International Journal of General Systems
Vol. 38, No. 7, October 2009, 777–791

A shortest path routing algorithm using Hopfield neural network with an improved energy function

Dong-Chul Park[a]* and Kyo-Reen Keum[b]

[a]*Department of Information Engineering, Myongji University, Yongin, South Korea;* [b]*Department of Marine Diesel Engine Shop Test, STX Engine Co., Seoul, South Korea*

(*Received 1 February 2006; final version received 16 April 2008*)

A shortest path routing algorithm using the Hopfield neural network with a modified Lyapunov function is proposed. The modified version of the Lyapunov energy function for an optimal routing problem is proposed for determining routing order for a source and multiple destinations. The proposed energy function mainly prevents the solution path from having loops and partitions. Experiments are performed on 3000 networks of up to 50 nodes with randomly selected link costs. The performance of the proposed algorithm is compared with several conventional algorithms including Ali and Kamoun's, Park and Choi's, and Ahn and Ramakrishna's algorithms in terms of the route optimality and convergence rate. The results show that the proposed algorithm outperforms conventional methods in all cases of experiments. The proposed algorithm particularly shows significant improvements on the route optimality and convergence rate over conventional algorithms when the size of the network approaches 50 nodes.

Keywords: Hopfield neural network; shortest path; routing

1. Introduction

Multicast communication services in many distributed networks have grown rapidly in accordance with recent trends and technological achievements in broadcasting industries in relation to TV, Internet and computer communications (Atwood 2004, Ganjam and Zhang 2005). Issues related to multicast communication include fast delivery of messages from a source node to more than one destination node in a connected communication network with constraints including the bandwidth, the link capacity, the delay bound and traffic loads among nodes (Kumar and Jaffe 1983, Kompella *et al.* 1993, Lin and Ni 1993). In communication network services, efficient routing of data or information has a significant impact on the quality of services and the performance of the communication network. An ideal routing algorithm should find the shortest path within a specified time period so as to satisfy the user's demand for minimising the data transmission delay.

This paper deals with an optimal routing problem with multiple destinations in communication networks. Providing the optimal routing problem can be formulated as a nonlinear multicommodity flow problem, the solution involves the shortest path computation that has to be carried out with minimal computational effort (Misra and Oommen 2005). This problem is also known as a constrained combinatorial optimisation problem. Recent successes in computational intelligence research provide us with a wide

*Corresponding author. Email: parkd@mju.ac.kr; parkd@dreamwiz.com

ISSN 0308-1079 print/ISSN 1563-5104 online

DOI: 10.1080/03081070902880433
http://www.informaworld.com

variety of approaches for constrained combinatorial optimisation problems. Among them, evolutionary algorithms including genetic algorithms (GA), simulated annealing (SA) approaches and neural networks have shown promising results.

Evolutionary algorithms develop a population of genetically encoded competing candidate solutions, and the next generation evolves from the current population by selection, recombination and mutation. By the genetic structures and the genetic operators for generating new variants, evolutionary algorithms can be classified as GA, evolution strategies (ES), evolutionary programming (EP) and genetic programming (GP) (Fatta *et al.* 2003). Compared to traditional search algorithms, a GA, similar to other evolutionary algorithms, is often able to automatically acquire and accumulate implicit knowledge about the search space during its search process and self-adaptively control the search process through a random optimisation technique. When applied to TSP, the GA based routing algorithm takes a few seconds to run the first 100 generations for a neural network (Wu *et al.* 2004).

Chen and Aihara proposed a SA scheme. The scheme starts with a sufficiently large negative self-coupling in the Aihara–Takabe–Toyota network and gradually decreases the self-coupling, thereby obtaining a transiently chaotic neural network and stochastic SA. Their computer simulations showed that SA leads to good solutions for the TSP that can be much more readily compared to the Hopfield–Tank approach (Chen and Aihara 1995). It is well known that SA can find high-quality solutions, if the annealing parameter (temperature) is reduced exponentially but with a sufficiently small exponent. The bottleneck of the computation complexity of SA, however, is the selection process. In this selection process, the whole population has to be sorted. The computational complexity of the best sorting algorithm is $O(P \log P)$ where P is the total number of individuals in each generation (Yip and Pao 1995). Typically, GA requires thousands of generations. For a 10-city TSP, SA uses at least 86 s (x86 Family 6 Model 8 Stepping 6, AT/AT Compatible) to find the global optima (Wang *et al.* 2004). This implies that SA is not practically applicable to real time operation.

As can be seen from the hardware implementation of neural network models such as Kocak *et al.* (2003), the potential of the Hopfield–Tank type neural network approach for TSP surpasses that of other newly developed approaches including the SA and GA. With respect to real time applications, an approach based on neural networks, especially the Hopfield neural network (HNN), is a good candidate for implementing the shortest path computations involved in the routing problem, primarily owing to the potential of the neural network hardware approach for high speed computation.

The use of neural networks to find the shortest path between a given source and destination was first introduced by Rauch and Winarske (1988). The approach by Rauch and Winarske is limited by the requirement of prior knowledge of the number of nodes forming the shortest path. However, the number of nodes forming the shortest path is usually unknown for each case. That is, this routing algorithm does not find the shortest path among all possible combinations of paths, but it can find the shortest path among the paths that consist of all the predetermined specific nodes. Therefore, the algorithm proposed by Rauch and Winarske finds only a suboptimal solution to the shortest path problem. In order to overcome the problem in the algorithm proposed by Rauch and Winarske, Zhang and Thomopoulos (1989) proposed another approach that makes it possible to visit all the nodes in a network. However, the neural network designed by this algorithm finds the shortest path between only a given source-destination pair. For another source-destination pair, the neural network configuration has to be changed. Furthermore, the approach proposed by Zhang and Thomopoulos is not suitable for use in practical traffic routing problems because the link costs in a communication network are usually time varying while the network configuration by this algorithm is more static and more suitable for fixed link costs. Ali and Kamoun (1993) introduced an adaptive routing algorithm for different network topologies. This algorithm shows very good convergence and scaling properties with relatively low programming complexity for a small scale network. However, the convergence and the quality of solutions for larger scale networks are not satisfactory, even with 20 nodes, because the routing path has loops or partitions. Park and Choi (1998) proposed an algorithm wherein a new term is added to the energy function that can remove the loop or the partition occurring in the approach proposed by Ali and Kamoun for large scale networks. The performance of this approach, however, depends on the network topologies. In addition, the convergence speed of the algorithm should be improved. More recently, Ahn and Ramakrishna (2002) added two new terms to Park and Choi's energy function and proposed a new algorithm that can accelerate the convergence while improving route optimality. However, the solution quality with the new terms in the energy function of Ahn and Ramakrishna's algorithm is questionable for networks with more than 20 nodes.

In this paper, we propose a modified version of the Lyapunov energy function for an optimal routing problem developed in a large scale network for determining routing order for a source and multiple destinations. Applications of the HNN to solve combinatorial optimisation problems are defined in Section 2. The problem of routing according to the destination set is defined in Section 3. Section 4 compares the existing routing algorithms using a HNN in terms of their energy functions. Section 5 proposes an improved routing algorithm. In Section 6, the proposed HNN algorithm and other algorithms are applied to problems with various scales and the results are compared. The results of this study are summarised in Section 7.

2. Definition of the problem

With the definition of an undirected graph, the underlying topology of the communication networks can be given with $\mathbf{G} = (\mathbf{V},\mathbf{E})$, where $\mathbf{V}$ is the set of vertices (N nodes), and $\mathbf{E}$ is the set of its edges (Ahuja *et al.* 1993). Further, a link cost matrix $\mathbf{L} = [L_{ij}]$, where L_{ij} is the cost from node i to node j, s is the source node and d is the destination node. For multiple destination problems, d_x is defined as the set of destination where $d_x \in \mathbf{V}$ and $x = \{1, \ldots, N\}$.

For each link (i, j), there exists a nonnegative number L_{ij}, called the cost including the time delay, the bandwidth, and the traffic load of the link from node i to node j. If there is no link from node i to j, L_{ij} is set to a very large value in order to exclude it from the routing path. Note that link (i, j) is symmetric with link (i, j) and $L_{ij} = L_{ji}$ since the network is given by an undirected graph.

If we define an undirected path P^{sd} for a routing problem, an ordered sequence of nodes connecting s to d can be written by:

$$P^{sd} = (s, a, b, \ldots, i, d),$$

where the route is as follows:

$$(s \rightarrow a \rightarrow b \rightarrow \cdots \rightarrow i \rightarrow d).$$

Then the total cost TC_{sd} of this path becomes:

$$\mathrm{TC}_{sd} = L_{sa} + L_{ab} + \cdots + L_{id}.$$

Our problem can then be summarised as follows:

Find a path that minimises TC_{sd}.

Therefore, we can also represent the problem of routing for multiple destinations d_x with respect to an undirected path P^{sd_x} as follows:

$$P^{sd_x} = (s, a, b, \ldots, d_1, \ldots, d_2, \ldots, i, d_x),$$

where the route is

$$(s \to a \to b \to \cdots \to d_1 \to \cdots d_2 \to \cdots \to i \to d_x).$$

The total cost TC_{sd_x} of this problem then becomes: $\mathrm{TC}_{sd_x} = L_{sa} + L_{ab} + \cdots + L_{id_x}$.

To formulate the above problem, a $(n \times n)$ matrix with all diagonal elements set to zero is used, where n is the number of nodes. Each element in the matrix is represented by a neuron which is denoted by two indices (i, j). The state of a neuron, V_{ij}, is defined as follows:

$$V_{ij} = \begin{cases} 1, & \text{if the link from node } i \text{ to node } j \text{ exists in solution,} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

A matrix $\Gamma = [\gamma_{ij}]$ is defined for the link connection (or edge) as follows:

$$\gamma_{ij} = \begin{cases} 1, & \text{if the link from node } i \text{ to node } j \text{ does not exist,} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

By using the above definitions, the problem of multiple destinations can be represented by the logical algebraic expressions of constrained combinatorial optimisation problems as follows:

$$\text{Minimise} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} L_{ij} V_{ij} \quad \text{for all } (s \to d_x), \ x \in \ \{1, 2, \ldots, N\}, \tag{3}$$

$$\text{Subject to} \left(\sum_{\substack{j=0 \\ j \neq i}}^{N-1} V_{ij} - \sum_{\substack{k=0 \\ k \neq i}}^{N-1} V_{ki} \right) = \begin{cases} 1, & \text{if } i = s, \\ -1, & \text{if } i = d_x, \ i = \{1, 2, \cdots, N\}, \\ & \quad x \in \mathbf{X}, \text{ where } \mathbf{X} \text{ is the set of destination,} \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

3. Hopfield neural networks for routing problems

Application to the constrained combinatorial optimisation problems using neural networks was first introduced by Hopfield and Tank in an attempt to find good solutions if not the best solution within a permissible time period by applying their model, the HNN, to the travelling salesman problem (TSP) (Hopfield and Tank 1986, Tank and Hopfield 1986, Funabiki and Takefuji 1993). The convergence of the nonlinear dynamic system for symmetric connections was verified by introducing the Lyapunov energy function. Since then, the HNN has been successfully used to solve various optimisation problems known as *NP-complete* (Yip and Pao 1995). The energy function of the Hopfield model is defined as follows:

$$E = \frac{1}{2}\sum_{i=1}^{P}\sum_{j=1}^{P} L_{ij} V_i V_j + \sum_{i=1}^{P} \theta_i V_i, \tag{5}$$

where P, V_i and θ_i, are the number of neurons, the output and bias of the ith neuron, respectively.

Later, Rauch and Winarske (1988) introduced an algorithm for finding the shortest path using neural networks. To improve the applicability of Rauch and Winarske's algorithm to practical problems, Zhang and Thomopoulos (1989) proposed a more practical algorithm by allowing the M value to the total number of nodes in the network. Later, Ali and Kamoun proposed an adaptive algorithm with a modified energy function that is designed to be minimised as the HNN evolves with time and find the optimal solutions for the shortest path. The energy function proposed by Ali and Kamoun is as follows:

$$\begin{aligned} E = {} & \frac{\mu_1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} L_{ij} V_{ij} + \frac{\mu_2}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} \gamma_{ij} V_{ij} + \frac{\mu_3}{2}\sum_{i=1}^{N}\left(\sum_{\substack{j=1\\ j\neq i}}^{N} V_{ij} - \sum_{\substack{j=1\\ j\neq i}}^{N} V_{ji}\right)^2 \\ & + \frac{\mu_4}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} V_{ij}(1 - V_{ij}) + \frac{\mu_5}{2}(1 - V_{ds}), \end{aligned} \tag{6}$$

where μ_1 minimises the total link cost of a route by taking into account the cost of existing links and the μ_2 term prevents nonexistent links from being included in the chosen routing path. The μ_3 term is zero if the number of incoming direction links equals the number of outgoing direction links while the μ_4 term enforces the state of the neural network to converge to a valid route. The μ_5 term includes the source node and the destination node for a formed path.

However, the algorithm given by Equation (6) has its own limitation to the size of the network for this algorithm to yield a proper solution as it does not converge well when the size of the network is larger than a certain size, roughly 20 nodes. When this algorithm fails to find valid routing paths, it typically yields a routing path that includes a loop (or partition) (Park and Choi 1998). In order to prevent the routing solution from including loops or partitions for large scale networks, Park and Choi proposed an improved routing algorithm with the following energy function:

$$\begin{aligned} E_{PC} = {} & \frac{\mu_1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} L_{ij} V_{ij} + \frac{\mu_2}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} \gamma_{ij} V_{ij} \\ & + \frac{\mu_3}{2}\sum_{i=1}^{N}\left(\sum_{\substack{j=1\\ j\neq i}}^{N} V_{ij} - \sum_{\substack{j=1\\ j\neq i}}^{N} V_{ji} - \phi_i\right)^2 + \frac{\mu_4}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} V_{ij}(1 - V_{ij}) \\ & + \frac{\mu_5}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N} V_{ij} V_{ji}, \end{aligned} \tag{7}$$

where, ϕ_i is defined as following:

$$\phi_i = \begin{cases} 1, & \text{if } i = s, \\ -1, & \text{if } i = d_x, \quad \forall i \in 1, \ldots, n, \quad x \in X, \\ 0, & \text{otherwise,} \end{cases} \tag{8}$$

where X is the set of destinations and n is the number of destination nodes.

The μ_5 term, which is newly introduced in Park and Choi's algorithm, ensures that the flow vector $V = [V_{ij}]$ is one-directional for each destination. That is, $V_{ij}V_{ji}$ is always zero for all (i, j) when the flow vector $V = [V_{ij}]$ is only one-directional when compared with the energy function in Ali and Kamoun's algorithm. Park and Choi reported that their algorithm with two preprocessing procedures showed more promising results compared to Ali and Kamoun's algorithm.

As Ahn and Ramakrishna pointed out, however, Park and Choi's algorithm sometimes suffers from instability and its convergence is heavily dependent on the network topologies. In order to enhance the speed of convergence and improve the quality of the solution, Ahn and Ramakrishna proposed a new energy function as follows:

$$E_{AR} = E_{PC} + \frac{\mu_6}{2}\left[\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}\left(\sum_{\substack{k=1\\ k\neq i,j}}^{N} V_{ik} - 1\right)V_{ij}^2\right] + \frac{\mu_7}{2}\left[\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}\left(\sum_{\substack{k=1\\ k\neq i,j}}^{N} V_{kj} - 1\right)V_{ij}^2\right]. \tag{9}$$

In Equation (9), the algorithm introduces new terms, the μ_6 and μ_7 terms, to eliminate the possibility of loops and partitions in a path and to have speedy and precise convergence to an optimal solution. The μ_6 and μ_7 terms work in a fashion such that if there is a tendency to form a routing path between one node and another node in the early stage of the evolving process of a neuron, then the neuron will be enforced to have a value of '1'. This results in fast convergence to a solution. This works well when there exists only an obvious choice of a link to choose for a node. On the other hand, however, when there exists less obvious routes to choose for a node, it is very dangerous to choose a route for a node in the early stage of evolution for a neuron. Therefore, the experimental results obtained in our experiments conflict with those of Ahn and Ramakrishna (2002).

4. The proposed algorithm

In order to improve the convergence of the solution and obtain a quality route for a given problem, a modified energy function is proposed in this paper as follows:

$$\begin{aligned} E = {} & \frac{\mu_1}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}\left(V_{ij}\sum_{\substack{k=1\\ k\neq i}}^{N}(1 - C_{ik})C_{ij}\right) + \frac{\mu_2}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}\gamma_{ij}V_{ij} \\ & + \frac{\mu_3}{2}\sum_{i=1}^{N}\left(\sum_{\substack{j=1\\ j\neq i}}^{N}V_{ij} - \sum_{\substack{j=1\\ j\neq i}}^{N}V_{ji} - \phi_i\right)^2 + \frac{\mu_4}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}V_{ij}(1 - V_{ij}) \\ & + \frac{\mu_5}{2}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}\left(V_{ij}\sum_{\substack{k=1\\ k\neq i}}^{N}V_{ki}\right). \end{aligned} \tag{10}$$

In Equation (10), the μ_1 term, newly introduced in this paper, not only minimises the total link cost of a routing path by taking into account the costs of existing links, but also takes account of all cost values around the node. The μ_2, μ_3 and μ_4 terms are identical to those in Equation (7). The μ_5 term is modified from Equation (7) so that the flow vector is $V = [V_{ij}]$ and is one-directional for all destinations. This means that if a routing path is formed from node i to node j, then the μ_5 term prevents any path from node j to node i while preventing any path from all nodes around node i. The μ_5 term, therefore, prevents the solution path from having loops or partitions.

The dynamics of the proposed HNN is summarised as follows:

$$\frac{\partial U_{ij}}{\partial t} = -\frac{U_{ij}}{\tau} - \frac{\partial E}{\partial V_{ij}} = -\frac{U_{ij}}{\tau} - \frac{\mu_1}{2}\sum_{\substack{k=1\\k\neq i}}^{N}(1 - C_{ik})C_{ij} - \frac{\mu_2}{2}\gamma_{ij} - \mu_3\left(\sum_{\substack{k=1\\k\neq i}}^{N}(V_{ik} - V_{ki}) - \Phi_i\right)$$
$$+ \mu_3\left(\sum_{\substack{k=1\\k\neq j}}^{N}(V_{jk} - V_{kj}) - \Phi_i\right) - \frac{\mu_4}{2}(1 - 2V_{ij}) - \frac{\mu_5}{2}\sum_{\substack{k=1\\k\neq i}}^{N}V_{ki}. \tag{11}$$

For implementing a HNN model from Equation (14), the connection weight $T_{ij,kl}$ and the bias I_{ij} are obtained as:

$$T_{ij,kl} = -\mu_3\delta_{ik} + \mu_3\delta_{jk} + \mu_3\delta_{li} - \mu_3\delta_{jl} + \mu_4\delta_{ik}\mu_3\delta_{jl} - \frac{\mu_5}{2}\sum_{\substack{k=1\\k\neq j}}^{N}\delta_{jk}, \tag{12}$$

$$I_{ij} = -\frac{\mu_1}{2}\sum_{\substack{k=1\\k\neq i}}^{N}(1 - C_{ik})C_{ij} - \frac{\mu_2}{2}\gamma_{ij} + \mu_3\ \Phi_i - \mu_3\ \Phi_j - \frac{\mu_4}{2}, \tag{13}$$

where the Kronecker delta function is defined by

$$\delta_{ij} = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{if } i \neq j. \end{cases} \tag{14}$$

By applying the first order Euler technique to Equation (11), we obtain the following:

$$U_{ij}(t + \Delta t) = U_{ij}(t) + \Delta t\left(-U_{ij}(t) - \frac{\partial E}{\partial V_{ij}}\right). \tag{15}$$

As shown in Equation (10), an advantage of this model is that it maps the link costs into the biases rather than into the neural interconnections. It also has flexibility, which is reflected by the attribute that the link cost and the network topological information can be changed through the biases. In other words, the interconnection weights do not depend on a particular source or destination. This will make the proposed neural network routing algorithm very attractive to operate in real time. In addition, by using the information on surrounding neurons, the proposed model is more effective in solving large scale networks. Because the proposed neural network model is adaptive according to any change in network topology, it is also more applicable to practical problems.

5. Experiments and results

5.1 *Proposed approach for multiple destination problem*

The proposed routing algorithm is applied to a multi-destination problem. In the multi-destination problem, a HNN with the proposed energy function (PK-HNN) is used as a routing tool for deciding the routing order among the source node and several destination nodes as well as a routing tool for each segment of source-destination pairs. For example, when we are given a three destination problem, that is, one source (S) and three destinations (D_a, D_b, D_c), the proposed approach uses the PK-HNN for deciding the routing order for the destinations, D_a, D_b and D_c, from S. Assume that the routing order is found as: $S \rightarrow D_c \rightarrow D_a \rightarrow D_b$. In the next step of the proposed approach for the multi-destination routing algorithm, the PK-HNN is applied to each pair of the routing orders: $S \rightarrow D_c$, $D_c \rightarrow D_a$ and $D_a \rightarrow D_b$. By applying the PK-HNN to two steps of the multi-destination routing algorithm, an optimal routing path can be obtained as long as the PK-HNN finds the optimal routing path in each pair of the source and destination nodes. Table 1 describes a pseudo code of the proposed approach for the multiple destination routing problem.

When designing a HNN, the importance of choosing appropriate parameter values for the network cannot be underestimated, because the performance of a HNN is highly sensitive to the parameter values. Although it is possible for the neural state in a HNN to fall into a local minimum state, it will be shown that, by proper tuning of the weight coefficients including the parameters, such a possibility can be reduced if not eliminated. Therefore, when selecting the parameters, a comparatively large value of μ_1 is first chosen. However, the μ_1 value is not chosen to be larger than the μ_2 value in order to prevent the network from forming a path with nonexistent links. A large value of μ_2 ensures that the solution quality is high while a large value of μ_3 guarantees that the network converges to a valid solution. A large value of μ_4, meanwhile, helps obtain fast convergence to a solution, but it opens up the possibility of falling into a local minimum from time to time. Therefore, a small value of μ_4 is chosen in our experiment for a stable, quality solution. In addition, the time constant τ of each neuron is set to 1 without any loss of generality and the simulation has shown an appropriate value of Δt is 10^{-4}. Since the neural transfer parameter λ is sensitive to performance, accuracy and computational time, we have chosen $\lambda = 1$ in order to allow the neurons' dynamics for finding the global minimum. Taking into account all of these requirements, the set of parameters for the proposed algorithm shown in Table 2 is chosen for our experiments. Table 2 also shows sets of parameters for other algorithms used in experiments. Note that the parameters, $\mu_6 = 100$ and $\mu_7 = 100$, for Ahn and Ramakrishna's algorithm used in our experiments are different from the one suggested in Ahn and Ramakrishna (2002) because the Ahn and Ramakrishna's HNN model hardly converges to valid solutions with the parameter value, $\mu_6 = \mu_7 = 400$, suggested in Ahn and Ramakrishna (2002).

Table 1. Pseudo-code for the multiple destination routing algorithm.

```
Set all parameters for a given network
Find the routing order of the source and destination nodes
  by using the PK-HNN
Initialise all the units and parameters
  For each pair of the source and destination nodes, do
    Apply the proposed PK-HNN
  End for
```

Table 2. Parameters used in experiments.

Parameter	Symbol	Algorithms: Ali and Kamoun	Park and Choi	Ahn and Ramakrishna	Proposed
Time constant	τ	1	1	1	1
Slope of logistic function	λ	1	1	1	1
Incremental time step	δ	0.0001	0.0001	0.0001	0.0001
Weight coefficients	μ_1	550	550	550	950
	μ_2	2550	2550	2550	2500
	μ_3	1950	1950	1950	1900
	μ_4	250	250	475	100
	μ_5	1350	1350	500	500
	μ_6	–	–	400–40	–
	μ_7	–	–	400–40	–

In this paper, the proposed algorithm is compared with the routing algorithms of Ali and Kamoun, Park and Choi and Ahn and Ramakrishna. As a baseline method for the optimality comparison, Dijkstra's algorithm (Stalling 1998) is selected and routing results from various algorithms are evaluated in terms of their optimality levels.

5.2 *A multi-destination problem with 40 nodes*

In order to evaluate the performance of the proposed algorithm for a multi-destination routing problem, an example network with 40 nodes, shown in Figure 1, is used. In Figure 1, the numbers on lines represent corresponding link costs. The source is given as node #1 and the destination nodes are #11, #18, #22, #35 and #38. First, we have to find the routing order between the source and destinations with respect to the node-to-node distance using the PK-HNN. In this experiment, the optimal routing order from the source to the multiple destinations is found as: ($S \longrightarrow D_1 \longrightarrow D_2 \longrightarrow D_3 \longrightarrow D_4 \longrightarrow D_5$). This routing

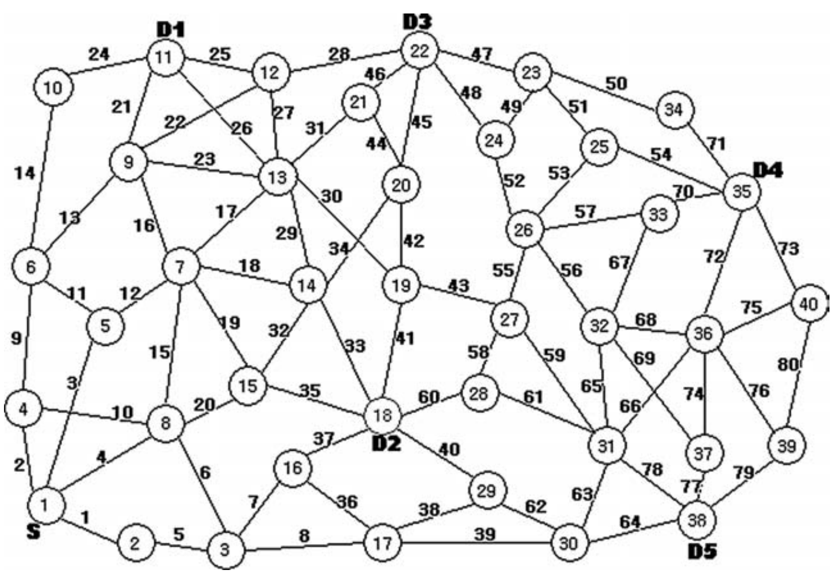

Figure 1. Example network for experiments (the numbers on lines represent link cost: $\times 10^{-4}$).

Table 3. Total cost comparison.

Algorithm	$S \rightarrow D_1$	$D_1 \rightarrow D_2$	$D_2 \rightarrow D_3$	$D_3 \rightarrow D_4$	$D_4 \rightarrow D_5$	Total
Ali and Kamoun	2535	1190	2336	5358	732	12,151
Park and Choi	2525	1190	1835	5411	732	11,693
Ahn and Ramakrishna	2525	1190	1835	2975	732	9257
Proposed	516	948	1835	1270	732	5301

path from the source to the destination nodes is used for the experiments with all the algorithms evaluated for performance comparison purpose in this paper.

Optimal paths that go through the destination path ($S \rightarrow D_1 \rightarrow D_2 \rightarrow D_3 \rightarrow D_4 \rightarrow D_5$) are obtained by applying the PK-HNN to each source-destination pair: $S \rightarrow D_1$, $D_1 \rightarrow D_2$, $D_2 \rightarrow D_3$, $D_3 \rightarrow D_4$ and $D_4 \rightarrow D_5$. The final routing path obtained by the proposed algorithm is compared in Table 3 with the results of the algorithms proposed by Ali and Kamoun, Park and Choi and Ahn and Ramakrishna in terms of total link costs and in Figure 2. Figure 3(a) shows the routing path from the Ali and Kamoun algorithm. Figure 3(b)–(d) shows the routing paths from Park and Choi, Ahn and Ramakrishna and the proposed algorithm, respectively. As shown in Figure 2 and Table 3, the proposed algorithm provides a quality routing solution.

5.3 *Experiments with random network topologies*

By generating 1000 networks for each group with different number of nodes; Group 1 (12–25 nodes), Group 2 (26–40 nodes) and Group 3 (41–50 nodes), experiments on a total

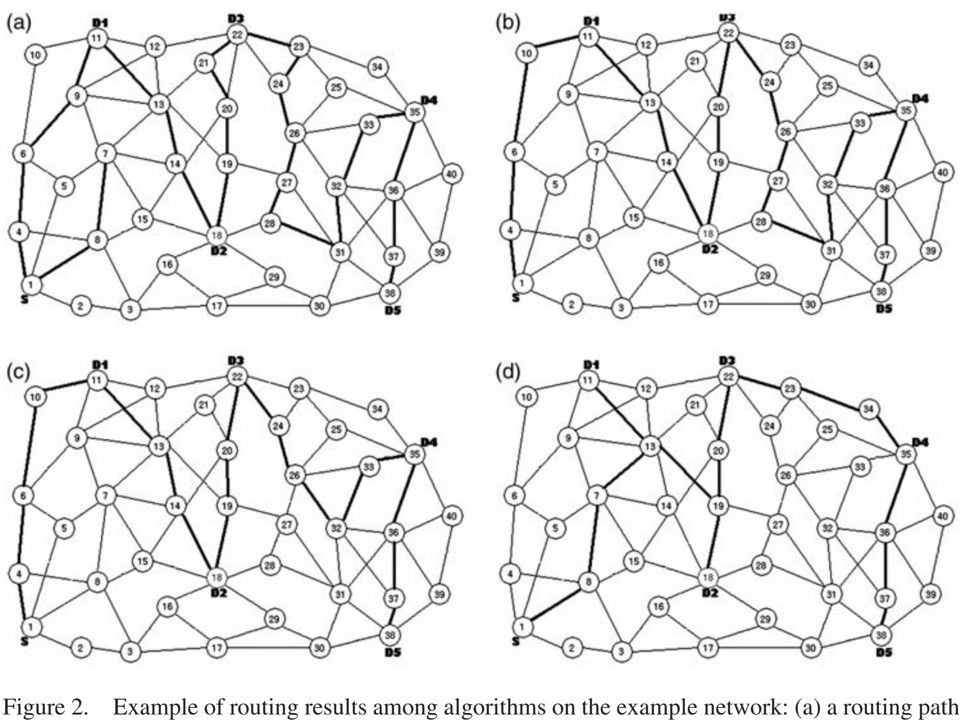

Figure 2. Example of routing results among algorithms on the example network: (a) a routing path of Ali and Kamoun's algorithm, (b) a routing path of Park and Choi's algorithm, (c) a routing path of Ahn and Ramakrishna's algorithms and (d) a routing path of the proposed algorithm.

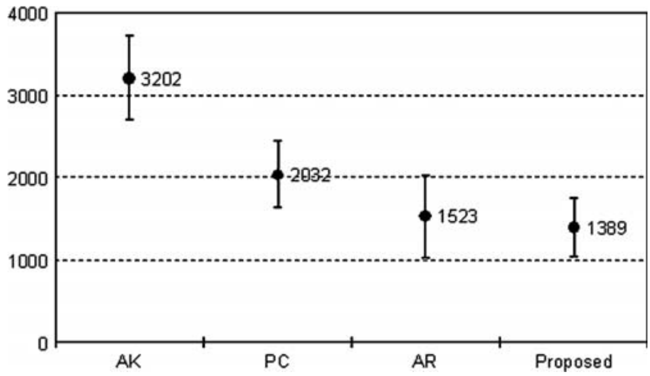

Figure 3. Comparison of computational complexities among different algorithms.

of 3000 networks are performed. Each link cost is generated randomly in each experiment and each network configuration.

5.3.1 *Convergence issue*

Table 4 shows convergence rates for different algorithms. Each convergence rate was calculated with 3000 different configurations of randomly generated link cost sets. In each experiment, an algorithm is considered to be convergent if there exists a link on the routing path: $(S \rightarrow D_1 \rightarrow D_2 \rightarrow D_3 \rightarrow D_4 \rightarrow D_5)$. As can be seen in Table 4, the proposed algorithm does not suffer significantly from degradation of convergence rates unlike the other algorithms. Ahn and Ramakrishna's algorithm suffers the most as the network size grows. This had been anticipated because Ahn and Ramakrishna's algorithm prioritises fast convergence without consideration of the routing path quality. Note that the parameter set, $\mu_6 = 100$ and $\mu_7 = 100$, for Ahn and Ramakrishna's algorithm used in our experiments are different from the one suggested in (Ahn and Ramakrishna 2002) and the convergence rate reported in Table 4 for Ahn and Ramakrishna's algorithm should have been much lower than that if the parameter set value, $\mu_6 = \mu_7 = 400$, suggested in Ahn and Ramakrishna (2002) were used.

5.3.2 *Computational complexity issue*

Figure 3 shows the computational complexities among the different algorithms. The computational complexity of each algorithm was calculated by the number of iterations after a link was obtained on the routing path: $(S \rightarrow D_1 \rightarrow D_2 \rightarrow D_3 \rightarrow D_4 \rightarrow D_5)$. That is, the computational complexity was calculated only when an algorithm is convergent. In Figure 3, the middle points and bars show the averages and standard deviations of iteration

Table 4. Summary of convergence rate (%) comparison.

	No. of nodes		
	Group 1 (12–25) (%)	Group 2 (26–40) (%)	Group 3 (41–50) (%)
Ali and Kamoun	99.2	98.8	42.0
Park and Choi	99.5	99.2	46.1
Ahn and Ramakrishna	98.8	76.2	13.3
Proposed	99.8	99.4	94.2

Table 5. Summary of routing optimality (%) comparison.

No. of nodes	Ali and Kamoun		Park and Choi		Ahn and Ramakrishna		Proposed	
	m (%)	σ (%)	*m* (%)	σ (%)	*m* (%)	σ (%)	*m* (%)	σ (%)
12–25	82.6	16.7	82.6	16.8	65.8	21.0	87.9	12.0
26–40	43.7	9.2	46.1	15.5	52.6	16.4	82.2	9.1
41–50	15.0	3.7	24.2	14.2	11.2	11.0	80.0	9.8

numbers over 3000 tries. The proposed algorithm shows very similar results with the Ahn and Ramakrishna's algorithm in terms of convergence speed. Note that the results were calculated only for convergent cases.

5.3.3 *Optimality issue*

Table 5 summarises results on the optimality comparison for each group of network sizes among the routing algorithms. In each case, the route optimality is calculated based on the ratio between the total cost of the route obtained by the algorithm of interest and the total cost of the route by Dijkstra's algorithm. As the routing path becomes shorter, the optimality is accordingly increased. As summarised in Table 5, the proposed algorithm's optimality does not suffer from sudden degradation of optimality as the number of nodes is increased in the network. Note that the optimality of Ahn and Ramakrishna's algorithm for the Group 3 networks (41–50 nodes) is 11.21% on average. This implies that this algorithm is essentially not applicable to practical problems.

Experiments in this paper are based on many different sizes of networks with random assignment of link costs. That is, the results shown in Tables 4 and 5 indicate that the

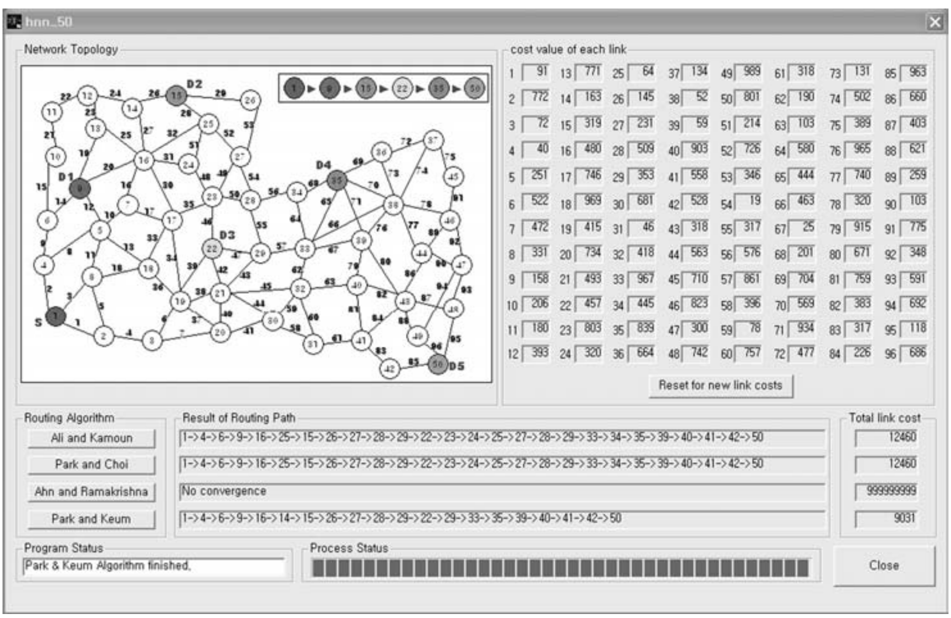

Figure 4. A screen image of the simulation package.

proposed algorithm is an effective tool for routing of communication networks with various sizes when convergence rate and optimality are considered.

An interactive simulation packages to demonstrate the proposed algorithm and other algorithms used in this paper for 20-, 40- and 50-node problems are available at the following web site: http://icrl.mju.ac.kr/NNSimulator/PK-HNN/index_e.html.

Readers can perform the experiments by downloading the simulation package. The package also includes all the source codes used for the experiments. Figure 4 shows a screen image of the simulation package. In this case, it has 50 nodes and 5 destinations with randomly chosen link costs. The simulation package can assign the link costs randomly by clicking a button. The total link cost from the routing path found by each algorithm is displayed on the screen.

6. Conclusions

A routing algorithm for communication networks is proposed in this paper. The proposed algorithm modifies the Lyapunov energy function in order to utilise all the information on the links connected with the node of interest. Simulations are performed extensively on 3000 networks with up to 50 nodes with randomly assigned link costs. The results show that the proposed algorithm in this paper exceeds the conventional algorithms both in convergence rate and route optimality.

Experiments show that the proposed routing algorithm yields acceptable results even when the network size is large. When the number of nodes is less than 20, the performances of all the algorithms are virtually the same. However, when the size of the network approaches 50 nodes, the simulation results reveal that the proposed algorithm gives on average at least a 94.23% convergence rate while Ahn and Ramakrishna's algorithm, the most recent algorithm presented on this subject, yields a convergence rate as low as 13.34%. Regarding the solution quality, the proposed algorithm shows 79.99% route optimality performance for networks with 41–50 nodes while other algorithms show 11.21–24.20% route optimality. This confirms that the proposed algorithm is an effective tool for the routing of communication networks when convergence rate and route optimality are considered.

Acknowledgements

This work was supported by the Korea Science and Engineering Foundation (KOSEF) grant funded by the Korean government (MOST)(Grant No.: R01-2007-000-20330-0).

Notes on contributors

Dong-Chul Park received his BS degree in Electronics Engineering in 1980 from Sogang University, Seoul, Korea, and his MS degree in Electrical and Electronics Engineering in 1982 from Korea Advanced Institute of Science and Technology, Seoul, Korea. His PhD degree in Electrical Engineering was received from the University of Washington, Seattle, in 1990, on System Identification Using Neural Networks.

During the period between 1990 and 1994, Dr Park was with the Department of Electrical and Computer Engineering at Florida International University, The State University of Florida at Miami. Since 1994, he has been with the Department of Information Control Engineering at Myong Ji University, Korea, where he holds the title of Professor. In 2000–2001, he was a Visiting Professor at the University of Washington, Seattle.

Dr Park is a pioneer in the area of electric load forecasting using NN. He was a committee member for the First International Forum on Applications of NN to Power Systems (Seattle, 1991). He was a member of the Editorial Board for *IEEE Transactions on Neural Networks* until 2000.

Dr Park has published over 60 papers including 30 archival journal papers in the areas of Neural Network algorithms and NN applications. His current research interest includes development of NN algorithms and their applications to various engineering problems including digital communication, image compression, speech recognition, and pattern recognition.

Kyo-Reen Keum received his BS and MS degrees in Control and Instrument Engineering from Myong Ji University, Korea, in 2002 and 2004, respectively. Since 2005, he has been with the Diesel Engine R & D Centre of STX Engine Co., and has participated in developing electric safety devices for diesel engines. Mr Keum's research interests include control systems using neural networks, system identification, routing algorithm, and pattern recognition.

References

Ahn, C.W. and Ramakrishna, R.S., 2002. Neural network based near-optimal routing algorithm. *Proceedings of international conference on neural information processing*, 4, 1771–1776.

Ahuja, R.K., Magnati, T.L., and Orlin, J.B., 1993. *Network flows theory, algorithms and applications*. Englewood Cliffs, NJ: Prentice Hall.

Ali, M.K. and Kamoun, F., 1993. Neural networks for shortest path computation and routing in computer networks. *IEEE transactions on neural networks*, 4 (6), 941–953.

Atwood, J.W., 2004. A classification of reliable multicast protocols. *IEEE network*, 18 (2), 24–34.

Chen, L. and Aihara, K., 1995. Chaotic simulated annealing by a neural net work model with transient chaos. *Neural networks*, 8 (6), 915–930.

Fatta, G., *et al.*, 2003. A genetic algorithm for the design of a fuzzy controller for active queue management. *IEEE transactions on systems, man, and cybernetics-part-C: applications and reviews*, 33 (3), 313–324.

Funabiki, N. and Takefuji, Y., 1993. A parallel algorithm for broadcast scheduling problems in packet radio networks. *IEEE transactions on communications*, 41 (6), 828–831.

Ganjam, A. and Zhang, H., 2005. Internet multicast video delivery. *Proceedings of the IEEE*, 93 (1), 159–170.

Hopfield, J.J. and Tank, D.W., 1986. Neural computations of decisions on optimization problem. *Biological cybernetics*, 52, 141–152.

Kocak, T., Seeber, J., and Terzioglu, H., 2003. Design and implementation of a random neural network routing engine. *IEEE transactions on neural networks*, 14 (5), 1128–1143.

Kompella, V.P., Pasquale, J.C. and Polyzos, G.C., 1993. Multicast routing for multimedia communication. *IEEE/ACM transactions on networking*, 1 (3), 286–292.

Kumar, K.B. and Jaffe, J.M., 1983. Routing to multiple destinations in computer networks. *IEEE transactions on communications*, 31 (3), 343–351.

Lin, X.L. and Ni, L.M., 1993. Multicast communication multicast in micro computer networks. *IEEE transaction on parallel and distributed systems*, 4 (10), 1105–1117.

Misra, S. and Oommen, B.J., 2005. Dynamic algorithms for the shortest path routing problem: learning automata-based solutions. *IEEE transactions on systems, man and cybernetics, part B*, 35 (6), 1179–1192.

Park, D.C. and Choi, S.E., 1998. A neural network based multi-destination routing algorithm for communication network. *In*: *Proceedings of joint conference on neural networks*, 1673–1678.

Rauch, H.E. and Winarske, T., 1988. Neural networks for routing communication traffic. *IEEE control systems magazine*, 8 (2), 26–30.

Stalling, W., 1998. *High-speed networks: TCP/IP and ATM design principles*. Englewood Cliffs, NJ: Prentice Hall.

Tank, D.W. and Hopfield, J.J., 1986. Simple neural optimization networks: an A/D converter, signal decision circuit, and a linear programming circuit. *IEEE transactions on circuits systems*, CAS-33 (5), 533–541.

Wang, L., Li, S. and Fu, X., 2004. A noisy chaotic neural network for solving combinatorial optimization problems: stochastic chaotic simulated annealing. *IEEE transactions on systems, man, and cybernetics-part B: cybernetics*, 34 (5), 2119–2124.

Wu, Q., *et al.*, 2004. On computing mobile agent routes for data fusion in distributed sensor networks. *IEEE transactions on knowledge and data engineering*, 16 (6), 740–753.

Yip, P.P.C. and Pao, Y.-H., 1995. Combinatorial optimization with use of guided evolutionary simulated annealing. *IEEE transactions on neural networks*, 6 (2), 290–295.

Zhang, L. and Thomopoulos, S.C.A., 1989. Neural network implementation of the shortest path algorithm for traffic routing in communication networks. *In*: *Proceedings of the international joint conference on neural networks*.