Design and Evaluation of Routing Concepts for QoS in Mobile Ad hoc Networks by using Hopfield neural network

Hind Ja'afar Aghati Hussain
Al-Mansour University Collage

Abstract:

Ad hoc networking is a concept in computer communications, which means that users wanting to communicate with each other forming a temporary network, without any form of centralized administration. Each node participating in the network acts both as host and as a router and must therefore is willing to forward packets for other nodes. For this purpose, a routing algorithm is needed. The most important characteristic in ad hoc network is the dynamic topology, which is a consequence of node mobility. Since direct communication is allowed only between adjacent nodes, distant nodes communicate over multihops.

Quality of Service (QoS) routing in an ad hoc network is difficult because the network topology may change constantly and the available state information for routing is inherently imprecise. An algorithm LARHNN (Lagrange relaxation method based on Hop field neural network) is proposed to solve QoS routing in ad hoc networks, the algorithm depends on Lagrange relaxation method to solve the QoS routing problem Delay Constrained Least Cost (DCLC) which is considered as NP-complete, the proposed algorithm selects a network path with sufficient resources to satisfy Delay Constrained Least Cost path in unicast routing and constrained minimal tree in multicast routing. The proposed algorithm uses the Hopfield neural network model of artificial neural networks to find the needed shortest path in execution of the proposed algorithm. And Hopfield neural network is selected from the other shortest path algorithms since it has good facilities to work in real time and adapt topology changes, Symmetrical connection, distributed asynchronous control, content addressable memory, in addition to its computation time, which depend on the speed of used hardware.

1- Introduction

The main function of the network layer is routing packets from the source machine to the destination machine. In most subnets, packets will require multiple hops to make the journey. The routing algorithm is that part of the network layer software responsible for deciding which output line an incoming packet should be transmitted on. Routing algorithms can be grouped into two major classes: non adaptive and adaptive [1]. Non-adaptive algorithms do not base their routing decisions on measurements or estimates of the current traffic and topology. Instead, the choice of the route to use to get from *I* to *J* (for all *I* and *J)* is computed in advance, off-line, and downloaded to the routers when the network is booted. This procedure is sometimes called static routing. Examples of such algorithms are flooding algorithm and random walk algorithm [2].

Adaptive algorithms, in contrast, change their routing decisions to reflect changes in the topology, and usually the traffic as well. Adaptive algorithms differ in where they get their information (e.g., locally, from adjacent routers, or from all routers), when they change the routes, and what metric is used for optimization (e.g., distance, number of hops, or estimated transit time). Good examples of adaptive routing are Distance Vector and Link State routing [2].

The routing algorithms can be divided further into the three types Isolated each router makes its routing decisions using only the local information it has on hand.

Centralized a centralized node makes all routing decisions. Specifically, the centralized node has access to global information. Distributed algorithms that use a combination of local and global information [3].

2 - Routing in Wireless Ad hoc Networks

An ad hoc network is a multi hop wireless network without fixed infrastructure or central supervision. It's very clear that ad hoc routing protocols are far different from those for wired networks, though some of the ad hoc protocols are derived from wired networks. In wireless networks, routing algorithms

should not only calculate the shortest path from the source to the destination, it should be combined with QoS (maximum delay, load balance) and energy conservation. Multi path routing is also an interesting technique to improve the performance in ad hoc networks with high mobility [3].

As to the wireless ad hoc network, the traditional routing algorithms does not perform well because there is no fixed infrastructure in ad hoc and the network has high mobility and fast changing topology. The transmission range of each node in wireless communication is limited, and each node must take responsibility to relay other nodes' data. Because of the mobility of the whole network, a route may be stale in a very short time, so the chain of intermediate nodes for a special session has to been modified frequently [2, 1].

Ad hoc routing algorithms can be classified into two types: table-driven and on-demand. In table-driven routing, each node sustains a routing table including its routing information to every other node. on-demand routing algorithm does not have routing information for every other node, The advantage is obvious: small overheads and low usage of capacity. The disadvantage is that on-demand routing need to use some time to find the special routes for an assigned destination, so it cannot respond to a data transmission requirement right away as in table-driven routing [1].

3- Quality of Service Routing

3.1 Basic Concepts [4]

A network is modeled as a graph (V, E). Vertices (V) of the graph represent switches, routers and hosts. Edges (E) represent communication links. The edges can be symmetric or asymmetric. A symmetric link has the same properties (capacity, propagation delay) and the same traffic volume in both directions. Each link is associated with some state. These states are used to specify the QoS constraint and are called metrics. These metrics can be broadly classified in three groups:

***Additive metric*: If m (s_1, s2) is the value of metric for a link s_1-s_2 and if the path p consists of links (s_1-s_2, s_2-s_3... s_{n-1}-s_n) then in case of additive metrics the**

value for whole of the path is actually the sum of all the values along the path. That is

$m(p) = m(s_1,s_2) + m(s_2,s_3)\ldots\ldots+m(s_{n-1}, s_n)$……(1)

Examples are hop count, delay, delay-jitter, and cost.

***Multiplicative metric*: The value for the path is product of all the values of the links along it.**

$m(p) = m(s_1,s_2)*m(s_2,s_3)\ldots*m(s_{n-1}, s_n)$……..(2)

Examples are reliability expressed in terms of probability between 0 and 1.

***Concave metric*: Value for path is actually the value of single bottleneck link along the path.**

$m(p) = \min[m(s_1,s_2),m(s_2,s_3)\ldots\ldots m(s_{n-1}, s_n)]$……..(3)

Examples are bandwidth, buffer space, and CPU time.

3.2 Maintaining State

The states that each node keeps are [5]:
***Local State :* Information about the outgoing / incoming link bandwidth, delay, buffer space, etc.**
***Global State*: Local state of the entire nodes in the network is collected either by link state protocols or distance vector protocols.**

3.3 Requirement Specification

The routing problems can be divided into two major classes: *unicast routing and multicast routing*. The unicast routing problem is defined as follows: given a source node *s*, a destination node *d*, a set of QoS constraints *C* and possibly an optimization goal, find the best feasible path from *s* to *d*, which satisfies *C*. The multicast routing problem is defined as follows: given a source node *s*, a set *R* of destination nodes, a set of constraints *C* and possibly an optimization

goal, find the best feasible tree covering s and all nodes in *R*, which satisfies constraints *C*.

In Unicast routing: two types of problems are defined. One is called link-optimization routing where the metric value for the path is optimized. The other problem is called link-constrained routing where a bound is given on the metric [5].

4- QoS Routing in Ad hoc Networks

The QoS routing algorithms for wire line networks cannot be applied directly to ad hoc networks. *First*, the performance of most wire line routing algorithms relies on the availability of precise state information. However, the dynamic nature of an ad hoc network makes the available state information inherently imprecise. Though some recent algorithms [9] [10] were proposed to work with imprecise information (e.g., the probability distribution of link delay), they require the precise information about the network topology, which is not available in an ad hoc network. *Second,* nodes may join, leave, and rejoin an ad hoc network at any time and any location; existing links may disappear and new links may be formed as the nodes move.

5- Problem Formulation [6, 7]

To formulate the shortest path problem in terms of the Hopfield neural network model, a suitable representation scheme must be found so that the shortest path can be decoded from the final stable state of the neural network.

$$V_{xi} = \begin{cases} 1 & \left(\textit{if the arc} (x,i) \textit{ is part of the shortest path} \right). \end{cases} \quad \textbf{((4)}$$

Also define r_{xi} **as:**

$$r_{xi} = \begin{cases} \\ \\ \end{cases} \quad 0 \quad \left(\textit{ if the arc } (x,i) \textit{ exist } \right). \qquad \textbf{(5)}$$

In addition the cost of an arc from node *x* to node *i* will be denoted by *Cxi*, a finite real positive number. For non-existing arcs this cost will be assumed to be zero:

$$\mathbf{C}_{xi} = \begin{cases} \\ \\ \end{cases} \quad \text{Positive real number } \left(\textit{ if the arc} (x,i) \textit{ exist } \right). \qquad \textbf{(6)}$$

6- The Energy Function and the Motion Equation [8]

In order to solve the shortest path problem, using the Hop field model, first an energy function must be defined whose minimization process drives the neural network into lowest energy state.

$$E = \frac{m\,1}{2} \sum_{\substack{x=1 \\ (x,i)\neq(d,s)}}^{n} \sum_{\substack{i=1 \\ i\neq x}}^{n} C_{xi} \cdot V_{xi} + \frac{m\,2}{2} \sum_{\substack{x=1 \\ (x,i)\neq(d,s)}}^{n} \sum_{\substack{i=1 \\ i\neq x}}^{n} r_{xi} \cdot V_{xi} + \frac{m\,3}{2} \sum_{x=1}^{n} \left\{ \sum_{\substack{i=1 \\ i\neq x}}^{n} V_{xi} - \sum_{\substack{i=1 \\ i\neq x}}^{n} V_{ix} \right\}^2$$

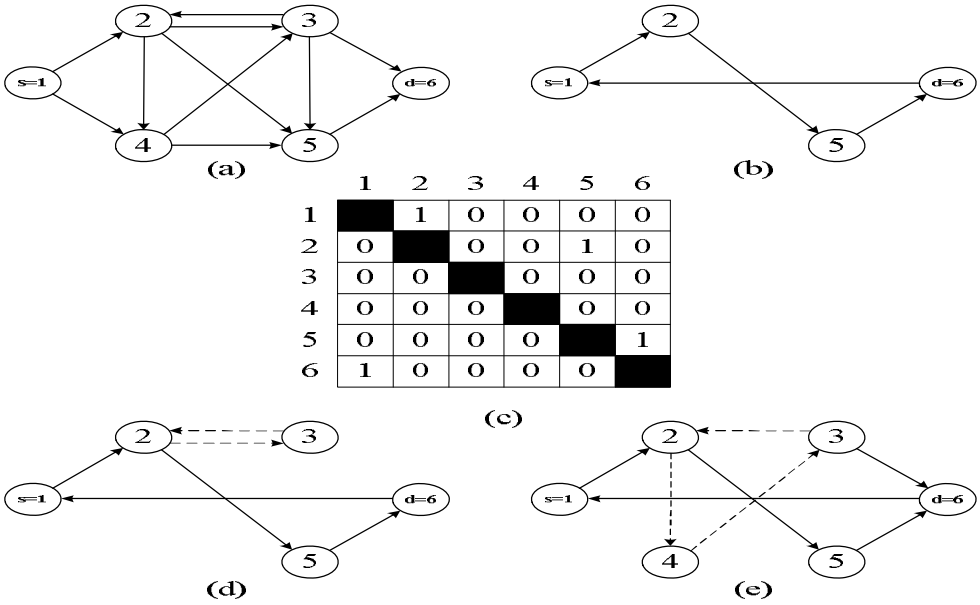
$$+ \frac{m\,4}{2} \sum_{i=1}^{n} \sum_{\substack{x=1 \\ i\neq x}}^{n} V_{xi} \cdot (1 - V_{xi}) + \frac{m\,5}{2} (1 - V_{ds}) \quad \text{.....} \quad \boxed{7}$$

In equation (7):

- **The term including $m\,1$ minimizes the total cost of a path by taking into account the cost of existing links.**
- **The term including $m\,2$ prevents nonexistent links from being included in the chosen path.**

- **The term including $m3$ is zero if for every node in the solution; the number of incoming arcs equals the number of outgoing arcs. This makes sure that if a node has been entered it will also be exited by a path.**
- **The term including $m4$ pushes the state of the neural network in the range defined by $V_{xi} \in \{0,1\}$.**

The term including $m5$ is zero when the output of the neuron at location (*d, s*) settles to *1*.



	1	2	3	4	5	6
1		1	0	0	0	0
2	0		0	0	1	0
3	0	0		0	0	0
4	0	0	0		0	0
5	0	0	0	0		1
6	1	0	0	0	0	

Figure (1): (a) The Original Network. (b) The Primary Loop Forming the SP Solution. (c) The Neural Output Representation. (d) A Secondary Loop (2, 3, 2). (e) Another Secondary Loop (2, 4, 3, 2).

7- The Connection Matrix and the Biases [9]

The most important task at hand is to find an appropriate connection weight matrix. It is constructed taking into account the requirement that non-valid paths should be prevented and valid paths should be preferred. A consideration in this regard is that, In this context, the *Kronecker delta*

function is used to facilitate a simple notation. The *Kronecker delta* function has two arguments, which are given usually as subscripts of the symbol δ. By definition, δ_{ab}, has value 1 if a = b, and 0 if a ≠ b. That is, the two subscripts should agree for the *Kronecker delta* to have value 1. Otherwise, its value is 0. Two subscripts are used to refer the neurons, one for the node it refers to, and the other for the order of that node in the path.

$$\frac{dU_{xi}}{dt} = -\frac{U_{xi}}{t} + \sum_{y=1}^{n}\sum_{\substack{j=1 \\ j\neq y}}^{n} W_{xi,yj} \cdot V_{yj} + I_{xi} \quad \text{.....(8)}$$

$$\frac{dU_{xi}}{dt} = -\frac{U_{xi}}{t} - \frac{\partial E}{\partial V_{xi}} \quad \text{..... (9)}$$

$$V_{xi} = g_{xi}(U_{xi}) = \frac{1}{1+e^{-l_{xi}\cdot U_{xi}}} \quad \text{..... (10)}$$

$$\forall (x,i) \in n \times n / x \neq i$$

By substituting (7) in (9), the equation of motion of the neural network is readily obtained [11]:

$$\frac{dU_{xi}}{dt} = -\frac{U_{xi}}{t} - \frac{m1}{2} \cdot C_{xi} \cdot (1 - d_{xd} \cdot d_{is}) - \frac{m2}{2} \cdot r_{xi} \cdot (1 - d_{xd} \cdot d_{is})$$

$$- m3 \cdot \sum_{\substack{y=1 \\ y\neq x}}^{n} (V_{xy} - V_{yx}) + m3 \cdot \sum_{\substack{y=1 \\ y\neq i}}^{n} (V_{iy} - V_{yi}) - \frac{m4}{2} \cdot (1 - 2 \cdot V_{xi})$$

$$+ \frac{m5}{2} \cdot d_{xd} \cdot d_{is} \quad \text{.....(11)}$$

$$\forall (x,i) \in n \times n / x \neq i$$

Where d is the Kronecker delta defined by:

If (a=b)

$$d_{ab} = \begin{cases} 1 \\ \\ 0 \end{cases} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (12)$$

By comparing the corresponding coefficients in (8) and (11), the connection strengths and the biases are derived :

$$W_{xi,yj} = m\,4 \cdot d_{xy} \cdot d_{ij} - m\,3 \cdot d_{xy} - m\,3 \cdot d_{ij} + m\,3 \cdot d_{jx} + m\,3 \cdot d_{iy} \ldots\ldots\ldots\ldots \quad (13)$$

$$I_{xi} = -\frac{m\,1}{2} \cdot C_{xi}(1 - d_{xd} \cdot d_{is}) - \frac{m\,2}{2} \cdot r_{xi}(1 - d_{xd} \cdot d_{is}) - \frac{m\,4}{2} + \frac{m\,5}{2} d_{xd} \cdot d_{is}$$

$$I_{xi} = \begin{cases} \dfrac{m_5}{2} - \dfrac{m_4}{2} & \textit{(If } (x,i) = (d,s)). \\ \\ \dfrac{-m_1}{2} C_{xi} - \dfrac{m_2}{2} r_{xi} - \dfrac{m_4}{2} \end{cases} \quad \ldots\ldots \quad (14)$$

$$\forall (x \neq i), \forall (y \neq i)$$

8- QoS Routing Scheme for Ad hoc Networks

The routing problem which studied is the *delay-constrained least-cost routing*; it is NP-complete. A path, which satisfies the delay constraint, is called a *feasible path.* The algorithm considers not only the QoS requirements but also the optimality of the routing path

1. **The proposed ad hoc network system model to suit with QoS routing problem.**
2. **The proposed LARHNN algorithm that based on HNN as shortest path solution and Lagrange Relaxation Aggregated Cost (LARAC) as cost function modifier to solve DCLC QoS routing problem in ad hoc network**

when a connection request arrives at source node.

3. **Using HNN in dynamic path maintenance to deal with the ad hoc network topology changes.**

8.1 QoS Routing Problem

Given a source node s, a destination node t, and a delay requirement Δdelay, the problem of delay-constrained routing is to find a feasible path P from *s* to *d* such that delay(p) ≤ Δdelay.

When there are multiple feasible paths, the one with the least cost is to be selected. Finding the *delay-constrained least-cost* (DCLC) path is an NP-complete problem since it concerns with two additive metrics delay and cost. The second part is to maintain the path when the network topology changes [8].

8.2 Imprecise State Model

The following end-to-end state information is required to be maintained at every node *i* for every possible destination *t*.

1) Delay: $D_i(t)$ keeps the minimum end-to-end delay from *i* to *t*, i.e., the delay of the least-delay path.

2) Bandwidth: B*i*(*t*) keeps the maximum end-to-end bandwidth from *i* to *t*, i.e., the bandwidth of the largest-bandwidth path.

3) Cost: C*i*(*t*) keeps the least end-to-end cost from *i* to *t*, i.e., the cost of the least-cost path.

9- The Proposed Simulation System

Figure (2) shows the flowchart of the proposed QoS routing scheme, some considerations will be took into account as follows:

1. **Using BEACONS can be done in practical world as part of network layer software and can be simulated in simulation system to be the method that**

2. **The "Wait for connection request" stage in the proposed scheme is to keep the routing algorithm software active while the computer is running and ready to deal with any application request for sending data to another node.**
3. **The Cost matrix, Delay matrix are constructed to map the link metric of the ad hoc network with $n \times n$ dimensions; n is the number of nodes of the network.**

4. **The user can enter just the upper triangle of Cost matrix, Delay matrix since the links are assumed to symmetric links. So the reading of links' costs, delays will be between such node and all the upward remaining nodes. Links' costs, delays of non-existing links are entered as zero.**

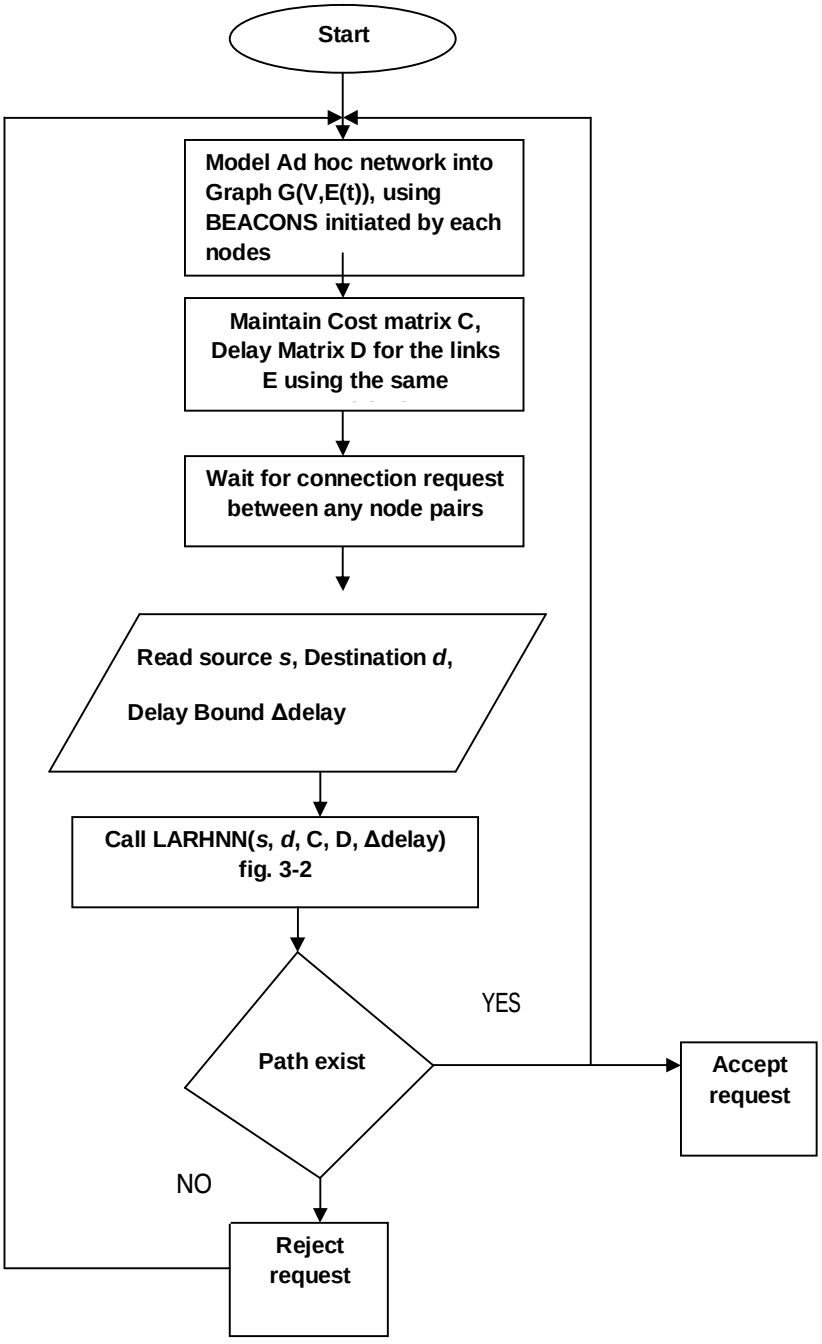

Start
Model Ad hoc network into Graph G(V,E(t)), using BEACONS initiated by each nodes
Maintain Cost matrix C, Delay Matrix D for the links E using the same
Wait for connection request between any node pairs
Read source s, Destination d, Delay Bound Δdelay
Call LARHNN(s, d, C, D, Δdelay) fig. 3-2
Path exist
YES
Accept request
NO
Reject request

Figure (2): A Proposed Scheme to adapt DCLC QoS Routing in Ad hoc Network

In next section LARHNN will be presented as a QoS routing algorithm to solve the DCLC QoS routing problem in ad hoc network.

10- The Proposed LARHNN Algorithm Overview [10]

In order to solve DCLC QoS routing problem in ad hoc network based on delay, cost metrics of the outgoing links at each node, and as described "LARAC" algorithm is one of techniques to deal with two metrics, the proposed "LARHNN" algorithm is presented below:

1. The LARHNN is called with the following parameters:
   ***s*: source node**
   ***d*: destination node**
   C: cost matrix that have the links' costs between all network nodes.
   D: delay matrix that have the links' delays between all network nodes.
   Δdelay: the delay requirement or bound that must be satisfied by the resulting path if there is such path.

2. It calculates shortest path on original cost metric with HNN algorithm as will be described later.

3. Otherwise the algorithm stores the path as the *best path* that does not satisfy ∆delay (this path is denoted by Pc in the following), and checks whether an appropriate solution exists or not: Calculates shortest path on delay using HNN. If the obtained path suits the delay requirement a proper solution exists, so the algorithm stores this path as the best appropriate path found till now (denoted by Pd). Otherwise there is no suitable path from *s* to *d* that can fulfill the delay requirement, so the algorithm stops.

The multiplier α is used to get aggregated cost metric based on the following

equation:

cα =d+ α .c ………………………………………….(15)

Since both d "delay" and c "cost" are additive metrics then cα also additive. It should be noted that is proposed based on the Lagrange relaxation technique to find the lower bound as described.

4. From the paths Pc and Pd that have been found in step 1, 2. Cost and delay of both paths are used to get α according to the formula that derived from equation and based on Lagrange relaxation described to get aggregated cost from both metrics (delay, cost):

$$a = \frac{d(p_c) - d(p_d)}{c(p_d) - c(p_c)} \text{ ………………………………..(16)}$$

By using the original formula to find α which is equal to $a = \frac{c(pc) - c(pd)}{d(pd) - d(pc)}$**, the solution will be not accurate since in most practical cases cost of Pc is less than cost of Pd, delay of Pc is greater than delay of Pd, this led to adapt the proposed formula in the proposed algorithm and this change does not violet the theory of the original algorithm, Figure (3) demonstrates the proposed algorithm which is called by the QoS routing scheme to solve DCLC QoS routing problem, also the LARHNN algorithm calls HNN to find shortest path when is needed inside the algorithm.**

Procedure LARHNN(*s*, *d*, c, d, Δ*delay*)

Pc := HNN(*s*, *d*, c)

if *d*(*Pc*) ≤ Δ*delay* then return *Pc*

Pd := HNN(*s*, *d*, d)

if *d*(*Pd*) > Δ*delay* then return "There is no solution"

repeat

$$a = \frac{d(p_c) - d(p_d)}{c(p_d) - c(p_c)}$$

r := HNN(*s*, *d*, *cα*)

if *cα*(*r*) = *cα*(*Pc*) then return *Pd*

else if *d*(*r*) ≤ Δ*delay* then *Pd* := *r*

Figure (3): The Proposed LARHNN to solve DCLC QoS Routing problem

10.1 Hopfield Neural Network Algorithm [11]

The shortest path between any two nodes based on local state information at each node which is submitted during the calling by LARHANN algorithm can be found using the following steps:

Step1: **Construct *n*.(*n-1*) neurons arranged in an (*nxn*) two dimensional array with diagonal elements eliminated, *n* is number on nodes of ad hoc network which can be entered by user during connection request.**

Step2: **For each entry (*x*, *i*) of the (*nxn*) array; (*x* ≠ *i*), map the cost matrix entered during calling HNN into that array. Let call this array by cost matrix C_{xi}. The cost of non-existing link is assumed to be equal to zero.**

Step3: **From the cost matrix, calculate the link matrix r_{xi} using equation (5).**

Step4: **Assuming that the sigmoid activation function ($g_{x,i}(U_{x,i})$) and amplifier gain ($l_{x,i}$), of the Hopfield neural network given by equation (10), are the same for all neurons.**

Step5: **The initial value of neurons output voltage is determined from equation (10). Also set the neural network parameters $m\,i's$, l, accuracy ($\Delta V_{threshokd}$), and step size (Δt) to their appropriate values.**

Step6: **To find the shortest path between a source node, *s,* and destination node *d* in the computer network under consideration, the dynamic of the neuron (11) must be solved iteratively until all neurons reach to their stable states.**

Equation (11) is a differential equation and can be solved using numerical methods.

The differentiation of neuron input voltage $U_{x,i}$ in equation (11) is with respect to time; therefore a function constructed depending on both time and $U_{x,i}$ from equation (11):

$$\frac{dU_{xi}}{dt}=F(t,U_{xi})=-\frac{U_{xi}}{t}-\frac{m\,1}{2}\cdot C_{xi}\cdot(1-d_{xd}\cdot d_{is})-\frac{m\,2}{2}\cdot r_{xi}\cdot(1-d_{xd}\cdot d_{is})$$

$$-m\,3\cdot\sum_{\substack{y=1\\ y\neq x}}^{n}(V_{xy}-V_{yx})+m\,3\cdot\sum_{\substack{y=1\\ y\neq i}}^{n}(V_{iy}-V_{yi})-\frac{m\,4}{2}\cdot(1-2\cdot V_{xi})+\frac{m\,5}{2}\cdot d_{xd}\cdot d_{\,is}$$

$$\forall(x,i)\in n\times n/\,x=i \quad \text{.....(17)}$$

$$t^{(v+1)}=t^{(v)}+\Delta t \quad \text{.....(18)}$$

$$U_{xi}^{(v+1)}=U_{xi}^{(v)}+\frac{1}{6}\left(k1+2\cdot k2+2\cdot k3+k4\right) \quad \text{.....(19)}$$

where (Δt) is the step size.

$$k1=\Delta t\cdot F\left(t^{(v)},U_{xi}^{(v)}\right) \quad \text{.....(20)}$$

$$k2=\Delta t\cdot F\left(t^{(v)}+\frac{\Delta t}{2},\ U_{xi}^{v}+\frac{k1}{2}\right) \quad \text{.....(21)}$$

$$k3=\Delta t\cdot F\left(t^{(v)}+\frac{\Delta t}{2},\ U_{xi}^{v}+\frac{k2}{2}\right) \quad \text{.....(22)}$$

$$k4=\Delta t\cdot F\left(t^{(v)}+\Delta t,\ U_{xi}^{v}+k3\right) \quad \text{.....(23)}$$

Step7: **After each iteration, if $\left|V_{xi}^{v}-V_{xi}^{v+1}\right|<accuracy$, for all neurons, then stop the iterations of solving equation (11), and go to step 8. Otherwise return to step 6.**

Step 8: **For all neurons, compare the final output voltage with a threshold level. Since the output voltage of the neuron is in the range [0, 1] therefore the threshold level is (0.5). If $V_{xi}^{final}\geq 0.5$ then *Vxi*=1; otherwise *Vxi*=0.**

Step9: **Now a two dimensional array obtained representing the neurons output voltage, with values of one or zero which corresponds to the mapping of existing links in the shortest path between *s* and *d*. Assume that the following neuron output voltage matrix obtained for six node subnet to find the shortest path between source node *a* and destination node *f*:**

10.2 Hopfield Neural Network Design Parameters [12, 13]

One major issue related to the efficiency of the Hop field model in solving combinatorial optimization problem is the lack of rigorous guidelines in selecting appropriate values of the energy function coefficients. also of minimum length found in, are:

$$2 \cdot m\,3 - m\,4 > 0 \quad \text{.....(24)}$$

$$m\,5 >> m\,1 \cdot (C_{xi})_{\max} \quad \text{..... (25)}$$

$$m\,2 = m\,5 \quad \text{..... (26)}$$

$$m\,1 < 2 \cdot \frac{m\,3}{(C_{xi})_{\max}} \quad \text{..... (27)}$$

Another parameter that must be selected is the *neural transfer parameter* l . Experimentally it has found that there is a compromise between choosing a small or a large for l . While a large l gives rise to a fast neural response for which the solution is not always the global minimum, a small l yields a slower response which can guarantee an optimum solution. In fact, it was observed that the ability of the SP neural algorithm to separate between an optimum and a very good solution is also enforced by the smoothness of the neural transfer function *g* (which corresponds to a low neural transfer parameter l). This is explained by the fact that a large l (such as *10* or *100*) may not allow enough time for a neuron to optimize its performance, as the neural response time becomes very fast and hence less accurate. Therefore, in order to allow the neurons' dynamics to wander freely in their state space, in the search for the global minima, a small value must be chosen for l .

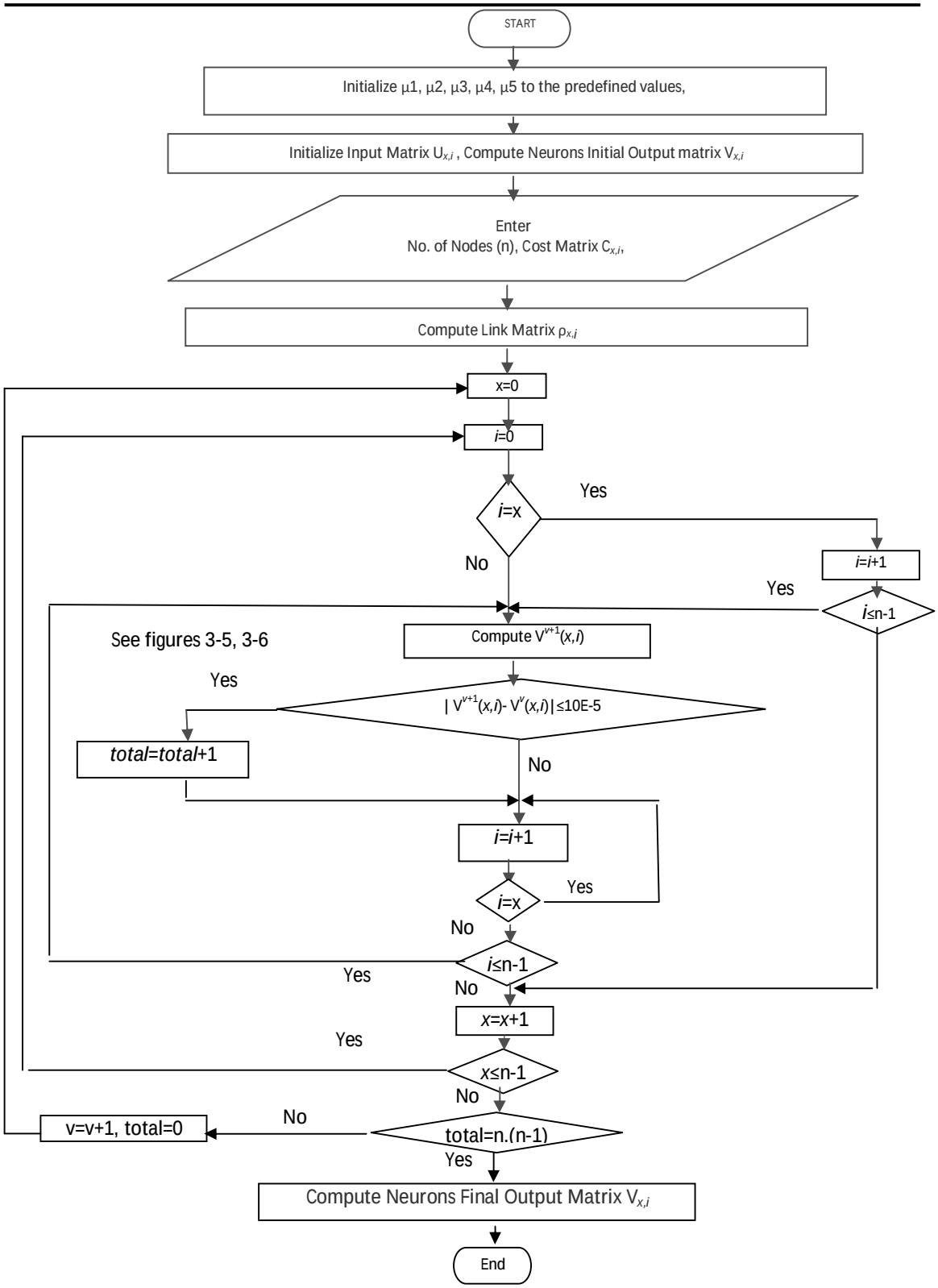


Figure (4): Hopfield Neural Network Shortest Path Computation Algorithm

No

To demonstrate the HNN algorithm, the following example is presented.

Example: Consider the five nodes weighted, symmetric computer network shown in figure (8). The shortest path between the source node 0 and destination node 4 can be found using the steps described early in this section.

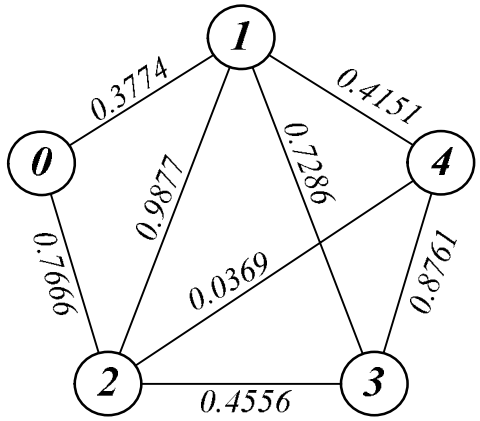


Figure (7): Five Nodes Network

Step1: **Number of nodes (*n*) is equal to (*5*), therefore the number of the neurons that must be arranged in (*5x5*) two dimensional array is equal to (*20=5(5-1)*).**

Step2: **From figure 7, the cost matrix can be found by placing the cost between any two nodes in its corresponding cell in the cost matrix.**

	0	*1*	*2*	*3*	*4*
0		.3774	.7666	0	0
1	.3774		.9877	.7286	.4151
2	.7666	.9877		.4556	.0369
3	0	.7286	.4556		.8761
4	0	.3774	.0369	.8761	

Step3: **Using equation (5), the link matrix can be computed from the cost matrix.**

	0	1	2	3	4
0		0	0	1	1
1	0		0	0	0
2	0	0		0	0
3	1	0	0		0
4	1	0	0	0	

Step4*:* Initializing the neurons input voltage matrix and calculating the neurons output voltage matrix using the sigmoid function described by equation (10). The values of the neural network parameters are:

l ***=1, accuracy=1E-5***, Δt ***=1E-5***,

$m1=1500$, $m2=5000$, $m3=4500$, $m4=1000$, $m5=5000$.

The values of $m\,i's$ **are calculated according to guidelines**

Step5: **By setting the source node (*s=0*) and destination node (*d=4*), the shortest path between them can be found by solving equation (11) iteratively until all neurons reach the stable states.**

Step6: **For each iteration (*v*), if and only if** $\left|V_{xi}^{v}-V_{xi}^{v+1}\right|<10^{-5}$ **for all neurons, then stop the iterations. If for all neurons the condition is not true return to step 5. The stable state is reached after *2012* iterations for the five-node network.**

Step7: **Now the final matrix of the neurons output voltage will be compared with (*0.5*) and assign (*1*) for each value greater than (*0.5*) and assign (*0*) otherwise.**

	0	1	2	3	4
0		1	0	0	0
1	0		0	0	1
2	0	0		0	0
3	0	0	0		0
4	1	0	0	0	

Step8: **Now the final solution matrix will be mapped to the five-node network.**

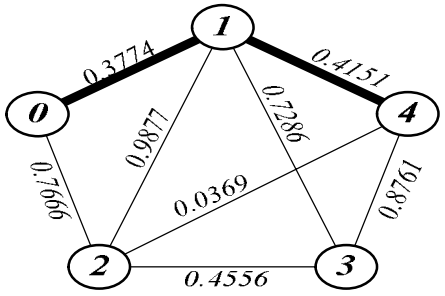


Figure (8): Shortest Path between Node 0 and Node 4

11- Results Discussion

1. **In LARHNN algorithm example, success of finding feasible path depends on delay bound and links' metrics values, no success to find feasible path while bound is very small, increasing delay bound value give chance to find feasible path.**
2. **Effect of mobility is show that LARHNN is suitable for soft QoS. This means for low mobility it is possible to find feasible path since time of running the algorithm is less than movement time, while in high mobility the algorithm fails to find feasible path at time.**
3. **Proposed algorithm can find constrained multicast tree in addition to feasible path. In some cases this tree may be not the optimal multicast tree.**
4. **Effect of changing HNN parameters on the number of iterations by HNN to converge to the solution, increasing gain of sigmoid equation decreases the number of iterations as explained, also using another sigmoid function decreases the number of iterations. The goal not always to decrease the number of iterations but also to keep HNN converges to the suitable solution.**

12- Conclusions

Through the demonstration example of the proposed algorithm LARHNN presented, few conclusions can be obtained as follows:

1.Since Hopfield Neural Network is categorized as isolated algorithm, these make the proposed LARHNN algorithm without need to exchange routing tables between nodes. The computation of feasible path is done on source node based on the network links' information that is got by prior process "BEACONING". The computation initiated by connection request, no need for any extra control messages during feasible path computation.

2.The proposed algorithm can find only single feasible path due to the Hopfield Neural Network features and the iteration process of Lagrange relaxation.

3.Success of finding feasible path is proportional to delay bound so that large delay bounds give chance to find feasible path more than small delay bounds. While guarantying QoS provision depends on mobility so that the proposed algorithm fails with high mobility and success with moderate and slow mobility.

4.Network size has significant effect on the execution time of the proposed algorithm and this was clear from the examples presented. As noticed increasing number of nodes in the network leads to more HNN iterations to reach stable state and thus more execution time.

5.Execution time will be also function of iteration process of Lagrange relaxation if the delay of path Pc did not be less than delay bound.

13- References

[1] L.D. Aronson, "Networks and routing - A survey", Report 94-112, Technical Mathematics and Informatics, Netherlands, 1994.

[2] S. Ackmer, U. Bilstrup, L. Svalmark, "Routing Protocol for Wireless Real time Multihop Network", M. Sc. Thesis, Halmstad University, Sweden, 1999.

[3] F. F. Wahhab, "Multi-Path Routing Protocol for Rapidly Deployable Radio Networks", M. Sc. Thesis, University of Jordan, 1994.

[4] L.D. Aronson, "Networks and routing - A survey", Report 94-112, Technical Mathematics and Informatics, Netherlands, 1994.

[5] C. E. Perkins, "Ad hoc Networking", Addison-Wesley, 2001.

[6] M. K. M. Ali, F. Kamoun, "Neural networks for shortest path computation and routing in computer networks", IEEE transactions on Neural Networks, vol. 4, No. 6, 1993.

[7] P. Samuel, H. Said, W. G. Probst. "An Artificial Neural Network Approach for Routing in Distributed Computer Networks", Elsevier Science Ltd, Canada, 2001.

[8] S. Chen, K. Nahrastedt, "An Overview of Quality of Service Routing for the Next Generation High Speed Networks: Problems and Solutions", Technical Report, University of Illinois, 1998.

[9] S. Chen, K. Narrstedt, "On finding Multi-Constraint Paths", IEEE International Conference on Communications, University of Illinois, June, 1998.

[10] R. Guerin, A. Orda, "QoS-based Routing in Networks with Inaccurate Information: Theory and Algorithms", IEEE INFOCOM '97, Japan, April, 1997.

[11] B. Hughes, "State of the Art Review of Quality of Service in Ad hoc Wireless Networks", M. Sc. Thesis, University of Dublin, Ireland, 2002.

[12] D. Lang, "A Comprehensive Overview about Selected Ad hoc Networking Routing Protocols", M. Sc. Thesis, March, 2003.

[13] I. Jawhar, J. Wu, "Quality of Service Routing in Mobile Ad hoc Networks", Florida Atlantic University, Kluwer Academic Publishers

A. S. Tanenbaum, "Computer Networks", Third Edition, Prentice-Hall, 1996.

تصميم وتقييم مفاهيم تحديد المسار الخاضعة لمعيار جودة الخدمة للشبكات المتنقلة المنشأة لغرض خاص

م.م. هند جعفر اغاتي

كلية المنصور الجامعة

المستخلص :

الشبكات اللاسلكية المُنشأة لغرض خاص هي احد مبادئ شبكات الحاسبات والتي تعني الاتصال بين مستخدمي الحاسبات ضمن شبكة مؤقتة بدون ادارة مركزية لهذه الشبكة. كل عقدة node في هذه الشبكة تمثل محدد المسار router والمضيف host في ان واحد ويجب ان تتولى مسؤولية نقل البيانات الى العقد الاخرى ولهذا فهناك حاجة لخوارزمية تحديد مسار ملائمة لـ ئروف هذه الشبكة. أحد أهم خصائص هذه الشبكة هو البنيوية المتحركة dynamic topology بسبب حركة العقد. العقد القريبة يمكنها الأتصال فيما بينها مباشرة أما العقد البعيدة فتتصل عبر مسار من العقد الوسيطة.

ان تحقيق معيار الجودة لخوارزميات تحديد المسار Quality of Service routing في هذا النوع من الشبكات ليس بالامر السهل لان بنيوية الشبكة متغيرة باستمراروامعلومات حالة الشبكة غير دقيقة.

تم اقتراح خوارزمية تحديد مسارLARHNN (طريقة تراخي لاكرانج المعتمدة على الشبكة العصبية) لحل مشكلة المسار المُحدد التأخيرمع الكلفة الدنيا DCLC في الشبكات اللاسلكية المُنشأة لغرض خاص. تعتمد الخوارزمية على طريقة التراخي للاكرانج Lagrange relaxation لحل احدى مشاكل تحديد المسار المراعي لمعيار الجودة والتي هي مشكلة معقدة الحل وتقوم الخوارزمية بايجاد مسار ذو مصادرresources كافية لتحقيق مسار يوفر متطلب التأخيرمع الكلفة الدنيا Delay Constrained Least Cost في تحقيق مسار منفرد unicast و شجرة المسارات المشروطة القصيرة constrained minimal tree في تحقيق مسارات عديدة multicast. الخوارزمية المقترحة مبنية على الحساب بواسطة الشبكات العصبية Hop field Neural Network لايجاد المسار الاقصر shortest path الذي تحتاجه الخوارزمية المقترحة في عملها. وتم اختيار الشبكات العصبية من بين مجموعة خوارزميات للمسار الاقصر لما لها من خصائص للعمل في بيئة العمل الحقيقي real time وتبينها تغيرات البنيوية topology changes ووقت تنفيذ يعتمد على الكيان الصلب للحاسبة hardware التي التنفيذ عليها,والسبب الاخر تعتبر خوارزميه ذات ربط متماثل ولها سيطرة موزعه غير متزامنه واخيرا تحتوي ذاكرة معنونه ومناسبه لتمثيل مثل هكذا جانب.