The Combination of Genetic Programming and Genetic Algorithm for Neural Networks Design and Training

Ahmed Badri Muslim
Babylon University
College of Science for Women
Computer Department

Ali Khalid Mohamed Ali
Foundation of Technical Education
Babylon Technical institute
Computer Systems Department

Abstract

We present in this paper using of two evolutionary computations tools for design and training feed-forward neural networks . We use genetic programming algorithm to discover suitable design for neural network that modeled the selected problem . This discover design it have features that make neural network less cost in structure (the smaller network topology) that give desired output for problem . Genetic programming it is a search algorithm that dell with population of tree structures , each of these tree structure it use as suggested design for neural network . The optimization is done throw minimization number of hiding layers and number of neurons in each layer with less connectivity with other neurons . Each new generated neural net send to genetic algorithm for training . Genetic algorithm work as learning algorithm that specify the training set of weights that linked with neural network connection . This work represent global approach that give promising result for some problems .

الخلاصة

يتناول هذا البحث استخدام اثنان من أدوات التحاسبات التطورية لتصميم وتدريب الشبكات العصبية ذات التغذية الأمامية . لقد استخدمنا خوارزمية البرمجة الجينية لاكتشاف التصميم المناسب للشبكة العصبية المستخدمة لنمذجة مشكلة معينة . إن التصميم المكتشف يمتلك خاصية بأن تكون الشبكة اقل كلفة من حيث الهيكلية والتي تعطي الإخراج المطلوب . البرمجة الجينية هي خوارزمية بحث والتي تتعامل مع مجتمع من الهياكل الشجرية , كل واحد من هذه الهياكل الشجرية يعتبر تصميم مقترح للشبكة العصبية . عملية التحسين تتم من خلال تقليص عدد الطبقات المخبئة وعدد العقد المخبئة في كل طبقة مع اقل عدد من الترابطات مع العقد العصبية الأخرى . كل تصميم من التصاميم الجديدة يرسل إلى الخوارزمية الجينية للتدريب . الخوارزمية الجينية في هذا البحث تعمل كخوارزمية تعلم والتي تحدد مجموعة من الأوزان المدربة والتي تربط فيما بعد إلى ترابطات الشبكة العصبية . هذا العمل يعرض طريقة عامة والتي تعطي نتائج واعدة لمجموعة من المسائل المختبرة .

1-Introduction

The human brain is organized as a huge network of numerous very simple computational units , called neurons . During the past half century , the study of artificial networks ,modeled after those . 'Natural prototypes' has become more and more popular [Fel 94] . ANNs are leaning system that have solved a large amount of complex problems related to different area (classification , clustering , regression , etc) [Riv 2006] . Many researchers use this technique in different fields of science . But the use of artificial neural networks has some problem mainly in development process . These problem can be divide to find suitable design and training the network for the problem that work on .In design process the researchers use the expert to find the network architecture and then train it to find the result if the result not satisfy them he go back to change the architecture until he find the best result . This process is slow in performance , the slow is mean that the researcher may be find the suitable architecture in some days , weeks or less from this time but the design is so complex , and the complexity in design have long time in training process . Many researchers (just like Geoffier , Todd and Hgde in 1989)

make automatic search for neural network design. First of this works by using genetic algorithm that represent the networks as two dimensional array of binary digit . After that in 1990 Kitan seen the process of direct encoding of array is be more complex when the design is big , he suggest a grammatical encoding for network by using some of grammatical rules see [Mit. 96] . Genetic programming is a technique to automatically discover computer programs using principles of Darwinian evaluation [Koz. & Ric.] . Genetic programming can be used as an automated invention machine to synthesize design for complex structures [O're 2005] . Genetic programming it use for neural network design architecture , this work firstly done by Koza in 1992 . Another researchers come after that by Marylyn and his Group in 2003 . They use genetic programming as tool for finding neural topology by representing neural structure as a binary tree . This work is design and training . Each tree have in structure the number of hiding layers with some of activation function and the terminal for tree is may be random weights or inputs neurons with one output neurons see [Mar. 2003] for details . River and his group use genetic programming as graph-based codification to represent ANN in genotype with no any cycles . This work use non binary tree to represent the network with generating sub trees inside it with binary operation to representing weights that connected to network connections see [Riv. 2006] . Our work use genetic programming for designing neural network by representing it as a tree structure . Each tree specify number of hiding layers and number of neurons in each layer . Each tree level we consider it as neural network layer . The connection with neurons are represented as dynamic array to store tree connection nodes . Each new neural network that generated will be send to genetic algorithm that train the received network . We use genetic algorithm as a learning algorithm instead of bag propagation algorithm , because genetic algorithm is fast from it see[Mit. 96] . The results that we find compared with other researchers mentioned previously are promising .

2-Genetic Programming As Automatic Modular for Neural Network Design

Genetic programming is a domain-independent method that genetically bread a population of computer programs to solve the problem . Specifically , genetic programming iteratively transform a population of computer programs into a new generation of programs by applying analogs of naturally occurring genetic operators [Koz. 94][Koz. 98][Koz. 2003] . Genetic programming can automatically create , in a single run , a general (parameterized) solution a problem in the form of a graphical structure whose nodes or edges represent components and where the parameter value of the components are specify by mathematical expression containing free variables . In a parameterized topology , the genetically evolved graphical structure represents a complex structure (e.g electrical circuit , controller , network of chemical reaction , antenna , programming determines the graph's size (it number of node) as well as the graph's connectivity (specifying which node are connected to each other [Koz. 2003] . Neural network is a natural model consist of a number of simple computational units (also called neurons or some times call nodes) connected with each other . Associated with each connection is a so-called weight which corresponds to a synapse in the biological model . The connection between units call network topology [Ben. 96][Fel. 94] . One of this network topologies feed-forward network .A feed-forward network has a layer structure .

Each layer consist of units which receive their input from a layer directly below and send their output to units directly above the unit .There are no connection with in a layer . The *Ni* input are feed into the first layer of *N h,1* hidden units . The input units are merely fan-out unit ; no processing takes place in these units . The activation of a hidden units is a function *Fi* if the weighted inputs plus a bias . The output of the hidden units is distributed over the next layer of *N h,2* hidden units, until the last layer of hidden units, of which the outputs are feed into a layer of no output units [Ben. 96][Jai. 96]. Using genetic programming as automatic tool for design feed-forward neural network architecture it firstly came by koza and rice in 1991 [Koz. 91] . This study was implement a genetic programming for optimizing neural network structure and compare its ability model and gene-gene interaction with traditional back propagation neural network . We show some researchers works in this field in our introduction . Our research use genetic programming for designing feed-forward neural network as simulated trees structures models . Each tree model give new design for neural network . The newer design that generated by genetic programming evaluation is send to genetic algorithm for training . Our proposal way its represent more general method for design active neural network that give the desired output with less neurons and connectivity for the problem that deal with it .

3-The Genetic Algorithm for Training Neural Networks :

When training a feed-forward neural network such as multilayered perceptron , back propagation is often employed . Back propagation is a local search method which performs approximate steepest gradient descent in the error space . It is thus susceptible to two inherent problems : it can get stuck in local minima a problem which becomes heightened when the search space is particularly complex and multimodal , and it requires a differentiable error space to work efficiently . In addition , it has been found that back propagation does not perform well with network with more than two or three hidden layers and it have long time for training the network [Mit. 96][Ebe. 98][Cha. 2001] . Genetic algorithm is a biological technique for optimization , we will use it as a training tool for neural network because genetic algorithm able to find global minima in complex , multimodal spaces , not requires a differentiable error function and they are more flexible . We use it in our research for reason mentioned before .And we represent each set of neural real values weights in chromosome directly . Each chromosome of real set is evolved in genetic algorithm to give the training set of weights .

4-The proposal of combination genetic programming and genetic algorithm

We suggest in this paper a method for designing and training neural network . Our work deal with global approach for designing neural network topology that evolved in each generation by genetic programming and then each new design is send to genetic algorithm that specify training set of weights for this design . Here we demonstrate each of the two algorithms :

4-1 The proposed genetic programming algorithm :

We use steady state genetic programming (ssGP) algorithm , that keep the diversity of solutions in population [Mit. ,96] [Gol. ,89] . ssGP work to yield one child in one generation and then replace it in population . The ssGP algorithm steps demonstrate in the next flow chart :

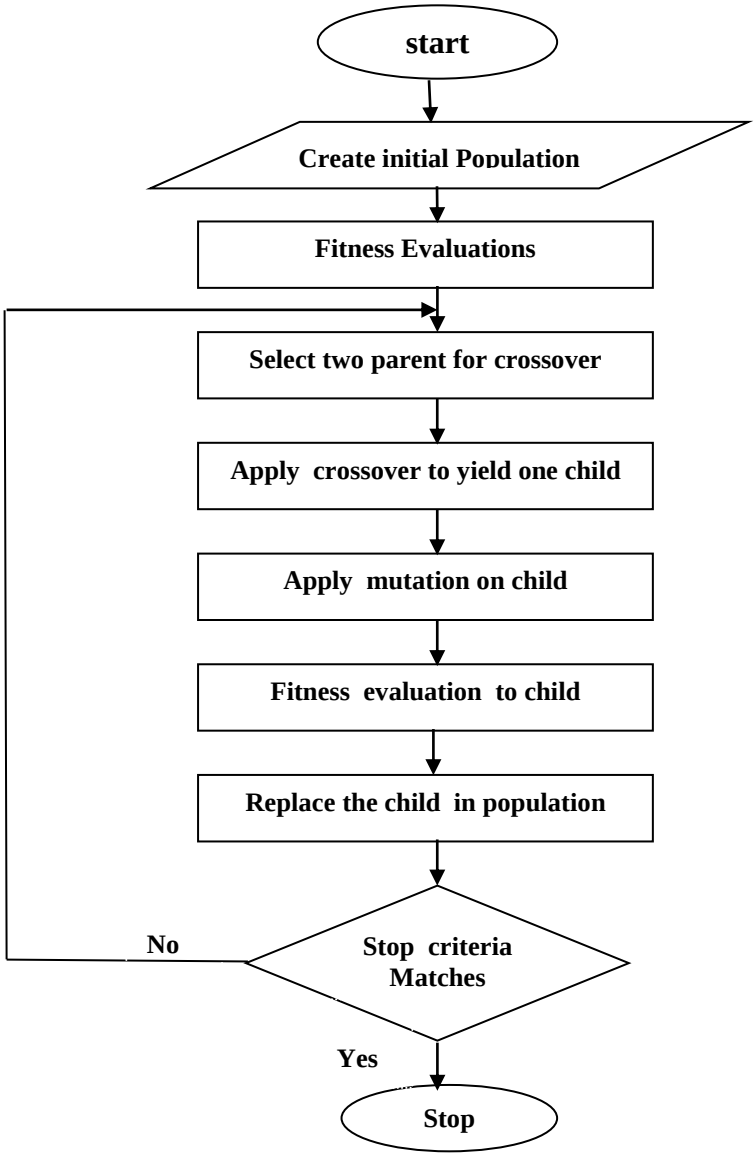


Figure (1) :Flow chart of ssGP

4-1-1 Create initial Population

The population of genetic programming algorithm is composed from number of trees structures . we use 30 tree as a population size . In this work we generate the tree structures for GP by number of steps . These tree its represent a solution for a candidate neural network design . We decode trees as one dimensional array of dynamic array .Our representation deal with non binary tree ,this representation is developed from binary tree representation that mention in[Hor. 87] . Each tree is represent feed-forward neural network by some rules as follow :

1- if the problem that represented by the neural network have one output neurons , then the first node in tree it's output node .Else (much than 2 neurons) the second level of tree represented to be the output layer and the first node is just be root node no more .

2-The N-hidden layer can represented by tree throw considering the levels from 2 or 3 to m-1 as hidden layers (Where m is a permitted number of tree levels) .

3-The last level of tree (level m) is consider as input layer for the network .

4-The connectivity between nodes it must be matches rules of neurons connections as a feed-forward neural networks , the output node is connected to hidden node or input node and the hidden node is connected to node in the next level and input node it has no next level to connect its call terminal node (these connection is store in embedded dynamic array from the location 2 and the location 1 is used for neuron) .

if we suppose the problem that neural network deal with it have much than one output neurons then the algorithm of generating trees are the follow :

1-Specify the location in array as root node that be outside of design

2-Specify the locations from 2 to k as output layer (k is number of output neurons) .

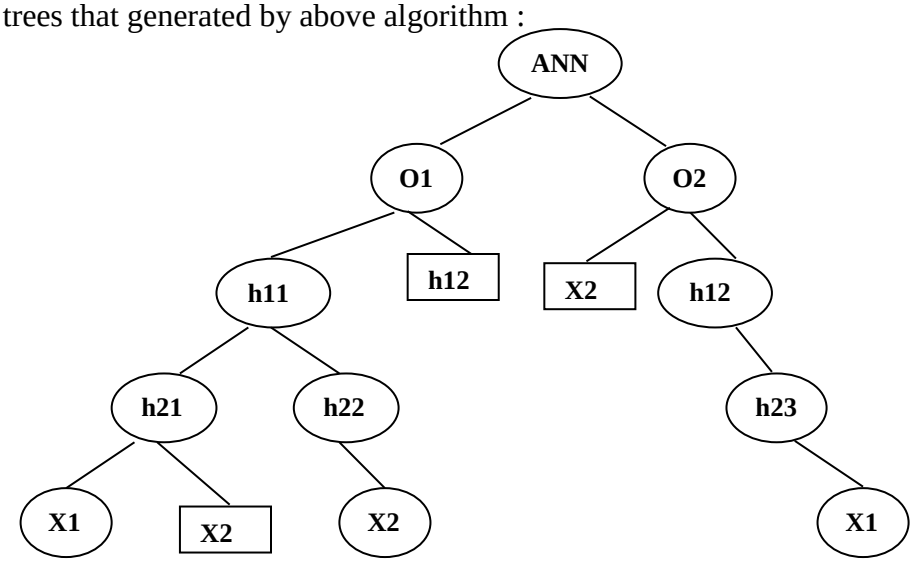
3-Loop from the location k+1 to (2^{m-1})-1 (where m is maximum number of tree levels) and then specifying each location as hidden node or but nil randomly .

4-inspecting all hidden node that connected to nil and make it connected to input node that selected randomly .

5-Estableisheed nodes connections randomly by storing in the dynamic array from location 2 to R (were R random number between 0 and the number of node in down level for this node) names of nodes in the down level. But in this step we must follow the rule of connection that we mention above.

The step two in this algorithm specify if the trees that generated is binary or non , this is the problem have 2 or more output neurons .Here if we suppose we work with XOR

parity problem ,this problem have 2 input and 2 output digits , then here an example of one trees that generated by above algorithm :



Figure(2) :Example of random Generated tree

From above figure we see that the nodes O1 ,O2 are output nodes , the nodes h11,h12 are hidden node in first layer , the nodes h21,h22 and h23 are second hidden layer and the nodes X1 and X2 are inputs nodes in input layer. The square nodes mean this is a connection but not real node (neuron) and the square refer to dynamic array .Our

representation for trees it use as maximum tree levels equal to 6 levels. This tree is decoded as one dimensional array according to some of calculation rules :

1-The father of node *i* is in the location $\lfloor i/2 \rfloor$ when $i \neq 1$, else the node *i* don't have a father (root node).

2-The left child for node *i* is in the location 2*i* when $2i \leq n$ (n is size of array) , else the node *i* don't have a child (terminal node) .

3-The right child for node *i* is in the location 2*i*+1 when $2i+1 \leq n$,else the node *i* don't have a child (terminal node) .

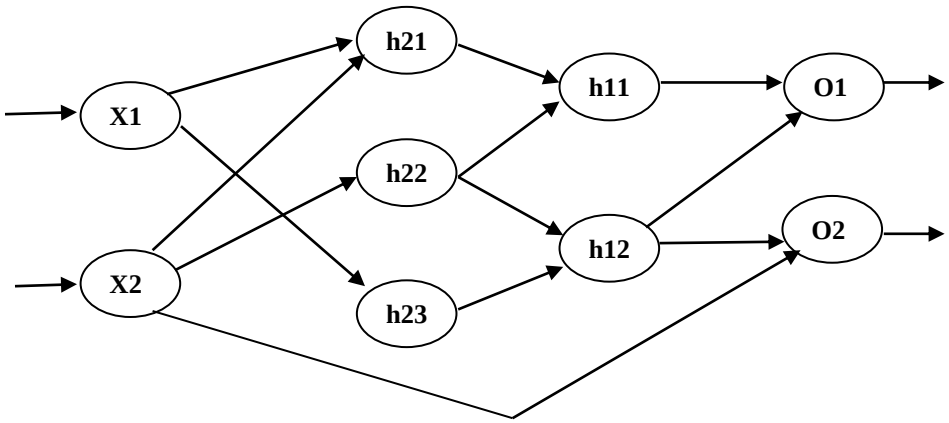
Then the above tree in figure(2) can be decoded according to this representation as follow :

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Ann	O1	O2	h11	Nil	h12	Nil	h21	h22	nil	nil	h23	nil	nil	Nil	X1	nil

18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
X2	nil	Nil	nil	Nil	nil	X1	nil	nil	nil	nil	nil	nil	nil	nil

Figure(3):Tree representation as one dimensional array

The tree in figure(2) that represented in figure(3) it equivalent to feed-forward neural network as in figure(4) :



Figure(4): Equivalent feed-forward neural network for above tree

The other details of ssGP algorithm in figure(1) it well be demonstrated in table below :

Table(1) : Details of Genetic programming operators

GENETIC PROGRAMMING OPERATORS	ITS DETAILS
Fitness evaluation (fitness function)	Each individual in genetic programming send to genetic algorithm to calculate it fitness as follow: Fitness =MSE + N * P ,where MSE : mean squares of errors for the desert output and the network output . N: number of hidden neurons + network connections ,this important factor that effect on fitness value when N increase fitness value is decrease and verse versa . P : penalty number , this value is terming throw experiments , we find the small value (0.0001) is better from large value 0.1 because the large value enable to create small size networks with high error , but our small value(0.0001) enable to creation large enough network for problem to be solved .
Selection of parents	Tournament selection : this method select two parents randomly form population and then take the best one [Mit. ,96] [Sch. ,97].
Crossover	Branch crossover : this done by choosing two nodes randomly in two selected parent and then cut each branch form selected nodes and swap this branches to create new Childs . [Ash. ,2006] [Koz. ,98] .
Mutation	Node mutation : it achieve by select one node in tree structure randomly and then replaced it by another one selected randomly from same type [Ash. ,2006] [Koz. ,98].
Stopping criteria	We use two criteria : 1-The number of generations are reach to the maximum number (we use 400 as Max. generations). 2-the average fitness for population not change for number of generations *cycle* (we use *cycle*=30) .

4-2 The proposed genetic algorithm

We use steady stead genetic algorithm that deal with population chromosomes . Each chromosome in genetic algorithm is decoded as one dimensional array of real numbers , these real numbers represent the set of weights for the received neural network from genetic programming .The flow chart of this algorithm we demonstrated it in figure(1) .Then the detail of this algorithm is as follow :

4-1-1 Create initial Population

We use 30 individuals in population .Each individual is length depend on the number of connections in neural network . Then the individual fill with real numbers that generated randomly in the rang [-3..3] .

The other detail of ssGA it well demonstrated in table(2) as follow :

Table(2) : Details of Genetic Algorithm operators

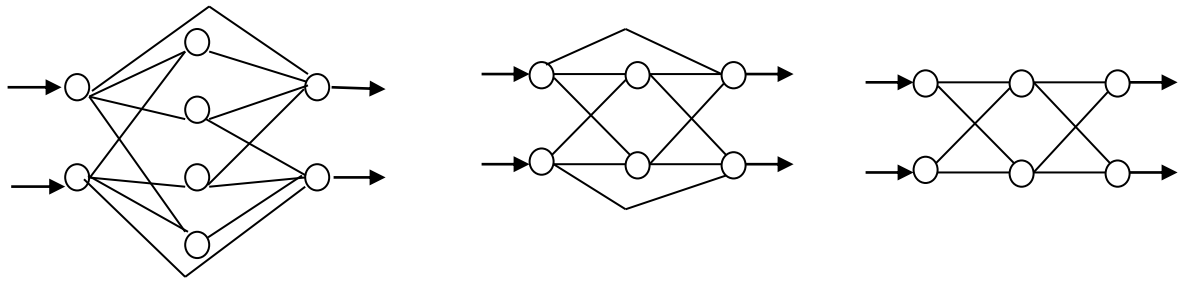
GENETIC ALGORITHM OPERATORS	ITS DETAILS
Fitness evaluation (fitness function)	Fitness =MSE , MSE : mean squares of errors for the desert output and the network output . MSE=SSE / M , where SSE is sum squares of errors for the desert output and the network output . M : number of data set.
Selection of parents	Tournament selection : this method select two parents randomly form population and then take the best one [Mit. ,96] [Sch. ,97].
Crossover	Uniform Crossover : it done by swapping genes between parents according to probability equal to 0.5 [Mit. ,96] [Sch. ,97].
Mutation	It work by adding little random real number in rang [-0.01..0.01] in probability equal to 0.1 .
Stopping criteria	We use two criteria : 1-The number of generations are reach to the maximum number (we use 400 as Max. generations). 2-the average fitness for population not change for number of generations *cycle* (we use *cycle*=30) .

5-The Results

We apply our genetic work on two non linear problems , these selected problems are conceder from hard problems .

5-1 XOR parity problem

This problem it have two input and two output digit .We ran our composed algorithms on the parameters of this problem . After ran the proposed algorithm we get number of training neural designs , Here we display the best three designs (models) that have one hidden layer with fitness values from model 1 to 3 sequentially (0.001,0.0012,0.0018) as follow :

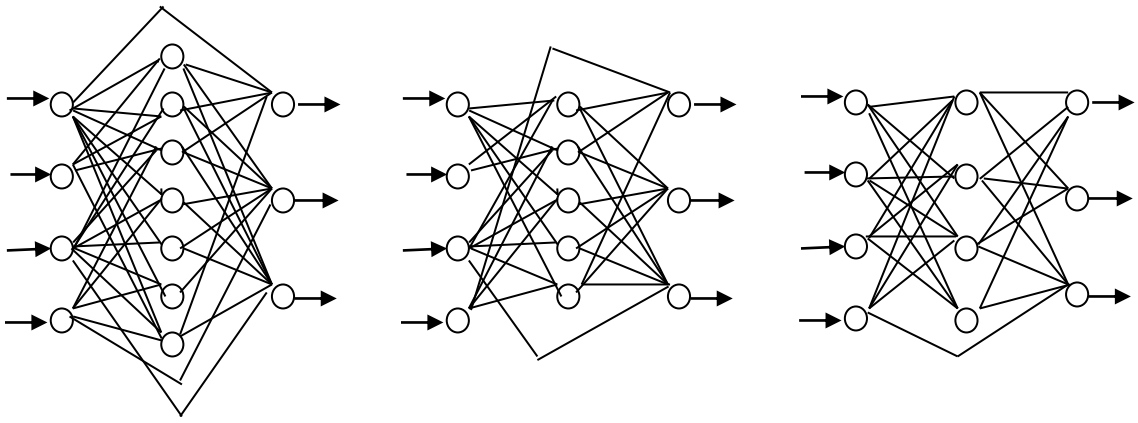


Model 3 ***Model 2*** ***Model 1***

Figure(5) : The best three models for XOR parity problem

5-2 Adder Problem

This problem have four input and three output ,2 bit for one input digit, therefore this problem is conceder more complex from XOR parity problem .here we display our three best models with fitness values from model 1 to 3 sequentially (0.0031,0.0033,0.0041) throw our run algorithm as follow :



Model 3 ***Model 2*** ***Model 1***

Figure(6) : The best three models for Adder problem

6- Conclusion

1-Our research is conceder global approach for minimization neural design and training, but our approach increase in run time while the problems that work on its be more complex .

2-The representation that we suggest for trees structure give us good exploitation for memory by using dynamic array for representing nodes connections , there for if we use static array (this mean two dimensional static array for tree representation) our approach it well be suffer from to be global method .

7-Future work

1- Study the effect of using another crossover operations and mutations .

2-We suggest apply our approach on more complex problems.

3-Using of bag propagation algorithm to work as slow finishing after run of genetic algorithm .

8-Refrences

[Ash. 2006] Ashlock D. [2006] , *Evolutionary Computation for Modeling and Optimization* ,Spring Science + Business Media , Inc.

[Ben 96] Ben K. and Patrick V.D.S [1996] ,*An Introduction to Neural Networks*, Spring.

[Cha. 2001] Chambers L. [2001], *The practical hand book of genetic algorithm applications* , Second Edition , acid-free paper.

[Ebe. 98] Eberhart R. C., Shi Y. [1998], *Evolving artificial neural networks*, international conference on neural network and brain processing (ICNN&B98) .

[Fel. 94] Felzer T. [1994] , *Artifitial Neural Networks* ,Spring.

[Han. 97] Hany I. F, George D.and Christos D.[1997], Application of Neural Networks and Machine Learning in Network Design , IEEE Journal on Selected areas in Comunication, VOL. 15, NO. 2.

[Jai. 96] Jain A. K. and Mao J [1996] ,*Artificial Neural Networks : A tutorials* ,IEEE.

[Hor. 87] Horowitz E. and Shahani S. [1987] , *Fundamentals of Data Structures In Pascal* , New York .

[Koz. 91] Koza JR and Rice JP[1991]: *Genetic generation of both the weights and architecture for a neural network IEEE Press* ,II**:**

[Koz. 94] Koza J. R. [1994], *Genetic Programming II: Automatic Discovery of Reusable Programs* .

[Koz.& Ric.] Koza J. R. and Ricardo P. , *A Genetic Programming Tutorial* .

[Koz. 98] Koza J. R. [1998] , *Genetic Programming: On The Programming of Computers Programs by Means of Natural Selection*, A Bradford Book , MIT Press .

[Koz. 2003] Koza J. R.[2003],*Genetic Programming IV,* Springer Science+Business Media, Inc.

[Mar. 2003] Marylyn D. R., Bill C.and Lance W. [2003], *Optimization of neural network architecture using genetic programming improves detection and modeling of gene-gene interactions in studies of human diseases,* BMC Bioifomatics .

[Mit. 96] Mitchell M. [1996] , *An Introduction to Genetic Algorithm* , A Bradford Book , MIT Press .

[O're. 2005] O'Reilly M.,Yu T.,Riolo R.and Worzel B[2005]., *Genetic Programming Theory and Practice II* ,Springer.

[Riv. 2006] Rivero D., Julian D. ,Juan R. and Javier P. [2006] , *Artificial Neural Network Development by means of Genetic Programming with Graph Codification* , International Journal of Applied Mathematics and Computer Sciences Volume 1 Number 1.

[Sch. 97] Schmidt M. and Stidsen T. [1997] , *Hybrid Systems : Genetic Algorithm , Neural Networks and Fuzzy Logic* , DAIMIIR .