Critical Analysis of Hopfield's Neural Network Model and Heuristic Algorithm for Shortest Path Computation for Routing in Computer Networks

Farah Sarwar, Abdul Aziz Bhatti
University of Management and Technology, Lahore, Pakistan

Abstract— **Shortest path routing and its computation is a crucial point in computer networks, and has significant impact on overall network's performance. Being an issue of salient importance, many algorithms were proposed for shortest path computation and are still under research for more enhancements. Hopfield proposed a neural network based architecture for such optimization problems. Mehmet and Park Keum suggested improved energy functions for this neural network to implement it for routing in computer networks. A* search algorithm is a heuristic based approach, with the properties of Dijikstra algorithm and is used for same purpose. Performances of both approaches are compared and results are analyzed.**

I. INTRODUCTION

Routing has significant impact on overall network's performance as it can enhance the capability of communication system. Furthur, it is known that ideal routing system follow optimum shortest path. For this purpose different algorithms were proposed and are still under research for more enhancements. Some of these algorithms are implemented at network layer [2] where a selected cost is assigned to each link and a shortest path algorithm is used to find either minimal cost path or minimal length path.

The problem of finding the shortest path (SP) from a single source to a single destination in a graph arises as a sub problem to many broader problems, including routing in computer networks. This problem has some well-known polynomial algorithmic solutions, namely Bellman Ford's or Dijkstra's. The Dijikstra's algorithm is considered as localized algorithm [21]. Each node has its own routing table and keeps on updating it , depending upon the connectivity with neighboring nodes. Change in cost of one link will let each node to update its routing table. It is preferred over centralized algorithm.

Bellman-Ford algorithm performs the same operation using distributed routing approach as each node updates its routing table based on the local information from neighboring nodes [21]. Deadlocks or looping of packets might happen due to inconsistent routing paths so it is less efficient than localized algorithm.

A* algorithm is used in path finding and is an extension to Dijiksta's work[6]. However it is more efficient as it uses heuristics and is commonly used for one pair of nodes or routers. For monotone heuristics it is considered to be equivalent to Dijikstra's algorithm.

Because of the high computational speed, nonlinear analog response and large connectivity of parallel processing systems, neural networks are considered to provide best option for shortest path computation [8][15]. The major feature which neural networks offer is feedback connectivity, parallelism and collective analog mode-with each neuron summing up the inputs of hundreds or thousands of others to provide graded output. Analog systems are preferred over digital because of the ability to change the interacting variables simultaneously and self-consistently, but are less accurate than digital. Neural networks have previously provided solution to many optimization problems. Travelling Sales Man problem, analog to digital conversion, optimum traffic flow control in routing, multihop radio routing without interference etc are among few examples.

Rauch and Winarske [18] introduced the use of neural network to find shortest path in routing and their work was based on Hopfield's neural network. Then Zhang and Thomopoulos [23] proposed little bit more practical algorithm, Ali and Kamoun's proposed algorithm [2] was adaptive and adjusted itself if the cost value changed. Park and Choi [16] figured out looping problem in previously proposed algorithm so they tried to fix this problem with an improved energy function but Ahn and Ramakrishna [1] found the instability problem in their work too. Recently Park and Keum [17] claimed to solve all of the above mentioned problems with an improved energy function. However there are some ambiguities as the implementation of their neural network is not mentioned. Even with a different step size the output of neural network can change and show drastic variations from the stable output.

Problem of shortest path computation in routing, under discussion, is explained in section II. Hopfield's neural network model and heuristic algorithm are discussed in section III and IV respectively. Section V will show the simulation results and conclusion is presented in section VI.

II. PROBLEM FORMULATION

Consider an undirected graph G(V,E), where V defines the set of vertices (nodes or routers) and E is the set of edges. As we are considering undirected path so cost of forward and backward flow will be same. As in routing, delay factor, link capacity, average flow, routing computation etc. are important factors too so all of these are combined in one term and we call it link cost. Link cost

978-1-4577-1929-5/12/$26.00 ©2011 IEEE

from router i to router j is be defined as C_{ij}, considering the fact that $C_{ij}=C_{ji}$ and C_{ij} is zero for non-existent links.

The objective function of this shortest path routing problem is to minimize the total cost from given source node s to destination node d. Neural network and heuristic algorithm will tackle same objective function but with different methodology.

III. HOPFIELD'S ORIGINAL MODEL

Hopfield proposed electrical model based on resistors, capacitors and operational amplifiers having functionality of neurons of human brain, shown in Fig 1. Only two possible states exist for this network i.e., V^0_i and V^1_i for representing 0 or 1 states respectively [10]. Neurons have sigmoid input-output relation. Although output V_i of each neuron can vary between 0 and 1 but it should finally settle down to either 0 or 1 either after comparison with a threshold value or after a given number of iterations.

$$V_i = \begin{cases} 0, & V_i < threshold\ value \\ 1, & V_i \geq threshold\ value \end{cases} \quad (1)$$

Sigmoid function used to specify input-output relation is:

$$\mathrm{Vi} = \mathrm{g(ui)} = \frac{1}{1+e^{-\lambda_i . U_i}} \quad (2)$$

Here λ provide the steepness of sigmoid function and U_i is the respective input voltage for ith neuron. The input of each neuron came from two sources, external bias current I_i and outputs of other neurons.

$$\mathrm{U_i} = \sum_{\substack{j=1 \\ j\neq i}}^{N} T_{ij}V_j + I_i \quad (3)$$

Whereas

T_{ij} = Conductance between output of jth neuron and input of ith neuron

V_j = Output of jth neuron

I_i = External bias current of ith neuron, representing data provided by user

Operational amplifiers provide the input-output sigmoid relation, capacitors are used to show lagging nature of input u_i from instantaneous output V_j of other neurons and finite impedance is represented by T^{-1}_{ij} [10]. Thus this resistance-capacitance charging circuit determines the rate of change of input u_i. Using Kirchhoff's current law, system equation becomes

Input current = Output current

$$\sum_{\substack{j=1 \\ j\neq i}}^{N} T_{ij}V_j + I_i = C_i\left(\frac{du_i}{dt}\right) + \frac{u_i}{R_i} \quad (4)$$

$$C_i\left(\frac{du_i}{dt}\right) = \sum_{\substack{j=1 \\ j\neq i}}^{N} T_{ij}V_j + I_i - \frac{u_i}{R_i} \quad (5)$$

Whereas:

$$1/R_i = 1/\rho_i + \sum_j 1/R_{ij} \quad (6)$$

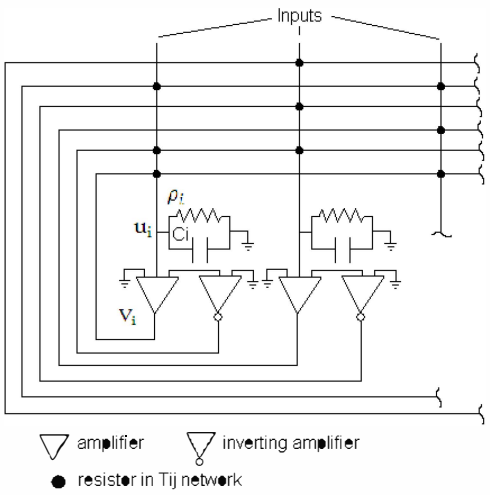

Figure 1. Hopfield's proposed analog neural network circuit

T_{ij} is thus the synapse efficacy and its magnitude is given by $1/R_{ij}$, where R_{ij} is the resistor connecting the output of j to the input line i. ρ_i, R_i, and Ci are input resistance of neuron i, total input resistance and capacitance of the ith neuron respectively.

Energy function for this network is

$$E = -\frac{1}{2}\sum_{\substack{j=1 \\ j\neq i}}^{N} T_{ij}V_iV_j + \sum_i \left(1/R_i\right)\int_0^{V_i} g^{-1}(V)dV - \sum_{i=1}^{N} I_iV_i \quad (7)$$

Its time derivative for symmetric T is

$$\frac{dE}{dt} = -\sum_{i=1}^{N}\frac{dV_i}{dt}\left(\sum_{\substack{j=1 \\ j\neq i}}^{N} T_{ij}V_j - u_i/R_i + I_i\right) \quad (8)$$

From Eq. 5 and Eq. 8, Eq. 8 becomes

$$\frac{dE}{dt} = -\sum_{i=1}^{N} C_i\left(\frac{dV_i}{dt}\right)\left(\frac{du_i}{dt}\right) \quad (9)$$

$$= -\sum_{i=1}^{N} C_i\, g^{-1'}(V_i)\left(\frac{dV_i}{dt}\right)^2 \quad (10)$$

Since $g^{-1}(V_i)$ is a monotone increasing function and C_i is positive, each term of Eq. 10 is nonnegative. Therefore:

$$\frac{dE}{dt} \leq 0, \frac{dE}{dt} = 0 \rightarrow \frac{dV_i}{dt} = 0\ for\ all\ i$$

Above equation proves that energy function finally converges to minima. The final solution may not be the most precise one but will always be a stable and valid solution because of this convergence property. So it promises a stable output if and only if the actual problem is mapped properly on this network as a different way can cause a problem and results may not be satisfactory.

Explaining it further, any optimization problem which is needed to be solved with the help of Hopfield neural network needs to be mapped on it in terms of connection matrix and input biases. Even small variations in this mapping activity can lead to drastic changes in the final solution. Many researchers have tried to solve optimization problems using this neural network but not all of

them get the satisfactory results [19] [5]. Behrooz Shirazi and Sue Yih explained in [19] the problems of invalid results. Considering their mentioned points, a properly mapped problem will converge to stable and valid solution, if not best, as neural network itself has converging property. However Wilson and Pawley pointed out in [5] that although Hopfield and Tank provided a great method for solving optimization problems yet it needs many modifications for proper implementation as Hopfield did not mention properly the method of solving which he used to get stable results. The dynamic equation used to keep system moving is

$$\frac{dU_i}{dt} = -\frac{U_i}{\tau} + \frac{\partial E}{\partial V_i} \tag{11}$$

Whereas τ is the time constant and its value is equal to the product RC and is considered as unity.

For neural network the routing problem is formulated further in the following way:

"If C_{ij} represents the cost from router i to router j and a packet is to be transmitted from source router s to destination router d then a directed link form s to d becomes:

$$L^{sd} = (s, i, j, k, \ldots, r, d).$$

Thus the objective function of this problem is to find minimum cost path from router s to d.

$$C^{sd} = C_{si} + C_{ij} + C_{jk} + \ldots. C_{rd} \tag{12}$$

Whereas, C^{sd} represents the total cost of the path."

The energy function of the routing problem should be mapped properly in a way so as to make the neural network move towards stable state with the condition that objective function is achieved. Mehmet and Kamoun proposed following modified cost function for it [2]:

$$E = \frac{\mu_1}{2}\sum_{x=1}^{n}\sum_{\substack{i=1\\ i\neq x}}^{n} C_{xi}.V_{xi} + \frac{\mu_2}{2}\sum_{x=1}^{n}\sum_{\substack{i=1\\ i\neq x}}^{n}\rho_{xi}.V_{xi} + \frac{\mu_3}{2}\sum_{x=1}^{n}\left\{\sum_{\substack{i=1\\ i\neq x}}^{n} V_{xi} - \sum_{\substack{i=1\\ i\neq x}}^{n} V_{ix}\right\}^2 + \frac{\mu_4}{2}\sum_{i=1}^{n}\sum_{\substack{x=1\\ x\neq i}}^{n} V_{xi}.(1 - V_{xi}) + \frac{\mu_5}{2}(1 - V_{ds}) \tag{13}$$

Whereas:

$$V_{xi} = \begin{cases} 1, & if\ the\ edge\ from\ node\ x\ to\ node\ i\ is\ in\ the\ shortest\ path \\ 0, & otherwise \end{cases} \tag{14}$$

And

$$\rho_{xi} = \begin{cases} 1, & if\ the\ edge\ from\ node\ x\ to\ node\ i\ does\ not\ exist \\ 0, & otherwise \end{cases} \tag{15}$$

In Eq.13 μ_1 term minimizes the total cost of a path by taking into account the cost of existing links. The μ_2 term prevents the nonexistent links from being included in the chosen path. The μ_3 term is zero if incoming connections are equal to outgoing connections for router. The μ_4 term force the output of neural network to converge to either of the two states i.e. '1' or '0'. And the $\mu_5$term makes the packet to get back to source router after completing the path. The cost of this newly generated connection is not included in total cost of the path as it is a forced connection. The dynamic equation of this energy function is:

$$\frac{\partial U_{xi}}{\partial t} = -\frac{U_{xi}}{\tau} - \frac{\mu_1}{2}C_{xi}(1 - \delta_{xd}.\delta_{is}) - \frac{\mu_2}{2}\rho_{xi}(1 - \delta_{xd}.\delta_{is}) - \mu_3\sum_{\substack{y=1\\ y\neq x}}^{n}(V_{xy} - V_{yx}) + \mu_3\sum_{\substack{y=1\\ j\neq i}}^{n}(V_{iy} - V_{yi}) - \frac{\mu_4}{2}(1 - 2V_{xi}) - \frac{\mu_5}{2}\delta_{xd}.\delta_{is} \tag{16}$$

It has been figured out that the algorithm proposed by (16) does not converge well for larger networks. Even after 20 node network, loops start appearing in the solution and thus solution becomes invalid. So, researchers keep on trying to find an algorithm which has better converging capability for large number of nodes. Park and Keum then presented following algorithm:

$$E = \frac{\mu_1}{2}\sum_{x=1}^{n}\sum_{\substack{i=1\\ i\neq x}}^{n}\left(V_{xi}\sum_{\substack{j=1\\ j\neq x}}^{n}\left((1 - C_{xj})C_{xi}\right)\right) + \frac{\mu_2}{2}\sum_{x=1}^{n}\sum_{\substack{i=1\\ i\neq x}}^{n}\rho_{xi}V_{xi} + \frac{\mu_3}{2}\sum_{x=1}^{n}\left(\sum_{\substack{i=1\\ i\neq x}}^{n} V_{xi} - \sum_{\substack{i=1\\ i\neq x}}^{n} V_{ix} - \phi_x\right)^2 + \frac{\mu_4}{2}\sum_{x=1}^{n}\sum_{\substack{i=1\\ i\neq x}}^{n} V_{xi}(1 - V_{xi}) + \frac{\mu_5}{2}\sum_{x=1}^{n}\sum_{\substack{i=1\\ i\neq x}}^{n}\left(V_{xi}\sum_{\substack{j=1\\ j\neq x}}^{n} V_{jx}\right) \tag{17}$$

Where ϕ_x is defined as:

$$\phi_x = \begin{cases} 1, & if\ i = s, \\ -1, & if\ i = d \\ 0, & otherwise \end{cases} \tag{18}$$

In (17) μ_1 term performs two functions, it minimizes the total link cost of routing path as well as takes into account all cost values around the node. Function of μ_2, μ_3 and μ_4 terms are same as in (13). However, μ_5 term is modified to make the flow of routing unidirectional and it prevents loop in the network. This network has capability of solving multi destination problem, however, we'll solve problem of single destination only. The values of these parameters are selected in a relative manner but needs proper attention as performance of the neural network is highly dependent on them. Values of μ_2 and μ_3 should be large enough to avoid non-existent links to be part of route and for valid solution respectively. μ_1 should be less than μ_2 to let μ_2 take effect. μ_4 should be small for slowly converging stable solution. Its dynamic equation is:

$$\frac{\partial U_{xi}}{\partial t} = -\frac{U_{xi}}{\tau} - \frac{\mu_1}{2}\sum_{\substack{j=1\\ j\neq x}}^{n}(1 - C_{xj})C_{xi} - \frac{\mu_2}{2}\rho_{xi} - \mu_3\left(\sum_{\substack{j=1\\ j\neq x}}^{n}(V_{xj} - V_{jx}) - \phi_x\right) + \mu_3\left(\sum_{\substack{j=1\\ j\neq i}}^{n}(V_{ij} - V_{ji}) - \phi_j\right) - \frac{\mu_4}{2}(1 - 2V_{xi}) - \frac{\mu_5}{2}\sum_{\substack{j=1\\ j\neq x}}^{n} V_{jx} \tag{19}$$

IV. HEURISTIC ALGORITHM

Heuristic algorithm has lot of applications in finding shortest path in different fields, like travelling salesman problem, Hamiltonian tour, capacitated vehicle routing problem etc. It is also used for finding SP in routing too. Based on the knowledge of all available nodes and link costs a shortest path can also be found in routing and the shortest path in routing corresponds to path with lowest cost. This algorithm is considered to provide good answers in reasonable time but provide no guarantee of optimum solution [14]. The main abstract of this algorithm is "Find a subset T from a set S that minimizes an objective function f and satisfies some criteria".

A* search algorithm actually uses heuristics and also resembles Dijkstra algorithm if heuristics are monotone. The main difference between A* search algorithm and Dijikstra's algorithm is that the later one finds out SP from one node to all others, however it is not the case with A* algorithm as it finds the SP between one pair only [6] . It consider whole problem as a problem of graph and solve accordingly.

V. SIMULATION RESULTS

The values of the parameters used for Mehmet's and Park's method are listed in Table I. Euler's method is used to solve differential equation system of both networks as it is iterative process and values of U_{xi} need to be updated accordingly.

$$U_{xi}(t+\Delta t) = U_{xi}(t) + \Delta t\left(-U_{xi}(t) - {}^{\partial E}/_{\partial V_{xi}}\right) \quad (20)$$

Values of V_{xi} are updated using the sigmoid function when all U_{xi}'s are updated. U_{00} are generated randomly within this range $-0.002 \leq u_{xi} \leq 0.002$ to give it a start. As Park had not mentioned the starting technique so it may have affected our results. According to our simulation with these parameter values Park's algorithm is not able to produce proper and stable results and tries to find a direct path from source node to destination node within 1000 iterations and converges to zero if direct path does not exist.

TABLE I.
PARAMETER VALUES

Parameter	Symbol	Algorithms	
		Ali and Kamoun	Park and Keum
Time constant	τ	1	1
Slope	λ	1	1
Step size	Δt	0.0001	0.0001
Coefficients	μ_1	550	950
	μ_2	2550	2500
	μ_3	1950	1900
	μ_5	250	100
	μ_6	1350	500

We are using C_{ij} to be zero for nonexistent links but Park used it to be infinity. Even with the infinite values for such links this algorithm does not converge to any stable state with a network of size greater than 10 routers. However, Mehmet's proposed algorithm is working efficiently even with a network of 50 routers and accurately finds the path from source router to destination router. Results of Mehmet's algorithm correspond to Dijikstra's algorithm till 55 router networks but cost may increase to some extent for greater networks. As A* search algorithm resembles Dijikstra's algorithm so results of both of them are same.

Costs are generated randomly in a unit square. Results are shown in Table II.

VI. CONCLUSION

Shortest path routing is implemented on neural networks with two different energy functions and results are compared with Dijikstra algorithm and A* search algorithm. It has been found out that Mehmet's energy function is more reliable than Park's as results of later start diverting even for network of 10 nodes under the conditions provided in simulation results. As A* resembles Dijikstra alogrithm with a heuristic function involved in it so both give same results. Results of Mehmet's algorithm also resembles to A* till the network size of 55 nodes but diverges a little for greater sizes. However it can be improved further for greater number of routers. So far neural network is comparable with A* with a compromise in programming complexity and speed of convergence. Neural network has far less programming complexity and takes time to give stable output, on other hand A* is much more complex but give results in less time.

TABLE II.
RESULTS

No. of routers in network (N)	Cost generated		
	Ali and Kamoun	Park and Keum	A* algorithm
5	0.017423	0.017423	0.017423
10	0.351288	0.351288	0.351288
15	0.36921	0	0.36921
20	0.12558	0	0.12558
30	0.26866	0	0.26866
40	0.09745	Invalid route	0.09745
50	0.26158	0	0.26158
60	0.18537	Invalid route	0.087321

REFERENCES

[1] Ahn, C.W. and Ramarkrishna, R.S., "Neural Network Based Near-Optimal Routing Algorithm," *Proceedings of international conference on neural information processing,* vol.4, pp.1771-1776, 2002.

[2] Ali, M.K. and Kamoun, F. "Neural Networks for the Shortest Path Computation and Routing in Computer Networks," *IEEE transactions on neural networks,* vol.4, No.6, pp.941-953, 1993.

[3] Bizzarri, A. R., "Convergence Properties of a Modified Hopfield-Tank Model," *Biological Cybernetics* vol. 64, pp.293-300, 1991.

[4] David, 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, 1986.

[5] G.V. Wilson and G.S. Pawley, "On the Stability of the Travelling Salesman Problem Algorithm of the Hopfield and Tank," *Biological Cybernetics*, vol. 57, pp. 63-70, 1988.

[6] Hart, P. E.; Nilsson, N. J.; Raphael, B, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths,". *IEEE Transactions on Systems Science and Cybernetics SSC4,* vol.4, No.2, pp. 100–107, 1968.

[7] Helsgaun, K., "An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic," *European Journal of Operational Research*, vol.12, pp. 106-130, 2000.

[8] Jain, Anil k. and Mao, Jianchang, "Artificial Neural Networks: A Tutorial," Computer, vol. 29, no. 3, pp. 31-44, 1996.

[9] J.J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," *Proc. Natl. Acad. Sci.*, vol. 79, pp. 2554-2559, 1982.

[10] J.J. Hopfield, "Neurons with Graded Response have Collective Computational Properties like those of Two-state Neurons," *Proc. Natl. Acad. Sci.*, vol. 81, pp. 3088-3092, 1984.

[11] J.J. Hopfield and D.W. Tank, "Neural Computation of Decisions in Optimization Problems," *Biological Cybernetics*, vol. 52, pp.141-152, 1985.

[12] Kahng, Andrew B., "Traveling Salesman Heuristics and Embedding Dimension in the Hopfield Model," *IEEE INNS Int.'l Joint Conference on Neural Networks,* vol. 1, pp. 513-520, 1989.

[13] Kamgar-Parsi, Behzad and Kamgar-Parsi, Behrooz, "On Problem Solving with Hopfield Neural Networks," *Biological Cybernetics,* vol. 62, pp. 415-423, 1990.

[14] Lin, S. and Kernighan, B. W., "An Effective Heuristic Algorithm for the Traveling-Salesman Problem," *Operations Research,* vol. 21, No.2, pp.498-516, 1973.

[15] Lippmann, Richard, P., "An Introduction to Computing with Neural Nets," *IEEE ASSP Magazine,* pp. 6-22, 1987.

[16] Park, D.C. and Choi, S.E., "A Neural Network Based Multi-Destination Routing Algorithm for Communication Network," *Proceedings of joint conference on neural networks,* pp.1673-1678, 1998.

[17] Park, Dong-Chul and Keum, Kyo-Reen, "A Shortest Path Routing Algorithm using Hopfield Neural Network with an Improved Energy Function," *International journal of general systems,* vol. 38, No. 7, pp.777-791, 2009.

[18] Rauch, H.E. and Winarske, T., "Neural Networks for Routing Communication Traffic," *IEEE control systems magazine,* vol.8, No.2, pp.26-30, 1988.

[19] Shirazi, Behrooz and Yiu, Sue, "Critical Analysis of Applying Hopfield Neural Net Model to Optimization Problems," *IEEE International Conference on Systems, Man and Cybernetics*, vol.1, pp.210-215, 1989.

[20] Smith, Kate A., "Neural Networks for Combinatorial Optimization: A Review of More Than a Decade of Research," *INFORMS Journal on Computing,* vol. 11, No. 1, 1999.

[21] Wang, Chia-Jiu and Weissler, Paul N., "The Use of Artificial Neural Networks for Optimal Message Routing," *IEEE networks,* vol.9, No.2, pp.16-24, 1995.

[22] Xu, X. and Tsai, W.T., "An Adaptive Neural Algorithm for Traveling Salesman Problem," *Proc. Int. Joint Conf. Neural Networks*, vol. 1, pp. 716-719, 1990.

[23] Zhang, L. and Thomopoulos, S.C.A., "Neural Network Implementation of the Shortest Path Algorithm for Traffic Routing in Communication Networks," *Proceedings of the international joint conference on neural networks,* vol.2, 1989.