real-valid weight associated with each link.
4. A real-valued bias associated with each node.
5. A transfer function for each node which determines the state of a node as a function of bias, weights of its incoming links the states of the nodes connected to it by these links this transfer function usually lakes the form sigmoid or a step function.

Neural network has some properties which helps it to spread. These properties can be summarized as follow:

- The NN has the ability to adapt and learn.
- Information storage in a NN simply requires storing the different values of the weights
- A NN responds well in the presence of noise (small changes in input will not drastically affect the output).
- NN responds well to hardware failure (a change in the value of a certain weight will only affect certain outputs, not all of them).
- Neural network derives its computing power through its massively parallel- distributed structure.
- A neural network can be regarded as a black box that transforms input vector x from n-dimensional space to an output vector y in m-dimensional space.

A feed forward network is on e whoso topology has no closed paths. Its input nodes are the ones with no arcs to them, and its out put nodes have no arcs away from the. all other nodes are hidden nodes .when the states of all the input nodes are set , all the other nodes in the network can also set their states as values propagate through the network .the operation of a feedforward network consist of calculating outputs given a set of inputs in this manner . A layered feedforward network is one such that any path from an input node to an out put node traverses the same number of arcs. Layered feedforward networks have become very popular for a few reasons. For one, they have been found in practice to generalize well, i.e. when trained on a relatively sparse set of data points they will often provide the right output for an input not in the training set. Secondly, a training algorithm called back propagation exists which can often find a good set of weights (and biases) in a reasonable amount of tune [1].

Back propagation is a variation on gradient search .it generally uses a least-squares optimality criterion. the key to backpropegation is a method for calculating the gradient of the error with respect to the weights for a given input by propagating error backs through the network. there are some drawbacks to back propegation. For one, there is the "scaling problem".

Back propegation works well on simple training problems. however, as the problem complexity increases (due to increased dimensionality and/or greater complexity of the data),the performance of backpropegation falls off rapidly the performance degradation appears to stem from the fact that complex spaces have nearly global minima which are sparse among the local minima. Gradient search techniques tend to get trapped at local minima.

With a high enough gain (or momentum), back propagation can escape these local minima However, it leaves them without knowing whether the next one it finds will be better or worse. When the nearly global minima are well hidden among the local minima, back propagation can end up bouncing between local minima without much overall improvement, thus making for very slow training .A second shortcoming of back propagation is the following. To compute a gradient requires differentiability. Therefore, back propagation cannot handle discontinuous optimality criteria or discontinuous nods transfer functions. This precludes its use on some common node types and simple optimality criteria [2].

1.2 Neural Network Learning Algorithms

Neural network are trained by two main types of learning algorithms: supervised learning and unsupervised learning algorithms. in addition . There exists a third type known as reinforcement learning, which can be regarded as a special form of Supervised learning [3].

- Supervised learning

A supervised learning algorithm adjusts the strengths
Or weights of the interneuron connections according to the difference between the desired and actual network output corresponding to the given input .thus , supervised learning require a teacher or a supervisor to provide , or target ,output signal .

- Unsupervised learning

Unsupervised learning algorithm does not require the outputs to be known. During training, only input patterns are presented to the neural network, which automatically adapts the weights of its connections to cluster the input pattern into group with similar features.

- Reinforcement learning

Reinforcement learning is a special case of the supervised learning instead of using teacher to give target outputs reinforcement learning algorithm employs a critic only to evaluate the goodness of the neural network output corresponding to a given input.

1.3 Genetic Algorithm

Genetic algorithms are algorithms for optimization and learning based loosely on several features of biological evolution. They require five components:

- Away of encoding solutions to the problem on chromosome.
- An evaluation function that returns a rating for each chromosome given to it.
- A way of initializing the population of chromosome Operators that may be applied to parents when they reproduce to alter their genetic composition.
- Included might be mutation, crossover (i.e. recombination of genetic material) and domain –specific operators.
- Parameter settings for the algorithm, the operators and so forth [4].

Given these five components a genetic algorithm operates according to the following steps:

- Determine the representation scheme (how solutions are encoded as a chromosome).
- Create the group of chromosome (initial population of solutions) at beginning of the evolution.
- Determine the fitness measure (the metric for measuring the fitness of chromosome).
- Determine the parameters and variable for controlling the algorithm (e.g. population size, crossover rate, etc).
- Apply genetic operators (e.g. three basic operators, selection, crossover and mutation) to produce new sets of individuals.
- Determine a way of designating the results and criterion for terminating a run.

Representation

The parameters to be optimized are usually represented in a string from since genetic operators are suitable for this type of representation. The method of representation has a major impact on the performance of the GA. different representation schemes might causes different performances in terms of accuracy and computation time [5].

There are two common representation methods for numerical optimization problems. The preferred method is the binary string representation method. The reason for this method being popular is that binary alphabet offers the maximum number of schemata per bit compared with other coding techniques.

When a binary representation scheme, an important step is to decide the number of bits to encode the parameters to be optimized. Each parameter should be encoded with the optimal number of bits covering all possible solutions in the solution space. When too few or too many bits are used the performance can be adversely affected.

Genetic operator

There are basically three genetic operators, selection, crossover and mutation. It is not necessary to employ all of these operators in the GA because each operator independently of the others. The choice of the operator depends on the problem and the representation scheme employed. For instance, operator designed for binary string can not be directly used on strings coded with integer or real.

- **Selection**

Reproduction is a process in which individual strings are copied according to their objective function (fitness) values. The means the string with the higher value has a higher probability of contributing one or more offspring in the generation. [4]

The selection procedure has a significant influence on driving the search towards a promising area and finding good solutions in a short time, however the diversity of the population must be maintained to avoid premature convergence and reach the global optimal solution.
There are many methods in selecting the best chromosomes like: roulette wheel selection, rank selection and some others. The most important one is "Roulette Wheel" selection, since its mechanism is reminiscent of the operation of a Roulette Wheel. Fitness values of individuals represent the widths of slots on the wheel. After a random spinning of the wheel to select an individual for the next generation, sots with large width will have a higher chance to be selected [4].

- **Crossover**

Crossover (sexual recombination) allows new individuals to be created and tested. It is used to recombine two strings to get a better string. It is usually performed with a probability called crossover probability (Pc) usually chosen to be near one to preserve some of the good strings found previously. The most important types of crossover are single-point and multi-point crossover. Single-point is the simplest crossover operation. Two individuals are randomly selected as parents from the pool of individuals formed by the selection procedure and cut at a randomly selected point. The tails, which are the parts after the cutting point, are swapped and two new individuals 'children are produced .multi-point crossover which is "multiple crossover position chosen at random and the parts of the two parents after the crossover positions are exchanged to form two offspring" [6, 7].

- **Mutation**

The mutation operator is applied independently, but immediately following the crossover operator. A mutation is a random addition or deletion of a gene in a chromosome, and is governed by a present mutation rate (mutation probability (Pm) usually quite low.

Unlike the crossover, this is monadic operation. That is, a child string is produced from a single parent string. The mutation operator forces the algorithm to search new areas. Mutation helps the GA to avoid premature convergence, getting trapped at local optima and find the global optimal solution [8].

1.4 Genetic Algorithm to Create an Optimized Neural Network

This paper present an algorithm to train MLPS neural network based on genetic algorithms. The problem is to find the optimal network structure. It's known that the genetic algorithm can optimize a non-linear problem and find global optima. Our proved that the proposed algorithm can automatically determine appropriate network structures and network parameters. in order to find the networks optima structure the process modifies the number of neurons in the hidden layer. The performance of the algorithm is achieved by evolving the initial population and by using operator that alter the size of the network.

Our good of this research is to find a minimal number of neurons in the hidden layer and with this we have obtained a good training error for the solution of the problem.

The algorithm of TNNGA optimizing is as follows:

1- Create randomly an initial population for an m chromosome; each chromosome represents a complete ANN of pre-determined structure size.

2- Initialize the number of generation count.

3- For j=1 to population size.

*Determine the fitness of each chromosome

*Put in order the old population according to its fitness.

4- Select parents for reproduction based on their fitness.

5- Apply search operators (crossover and mutation)to parents and generate offspring. which form the next generation

6- If the generation counter is less than the maximum generation

*Increment the generation counter by one

*Replace the old population by new population

*Go to step (3) Else

7- Store the first chromosome which represents the optimum weights and hidden nodes of ANN.

The flowchart for this algorithm is shown in figure (1)

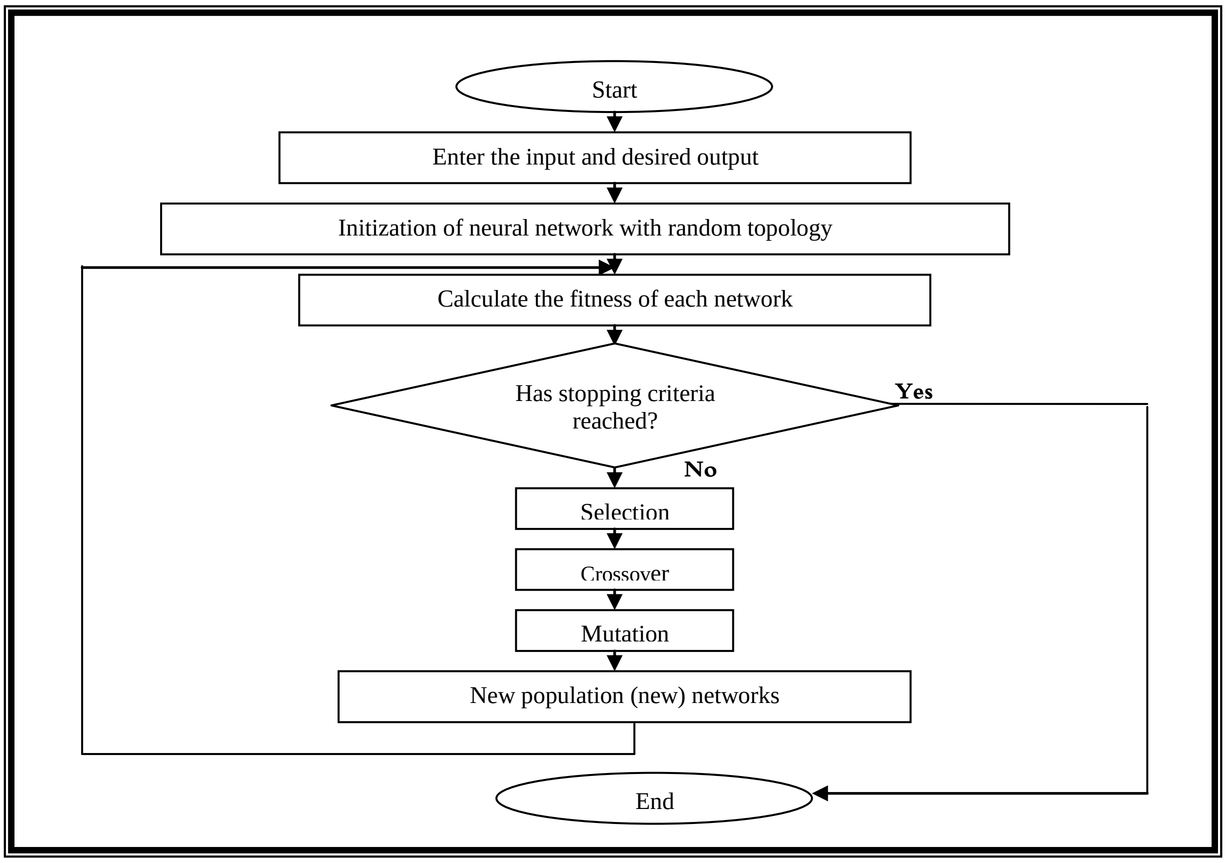
Start
Enter the input and desired output
Initization of neural network with random topology
Calculate the fitness of each network
Has stopping criteria reached?
Yes
No
Selection
Crossover
Mutation
New population (new) networks
End

Figure (1) Flowchart of Genetic Algorithm and Neural Network hybrid system for optimum network topology.

Experimental results

The input vector to the network was chosen to be XOR gate, full adder and Decoder circuit. The goal of our work is to prove that genetic algorithm can find appropriate networks for the given adjective functions; the operator's selection, crossover, and mutation are used in the experiments.

In the experiments the population size was chosen to be equal to 10, the initial population are chosen randomly. The crossover probability was 0.8 and the mutation probability was 0.04. The initial maximum number of neurons in the hidden layer was chosen to be 40 neurons. The maximum number of generation in our genetic algorithm was 400.

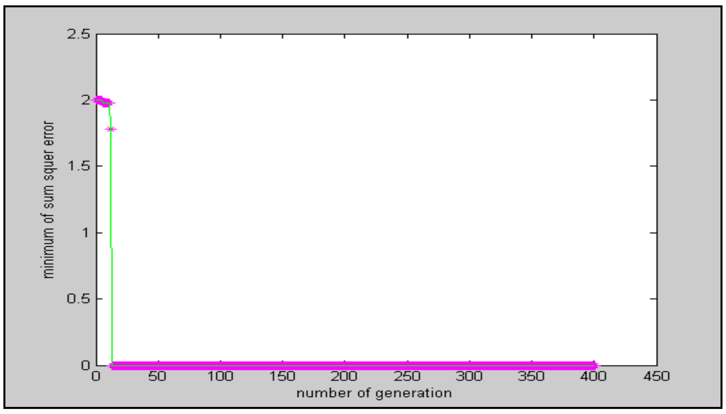

Figure (2) represent the relationship between SSE and the number of generation for NN trained by TNNGA system simulates the work of XOR gate.

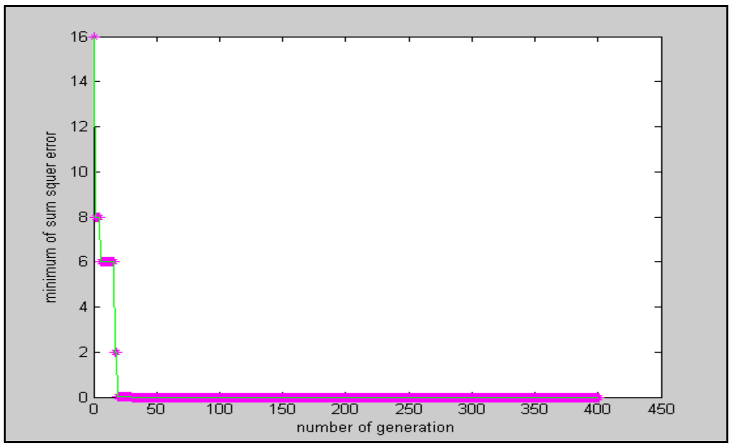

Figure (3) represent the relationship between SSE and the number of generation for NN trained by TNNGA system simulates the work of Full adder circuit.

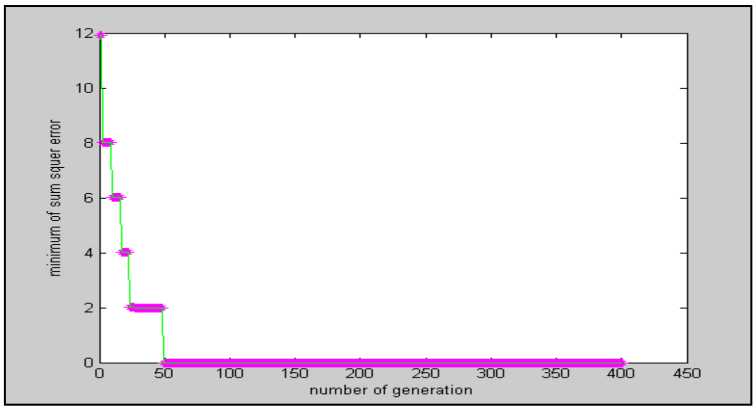

Figure (4) represent the relationship between SSE and the number of generation for NN trained by TNNGA system simulates the work of Decoder circuit.

The results obtained from figures (2), (3) and (4) show the ability to reach the error value (SSE=0), the automatically alter in the number of hidden nodes is shown in figures (5), (6) and (7).

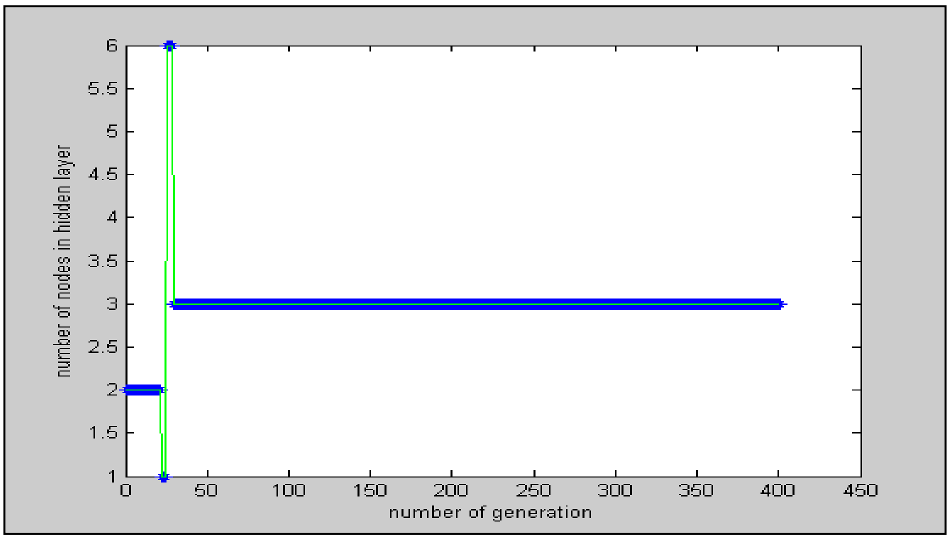

Figure (5) represent the relationship between number of hidden nodes and number of generation for NN trained by TNNGA system simulates the work of XOR gate.

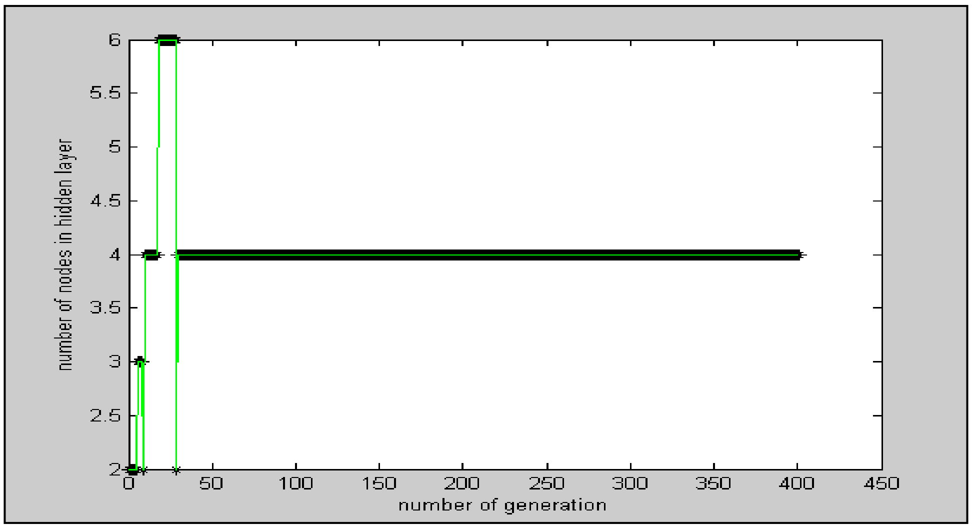

Figure (6) represent the relationship between number of hidden nodes and number of generation for NN trained by TNNGA system simulates the work of Full Adder circuit.

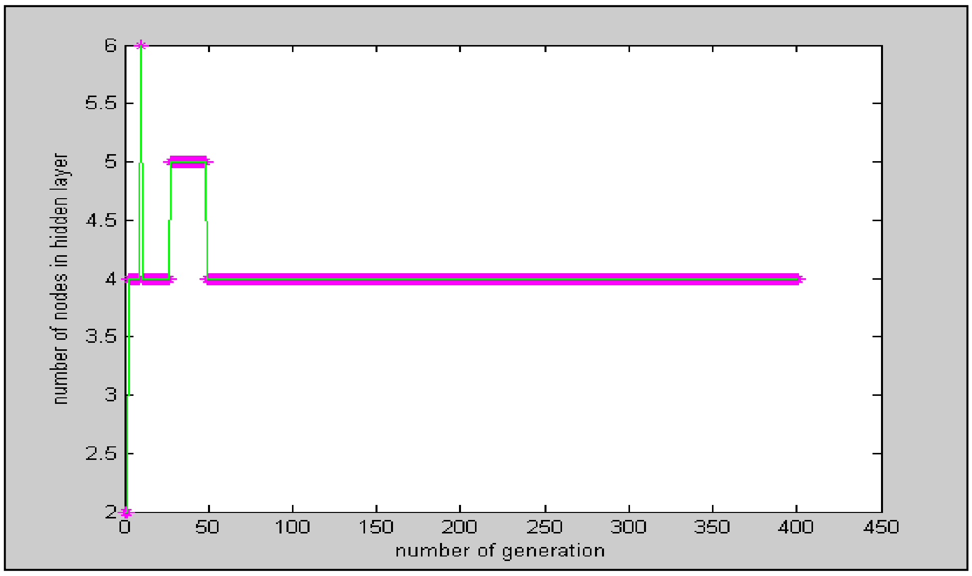

Figure (7) represent the relationship between number of hidden nodes and number of generation for NN trained by TNNGA system simulates the work of Decoder circuit.

Without using the trail and error method which is needed in the training process of neural networks trained by Bp algorithm as shown in figures (8),(9),(10),thus contributing the effort and shorten the time required for the process of solving the problems that we face during the process of designing systems**.**

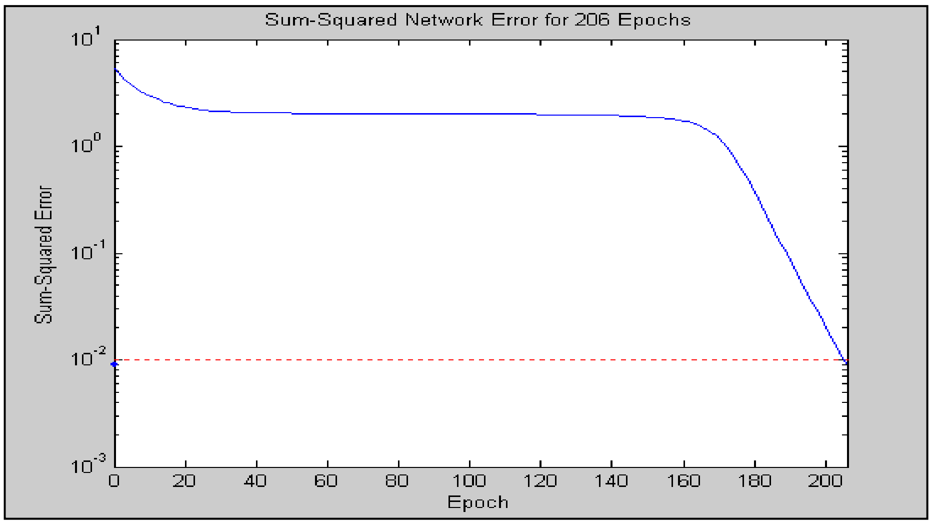

***Figure (8) represent the relationship between SSE and the number of generation for NN trained by Bp algorithm simulates the work of XOR gate*.**

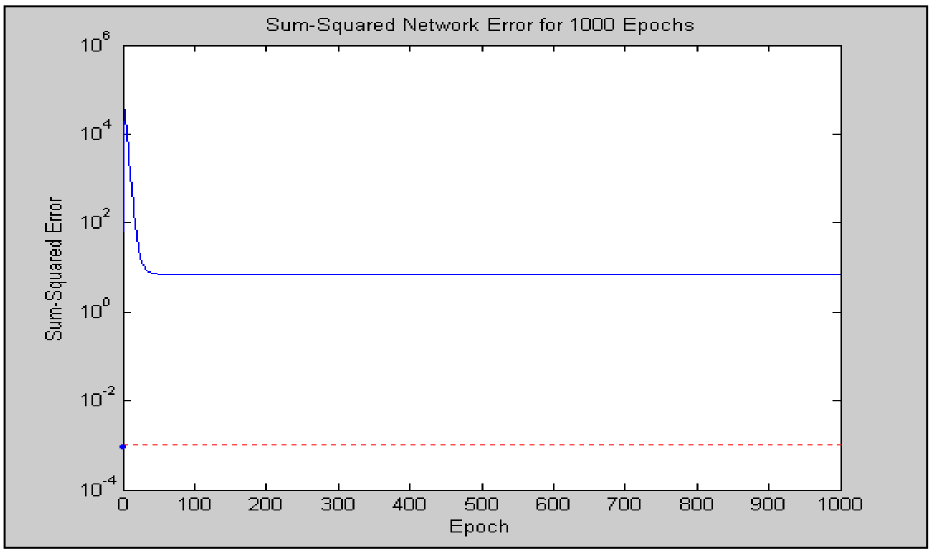

Figure (9) represent the relationship between SSE and the number of generation for NN trained by Bp algorithm simulates the work of Full Adder circuit.

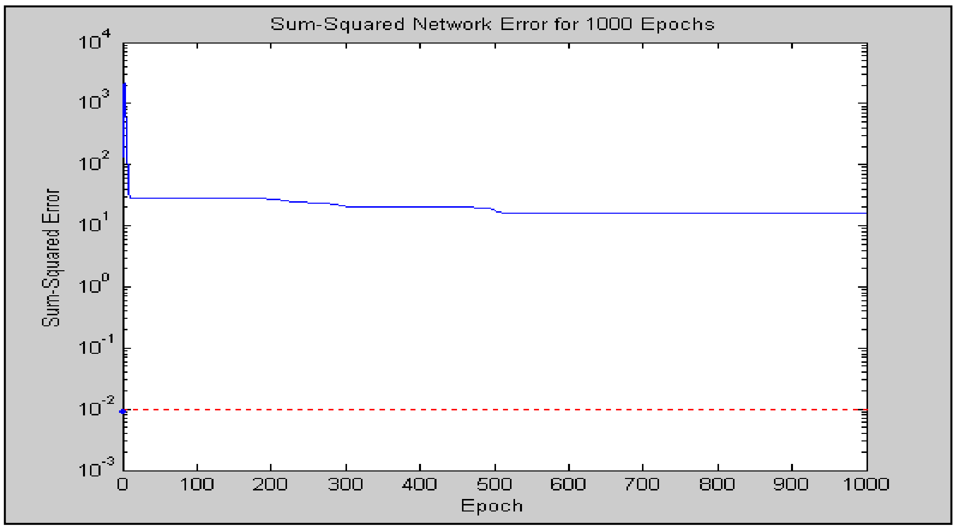

Figure (10) represent the relationship between SSE and the number of generation for NN trained by Bp algorithm simulates the work of Decoder circuit.

Note that through observation the figures above, they indicate the difficultly of obtaining a small values of SSE as occurred through the use of the new designed system.

Conclusion

1. The time required to train ANN using GA is less than time required to train the same ANN using classical algorithm.
2. Gas is free from local minima because in GA there are different types of random operation which can introduce good solution to the desired problem in a random method.
3. The optimal structure of ANN is obtained after executing small number of generations.

References:

1. Zurada J. M. "**Introduction to Artificial Neural System**", Jaico-Impression, 1996.
2. Rumelhart D. E. and McClelland J. L. "**Parallel Distribuited Processing: Exploration in the Microstructure of Cognition**", MIT Press, 1986.
3. Kinnebrock W. "**Neural Network, Fundamentals, Application, Examples**", Galogolia Publication PVT. , 1995.
4. Goldberg D. E. "**Genetic Algorithms in Search, optimization and Machine Learning**", Addison-Wesely, Reading, 1989.
5. Manfred F. and Yee L. "**A genetic- algorithms based evolutionary computational neural network for modelling spattial interaction data**", Springer- verlag , 1998.
6. Siddique M.N.H. and Tokhi M.O. "**Training Neural Networks*: Back* Propagation VS Genetic Algorithms** ", IEEE, 2001.
7. Azar D. "**Using Genetic Algorithms to Optimize Software Quality Estimation Models**", Ph. D. Thesis, McGill University, Montreal, 2004.
8. Obitko M. "**Introduction to Genetic Algorithms**", Czech Technical University, Prague, 1998.