OPTIMAL LINEAR SHORTEST DESTINATION BY USING HOPFIELD-TANK

Kyaw Swar Hlaing, Thaung Thaung Win
University of Computer Studies, Mandalay
lusoelayy84@gmail.com, thaung2winster@gmail.com

ABSTRACT

The Travelling Salesman Problem is a classical problem in the field of combinatorial optimization, concerned with efficient methods for maximizing or minimizing a function of many independent variables. Given the positions of N cities, which in the simplest case lie in the plane, to find the shortest closed tour in which each city can be visited once. In this system, the neural network model of Hopfield-Tank and Nearest Neighbor algorithm are applied to the Travelling Salesman Problem. Microsoft Visual C#.Net 2005 programming language is used to implement this system.

.

Keyword: Hopfield-Tank, Travelling Salesman Problem, Optimization by using Neural Network

1. INTRODUCTION

For last several decades, researchers have been trying to use Hopfield Neural (HN) network to solve combinatorial optimization problems because it could find an optimal solution faster than traditional methods. The HN network was proposed by physicist John J. Hopfield in1982 [4]. The Travelling Salesman Problem (TSP) consists in the problem of determining the shortest circuit which can be made visiting a list of cities, in such a way that each city is visited (once and only once). This is an "NP-complete" optimization problem, which means that, for a problem with a reasonable dimension, there are so many hypothesis to consider that it is generally unpractical to look for an optimum solution, because the computation cost is too high. That is why; finding a "good" solution in a short computation time is considered to be a good result. A neural net allows the fast determination of solutions that, when valid, are all among the best 30 or 40 of the hundreds of thousands of possible hypothesis. Choose among them the one that corresponds to the shortest traveled distance; obtain in a fast way a solution which, if it is not the best, it "closes to" the best [4].

A simple strategy for Traveling Salesman Problem is to enumerate all feasible tours: a tour is feasible if it satisfies the criterion that every city is visited but once to calculate the total distance for each tour, to pick the tour with the smallest total distance. This simple strategy becomes impractical if the number of cities is large. For example, if there are 10 cities for the traveling salesperson to visit, there are 10! = 3,628,800 possible tours, where 10! denotes the factorial of 10: the product of all integers forms 1 to 10 and is the number of distinct permutations of the 10 cities. This number grows to over 6.2 billion with only 13 cities in the tour, and to over a trillion with 15 cities.

2. MOTIVATION OF THE SYSTEM

This system is implemented to find the optimal path of travelling salesman problem by using Hopfield-Tank and Nearest Neighbor algorithm. The system uses Hopfield neural network architecture. Input for the system consists of the points of cities that the salesman traveled. The output is the shortest tour length. The input parameters can be changed variable and the output is changed according to the input parameters. The Nearest Neighbor algorithm generates the shortest path but the solutions can be more than one and so the Hopfield-Tank algorithm can specify the only the optimal path. The output result of the Hopfield-Tank algorithm can be analyzed with the result of shortest distance in each iteration.

3. HOPFIELD NETWORK

Hopfield network is a dynamic network, which iterates to converge from an arbitrary input state. The Hopfield Network works as minimizing an energy function. The Hopfield net is fully connected network. It is a weighted network where the output of the network is fed back and there are weights to each of this link.

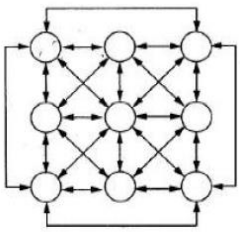

Figure 1. Fully Connected Hopfield Network for TSP for 3 cities

Here n^2 neurons are used in the network, where n is the total number of cities. The neurons here have a threshold and step function. The inputs are given to the weighted input node. The network then calculates the output and then based on Energy function and weight update function, converges to the stable solution after few iteration. The most important task on hand is to find an

appropriate connection weight. It should be such that invalid tours should be prevented and valid tours should be preferred. The output result of TSP can be represented as following. The example here is for 4 cities. The 4 cities TSP need 16 neurons.

	#1	#2	#3	#4
C1	0	1	0	0
C2	1	0	0	0
C3	0	0	0	1
C4	0	0	1	0

Figure 2. Tour Matrix Obtained as the Output of the Network

The corresponding visiting route, in this example is

City2→ City1→City4→City3→City2

So the total traveling distance is

$$D = D_{21} + D_{14} + D_{43} + D_{32} \quad (1)$$

Network Inputs

The inputs to the network are chosen arbitrarily. The initial state of the network is thus not fixed and is not biased against any particular route. If as a consequence of the choice of the inputs, the activation works out to give outputs that add up to the number of cities and initial solution for the problem, a legal tour will result. A problem may also arise that the network will get stuck to a local minimum. To avoid such an occurrence, random noise is generated and added. Also there are inputs that are taken from user. The user is asked to input the number of cities want to travel. Distance matrix in n*n square matrix whose principal diagonal is zero. The figure below shows a typical distance matrix for 4 cities.

	C1	C2	C3	C4
C1	0	10	18	15
C2	10	0	13	26
C3	18	13	0	23
C4	15	26	23	0

Figure 3. Distance Matrix

Energy Function

The Hopfield network for the application of the neural network can be best understood by the energy function. The energy function used should satisfy the following criterions

- The energy function should be able to lead to a stable combination matrix.
- The energy function should lead to the shortest traveling path. The energy function used for the Hopfield neural network is

$$E_1 = A_1 \Sigma_i \Sigma_k \Sigma_{j \neq k} x_{ik} x_{ij}$$
$$E_2 = A_2 \Sigma_j \Sigma_k \Sigma_{j \neq k} x_k i\, x_{ji}$$
$$E_3 = A_3 (n - \Sigma_i \Sigma_k x_{jk})^2$$
$$E_4 = A_4 \Sigma_k \Sigma_{j \neq k} \Sigma_i d_{kj} x_{ki} (x_{j,(i+1)} + x_{j,(i-1)})$$
$$E_{total} = E_1 + E_2 + E_3 + E_4 \quad (2)$$

Here A_1, A_2, A_3, A_4 are positive integers, the setting of these constants are critical for the performance of Hopfield network. X_{ij} is the variable that city i is the jth city visited in a tour. Thus X_{ij} is the output of the j^{th} neuron in the array of neurons corresponding to the i^{th} city. There have n^2 such variable and their value will finally be 0 or 1 or very close to 0 or 1.

Weight Matrix

The network here is fully connected with feedback and there are n^2 neurons, thus the weight matrix will be a square matrix of $n^2 * n^2$ elements. According to the Energy function the weight matrix can be set up as follows

$$W_{(jk)(lj)} = -A_1 \delta_{il}(1 - \delta_{kj}) - A_2 \delta_{kj}(1 - \delta_{il}) - A_3 - A_4 d_{il} (\delta_{j,(k+1)} + \delta_{j,(k-1)}) \quad (3)$$

Here the value of constants A_1, A_2, A_3, A_4 is same as have it in the Energy function. Weights are also updated keeping into mind various constraints to give a valid tour with minimum cost of travel.

Activation Function

The activation function also follows various constraints to get a valid path. Hence the activation function can be defined as follows.

$$\Delta a_{ij} = \Delta t \,(Term1 + Term2 + Term3 + Term4) \quad (4)$$
$$Term1 = - aij / \tau$$
$$Term2 = - A_1 \Sigma_{k \neq j} x_{ik}$$
$$Term3 = - A_2 \Sigma_{k \neq i} x_{kj}$$
$$Term4 = - A_3 (\Sigma_i \Sigma_k x_{jk} - m)$$
$$Term5 = - A_4 \Sigma_{k \neq i} d_{kj} (x_{k,(j+1)} + x_{k,(j-1)})$$

Denote the activation of the neuron in the i^{th} row and j^{th} column by a_{ij}, and the output is denoted by x_{ij}.

- A time constant t, is also used.
- A constant m is also another parameter used.
- The first term in activation function is decreasing on each iteration.
- The second, third, fourth and the fifth term give the constraints for the valid tour.

The activation is updated as

$$a_{ij}(new) = a_{ij}(old) + \Delta a_{ij} \quad (5)$$

Output Function

This a continuous Hopfield network with the following output function

$$x_{ij} = (1 + \tanh(\lambda a_{ij})) / 2 \quad (6)$$

Here X_{ij} is the output of the neuron. Ideally the output is wanted either 1 or 0. But the hyperbolic tangent function gives a real number and settles at a value that is

very close to desired result, for example, 0.956 instead of 1 or says 0.0078 instead of 0.

Nearest Neighbor Method for Traveling Salesperson Problems

Nearest Neighbor algorithm finds a suboptimal solution to the traveling salesperson problem; that is the algorithm finds a Hamiltonian circuit in a complete graph with positive weights, but does not guarantee finding the one with least weight. The path is represented as a list of vertices. The weight is assigned to the variable w.

4. SYSTEM DESIGN

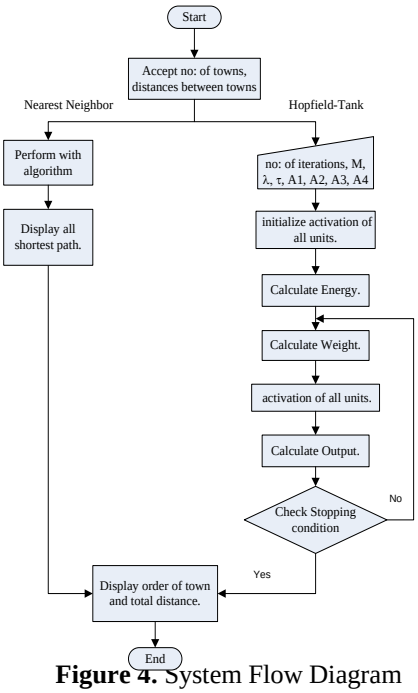

Figure 4. System Flow Diagram

4.1.1 Parameters

The parameter settings in the Hopfield network are critical to the performance of the Network. The various parameter used and their intial value set are as follows: A_1 : 1.0, A_2 : 1.0, A_3 : 1.0, A_4 : 1.0, λ : 1.0, τ : 1.0 and M : 0.01.

4.1.2 Output Result of TSP Program

Hopfield neural network is efficient and it can converge to stable states in hundreds times iterations. The state values are analog to facilitate the finite differential calculation. The output first gives the inputs that are taken from the user. i.e. the number of cities and their distance in the form of distance matrix. Then for those cities the output that is generated is displayed in the form of Tour Matrix, Tour Route and Total Distance Traveled. The solution is optimal or near optimal.

The output graph of Hopfield-Tank can be seen compare with the output of the Nearest Neighbor algorithm output.

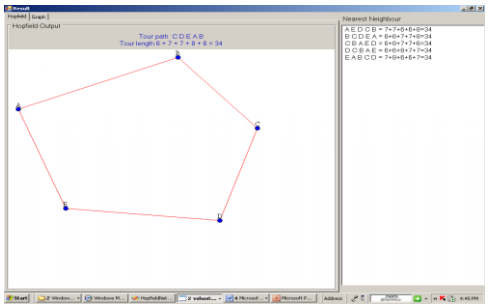

Figure 5. Output Result

4.1.3 Discussion of Result and Comparison

TSP, one of the most famous NP-complete problems, has been simulated using Hopfield Network. The simulation is satisfactory. The result are tested along with the code are for 4, 5, 6, 7, 8, …., 26 cities respectively. The number of iteration required to converge the network in each case can be summarized as follows.

- The results show that as the number of cities increases the number of iteration required increases sharply. The increase is not a linear increase.
- One more thing that is noticed is that the number of iteration required for the convergence does not remain same for any particular city. For example for 5 cities the network usually converged after 80 to 110 iterations, but on few occasions it took around 60 iterations while in few cases it didn't converge at all or took more than 250 iterations. This is because the initial network state is randomly generated. This may sometimes result to no convergence also.
- Many times the result converges to local minimum instead of global minimum. To avoid these random weights are added to the initial inputs.
- The algorithm developed in nondeterministic. Thus it does not promise an optimal or near optimal solution every time. Though it does gives near optimal solution in most of the cases, it may fail to converge and give a correct solution.
- Many times when the energy of the system is calculated, it was found to increase instead of decreasing. Thus the algorithm failed in few cases. This again is the consequence of the random initial state of the network.
- In 50 % of test cases the algorithm converged, while in 45 % algorithm failed to converge and in remaining 5% the energy of the system increased instead of decreasing.

There are various advantages of using Hopfield network though Nearest Neighbor algorithm approach.

- Hopfield neural network setup is very optimal for the solution of TSP. It can be easily used for the optimization problems like that of TSP.
- It gives very accurate result due to very powerful and complete Energy equation developed by Hopfield and Tank.
- This neural network approach is very fast compared to standard programming techniques used for TSP solution.

- With very few changes this algorithm can be modified to get the approximate solution for many other NP-complete problems.

This graph is the result of shortest distance based on every iteration.

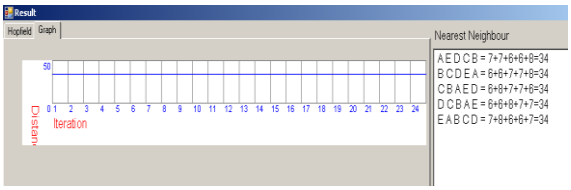

Figure 6. Analysis of Output by Iteration

5. CONCLUSION

In this system, the optimal shortest path of TSP has been found by using Hopfield-Tank algorithm. The Hopfield neural network is tested with variable number of point in the cities through A to Z, by varying the number of the iteration and other input parameters. The accuracy result is satisfactory in some cases in input points 4 to 6 and weak in number of points is above 6 compare with Nearest Neighbor algorithm. But when reducing the number of input parameter, the accuracy of the network depends on how important the variation of the parameters in finding the optimal path compare with Nearest Neighbor algorithm.

Artificial Neural Network (ANN) technology is still very new and is developing quickly. Humans are witnessing fast expansion of neural network-based intelligent machines. At the beginning of this thesis, the objective is to make this system for finding the optimal shortest path with Hopfield-Tank algorithm. However, in practice, calculation of neural network for finding any points of cities is convenient, but the results are always not optimal because of the input parameter setting when the number of cities is increased.

The network does not always give optimal solution though in most cases it's near optimal. Few changes or improvement can be made to energy function along with other functions like weight updating function and activation function to get better answer. Various values of constants (i.e. A_1, A_2, A_3, A_4) can be tried to get optimal or near optimal result in the present algorithm. Even if one of the distances between the cities is wrong the network has to start form the very first stage. Some way or method needs to develop so that this error can be handled. If a city is wanted to add or delete, the network has to be again started from the initial state with the required changes. Some equations can be developed so that these changes can be incorporated. Optimization of the result and modification can be done on the network, which will improve the result and the convergence of the result. The system can be extended by using other Neural Network algorithm like Self-Organizing Map and the output results are compared. The algorithm can be modified for solving other NP-complete problems.

REFERENCES

[1] D.W.Tank and J. J. Hopfield
"Simple Neural Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit", IEEE Transactions on circuits and systems, vol. 33, No.5.

[2] J.M.Zurada,
"INTRODUCTION TO ARTIFICIAL NEURAL SYSTEMS", Info Access Distribution Pte Ltd, 1992.

[3] J.Price, M.Gunderloy,
"MASTERING VISUAL C#.NET", SYBEX Inc, 1151 Marina Village Parkway, Alameda, CA 94501,2002.

[4] J. J. Hopfield and D. W. Tank
"Neural Computation of Decisions in Optimization Problems," Biological Cybernetics, 1985, vol. 52.

[5] J.Hertz, A. Krogh and R.G. Palmer
"Introduction to the Theory of Neural Computation", Addison-Wesley Publishing Company

[6] R.Gandhi
"Implementation of Travelling Salesman's Problem Using Neural Network" ECE 559: Traveling Salesman's Problem's Solution using Hopfield NN. (Fall 2001)

[7] R.Collan,
"THE ESSENCE OF NEURAL NETWORKS", Prentice Hall, Inc…., 1999.

[8] R.Rojas
"The Hopfield Model: Neural Networks", Springer-Verlag, Berlin, 1996

[9] S.Haykin,
"NEURAL NETWORKS: A COMPREHENSIVE FOUNDATION", Second Edition, Prentice Hall, Inc…., 1999.