DEPARTMENT OF ELECTRONICS AND COMMUNICATION
B M S COLLEGE OF ENGINEERING
(AUTONOMOUS COLLEGE UNDER VTU, BELGAUM)
BANGALORE – 560019

2017-18

6TH SEMESTER ALTERNATE ASSESSMENT
IN

COMPUTER COMMUNICATION NETWORK
(16EC6DCCCN)

ROUTING IN COMPUTER NETWORKS USING ARTIFICIAL NEURAL NETWORKS

BY

AKSHAY. S. RAO | 1BM17EC007

Course Instructors
Dr G.POORNIMA
K.SUJATHA
B.HARSHITHA

CONTENTS

INTRODUCTION

The report proposes a heuristic approach based on Hopfield model of neural networks to solve the problem of routing which constitutes one of the key aspects of the topological design of computer networks. Adaptive to changes in link costs and network topology, the proposed approach relies on the utilization of an energy function which simulates the objective function used in network optimization while respecting the constraints imposed by the network designers. This function must converge toward a solution which, if not the best is at least as close as possible to the optimum. The simulation results reveal that the end-to-end delay computed according to this neural network approach is usually better than those determined by the conventional routing heuristics, in the sense that the proposed routing algorithm realizes a better trade-off between end-to-end delay and running time, and consequently gives a better performance than many other well-known optimal algorithms.

LITERATURE SURVEY

As illustrated in Fig. 1, a typical packet switching network consists of a set of nodes connected by communication links. The term node refers to terminals, printers, servers or hosts, and switches. A reliable communication network should permit two hosts to communicate with each other, as well as to tolerate certain breakdowns of nodes and links, with a possible reduction in performance.

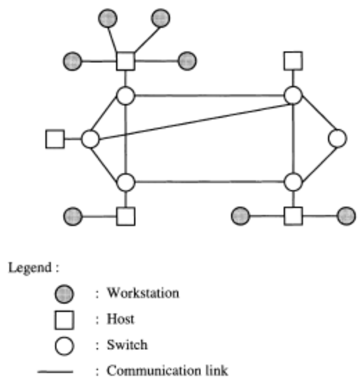


Fig. 1. A typical packet switching network.

The nodes through which the packet is transmitted from source to destination constitute a path or a route; the mechanism used to select one route from various alternatives to link each source-destination pair is called a routing strategy.

The objective of a routing strategy is essentially to minimize the mean delay of the packets in a network, subject to some reliability or capacity constraints. There exist some exact mathematical programming methods to solve problem of optimal routing but they are complex to implement and has lengthy calculations. As a result, heuristic routing procedures have been used in order to determine, within reasonable computation time, the routes along which the packets must travel without causing network congestion.

Routing procedures are classified into three categories

1. Static routing: It consists of defining the paths to be followed by the various packets based on the general characteristics of the network, such as the topology and anticipated mean traffic on the communication links. The traffic matrix gives the average number of packets per second exchanged between each node pair of the network. The routing procedures determine the link flow f which refers to the effective number of bits per second (bps)

carried by a link; the link capacity C denotes the maximum number of bits per second carried by this link

2. Adaptive routing: Adaptive routing is based on the dynamic characteristics of the network in order to manage the traffic requirements expressed by the traffic matrix. Among these characteristics, consideration must be given to the utilization of the links (f /C), their residual capacity (C - f), the cost of paths, and so on.

Despite their simplicity of implementation and their speed of execution, the heuristics of static routing do not generally take into account the level of traffic in the network, and thus do not guarantee a good solution to the problem. On the other hand, the adaptive routing procedures take into account the level of traffic, but the storage requirements of the current state of the network at each node may lead to the eventual saturation of the buffer at these nodes.

3. Optimal routing: They are practically never used, due to their complexity and relatively high running time.

Nomenclature

w_{ij}	Resistive connection between neurons i and j, element of the connection matrix of the network
τ	A circuit's time constant of a neuron
I_i	External bias provided by the user to the neuron i
U_i	Input of the ith neuron
V_i	Output or state of the ith neuron
E	Energy function of the Hopfield model
L_{sd}	Length of the route between node s and node d
C_{ij}	Cost, distance, or time transmission between node i and node j
V_{xi}	Output of the neuron at location (x, i)
μ_i	ith weighting coefficient of the energy function
f_k	Flow of link k, in kilobits per second (kbps)
C_k	Capacity of link k, in kilobits per second (kbps)
m	Number of links in the network topology
γ_{ij}	Average traffic between node i and node j of the computer network, in packet per second
γ	Total traffic in the computer network, in packet per second
T	End-to-end delay of a computer network, in second per packet
$T(\theta)$	End-to-end delay of a computer network topology θ, in second per packet
t_p	Propagation delay, equal to L/c where L is the length of the link, and c the speed of light
t_{run}	Running time in seconds required for determining the routing matrix and the end-to-end delay associated with a computer network topology.

This raises the question of considering the possibility to design a routing algorithm which, while being simple to implement and fast to execute, also takes into account the level of traffic on each link for efficient operation and for a solution approaching the optimum. The Hopfield model of neural networks appears as a suitable answer to this question. The paper proposes a heuristic approach based on Hopfield neural networks to solve the routing problem in a topological design context.

The proposed routing method

Outline of the concept of Hopfield networks.

The Hopfield networks constitute a class of artificial neural networks which consist of n neurons completely connected, each neuron having two possible states: $V_i = -1$ and $V_i = 1$ (Hopfield used 0 and 1, but this is equivalent). The connection of neuron i to neuron j is denoted by w_{ij}, and the total entry of a neuron i is equal to

$$\sum_j w_{ij} V_j$$

The network has a dynamic functioning assumed to be sequenced by a clock. Hopfield introduces the concept of energy of a neural network at a given time t. If the states of the network are $V_i \in [-1, 1]$ then the energy of the network is defined as:

$$E = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} V_i V_j \quad - (1)$$

The dynamics of Hopfield networks may be described by the following relation:

$$\frac{dU_i}{dt} = \sum_{j=1}^{n} w_{ij} V_j - \frac{U_i}{\tau} + I_i \quad - (2)$$

For a symmetric connection matrix, and for a sufficiently high gain of the amplifiers $(\lambda_i \to \infty)$ the dynamics of the neurons follow a decreasing gradient descent of the quadratic energy function E:

$$E = -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} V_i V_j - \sum_{i=1}^{n} I_i V_j \quad - (3)$$

In terms of energy function, the following relation describes the dynamics of the ith neuron

$$\frac{dU_i}{dt} = -\frac{U_i}{\tau} - \frac{\partial E}{\partial V_i} \quad - (4)$$

In order to find the shortest path between any node pair of a computer network, the paper proposes adaptation of this Hopfield model

Network Representation

The communication networks considered can be modelled by a graph G = (N, A), where N denotes the set of nodes and A the set of arcs or links. Each arc is associated with a non-negative number C_{ij} that can represent the cost, distance or time of transmission between node i and node j. The length of the route between the nodes s and d is defined as follows:

$$L_{sd} = C_{si} + C_{ij} + C_{jk} + \cdots + C_{rd}$$

The problem is to find through the network, the route with the minimal cost (shortest length) between the two nodes s and d. Based on the Hopfield networks, the model consists of $n(n-1)$ neurons represented by a matrix n×n where all the neurons on the diagonal are eliminated. The coordinates of the neurons are (x, i), where x denotes the rows, and i the columns. The neuron at (x, i) is characterized by its output V_{xi} and defined as follows:

$$V_{xi} = \{ \begin{matrix} 1 & \text{if the arc}(x,i) \text{ is part of the route} \\ 0 & \text{if not} \end{matrix}$$

And the variable ρ_{xi} defined as follows: $\rho_{xi} = \{ \begin{matrix} 0 & \text{if the arc}(x,i)\text{exists} \\ 1 & \text{if not} \end{matrix}$

The cost of the arc (x, i) is denoted by C_{xi} which is a real positive variable. Null costs are assigned to the non-existent arcs. To solve the problem, it is necessary to define the energy function used for the minimization process of the neural networks. For the purposes of numerical manipulation associated with calculations of the derivatives, but without loss of generality, the paper adopted the function proposed by Mehmet and Kamoun.

$$E = \frac{\mu_1}{2}\sum_{x=1}^{n}\sum_{i=1}^{n} C_{xi}V_{xi} + \frac{\mu_2}{2}\sum_{x=1}^{n}\sum_{i=1}^{n} \rho_{xi}V_{xi}$$

$$(x,i) \neq (d,s)$$

$$+\frac{\mu_3}{2}\sum_{x=1}^{n} \{\sum_{\substack{i=1 \\ i\neq x}}^{n} V_{xi} - \sum_{i=1}^{n} V_{ix}\}^2 + \frac{\mu_4}{2}\sum_{i=1}^{n}\sum_{\substack{x=1 \\ x\neq i}} V_{xi}\cdot(1-V_{xi}) + \frac{\mu_5}{2}(1-V_{dS}) - (5)$$

In Eq. (5): The μ_1 term is utilized to minimize the total cost of the searched route by adding the cost of existing arcs. The μ_2 term prevents and prohibits the inclusion of nonexistent arcs. The μ_3 term is zero if each neuron (x, i) is linked only to two others. The μ_4 term forces the network to converge towards one of the 2^{n2-n} vertices (networks consisting of n^2 - n neurons) of the hypercube defined by $V_{xi} \in \{0; 1\}$.The μ_5 term is equal to zero if the output of neuron (d, s) is 1.

Determination of Routes:

As the neurons are organized into a two-dimensional table Eqs. (2) and (4) can be rewritten, respectively, as follows:

$$\frac{\mathrm{d}U_{xi}}{\mathrm{d}t} = \sum_{y=1}^{n}\sum_{\substack{j=1 \\ j\neq y}}^{n} w_{xiyi}\cdot V_{yj} - \frac{U_{xi}}{\tau} + I_{xi} - (6)$$

$$\frac{\mathrm{d}U_{xl}}{\mathrm{d}t} = -\frac{U_{xl}}{\tau} - \frac{\partial E}{\partial V_{xi}} - (7)$$

By substituting Eq. (5) in Eq. (7) and calculating the derivative $\partial E/\partial V_{xi}$,

$$
\frac{\mathrm{d}U_{xi}}{\mathrm{d}t} = -\frac{U_{xi}}{\tau} - \frac{\mu_1}{2} C_{xi}(1 - \delta_{xd} \cdot \delta_{is}) - \frac{\mu_2}{2} \rho_{xi}(1 - \delta_{xd} \cdot \delta_{is})
$$

$$
-\mu_3 \sum_{\substack{y=1 \\ y \neq x}}^{n} (\mathrm{V}_{xy} - \mathrm{V}_{yx}) + \mu_3 \sum_{\substack{y=1 \\ y \neq i}}^{n} (\mathrm{V}_{iy} - \mathrm{V}_{yi}) - \frac{\mu_4}{2}(1 - 2 \cdot \mathrm{V}_{xi}) + \frac{\mu_5}{2} \delta_{xd} \cdot \delta_{is}
$$

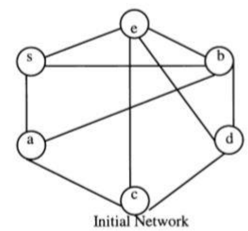
Where δ is the Kronecker symbol, which is defined as follows δab = 1 if a=b and δab = 0 if not. By comparing the coefficients of Eqs. (6) and (8), the w_{xiyi} of the connection matrix take the following values:

$$
w_{xi,yj} = \mu_4 \delta_{xy} \delta_{ij} - \mu_3 \delta_{xy} - \mu_3 \delta_{ij} + \mu_3 \delta_{jx} + \mu_3 \delta_{iy}
$$

$$
I_{xi} = -\frac{\mu_1}{2} C_{xi}(1 - \delta_{xd} \cdot \delta_{is}) - \frac{\mu_2}{2} \rho_{xi}(1 - \delta_{xd} \cdot \delta_{is}) - \frac{\mu_4}{2} + \frac{\mu_5}{2} \delta_{xd} \cdot \delta_{is}
$$

$$
= \begin{cases} \frac{\mu_5}{2} - \frac{\mu_4}{2} & \text{if}(x, i) = (d, s) \\ -\frac{\mu_1}{2} C_{xi} - \frac{\mu_2}{2} \rho_{xi} - \frac{\mu_4}{2} & \text{if not } \forall (x \neq i), (y \neq j) \end{cases}
$$

- The initial data of the neurons U_{xi} at time t are equal to zero, and the evolution of the state of the neural network is simulated by the solution of a system of n*(n - 1) differential equations where the variables are the neuron outputs V_{xi}.
- To solve this system, the numerical method of Runge–Kutta of the fourth order is used.
- The solution consists of observing the outputs of the neurons V_{xi} for a specific duration δt. The circuit's time constant τ for each neuron is initialized to 1.
- It is also noted that, in order to obtain good results, δt must be between 10^{-5} and 10^{-3}.
- To avoid bias in favour of any particular path, it must be assumed that all inputs U_{xi} are equal to 0. However, to help the network converge rapidly, while preventing it from adopting an undesirable state (for example, convergence of two different paths), small perturbations must be made to the initial inputs of the U_{xi} network.
- At the start, it is assumed that $-0.0002 \leq \delta U_{xi} \leq 0.0002$: The calculations cease when the network reaches a stable state, that is, when the difference between the outputs is less than 10^{-5}(V_{xi} $\leq$0.00001) from one update to another.
- When the network is in a stable state, the final values of V_{xi} are rounded off, that is, they are set to 0 if $V_{xi} < 0.5$; and to 1 otherwise.



Initial Network

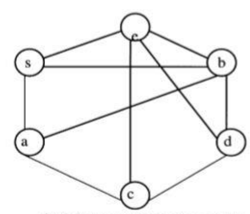
	s	a	b	c	d	e
s	0	0.4	1.6	0	0	0.3
a	0.4	0	0.8	0.2	0	0
b	1.6	0.8	0	0	0.1	0.5
c	0	0.2	0	0	0.1	0.7
d	0	0	0.1	0.1	0	0.5
e	0.3	0	0.5	0.7	0.5	0

Cost matrix

	s	a	b	c	d	e
s	1	0	0	1	1	0
a	0	1	0	0	1	1
b	0	0	1	1	0	0
c	1	0	1	1	0	0
d	1	1	0	0	1	0
e	0	1	0	0	0	1

Matrix ρ

Fig. 3. Neural network and its related cost matrix.



Path between the nodes s and d

	s	a	b	c	d	e
s	0	1	0	0	0	0
a	0	0	0	1	0	0
b	0	0	0	0	0	0
c	0	0	0	0	1	0
d	1	0	0	0	0	0
e	0	0	0	0	0	0

Output neurons matrix V_{xi}

Fig. 4. Best route between nodes *s* and *d*.

Here are the main steps in the search of a route:

Step 1. Obtain the network data: the number n of nodes, the matrix r of the links, and the matrix of link costs

Step 2. Initialize the matrix of the V_{xi}: the V_{xi}'s take random values between -0.0002 and +0.0002;

Step 3. Trigger the process of minimization in the neural network to solve the differential equations and to stabilize the network;

Step 4. Round off the values of the matrix of outputs and obtain the searched route.

The example in Fig. 3 concerns a network of six nodes, which has two corresponding matrices: the link matrix, and the cost matrix (expressed in km). The problem is to find all the routes between all node pairs in the network. Fig. 4 shows the paths between node s and node d; in the matrix of outputs, the 1's denote only the connections belonging to the path. The connections found form a cycle; however, in the construction of the path, it is necessary to eliminate the link between the source and destination nodes.

Routing and Delay Computation.

The ultimate goal of a routing procedure remains the flow and capacity assignment which allows us to compute the end-to-end delay of a given network topology. The routing matrix provides, for each source and destination pair, the best route along which the packets must travel through the network. From this matrix, we can calculate the flow carried by each link, by distributing the traffic between each node pair on the routes of the routing matrix. Then, a capacity value can be assigned to each link by using capacity options available on the marketplace, and to ensure that the link flow does not exceed the link.

delay T is calculated as follows

$$T = \frac{1}{\gamma} \sum_{k=1}^{m} \frac{f_k}{C_k - f_k}$$

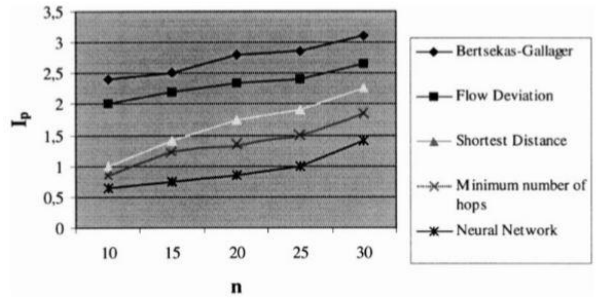
If the length L of the link is expressed in km, we then have

$$t_{\mathrm{p}} = \frac{10^{-5}}{3} L$$

APPLICATION AND CONCLUSION

Comparisons with optimal routing algorithms

Routing method	End-to-end delay T_{mean} (s)	Running time t_{run} (s)
Shortest distance	0.795	0.110
Minimum no. of "Hops"	0.627	0.110
Neural network	0.3450	0.114
Flow-deviation	0.190	5.54
Bertsekas–Gallager	0.187	13.94



- The method proposed in the paper to select the weighting coefficients relies on a set of hybrid constraints which are congruent with the approach proposed by Mehmet and Kamoun.
- With the values derived from this simulation for the weighting coefficients, the mean number of iterations needed to converge to states corresponding to valid solutions which are often global optima is equal to 100, compared to 5000 in the work of Mehmet and Kamoun .
- The analysis of the results reveals that the end-to-end delay computed according to the proposed method is, in most cases, better than those determined by conventional routing methods such as the Shortest Path and the Minimum Number of Hops.
- Compared with optimal algorithms such as the Flow Deviation method and the algorithm of Bertsekas– Gallager, the proposed method's delay results are slightly less favorable than the ones obtained with these algorithms; however, the method's execution times are considerably shorter.
- For a low traffic level, about 50% of the links have relatively great values of flow, leading to some great link capacities; for a high traffic level, only less than 22% of these links have truly high values of flow, and consequently high-speed. Such a reduction in the number of high capacity links possibly results in scale economy in terms of communication costs according to the traffic requirements.
- Finally, neural routing algorithm realizes the better trade-off between end-to-end delay and running time, and consequently gives a better performance than the other conventional algorithms considered.

REFERENCE

Routing in computer networks using artificial neural networks - S. Pierre, H. Said, W.G. Probst.