A Neural Network Approach to Multicast Routing in Real-Time Communication Networks

Chotipat Pornavalai, Goutam Chakraborty and Norio Shiratori
email:{chotipat,gbc,norio}@shiratori.riec.tohoku.ac.jp
Research Institute of Electrical Communication
Graduate School of Information Science
Tohoku University
2-1-1, Katahira, Sendai, 980-77 Japan

Abstract

Real-time communication networks are designed mainly to support multimedia applications, especially the interactive ones, which require a guarantee of Quality of Service (QoS). Moreover, Multicasting is needed as there are usually more than two peers who communicate together using multimedia applications. As for the routing, network has to find an optimum (least cost) multicast route, that has enough resources to provide or guarantee the required QoS. This problem is called QoS constrained multicast routing and was proved to be NP-complete problem. In contrast to the existing heuristic approaches, in this paper we propose a modified version of Hopfield neural network model to solve QoS (delay) constrained multicast routing. By the massive parallel computation of neural network, it can find near optimal multicast route very fast, when implemented by hardware. Simulation results show that the proposed model has the performance near to optimal solution and comparable to existing heuristics.

1 Introduction

Recently there are many efforts to design a packet-switching communication network which could guarantee Quality of Service (QoS) performance required by the applications e.g. video-conferencing. For example, one of the most critical QoS parameter for multimedia applications is a lower limit to the end-to-end delay of data transmission. [12] described 5 important issues which have to be considered to design network that can support real-time communication or can guarantee QoS for multimedia applications. Those are flow specification, routing, resource reservation, packet scheduling, and admission control. In case of routing, we need a routing algorithm that is able to find a route which (1) is capable of providing resources as requested by the applications, (2) use existing resources very effectively to maximize the probability that new request in future can be fulfilled, and (3) can find the route in real-time [11]. Furthermore, most interactive multimedia applications are multi-party applications and they require group communication or multicast support from the network.

The benefits of multicast are that it can reduce transmission overhead, and the delay time to receive the information [3]. The problem of finding the minimum cost multicast route is well known in graph theory as Steiner tree problem, proved to be a member of the NP-complete class. However, Steiner tree is the tree that has minimum total cost on the links of the tree without considering the constraints on the link i.e. QoS parameters. In order to find the multicast route that can guarantee QoS, we have to find a tree from the source node to the destination nodes, such that the total cost on the links in the tree is minimum and QoS on the path starting at the source to each destination is satisfied. This problem is named Constrained Steiner Tree (CST) problem and was proved to be NP-complete too [5]. There are already some heuristic proposals to solve this problem [5] [11]. Though the heuristics can find near optimal constrained multicast route within polynomial time, the search time depend on the number of nodes in the network. This is not so efficient in case of large networks.

Neural network is another approach for solving optimization problems. With the massive parallel computation power of neural network, it can find near optimal solution very fast, when implemented with hardware. Moreover, the computation time does not depend on the complexity of the problem, but only on the neural feedback interconnectivity [2]. Hopfield neural network model is used to solve many combinatorial optimization problems such as TSP, real-time scheduling, unicast routing problem [6] [2] [9]. In this paper we propose a modified version of Hopfield neural networks to solve the QoS constrained real-time multicast routing problem which is an extension from the model in [6]. Here we consider only one QoS constraint, the delay bound, which is the most important QoS parameter for most of the multimedia applications. The experimental results show that it can find near optimal delay constrained multicast route and its performance is comparable to other existing heuristics.

The organization of the paper is as follows. In section 2, we discuss the assumptions on the network topology used in this work and then the formal definition of QoS constrained real-time multicast routing problem. In section 3, we describe an overview of Hopfield neural network and its working principle. The proposed modified version of Hopfield neural network model model for solving delay constrained multicast routing is presented in section 4. In section 5, important considerations for simulation and results are reported and discussed. Concluding remarks are in section 6.

2 QoS Constrained Multicast Routing Problem

The objective of QoS constrained multicast routing is the same as the Steiner tree problem, with the imposition of QoS constraints for each destination. In the next two subsections, we describe the assumptions on the network that we will work with and give the formal definition of the QoS constrained multicast routing problem.

2.1 Assumptions

We assume that the network can be modeled as a directed graph and there is no multi-edge connection between two nodes. The vertices and edges in the graph represent nodes and links of the network. We suppose that the source node has the responsibility to establish QoS constrained multicast route which is a connection oriented channel to the specific destinations. Each link in the network has two characteristics, cost and QoS. It is possible that there are more than one QoS parameters associated with each link as,

$$\text{link characteristic} = (cost, QoS_1, QoS_2, ..., QoS_n) \quad (1)$$

where $QoS_1, QoS_2, ..., QoS_n$ are the different Quality of Service parameters. As in the other related works [5] [11], the cost value on the link may be used to represent the utilization of resources on the link. The lower the cost, the more resources are available. For delay constrained multicast routing problem, delay value on the link can be used to represent the sum of the switching and propagation delays on that link. Also the Cost and QoS (delay) are known to source node before it starts finding the multicast route. Finally, the obvious assumption is that the QoS on the path from the source to each destination will be maintained during the session, due to the resource reservation protocol.

2.2 Problem Definition

Consider a graph $G = (V, E)$, where V represents set of vertices or nodes, and E represents set of edges or links of the network. Each link has its properties, cost (C) and delay (L), where

$$C: E \rightarrow R^+, \textit{ a link cost function}$$

and

$$L: E \rightarrow R^+, \textit{ a link delay function}$$

where R^+ is any real positive value. A source node $s \in V$ and a set of destination nodes D, $D \subseteq V$, are given. The problem is to find a tree rooted at the source s and spanning to all the members of D such that (1) the total cost on the links of the tree is minimum, (2) the delay from source to each destination is not greater than the required delay constraint (Δ).

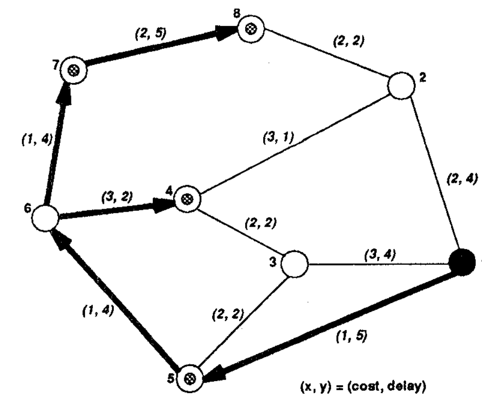

Figure 1: The optimal delay constrained multicast route when delay constraint (Δ) is 20

As mentioned earlier, in this paper, we consider only one QoS parameter, end-to-end delay bound. This is the most critical QoS requirement for real-time interactive multimedia applications. It is more complex to find the solution when there are more than one QoS constraints requested from the applications, such as delay, delay variance and bandwidth. It is even harder to solve when each destination has different QoS requirements [7]. However, by simple modifications of the proposed neural network model it should be possible to find the multi-QoS constrained multicast route, even when they are different for different destinations.

Fig. 1 shows an example of the network topology that satisfy the above assumptions (for simplicity we assume that every link has the same link characteristic in both directions). The link characteristic is represented as a pair *(cost, delay)*. A multicast tree, which is shown by thick arrow lines in the Fig.1, is an optimal delay constrained multicast route that satisfies the two required conditions. Here the tree is rooted from the source node 1 (solid circle) to the destination nodes (4, 5, 7, 8) (shaded circle) with delay constraint (Δ) 20. Total cost and maximum delay of the optimal tree are 8 and 18 respectively.

2.3 Related Works

There are two basic approaches to solve this problem, heuristic and neural network. The delay constrained multicast routing problem was first proposed by Kompella [5]. Many heuristic proposals based on Kou-Markowsky-Berman algorithm to approximate the optimal solution are also proposed in [5]. However, the definition of the delay function is not the same as defined in this work. Kompella used only positive integer values for the delay function where as we could use positive real numbers. The computation time of $CMCT_C$, the most powerful heuristic

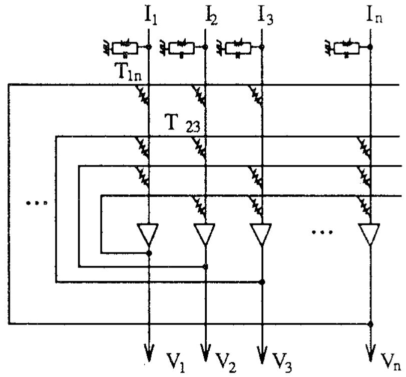

Figure 2: A circuit of Hopfield neural network model

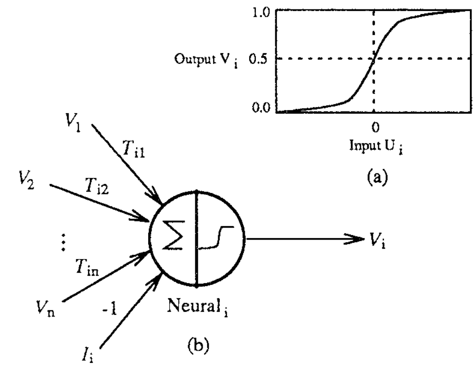

Figure 3: (a) A Hopfield neural. (b) The sigmoid transfer function.

in [5], is $O(n^3\Delta)$, where n is the number of nodes in the network. However, because Δ must be presented with $O(log\Delta)$ bits, the solution is exponential. Thus allowing real values for the delay (Δ) would increase the computation time enormously. [11] shows that even by reducing the granularity of Δ to a small value, the solution still is not practical. Widyono [11] also proposed many heuristics to this problem. One of his most efficient heuristic is a Constrained Adaptive Ordering heuristic (CAO). These two heuristics ($CMCT_C$ and CAO) will be used to compare the efficiency of the proposed neural network model.

Other related works are by using the neural network approach. The first proposal to the routing problem in communication network using neural network could be found in [8] where many restrictions were being imposed. In last few years, there are many proposals to improve the solution such as the works reported in [2] [6]. But all the existing works using neural networks are for *unconstrained unicast* routing problem, whereas we propose our model to solve the *delay constrained multicast* routing problem.

3 Hopfield Neural Network Overview

Hopfield neural network model is a recurrent network of elementary processing units. The energy of the ensemble converges to its minimum when relaxed. The basic Hopfield neural network model is shown in Fig. 2. Each neuron (shown by a triangle) in the network represents a non-linear device (op-amp) which has a monotonic increasing transfer function between its input U_i and the output V_i. The most popular function is the sigmoid transfer function as shown in Fig. 3(a).

$$V_i = g_i(U_i) = \frac{1}{1+e^{-\lambda_i \cdot U_i}} \tag{2}$$

The outputs of the neurons will be fed back to other neurons through the connection weight matrix $T = [T_{ij}]$, which is represented by the conductances. The input of the neuron (U_i) is the sum of all the feedbacks from other neurons and the external input current I_i (also known as bias current, see Fig. 3(b)), which usually represent the actual data provided by the user to the neural network [6]. Fig. 3(b) shows an example of a typical neuron that has two functions inside, summing all the inputs and transforming it by sigmoid function. The dynamics of the i^{th} neuron is

$$\frac{dU_i}{dt} = -\frac{U_i}{\tau} + \sum_{j=1}^{N} T_{ij}V_j + I_i \tag{3}$$

where τ is the time constant of the i^{th} unit's circuit, determined by the value of the resistance and capacitance by which the input is grounded. If the gains of the amplifiers are sufficiently high ($\lambda \to \infty$), then (3) follows a gradient descent of the quadratic energy function below

$$E = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} T_{ij}V_iV_j - \sum_{i=1}^{n} I_iV_i \tag{4}$$

From (3) and (4), the dynamics of i^{th} neuron could again be defined as

$$\frac{dU_i}{dt} = -\frac{U_i}{\tau} - \frac{\partial E}{\partial V_i} \tag{5}$$

4 Hopfield Neural Network for Delay Constrained Multicast Routing

In [6], we find the model for solving unconstrained unicast routing problem. In this work, the concept of finding continuous path from source to destination is adopted from [6]. We introduce several other modifications to make the model suitable for solving delay constrained multicast routing problem. To simplify the explanations the modifications are introduced in two steps.

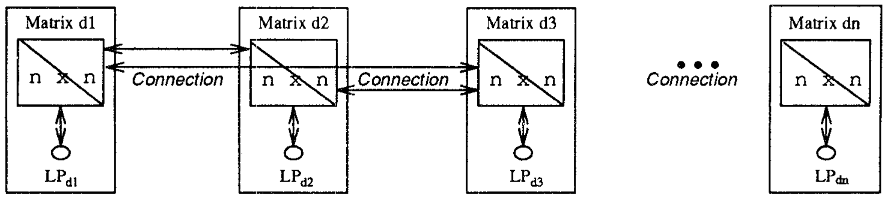

Figure 4: The Hopfield neural network model for delay constrained multicast routing

Step 1 Modification of **unconstrained** unicast routing Hopfield neural network to **delay constrained** unicast routing Hopfield neural network.

Step 2 Modification of delay constrained **unicast** Hopfield neural network to delay constrained **multicast** Hopfield neural network model.

Step 1 is realized by adding the delay constraint term to the energy function of the unconstrained unicast routing model. Extending it to delay constrained multicast route is much more tricky. It is done by combining each of the delay constrained unicast routes for individual destinations together. However, the cost of multicast route found in this way, may be much higher than the optimum, if there is no cooperation among the different unicast routes to find the cheapest multicast route. In step 2, we modified the energy function of delay constrained unicast routing Hopfield neural network, so that it will try to minimize the cost of multicast route, and not each unicast route independently. Thus the route selected by one destination will influence the routes of all other destinations. Finally we compose them together to form the delay constrained multicast route as a whole. A sketch of the proposed model is shown in Fig. 4. We use n matrices of n x n neurons where n is the number of nodes in the network. The q^{th} matrix is used to find the delay constrained unicast route for destination q. Fig. 4. shows the model with n matrices, and only k matrices out of all n matrices are actually used, where k is the number of destinations. Thus the matrices which are the member of set D are used. The diagonal line on each matrix depicts that the neurons on those position are ignored.

In each matrix i, there are connections from the neurons inside the matrix to a new separated neuron, called LP-type neuron [10] [9]. This neuron will take care of the specified delay constraint so that the delay of the unicast route in the matrix does not exceed the delay constraint. There are connections among the neurons that represent the same link in the network but are in different matrices. These connections were necessary due to defining a cost term in the energy function to take care of minimum cost for the whole multicast route, as explained before. Finally, we will have a delay constrained unicast route for individual destinations in the corresponding matrices, where the solution takes advantage of the links used for other destinations.

4.1 Problem Formulation

Each neuron in the matrix is described by double indices (x, i) , where x and i denote the row and column number respectively. The neuron at row x and column i represents the link from node x to node i in the network. The neurons at the diagonal are not used. Therefore $n(n-1)$ neurons are used for computation in each matrix. The meaning of the output voltage V_{xi} of each neuron, and other terms in the energy function are defined as follows.

$$V_{xi}^{m} = \begin{cases} 1 & \text{if the link from node } x \text{ to node } i \text{ is chosen in the delay constrained multicast tree to destination } m \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

$$P_{xi} = \begin{cases} 1 & \text{if the link from node } x \text{ to node } i \text{ dose not exist in the network} \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

$$C_{xi} = \text{finite real positive cost value on the link from node } x \text{ to node } i \quad (8)$$

$$L_{xi} = \text{finite real positive delay value on the link from node } x \text{ to node } i \quad (9)$$

4.2 Energy Function

From the above formulation, we define an energy function composed of cost term and constraint terms. The cost term is what we want to minimize. As mentioned earlier, we will minimize the total cost of the links in multicast tree. Many constraint terms are also needed to force neurons to go to only those stable states that would represent valid solutions only, by avoiding invalid solutions. By invalid solution we mean those routes which violate the delay constraint, or select non-existing links in the multicast tree. From the above statement, the energy function should be formalized so as to minimize the cost and consider all the following constraint terms.

Minimize:

1. The total cost of delay constrained multicast route

Constraints

2. At stable state, the output voltage of neurons are either 0 or 1.

3. There is only one unicast route from source to the destination in each matrix.

4. Prevent non-existing links in the network to be selected.

5. Delay from source to each destination must not be greater than the delay constraint.

Constraints 2, 3 and 4 are also needed in unconstrained unicast routing problem. So we can use the energy terms for those constraints as defined in [6].

Cost term

Here the cost term of energy function of each individual matrix corresponding to each destination (belonging to set D, denoted by superfix m) is defined as follows:

$$E_1^m = \sum_{x=1}^{n} \sum_{\substack{i=1 \\ i \neq x \\ (x,i) \neq (m,s)}}^{n} C_{xi} \cdot f_{xi}^m(V) \cdot V_{xi}^m \qquad (10)$$

$$f_{xi}^m(V) = \frac{1}{(1 + \sum_{\substack{j=1 \\ j \neq m, j \in D}}^{n} V_{xi}^j)} \qquad (11)$$

where $V = \{V_{xi}^k : k = 1, 2, ..., n, k \neq m, k \in D\}$. Eq. (10) is defined to minimize the total multicast cost through the function $f_{xi}^m(V)$. Outputs of the neurons from different matrices that represent the same link in the network try to cooperate together to minimize the cost of the whole multicast route through the connections represented by function $f_{xi}^m(V)$. The actual meaning is that, if a link is selected by other unicast routes, then the cost is reduced in proportion to the number of the unicast routes use that link. In other words, it implies that finally the cost of the link will be considered only once even though used by different matrices.

Constraint 2

$$E_2^m = \sum_{x=1}^{n} \sum_{\substack{i=1 \\ i \neq x}}^{n} V_{xi}^m (1 - V_{xi}^m) \qquad (12)$$

At stable state, when the energy goes to minimum, this energy term will force the output of neurons to a value 0 or 1.

Constraint 3

$$E_3^m = (1 - V_{ms}^m) + \sum_{x=1}^{n} \left\{ \sum_{\substack{i=1 \\ i \neq x}}^{n} V_{xi}^m - \sum_{\substack{i=1 \\ i \neq x}}^{n} V_{ix}^m \right\}^2 \qquad (13)$$

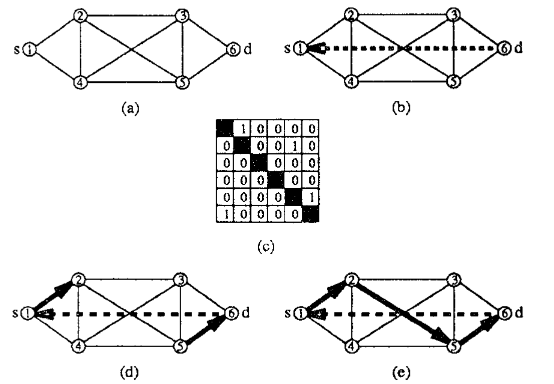

Figure 5: An Example shows how the two terms in E_3 construct a completed route. (a) The network. (b), (d), (e) The completed route construction steps. (c) The neural output representation.

The above equation is to make sure that each matrix will find a continuous route i.e. a completed route from source to a specific destination. The first term of E_3^m create a virtual link from destination m to source s. The second term of E_3^m will force the total voltage of the neurons that represent the incoming links equal to the total voltage of neurons that represent the outgoing links at any specific node, to minimize the energy. When the output voltage of the neuron goes to one or zero by E_2^m, the second term of E_3^m ensures that for every network node,

number of incoming links = number of outgoing links

Fig. 5 shows an example how these two terms force the Hopfield network to find a unicast route from source node 1 to destination node 6. The topology of the network is shown in Fig. 5(a). First, due to the first term $(1 - V_{ms}^m)$ of (13), the virtual link from node 6 to node 1 will be initiated (see Fig. 5(b)). Then the second term in E_3^m will initiate an outgoing link from node 1 and an incoming link to node 6. This will be continued till a completed route or closed loop route from source to the destination (see Fig. 5(d), (e)) is formed. The output values of the neurons in the matrix corresponding to the route in Fig. 5(e) is shown in Fig. 5(c).

Constraint 4

$$E_4^m = \sum_{x=1}^{n} \sum_{\substack{i=1 \\ i \neq x \\ (x,i) \neq (m,s)}}^{n} P_{xi} V_{xi}^m \qquad (14)$$

From (7), the definition of P_{xi}, this term will penalize or inhibit the neurons that represent non-existing links of the network. The virtual link, which is important to make a completed route, will not be inhibited.

Constraint 5

Constraint terms can be divided into equality constraint and inequality constraint. All the constraint terms explained till now are equality constraint terms. But the delay constraint is an in-equality constraint, because the delay of route from source to the destination must not be greater than the specified delay constraint.

$$\sum_{\substack{x=1 \\ (x,i)\neq(m,s)}}^{n} \sum_{\substack{i=1 \\ i\neq x}}^{n} L_{xi} V_{xi}^{m} \leq \Delta \quad , V_{xi}^{m} \in \{0,1\} \tag{15}$$

Original Hopfield network was first proposed with equality constraint only. There are two methods to deal with the in-equality constraint. One method described in [1], is to modify in-equality constraint to equality constraint. Another method is to use a kind of LP type neuron (LP stands for Linear Programming) [10] [4] [9]. This neuron will receive the delay of the current established route and the delay constraint value. Then it penalizes the neurons in the corresponding matrix when the delay of the route is greater than the delay constraint. $h(z)$ function below is the transfer function of LP-type neuron.

$$h(z) = \begin{cases} 0 & \text{if } z \leq 0 \\ z & \text{otherwise} \end{cases} \tag{16}$$

where

$$z = \sum_{\substack{x=1 \\ (x,i)\neq(m,s)}}^{n} \sum_{\substack{i=1 \\ i\neq x}}^{n} L_{xi} V_{xi}^{m} - \Delta \tag{17}$$

Thus the LP neurons contribute positively only when the delay constraint is violated. The delay constraint term using LP-type neurons can be defined as follows.

$$E_{5,LP}^{m} = H(z) \tag{18}$$

where

$$H(z) = \int h(z)dz \tag{19}$$

In this paper, we use the LP-type neuron as a method to take care the delay constraint of the unicast route. Thus the total energy function for the delay constrained multicast routing (E) is the sum of energy functions of delay constrained unicast routing and can be defined as

$$E = \sum_{\substack{m=1 \\ m\in D}}^{n} E^{m} \tag{20}$$

where E^m is the energy function of matrix m, which is used to find constrained unicast route from source node s to destination m and is defined as

$$E^{m} = E_{1}^{m} + E_{2}^{m} + E_{3}^{m} + E_{4}^{m} + E_{5,LP}^{m} \tag{21}$$

$$\begin{aligned} E^{m} = & \frac{\mu_1}{2} \sum_{\substack{x=1 \\ (x,i)\neq(m,s)}}^{n} \sum_{\substack{i=1 \\ i\neq x}}^{n} C_{xi} \cdot f_{xi}^{m}(V) \cdot V_{xi}^{m} \\ & + \frac{\mu_2}{2} \sum_{x=1}^{n} \sum_{\substack{i=1 \\ i\neq x}}^{n} V_{xi}^{m}(1 - V_{xi}^{m}) \\ & + \frac{\mu_3}{2}(1 - V_{ms}^{m}) \\ & + \frac{\mu_4}{2} \sum_{x=1}^{n} \left\{ \sum_{\substack{i=1 \\ i\neq x}}^{n} V_{xi}^{m} - \sum_{\substack{i=1 \\ i\neq x}}^{n} V_{ix}^{m} \right\}^{2} \\ & + \frac{\mu_5}{2} \sum_{\substack{x=1 \\ (x,i)\neq(m,s)}}^{n} \sum_{\substack{i=1 \\ i\neq x}}^{n} P_{xi} V_{xi}^{m} \quad + \frac{\mu_6}{2} H(z) \end{aligned} \tag{22}$$

where $\mu_1, \mu_2, \mu_3, \mu_4, \mu_5$ and μ_6 are the coefficient values used to specify and control the significance of each term in the energy function.

4.3 Network Dynamics

From the dynamic equation (5), the evolution of the input voltage for U_{xi}^{m} is

$$\frac{dU_{xi}^{m}}{dt} = -\frac{U_{xi}^{m}}{\tau} - \frac{\partial E^{m}}{\partial V_{xi}^{m}} \tag{23}$$

By substituting (22) in (23), we will get the network dynamics or the differential equations of neuron voltages as follows

$$\begin{aligned} \frac{dU_{xi}^{m}}{dt} = & -\frac{U_{xi}^{m}}{\tau} \\ & -\frac{\mu_1}{2} C_{xi} \cdot f_{xi}^{m}(V) \cdot (1 - \delta_{xm} \cdot \delta_{is}) \\ & -\frac{\mu_2}{2}(1 - 2V_{xi}^{m}) + \frac{\mu_3}{2}\delta_{xm} \cdot \delta_{is} \\ & -\mu_4 \sum_{\substack{y=1 \\ y\neq x}}^{n} (V_{xy}^{m} - V_{yx}^{m}) + \mu_4 \sum_{\substack{y=1 \\ y\neq i}}^{n} (V_{iy}^{m} - V_{yi}^{m}) \\ & -\frac{\mu_5}{2} P_{xi}(1 - \delta_{xm} \cdot \delta_{is}) \\ & -\frac{\mu_6}{2} L_{xi} \cdot (1 - \delta_{xm} \cdot \delta_{is}) \cdot h(z) \\ & \forall (x,i) \in \overline{N} \times \overline{N}/x \neq i \end{aligned} \tag{24}$$

where

$$V_{xi}^{m} = g_{xi}^{m}(U_{xi}^{m}) = \frac{1}{1 + e^{-\lambda_{xi}^{m} \cdot U_{xi}^{m}}} \tag{25}$$

$$\delta_{ab} = \begin{cases} 1 & \text{if } a = b \\ 0 & \text{otherwise} \end{cases} \tag{26}$$

5 Simulation and Results

5.1 Important considerations during simulation

The proposed Hopfield neural network model is simulated by using fourth order Runge-Kutta method with time step equal to 10^{-5}. Without any loss of generality, the time constant τ for each neuron is set to 1. For simplicity it is assumed that $\lambda_{xi}^m = 1$ and $g_{xi}^m = g$. In order to avoid the undesirable condition due to the symmetry of initial neuron state and network topology, small value of noise $-0.0001 \leq \delta U_{xi}^m \leq +0.0001$ is set as the initial value of each neuron. The simulation stops when the change of each neuron output from its previous iteration is not more than the threshold value of $\Delta V_{th} = 5 \times 10^{-5}$. At the stable state the neurons with $V_{xi}^m \geq 0.5$ are chosen to construct a path from source s to destination m. Though our algorithm works with real numbers as the cost and delay values, in this simulation we used random integers, so that we can compare our simulation results with the heuristic approaches, which could not practically work with real values for the delay. Also we assumed $C_{ab} = C_{ba}$ and $L_{ab} = L_{ba}$. The cost function C is a random integer function that will give the cost of the links in the range from 1 to 3. The delay function L is also a random integer function in the range from 1 to 5.

Selection of the coefficient values (μ's) is one of the hardest task dealing with Hopfield neural networks. Examples of generation of bad path due to wrong coefficient choice is shown in Fig. 6. Fig. 6(a) is an example that the Hopfield network could not construct a completed route. Fig. 6(b) shows a completed route which has a non-existing link. Fig. 6(c), (d) show solutions from Hopfield model that construct more than one completed route. We analyze the reasons to draw some guidelines to choose the proper range of coefficient values. We propose an adaptive method to change the coefficient values during updating, when the delay constraint is not satisfied. In the simulation, each neuron has different values of μ_1 and the values are changed during iteration, where $\mu_{1,xi}^{m,j}$ is μ_1 for neuron in matrix m at row x and column i during iteration j. μ_3 and μ_5 were chosen with the same priority. They were of very high values to prevent infeasible solutions due to the non-existing links and to speed up the construction of the valid route [6]. The value of μ_1 is selected not more than μ_4/C_{max}. The coefficient values are first set so that it will try to find the minimum cost CST without considering delay constraint. This means setting μ_2 and μ_6 to a low value. However, the value of μ_1 should not be set to a very high value. Else the relaxation will occur so fast that there is not enough time to have cooperation among the matrices before reaching the stable state. By these reasons, the coefficient values at the initial iteration are chosen as follows:

$$\mu_{1,xi}^{m,0} = 1000; \quad \mu_2 = 100; \quad \mu_3 = 50000;$$

$$\mu_4 = 15000; \quad \mu_5 = 50000; \quad \mu_6 = 2000;$$

During the relaxation, if the delay constraint condition, (27) below, is not satisfied then (29) is used to increase the cost coefficient of the link responsible for that invalid path. By the dynamics of the neurons, they will try to construct a new route so that delay constraint is satisfied.

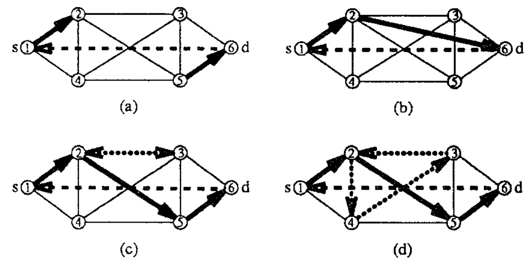

Figure 6: Examples showing the bad routes. (a) Non-completed route. (b) A completed route with non-existing link. (c), (d) 2 completed routes.

$$\sum_{x=1}^{n} \sum_{\substack{i=1 \\ i \neq x \\ (x,i) \neq (m,s)}}^{n} L_{xi} sgn(V_{xi}^m) \leq \Delta \tag{27}$$

where

$$sgn(z) = \begin{cases} 1 & \text{if } z \geq 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{28}$$

$$\forall (x,i) \in \overline{N} \times \overline{N}/x \neq i, V_{x,i}^{m,j} \geq 0.5,$$
$$\mu_{1,xi}^{m,j+1} = \mu_{1,xi}^{m,j} \cdot (1 + \tfrac{L_{xi}}{1000}) \tag{29}$$

5.2 Simulation Results

We compare the cost of the tree computed by using proposed modified Hopfield neural network model (NN) with other existing heuristics and optimal solution (Opt). The existing heuristics considered here are Independent heuristic (Ind), heuristic proposed by Kompella [5] denoted by $CMCT_C$ and CAO heuristic in [11]. The Independent heuristic is the simplest heuristic that construct CST just by adding independent constrained shortest paths for each destination. The comparison is shown in Table. 1 with different random graphs, all having 8 nodes with 4 destinations. Topology 1 is the same as shown in Fig. 1. Other topologies are created by generating random links with random cost and delay. The proposed model can find optimal solutions when Δ is 20 as well as Δ is 15. Fig. 7 shows the optimal CST when delay constraint is 15 and the cost is 10. We can see that proposed modified Hopfield model can find near optimal solution and the performance is also comparable to best known heuristics.

6 Conclusion

In this paper, we have proposed a modified version of Hopfield neural network model for delay constrained multicast routing which is necessary for multimedia applications. The problem is to find the least

Network	Δ	Ind	$CMCT_C$	CAO	NN	Opt
Topology (1)	15	10	11	10	10	10
	20	10	9	8	8	8
Topology (2)	10	15	12	12	9	9
	15	15	10	10	9	9
	20	15	8	8	8	8
Topology (3)	10	9	9	9	9	9
	15	9	8	8	9	8
	20	9	8	8	9	8
Topology (4)	10	12	10	10	10	10
	15	12	9	9	9	9
	20	12	9	9	9	9
Topology (5)	10	9	9	9	9	7
	15	9	6	6	7	6
	20	9	6	6	7	6

Table 1: Cost comparison on different 8 nodes network topologies among the existing heuristics, optimal solution and neural network solution

cost tree rooted at source node and span to the specific destinations, with delay from source to each of those destinations not exceeding some specified delay. The proposed model is the integration of delay constrained unicast routing from source to the different destinations, while each of these unicast route try to find minimum cost multicast route so that the delay constraint is satisfied at the same time. We also proposed an adaptive way to change the cost coefficient so that the delay constraint is not violated. The simulation results show that it can find near optimal route in randomly connected 8-nodes networks and the performance is comparable to the existing heuristics.

Acknowledgement

The authors would like to thank Dr. Ruck Thawonmas and Aung Aung Thein for their comments and help in computer simulation.

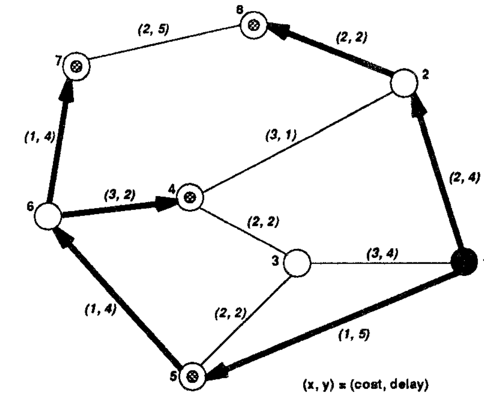

Figure 7: The optimal delay constrained multicast route when delay constraint (Δ) is 15

References

[1] S. Abe and J. Kawakami. Theories on the hopfield neural networks with inequality constraints. In *Proc. Int. Joint Conf. Neural Network*, pages I349–I352, Sandiego, 1989.

[2] S. Cavalieri. Optimal path determination in a graph by hopfield neural network. *Neural Networks*, 7(2):397–404, 1994.

[3] S. Deering and D. Cheriton. Multicast routing in internetworks and extended lans. *ACM Trans on Comp. Sys.*, pages 85–110, May 1990.

[4] M. P. Kennedy and L. O. Chua. Unify the tank and hopfield linear programming circuit and the canonial nonlinear programming circuit of chua and lin. *IEEE Trans on Circuits and Systems*, 34(2):210–214, 1987.

[5] V. Kompella. *Multicast Routing Algorithms for Multimedia Traffic.* PhD thesis, University of California, Sandiego, 1993.

[6] M. K. Mehmet and F. Kamoun. Neural networks for shortest path computation and routing in computer networks. *IEEE Trans on Neural Network*, 4(6):941–954, 1993.

[7] C. Partridge. *Gigabit Networking.* Addison-Wesley, 1993.

[8] H. E. Rauch and T. Winarske. Neural network for routing communication traffic. *IEEE Trans on Cont. Syst. Mag*, pages 26–30, April 1988.

[9] T. Ruck. *A Study on Application of Neural Networks for Real-time Systems.* PhD thesis, Tohoku University, Japan, 1993.

[10] D. W. Tank and J. Hopfield. Simple neural optimization networks: An a/d converter, signal decision circuit, and a linear programming circuit. *IEEE Trans on Circuits and Systems*, 33(5):533–541, 1986.

[11] R. Widyono. The design and evaluation of routing algorithms for real-time channels. Technical Report TR-94-024, International Computer Science Institute, Berkeley, June 1994.

[12] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. Rsvp: A new resource reservation protocol. *IEEE Network Magazine*, September 1993.