Noisy Chaotic Neural Networks For Delay Constrained Multicast Routing

Wen Liu, Student Member, IEEE, Lipo Wang, Senior Member, IEEE, Haixiang Shi

Abstract—The QoS constrained multicast routing is studied widely these years due to the development of multimedia applications such as video-conferencing and video-on-demand. We apply noisy chaotic neural networks (NCNN) on the delay constrained multicast routing problem. The NCNN has richer and more flexible dynamics, and therefore is more efficient compared with the conventional Hopfield neural network as the latter is often trapped at local minima.

I. INTRODUCTION

The multicast routing problem, also known as the Steiner tree problem, aims to minimize the total cost of a multicast tree and is NP-complete (nondeterministic polynomial time complete) [1]. In multicast services, data generated by the source flows through the multicast tree, traversing each tree edge exactly once.

In real-time communications, the network has to find an optimum multicast route, which has enough resources to guarantee the required quality of service (QoS) from the end user [2], [3]. This problem is called QoS constrained multicast routing problem or constrained Steiner tree (CST) problem [4]–[6]. Individual QoS parameters may be conflicting and inter-dependent, thus making the problem even more challenging [7].

The application of neural networks to the routing problem has been motivated by the powerful parallel computational ability of the neural network and the fact that a hardware-implemented neural network can achieve high response speeds [8], [9]. A modified Hopfield neural network model is proposed in [10] to solve the delay constrained multicast routing problem. In this paper, we modify the energy function and solve the problem using noisy chaotic neural networks (NCNN) [11], [12].

This paper is organized as follows. We introduce the delay constrained multicast routing problem in Section 2. The noisy chaotic neural network is presented in Section 3. The simulation result is presented and discussed in Section 4. Finally, we conclude this paper in Section 5.

II. THE DELAY CONSTRAINED MULTICAST ROUTING PROBLEM

A. Problem formulation

We use the formulation proposed in [13]. Considering an \( n \times n \) node communication network with \( D \) destinations, for which the neurons are arranged on \( D \times n \times n \) matrices, matrix \( m \) is used to compute the constrained unicast route from source node \( s \) to destination \( m \). Each element in one matrix is treated as a neuron and neuron \( \Delta \) describes the link from node \( x \) to node \( i \) for destination \( m \) in the communication network. \( \Delta \) is the output of the neuron at location \((x, i)\) in matrix \( m \): \( V^m_{xi} = 1 \), then the link from node \( x \) to node \( i \) is on the final optimal tree for destination \( d_m \). \( V^m_{xi} = 0 \), otherwise.

To characterize the connection topology of the communication network, we define \( P_{xi} \) as \( P_{xi} = 1 \), if the link from node \( x \) to node \( i \) does not exist; \( P_{xi} = 0 \), otherwise. The cost and delay of a link from node \( x \) to node \( i \) are assumed to be real non-negative numbers and are denoted by \( C_{xi}, L_{xi} \), respectively [10]. For non-existing arcs, corresponding entries will be zero. Link costs and link delays are assumed to be different independent functions. For example, costs could be a measure of buffer space or channel bandwidth used, and the delay could be a combination of propagation, transmission, and queuing delay.

B. Problem Definition

The delay constrained multicast routing problem [14], [15] is defined to construct a tree rooted at the source \( s \) and spanning to all the destination members of \( D = \{d_1, d_2, \ldots, d_m\} \) such that 1) the total cost of the tree is minimum; 2) the delay from the source to each destination is not greater than the required delay constraint:

\[
\sum_{x=1}^{n} \sum_{i=1, i \neq x}^{n} L_{xi} V^m_{xi} \leq \Delta \quad V^m_{xi} \in \{0,1\}
\]

where \( \Delta \) is the delay bound. \( V^m_{xi} \) denotes the neuron output of constrained unicast route for destination \( d_m \).

C. The energy function

When solving the optimization problem by neural networks, one needs to define a proper energy function which determines the associated weights between neurons as in (2). The energy function of a Hopfield neural network (HNN) decrease monotonically as neurons update [16]. Hence the energy function should be designed with its minima value corresponding to the optimal solution.

\[
-\frac{\partial E}{\partial V^m_{xi}} = \sum_{y=1}^{N} \sum_{j=1, j \neq y}^{N} w_{yj,xi} V^m_{yi}(t) + I_{xi}
\]
Where $w_{yjx}$ is the connection weight from neuron $(y, j)$ to neuron $(x, i)$ and $I_{xi}$ is the input bias.

Ali and Kamoun [13] proposed the energy function for the shortest path computation. Ponnavaith et al. [10] then extended it to suit the delay constraint multicast routing problem. We change the neuron update rule by using the average value of neuron outputs as the threshold to fire the neuron, i.e., to separate the final outputs into 0 or 1. In the conventional model, the outputs are forced to be 0 or 1 by an energy term $\sum_{x=1}^{n} \sum_{i=1 \neq x}^{n} V_{x}^{m} (1 - V_{xi}^{m})$.

The total energy function $E$ for the delay constrained multicast routing is the sum of energy functions $E_{m}$ ($m \in D$) for delay constrained unicast routing to every destination. Here $E_{m}$ is the energy function of matrix $m$, which is used to find the constrained unicast route from source node $s$ to destination $d_{m}$. It consists of a cost term $E_{1}^{m}$ and several constraint terms $E_{2}^{m}, \ldots, E_{5}^{m}$ [10]:

$$E_{1}^{m} = \sum_{x=1}^{n} \sum_{i=1 \neq x}^{n} C_{x} \frac{1}{1 + \sum_{j=1, j \neq m}^{n} V_{xj}^{m}} V_{xi}^{m}$$

$$E_{2}^{m} = (1 - V_{xms})$$

$$E_{3}^{m} = \sum_{x=1}^{n} \left\{ \sum_{i=1 \neq x}^{n} V_{xi}^{m} - \sum_{i=1 \neq x}^{n} V_{xi}^{m} \right\}$$

$$E_{4}^{m} = \sum_{x=1}^{n} \sum_{i=1 \neq x}^{n} P_{xy} V_{xy}^{m}$$

$$E_{5}^{m} = \sum_{x=1}^{n} \sum_{i=1 \neq x}^{n} L_{yi} V_{yi}^{m} - \Delta$$

where $C_{x}$ is the cost term, $P_{xy}$ are the connection weights. We control the execution time increases.

As described above, when the neural network reaches the minimal value of the energy function, all the constraints are fulfilled and the route cost is optimized at the same time.

### III. Simulation Results

Wang et al [11], [12], [18] proposed a stochastic chaotic simulated annealing (SCSA) method which can solve the traveling salesman problem (TSP) efficiently. The model is a noisy chaotic neural network (NCNN) resulted from adding decaying stochastic noise into the transiently chaotic neural network (TCNN) [19].

Substituting the energy function in the NCNN and TCNN model, we will get the neural network dynamics function. The algorithm is implemented in VC++ and run on a Linux cluster (16-node dual Xeon 3.06 GHz, Intel IA32). We control the iteration of the algorithm through the change in the energy function between two steps: $\Delta E = E(t) - E(t - 1)$. The output is accepted, i.e. iterations stop, if $\Delta E$ is smaller than a threshold (0.002) in three continuous steps.

Values for the weighting coefficients are chosen as follows based on [10]:

$$\mu_{1} = 200 \quad \mu_{2} = 5000 \quad \mu_{3} = 1500 \quad \mu_{4} = 5000 \quad \mu_{5} = 250$$

Without loss of generality [12], we choose the parameters of the neural network as follows: $c_{x}^{m} = \epsilon = 0.004$, independent of neuron location $(x, i, m)$, $k$ corresponds to time step $\Delta t$ used in the Euler approximation, and in our simulations $k = 0.9099$, i.e., $\Delta t = 0.0001$. In addition, $I_{0} = 0.65, \beta_{1} = \beta_{2} = 0.001, z(0) = 0.1$ and $A[u](0) = 0.06$. The initial value for self-feedback $z$ and the amplitude of random noise $n$ are set properly to keep balance of every term in the energy function. Initial inputs of neural networks $U_{x}(0)$ are randomly generated between $[-1, 1]$.

At the end of each iteration, we set each neuron on or off according to the average value $(V_{T})$ of the output matrix. If $V_{xi} \geq V_{T}$, the neuron is on, i.e., $V_{xi} = 1$, which means the link from $x$ to $i$ is chosen in the final optimal tree, and vice versa.

The simulation is carried out on an eight-node communication network [10]. The source node $s$ is labeled 1, and destination nodes are labeled 4, 5, 7 and 8, respectively. The algorithm finds three solutions when the delay constraint ($\Delta$) is 15. The final optimal route is presented in Fig.1. The output matrix of the neural network is also showed in Fig.1(d). We compare these three solutions in Table I (the solutions 1, 2, and 3 corresponds to (a), (b), and (c) in Fig.1, respectively).

When the delay constraint ($\Delta$) is 20, the NCNN finds the optimal tree shown in Fig.2, with total cost 8.

We run the algorithm 10000 times with randomly generated initial states for delay constraint $\Delta$ = 15 and 20, respectively, to compare with conventional Hopfield networks used in [10] and TCNN [19]. The result is listed in Table II. "Route opt" denotes the ratio that the algorithm achieves the optimal result in 10000 runs. The NCNN is capable to jump out of local minima and achieves the global optimal due to its complex dynamics, including chaos and stochastic noise. As a trade off, the execution time increases.
The effect of the damping factor $\beta_1$ and $\beta_2$ is analyzed and results are shown in Table III and IV. In applications, we can balance the “route opt” ratio and the execution time through parameters $\beta_1$ and $\beta_2$ to control the decay of chaotic dynamics and noise. The larger $\beta_1$ and $\beta_2$, the faster the NCNN converges, while the smaller the parameters, the more probable the NCNN can reach the global optimal.

IV. CONCLUSION

The equation governing dynamics of neural networks and the formulation of the delay constrained multicast routing problem have been discussed in this paper. The algorithm can be applied to directed topology, i.e., asymmetric communication networks or applications requiring different delay bounds for different destinations. Simulation results show that the noisy chaotic neural network is efficient in solving the delay constraint multicast routing problem.

REFERENCES

Fig. 2. The optimal delay constrained multicast route when $\Delta = 20$ (a) The optimal route; (b) the output matrix.

**TABLE III**

Performance comparison of the NCNN models with different $\beta_1$ on the 8-node network with $\Delta = 20$, $\beta_2 = 0.001$

<table>
<thead>
<tr>
<th>$\beta_1$</th>
<th>0.01</th>
<th>0.005</th>
<th>0.001</th>
<th>0.0005</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Time (s)</strong></td>
<td>mean</td>
<td>1.10</td>
<td>1.27</td>
<td>1.55</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>0.30</td>
<td>0.37</td>
<td>0.50</td>
</tr>
<tr>
<td><strong>Route opt %</strong></td>
<td>93.02</td>
<td>95.30</td>
<td>96.27</td>
<td>96.85</td>
</tr>
</tbody>
</table>

**TABLE IV**

Performance comparison of the NCNN models with different $\beta_2$ on the 8-node network with $\Delta = 20$, $\beta_1 = 0.001$

<table>
<thead>
<tr>
<th>$\beta_2$</th>
<th>0.01</th>
<th>0.005</th>
<th>0.001</th>
<th>0.0005</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Time (s)</strong></td>
<td>mean</td>
<td>0.89</td>
<td>1.01</td>
<td>1.55</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>0.45</td>
<td>0.45</td>
<td>0.50</td>
</tr>
<tr>
<td><strong>Route opt %</strong></td>
<td>94.04</td>
<td>94.70</td>
<td>96.27</td>
<td>97.23</td>
</tr>
</tbody>
</table>