Hybridization of Genetic Algorithm and Neural Network for Optimization Problem

Gaurang Panchal, Devyani Panchal

***Abstract*— The use of both, genetic algorithms and artificial neural networks, were originally motivated by the astonishing success of these concepts in their biological counterparts. Despite their totally deferent approaches, both can merely be seen as optimization methods, which are used in a wide range of applications. "Genetic algorithms (GA) are good at taking large, potentially huge search spaces and navigating them, looking for optimal combinations of things, solutions you would find difficult to accomplish." A genetic algorithm (GA) is an iterative search, optimization and adaptive machine learning technique premised on the principles of Natural selection. They are capable to finding solution to NP hard Problems. Neural Networks utilizing back propagation based learning have promisingly showed results to a vast variety of function and problems. TSP is one such classical problem for computation.**

***Index Terms*—Genetic Algorithm, Neural Network, Weight Optimization, Neural Network Parameter**

I. Introduction

Genetic Algorithms are search, optimization and machine learning techniques based on the mechanics of Natural Selection and Natural Genetics. Genetic Algorithms (GA)[1] are adaptive procedures of optimization and search that find solutions to problems by an evolutionary process inspired in the mechanisms of natural selection and genetic science[1-5]. A genetic algorithm is a search method that functions analogously to an evolutionary process in a biological system. They are often used to find solutions to optimization problems. GAs are Randomized search and optimization technique guided by the principle of natural genetic systems. Currently, these algorithms are being highly considered in those

Problems with complex solution spaces for those which we do not have good algorithms to solve them [1]. Genetic algorithms are algorithms that combine search algorithms with the genetics of nature. Past data and

Result is used to determine future results, in a 'survival of the fittest' kind of way. In a generation, the elements (the data represented as a string) that work the best move on to a new generation, with some mutations added in, just incase some important piece of information is lost through these changes. In a GA, a solution of our problem is called individual. In essence, it consists of maintaining a population of a given number of individuals, each one of them characterized by a genetic code (genotype)[10] that identifies it univocally; thus an evolution of such population is simulated during the course of time, based on the apparition of new individuals resulting from crossovers, mutations and direct reproductions of the parents. An evaluation or objective function plays the role of the environment to distinguish in each generation that relatively good solutions reproduce, and that relatively bad solutions die, to be replaced by offspring of the good. Basically, we can say that a GA is based on the following components, for any type of application: a "genetic" representation of solutions to the problem; a way to create an initial population of solutions; an evaluation function to measure the fitness of any solution, and plays the role of the environment, in which the better solutions may have greater probability of survival; "genetic" operators that effect the composition of children during reproduction; value for the parameters that the algorithm uses to guide its evolution: population size, number of generations, crossing and mutation probabilities, etc[6-9].GA,NN and FL ,each of the technologies, in their own right and merit, has provided efficient solution to a wide range of problem. Objective of the hybridization has been to overcome the weakness in one technology during its application, with the strengths of the others by appropriately integrate them. It investigating better methods of problem solving. Hybrid systems have a tremendous potential to solve problem. Inappropriate use of technology can backfire. It has ability to locate the neighborhood of the optimal solution quicker than other conventional search strategies.

II. Travelling Salesman Problem

A. *TSP – The Problem*

The Traveling Salesman Problem (TSP) is a classic graph-searching problem. In practical terms the problem can be thought of as that of a salesman who wishes to perform a circular tour of N cities, calling at each city once and traveling the minimum total distance possible (because he has a penny-pinching boss and has to keep fuel costs as low as possible). It is the most famous example of an intractable (NP-complete) problem. For a tour N cities, the number of possible solutions is N! /2N, or equivalently 1/2(N-1)! For just twenty-five cities the number of possible journeys is so immense that a computer evaluating a million possibilities per second would take 9.8 billion years- around two-thirds of the age of the universe - to search through them all. A genetic algorithm can be used to find a solution is much less time.

Manuscript received April, 2015.

Gaurang Panchal, *Department of Computer Engineering, Chandubhai S Patel Institute of Technology, Charotar University of Science and Technology, Changa, India,*

Devyani Panchal, *Department of Computer Engineering, Chandubhai S Patel Institute of Technology, Charotar University of Science and Technology, Changa, India,*

ISSN: 2278 – 1323

Although it probably will not find the best solution, it can find a near perfect solution in less than a minute. Each city has a set of co-ordinates (x, y), the distance between 2 cities, a & b is given by:

$$d_{ba} = d_{ab} = \sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}$$

Suppose there are N cities to be visited, first of all, let vectors X, Y contains the X and Y Coordinates of the N cities, which could be described as below:

X = [X1 X2 X3… XN]; Y = [Y1 Y2 Y3… YN];

Then let matrix City_dist denotes the distances between any two cities. The elements of City_dist, say, (City_dist) ij be the distance from city i to city j.

Notice that the matrix is symmetric, which means that for any two cities Ci and Cj, the distance from Ci to Cj is identical with the distance from Cj to Ci.

Now we can introduce the critical matrix for the representation of TSP. Matrix travel_sequence contains the order of all the cities that are traveled through, thus it defines a solution to the problem. Each column denotes which city is chosen to visit in each step. Suppose we are getting city sequence like:

C4 →C5→C3→C2→C6→C7→C1→C8

Thus, the total traveling distance will be:

d45+d53+d32+d26+d67+d71+d18

III. Implementation of GABPN

Now, we have developed a GABPN with real value coding for solving the Traveling Salesman Problem. We have developed TSP problem using Genetic Algorithm based Back Propagation Network in C++.

A. Parameter of GA

Genetic Algorithm (GA) depends on some parameters like population size, maximum generation number, probability of crossover, a goal condition and probability of mutation [2]. In our present study, we have taken the values of those parameters as follows: pop size = 20, crossover rate = 0.8, mutation rate = 0.10, maximum generation =100.

B. Initialization

For the Traveling Salesman Problem, here we have taken binary encoding to generate initial random population. Initial random population is depending upon the number of cities depending upon that we will initial population. The initialization of any component of a chromosome can be done by random initialization, as the boundary of each component is not specified in the problem. The BPN Network configuration is l-m-n (l input neurons, m hidden neurons and n output neurons) so the values of l, m and n is 5.Here the Number of weights are (l + n) m = 50, the Gene Length is d = 2 and Chromosome length L = (l + n) *md = 100 [2].

C. Random Population

The objective of this method is to start or create initial random population. The name of the method signifies the work it has to do in program life cycle. Its basic aim is to populate the chromosomes. There are no arguments to this method. In this method, first the random number generator is initialized be its appropriate seed [11-15]. Then for number of chromosomes and for number of bits time the module to generate the random number for the bit string object of parent chromosome and to store them in it is repeated. The method can be of form,
The Algorithm can be of the form:

1. Initialize the seed of random number generator.
2. for the entire parent chromosome do
3. For the entire bit single chromosomes do
3a. Generate random number according to the encoding scheme
3b. Assign this value to the bit of the chromosome.
4. End

D. Intermediate Population

Objective of this method is to select best parents for reproduction and generating new chromosomes. The name of the method signifies the work it has to do in program life cycle. Its basic work is to create an intermediate population, by choosing best parent chromosomes for reproduction process [2].
The Algorithm can be of the form:

1. for the entire parent chromosome do
1a. map the probability of selection on roulette wheel
2. for the entire parent chromosome do
2a. generate a random number
2b. for all parent chromosomes do
2b.1 if random number in the range of any parent chromosome
2b.1 (a) mark parent
2c. store the marked chromosome in intermediate population
3. End

E. Extract Weights

To determine the fitness values for each chromosome, we extract weight from each of chromosomes [2].

Wk =
$$\frac{+X\ kd+2 * 10(d-2) + X\ kd+2 * 10(d-2) + \ldots X\ (k+1)\ d}{10(d-2)}$$
If 5<= X kd+1 <= 9

Wk =
$$\frac{-X\ kd+2 * 10(d-2) + X\ kd+2 * 10(d-2) + \ldots X\ (k+1)\ d}{10(d-2)}$$
If 0<= X kd+1 <= 5

Let X1, X2…Xd …XL representing a chromosomes and Xkd+1, Xkd+2…X (k+1) d representing the kth gene in the chromosomes

F. Fitness Value

We first randomly generated the initial population Po of size P = 40 which represents the 40 chromosomes of Po. (Chromosomes are C1, C2…C40). Let W1, W2 …….W40 be the weight sets extracted from each of Ci (i = 1 to 40). For the fixed weight set Wi, the BPN is trained for the entire input instance given.

Let O1, O2, and O3 be the calculated outputs of the BPN. Compute,

$$E1 = [(T11 - O11) * (T11 - O11)] + [(T21 - O21) * (T21 - O21)]$$
$$E2 = [(T12 - O12) * (T12 - O12)] + [(T22 - O22) * (T22 - O22)]$$
$$E3 = [(T13 - O13) * (T13 - O13)] + [(T23 - O23) * (T23 - O23)]$$

The root mean square of the error is

$$E = Sqrt\ [(E1+E2+E3) / 3]$$

The fitness F1 for the chromosomes C1 is given by

$$F1 = 1 / E$$

G. Reproduction

In this phase, the mating pool is first formed, before the parent chromosomes reproduced to deliver offspring with better fitness. For the given problem, the mating pool is first formed by excluding those chromosomes C1 with the least fitness Fmin and replacing it with a duplicate copy of the chromosomes Ck reporting the highest fitness Fmax. That is, the best-fit individuals have multiple copies while fit individuals die off.

H. Mate Pool

Consider the initial population of chromosomes Po generated earlier, with Fi, where i = 1 to 40.Let Fmax = Fk and Fmin = Fl, be the maximum and minimum fitness values. We remove all chromosomes with a fitness values Fmin and replace it with copies of chromosomes whose fitness values is Fmax [2].

I. Convergence

For any problem, if the GA is correctly implemented, the population evolves over successive generation with the fitness value increasing towards the global optimum. Convergence is the progression constitutes the population share the same fitness values.

The population P1 now undergoes the process of selection, reproduction and crossover. The fitness values for the chromosomes in P1 as computed, the best individual replicated and reproduction carried out using two point crossover operators to form the next generation P2 of chromosomes. The process of generation proceeds until at one stage 95% of chromosomes in the population Pi converge to the same fitness values. At that stage, the weights extracted from the population Pi are the final weight to be used by the BPN.

J. Crossover

Objective of this method is to apply the main operator of reproduction (i.e. Crossover) to the chromosomes of intermediate population. The name of the method signifies the work it has to do in program life cycle [3]. Its basic work is to produce crossover between each paid of parent chromosomes. This is analogous to the crossover process in biological reproduction. The most general method is single point cross over. This method takes parent chromosomes from intermediate population, selected by roulette wheel or any other selection technique, and then at a particular random point, these chromosomes are crossed. So, the region after the cross point in both chromosomes are interchanged. This crossover process is Probabilistic. The maximum probability of selection will decide, whether there should be crossover or not.
The Algorithm can be of the form:

1. for the entire parent chromosome do
1a. toss the coin for crossover
1a.1 if random number below probability of crossover
1a.1 (a) generate random number for crossover point
1a.1 (b) for bits after cross point
1a.1 (b). (i) Interchange bits between parent chromosomes
2. End

K. CleanUp Operator

We present a solution to real world traveling salesman problems, which we have accomplished through the use of a more restrictive method for calculating distance between cities and through the introduction of a new genetic operator. This new operator Cleanup has been specifically designed for use in real world evolutionary TSP systems. The cleanup operator improves the convergence speed by reducing the number of epochs required to identify a near-optimal tour; in each instance a significant reduction is convergence time was observed. By using the cleanup operator we can easily eliminates the duplicate cities from out travel sequences. This clean up operator is useful when we are doing crossover. After the crossover when we merge the two travel sequences that time some of the cities will miss in new sequence at that time we can eliminate the duplicate cities from new travel sequence. The cleanup operator[17-19] improves the convergence speed by reducing the number of epochs required to identify a near-optimal tour.

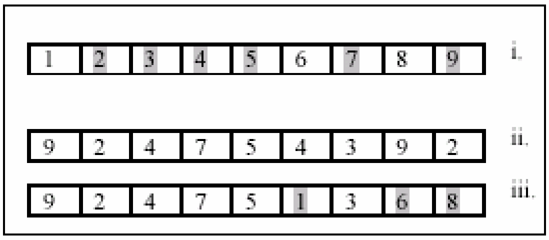

Fig. 1. Cleanup Operator

L. Mutation Operator

Objective of this method is to randomly change one of the bits in the parent chromosome bit string after crossover. The name

ISSN: 2278 – 1323

of the method signifies the work it has to do in program life cycle. Its basic work is to randomly change information encoded in the chromosome bit string structure. This is same as biological mutation, in which sometimes error is generated during the reproduction process there are mainly two different mutation techniques, which can be implemented in this method as its basic algorithm. The most general method is single bit mutation. In this method, the coin is tossed to get the probability of mutation. If it lies below the specified range, the parent chromosome's bit string is mutated at random point. Generally, the probability of mutation is kept very low. This is a very good operator, as it exploits Global Optimum
The Algorithm can be of the form:

1. for the entire parent chromosome do
1a. generate a random number for probability of mutation
1a.1 if number within range
1a.1 (a) generate random mutation point
1a.1 (b) mutate the bit
2. End

IV. Experiments and Results

We use the Genetic Algorithm based back propagation network to solve the TSP with 5, 8, 10, 12 cities. Here are the general results for the near-optimum solutions. We set the parameters, as Crossover Rate is 0.8, Mutation Rate is 0.3 and we are assuming the distance between two cities is 1.Here we have taken Unit Distance between two cities. Here we are getting results using simple Genetic Algorithm, and Genetic Algorithm based Back Propagation Network and Genetic Algorithm Based Back Propagation Network with Cleanup Operator.

Table 2. Distance measurements

Generation	GABPN with Cleanup Operator (Dist)	GABPN without Cleanup Operator (Dist)	GA (Dist)
20	846.43	934.69	846.64
40	826.77	846.64	907.23
60	826.24	826.24	847.95
80	738.95	740.80	871.30
100	610.62	739.28	650.62

If we compare Genetic Algorithm based Back propagation Network with other conventional method like shortest path algorithm, Branch and Bound algorithm then we will get better result as compare to them. Following is the comparison result with GABPN.

Table 3. Comparison of Algorithms

Cities	10	15	20
Branch and Bound Algorithm (Dist)	681.14	733.60	999.95
GABPN (Dist)	616.52	681.58	738.26

Some experiments show that the branch-and-bound algorithm provides good optimal solutions at reasonable computation time if numbers of cities are less. There is no way to determine how far we are from the optimal solution

V. Conclusion

Genetic algorithm based BPN can be used to find a solution is much less time. Although it probably will not find the best solution, it can find a near perfect solution in just 10 seconds. The great weakness of this branch-and-bound algorithm, in common with other heuristic approaches, there is no way to determine how far we are from the optimal solution. From all result obtained, we can conclude that the Genetic Algorithm based back propagation is best search method as compared to the other conventional methods.

References

[1] Goldberg, D.E. "Genetic algorithms in search, optimization, and machine learning", Pearson Education, 2001
[2] Vijyalakshmi Pai G.A. & Rajasekaran S. Neural networks, fuzzy logic and genetic algorithms, Synthesis and applications. Reading, Prentice-Hall of India, 2004
[3] Man Mohan, Gupta P.K . Problems in operations research, Methods and Solutions. Reading, Sultan Chand and Sons, 1992
[4] Tapojit Kumar, Susan M. Schilling, "Comp- arison of optimization techniques in large scale Transportation Problems"
[5] R. K. Gupta, A. K. Bhunia, "An Application of real-coded Genetic Algorithm for integer linear programming in Production-Transportation Problems with flexible transportation cost ", AMO-Advanced Modeling and Optimization, Volume 8, Number 1, 2006
[6] A. Ganatra, G. Panchal, Y. Kosta, C. Gajjar, "Initial classification through back propagation in a neural network following optimization through GA to evaluate the fitness of an algorithm," International Journal of Computer Science and Information Technology, vol. 3, no. 1, pp. 98–116, 2011.
[7] G Panchal, D. Panchal, A. Ganatra, Y Kosta, "Performance analysis of classification techniques using different parameters," Springer Verlag-Germany (Springer-LNCS), vol. 6411, 2010.
[8] G. Panchal, A. Ganatra, Y. Kosta, D. Panchal, "Searching most efficient neural network architecture using Akaikes information criterion (AIC)," International Journal of Computer Applications, vol. 1, no. 5, pp. 41–44, 2010.

[9] G. Panchal, "Forecasting Employee Retention Probability using Back Propagation Neural Network Algorithm," IEEE 2010 Second International Conference on Machine Learning and Computing (ICMLC), Bangalore, India, pp. 248–251, 2010.

[10] G Panchal, A Ganatra, Y Kosta, D Panchal, "Behaviour analysis of multilayer perceptrons with multiple hidden neurons and hidden layers," International Journal of Computer Theory and Engineering, vol. 3, no. 2, pp. 332–337, 2011.

[11] G. Panchal, A. Ganatra, P. Shah, D. Panchal, "Determination of overlearning and over-fitting problem in back propagation neural network," International Journal on Soft Computing, vol. 2, no. 2, pp. 40–51, 2011.

[12] G. Panchal, A. Ganatra, P. Shah, Y. Kosta, "Unleashing Power of Artificial Intelligence for Network Intrusion Detection Problem," International Journal of Engineering Science and Technology., vol. 2, no. 10, 2010.

[13] G. Panchal, D. Panchal, "Solving NP hard Problems using Genetic Algorithm," IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 6 (2), 1824-1827, 2015.

[14] G. Panchal, D. Panchal, "Efficient Attribute Evaluation, Extraction and Selection Techniques for Data Classification," IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 6 (2), 1828-1831, 2015.

[15] G. Panchal, A. Ganatra, D. Panchal, Y. Kosta, "Employee Retention Probability using Neural Network Based Back Propagation Algorithm," IEEE Xplore, vol. 248, 2010.

[16] G. Panchal, D. Panchal, "Forecasting Electrical Load for Home Appliances using Genetic Algorithm based Back Propagation Neural Network", International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), Vol. 4 (4), 2015.

[17] G. Panchal, Y. Kosta, A. Ganatra,D. Panchal, "Electrical Load Forecasting Using Genetic Algorithm Based Back Propagation Network," in 1st International Conference on Data Management, IMT Ghaziabad. MacMillan Publication, 2009.

[18] G. Panchal, A. Ganatra, "Classification and Optimization to Evaluate the Fitness of an Algorithm". Lambert Academic Publisher, Germany, 2012.

[19] G. Panchal, A. Ganatra, "Optimization of Neural Network Parameter using Genetic Algorithm". Lambert Academic Publishing, Germany, 2012.

Gaurang Panchal received the B.E. degree in Information Technology from the Saurashtra University, in 2003, and the M.E. degree in Computer Engineering from Dharmsinh Desai University, in 2007. Presently, he is working as an Assistant Professor at Chandubhai S Patel Institute of Technology, Charogar University of Science and Technology, Changa, India. His research interests include Soft Computing and its applications. Mr. Panchal is serving as a reviewer in more than 20 international journals including IEEE Transactions and Taylor & Francis. He has also been serving as an editorial board member in various international journals.

Devyani Panchal received the B.E. degree in Computer Engineering from the Sardar Patel University, in 200. Presently, she is working as an Assistant Professor in Charotar University of science and Technology, CHARUSAT, changa, India. Her research interests include Soft Computing and its applications.

ISSN: 2278 – 1323