'94 KACC (1994. 10. 17 ~ 20)

The Shortest Path Finding Algorithm Using Neural Network

Sun-Gi Hong°, Taeduck Ohm, Il-Kwon Jeong, and Ju-Jang Lee

Department of Electrical Engineering
Korea Advanced Institute of Science and Technology
373-1 Kusong-dong, Yusong-gu, Taejon 305-701, Korea
FAX: +82-42-869-3410 E-mail: sghong@odyssey.kaist.ac.kr

Abstract - Recently neural networks have been proposed as new computational tools for solving constrained optimization problems because of its computational power. In this paper, the shortest path finding algorithm is proposed by using a Hopfield type neural network. In order to design a Hopfield type neural network, an energy function must be defined at first. To obtain this energy function, the concept of a vector-represented network is introduced to describe the connected path. Through computer simulations, it will be shown that the proposed algorithm works very well in many cases. The local minima problem of a Hopfield type neural network is discussed.

I. INTRODUCTION

Generally, to find the shortest path on the network model is involved in many problems as subproblem. For this reason the shortest path finding problem has been the subject of intensive research for many years. According to these efforts, many algorithms have been proposed, for example, Dijkstra's algorithm and so on [1]. However, these algorithms have the shortcomings that the computational burden is getting larger as the number of nodes increases on the network model. Recently neural networks have been proposed as new computational tools for solving constrained optimization problems. Since the shortest path finding problem can be regarded as a constrained optimization problem, we propose the shortest path finding algorithm using a Hopfield type neural network in this paper. To design a Hopfield type neural network, an appropriate energy function must be defined at first so that the shortest path can be decoded from the final state of the neural network. To obtain this energy function, we assume that the network model is given in the form of vectors between the nodes without loss of generality. Using this vector-represented network model, the constraints for the shortest path is presented in this paper. After finding the dynamic equation of each neuron based on the gradient descent method, the electrical circuit can be implemented by using the obtained parameters as shown in Fig. 1. In this paper, through the numerical analysis, it will be shown that the proposed algorithm can solve the shortest path problems.

Generally, a Hopfield type neural network has the local minima problem because the dynamics of each neuron is obtained, based on the gradient descent method. In order to overcome these shortcomings, a surface of the energy function must be designed gracefully by proper tuning the weighted coefficients of the energy function. This tuning scheme has been proposed by Mustafa K. Mehmet Ali and Faouzi Kamoun. This will be discussed in this paper [2].

This paper is organized as follows:

The next section gives a brief review of the application of a Hopfield type neural network to solve discrete combinatorial optimization problems. In section III, a neural network architecture based on the Hopfield model is proposed for solving the shortest path finding problem. The main steps involved in the design of the proposed model are described. In section IV, it will be shown that the proposed model works well in many cases, through computer simulations. Finally we draw some conclusions in section V.

II. HOPFIELD TYPE NEURAL NETWORK

The use of neural networks to solve constrained optimization problems was initiated by Hopfield and Tank [3]. A Hopfield type neural network has been applied to solve discrete combinatorial optimization problems, (for example, the Traveling Salesman Problem (TSP)) because of its computational power. A Hopfield type neural network can be implemented by the electrical circuit as shown in Fig. 1. This circuit is based on the model of a biological neural network. Each neuron is modeled as a nonlinear device (operational amplifier) with a saturated monotonic increasing function relating the output V_i of the i th neuron to its input U_i . The example of a typical characteristic function of the nonlinear device is as follows:

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

Using the above equation, the output V_i is allowed to take on any value between 0 and 1. The synaptic connections between neurons are represented as resistive components which can be fully described through the matrix $T=[T_{ij}]$, also known as the connection matrix of the network. As shown in Fig. 1, each neuron is biased by an external current I_i, which could represent actual data provided by user to the neural network. The dynamics of a Hopfield type neural network is described as follows:

$$\frac{dU_i}{dt} = -\frac{U_i}{\tau_i} - \sum_{j=1}^{N} T_{ij} V_j - I_i \tag{2}$$

where τ_i is the circuit's time constant.

Hopfield [3] has shown that if the connection matrix T is symmetric and the gains of the nonlinear devices are sufficiently high (i.e. $\lambda_i \rightarrow \infty$) then the dynamics of the neurons moves to the stable state in the gradient descent direction of the quadratic energy function (3).

$$E = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} T_{ij} V_i V_j + \sum_{i=1}^{N} I_i V_i \tag{3}$$

Hopfield [3] has also shown that if the state of the space over which the circuit operates is the interior of the N-dimensional hypercube defined by V_i = 0 or 1, i.e. when $\lambda_i = \infty$, and the diagonal elements T_{ii} are 0, then the stable states of the network correspond to those locations in the discrete space consisting of the 2^N corners of this hypercube which minimize E (3). In terms of the generalized energy function , the dynamics of the i th neuron is described as follows:

$$\frac{dU_i}{dt} = -\frac{U_i}{\tau_i} - \frac{\partial E}{\partial V_i} \tag{4}$$

Therefore, to apply a Hopfield type neural network to the shortest finding problem the energy function is required.

III. THE SHORTEST PATH FINDING ALGORITHM

The vector-represented network model can be defined as a directed graph $G=(\overline{N}, \overline{A})$, with N nodes and l arcs. Corresponding to each arc (i, j) there is a nonnegative number C_{ij} which represents the cost value (length, transit time, etc.) from node i to node j. We assign the number to each node from 0 to N-1. Especially a source and a destination are assigned 0 and N-1 respectively. Let a dirçcted path P^{0N-1} be an ordered sequence of nodes connecting 0 to N-1 [2].

$$P^{0N-1} = (0,\ i,\ j, \cdots ;\ k,\ N-1) \tag{5}$$

Then the total cost of this path will be equal to $C_{0i}+C_{ij}+ \cdots + C_{kN-1}$. Hence, the shortest path can be found among all possible directed paths.

To formulate the shortest path finding problem in terms of a Hopfield type neural network, an appropriate energy function must be defined at first. To achieve this, the proposed Hopfield type neural network is organized in $N(N$-$1)/2$ neurons. Each neuron is described by double indices (i, j), where the subscript i and j denote node numbers. A neuron which represents (i, j) is characterized by its output V_{ij}, and is defined as follows:

$$V_{ij} = \begin{cases} 1 & \text{if } \textit{the arc from node i to node} \\ & \textit{j is in the minimum cost path} \\ 0 & \textit{otherwise} \end{cases} \tag{6}$$

To eliminate all nonexistent arcs from the solution the following parameter is introduced.

$$\rho_{ij} = \begin{cases} 1 & \text{if } \textit{the arc from node i to node} \\ & \textit{j does not exist} \\ 0 & \textit{otherwise} \end{cases} \tag{7}$$

To express the connecting condition for the shortest path, which means the feasible path must be perfectly connected through the nodes, the following assumption is required.

Assumption 1

A directed graph $G=(\overline{N}, \overline{A})$ is represented as the form of vectors between nodes.: the network model is given in the form of the vector- represented network.

Since each node of a directed graph has the coordinate for X-Y plane, the vector between the nodes can be easily obtained. Let (x_i, y_i) be the coordinate of node i, then the vector between node i and node j is defined as follows:

$$\overrightarrow{d_{ij}} = \begin{cases} (x_j, y_j) - (x_i, y_i) & \text{if } \textit{the arc from node } i \textit{ to node } j \textit{ exists} \\ \vec{0} & \textit{otherwise} \end{cases} \qquad (8)$$

To describe the vector-represented network, the directional factor α_{ij} is also defined as follows:

$$\alpha_{ij} = \begin{cases} 1 & \text{if } \textit{there is a vector from node } i \textit{ to node } j \\ -1 & \textit{otherwise} \end{cases} \qquad (9)$$

Using the vector $\alpha_{ij}\overrightarrow{d_{ij}}$, a vector-represented network can be described easily as shown in Fig. 2. The connecting condition is represented as the following equation using the parameters defined in equation (8), (9).

$$\sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} \alpha_{ij}\overrightarrow{d_{ij}}V_{ij} = \overrightarrow{d_{0N-1}} \qquad (10)$$

Equation (10) is not "if and only if" condition for representing the connected path from a source to a destination because the vector $\overrightarrow{d_{ij}}$ is not unique. To obtain "if and only if" condition, the following function is introduced.

$$F(i) = \sum_{k=0}^{i-1} V_{ki} + \sum_{k=i+1}^{N-1} V_{ik} \qquad (11)$$

The value of the function $F(i)$ means the total sum of the connected paths at node i when the dynamics of neurons reaches the stable state. If the following equation is satisfied, then the connected path between a source and a destination is generated.

$$\begin{cases} F(i)F(j)V_{ij} = 2 & \text{if } \textit{node } i \textit{ is a source or node } j \textit{ is a destination} \\ F(i)F(j)V_{ij} = 4 & \textit{otherwise} \end{cases} \qquad (12)$$

where $(i, j) \in [0, N-1]/i < j$.

The example of equation (12) is depicted in Fig. 3. Since a source node and a destination node must have the only one connected path respectively, the following condition is also needed in addition to equation (12).

$$\sum_{j=1}^{N-1} V_{0j} = 1 \quad \textit{at a source node} \qquad (13)$$

$$\sum_{i=0}^{N-2} V_{iN-1} = 1 \quad \textit{at a destination} \qquad (14)$$

Therefore, according to equation (10), (12), (13), (14) the energy function can be formulated as follows:

(i) If node i is the starting point or node j is the destination

$$\begin{aligned} E = {} & \mu_1 \sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} C_{ij}V_{ij} + \mu_2 \sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} \rho_{ij}V_{ij} \\ & + \mu_3 \left\| \sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} \alpha_{ij}\overrightarrow{d_{ij}}V_{ij} - \overrightarrow{d_{0N-1}} \right\|^2 \\ & + \mu_4 \Big(\sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} F(i)F(j)V_{ij} - 2 \Big)^2 \\ & + \mu_5 \Big(\sum_{j=1}^{N-1} V_{0j} - 1 \Big)^2 + \mu_6 \Big(\sum_{i=0}^{N-2} V_{iN-1} - 1 \Big)^2 \\ & + \mu_7 \sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} V_{ij}(1 - V_{ij}) \end{aligned} \qquad (15)$$

(ii) otherwise

$$\begin{aligned} E = {} & \mu_1 \sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} C_{ij}V_{ij} + \mu_2 \sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} \rho_{ij}V_{ij} \\ & + \mu_3 \left\| \sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} \alpha_{ij}\overrightarrow{d_{ij}}V_{ij} - \overrightarrow{d_{0N-1}} \right\|^2 \\ & + \mu_4 \Big(\sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} F(i)F(j)V_{ij} - 4 \Big)^2 \\ & + \mu_5 \Big(\sum_{j=1}^{N-1} V_{0j} - 1 \Big)^2 + \mu_6 \Big(\sum_{i=0}^{N-2} V_{iN-1} - 1 \Big)^2 \\ & + \mu_7 \sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} V_{ij}(1 - V_{ij}) \end{aligned} \qquad (16)$$

where $\|\cdot\|$ is the Euclidean norm.

In equation (15) and (16) the μ_1 term minimizes the total cost of a path by taking into account the cost of existing arcs. The μ_2 term prevents nonexistent arcs from being included in the chosen path. The μ_3, μ_4, μ_5, and μ_6 terms are zero if the shortest path is connected from a source node and a destination through nodes. The μ_7 term pushes the state of a Hopfield type neural network to converge to one of the $2^{N(N-1)/2}$ corners of the hypercube, defined by $V_{ij} \in [0, 1]$ [2].

Using equation (1) and (4) the dynamics of a Hopfield type neural network can be obtained as follows:

$$V_{ij} = g_{ij}(U_{ij}) = \frac{1}{1 + e^{-\lambda_{ij}U_{ij}}} \qquad (17)$$
$$\forall (i, j) \in [0, N-1]/i < j$$

(i) If node i is a source or node j is a destination

$$\begin{aligned} \frac{dU_{ij}}{dt} = {} & \frac{-U_{ij}}{\tau} - \mu_1 C_{ij} - \mu_2 \rho_{ij} - 2\mu_3 \{ (\sum_{i=0}^{N-1} \\ & \sum_{j=i+1}^{N-1} \alpha_{ij}d_{xij}V_{ij} - d_{x0N-1}) \alpha_{ij}d_{xij} + (\\ & \sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1} \alpha_{ij}d_{yij}V_{ij} - d_{y0N-1}) \alpha_{ij}d_{yij} \} \end{aligned}$$

$$
\begin{aligned}
&-2\mu_4\Big(\sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1}F(i)F(j)V_{ij}-2\Big)\{\\
&F(j)F(j)+F(j)V_{ij}+F(i)V_{ij}\}-2\\
&\mu_5\Big(\sum_{j=1}^{N-1}V_{0j}-1\Big)\delta_{0i}-2\mu_6\Big(\sum_{i=0}^{N-2}V_{iN-1}\\
&-1\Big)\delta_{jN-1}-\mu_7(1-2V_{ij})\\
&(i,\ j)\in[0,\ N-1]/i<j
\end{aligned}
\tag{18}
$$

(ii) otherwise

$$
\begin{aligned}
\frac{dU_{ij}}{dt} = &\frac{-U_{ij}}{\tau}-\mu_1C_{ij}-\mu_2\rho_{ij}-2\mu_3\{\Big(\sum_{i=0}^{N-1}\\
&\sum_{j=i+1}^{N-1}\alpha_{ij}d_{xij}V_{ij}-d_{x0N-1}\Big)\alpha_{ij}d_{xij}+\Big(\\
&\sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1}\alpha_{ij}d_{yij}V_{ij}-d_{y0N-1}\Big)\alpha_{ij}d_{yij}\}\\
&-2\mu_4\Big(\sum_{i=0}^{N-1}\sum_{j=i+1}^{N-1}F(i)F(j)V_{ij}-4\Big)\{\\
&F(i)F(j)+F(j)V_{ij}+F(i)V_{ij}\}-2\\
&\mu_5\Big(\sum_{j=1}^{N-1}V_{0j}-1\Big)\delta_{0i}-2\mu_6\Big(\sum_{i=0}^{N-2}V_{iN-1}\\
&-1\Big)\delta_{jN-1}-\mu_7(1-2V_{ij})\\
&(i,\ j)\in[0,\ N-1]/i<j
\end{aligned}
\tag{19}
$$

where $\overrightarrow{d_{ij}}$ is equal to $(d_{xij},\ d_{yij})$ and the function δ_{ij} is defined as follows:

$$
\delta_{ij}=\begin{cases}1 & \text{if } i=j\\ 0 & otherwise\end{cases}
\tag{20}
$$

Using equation (18) and (19) the connection matrix and the bias current can be obtained. The shortest path corresponds to the minimal state of the energy function which is defined in equation (15) and (16). Generally, a Hopfield type neural network has the local minima problem because the dynamics of neurons is based on the gradient descent method. To solve this local minima problem the energy surface is designed gracefully by proper tuning of the μ_i's weighting coefficients. In order to ensure that the energy function has only one low point and hence provides a graceful descent along the energy surface, the following equation is required [2].

$$
\frac{\partial^2 E}{\partial V_{ij}^2}>0 \qquad \forall(i,\ j)
\tag{21}
$$

Using equation (21) the μ_i's weighting coefficients can be obtained. The simulation results for the proposed algorithm will be presented in the next section.

IV. SIMULATION

To show the effectiveness of the proposed algorithm, the vector-represented network shown in Fig. 4 is used. Let the cost value C_{ij} be the distance between node i and node j, i.e. $|\alpha_{ij}\overrightarrow{d_{ij}}|$ where the coordinates of nodes are given in Table II. Then we can easily find the fact that a directed path $P^{04}=(0,\ 2,\ 4)$ is the shortest path. To obtain the parameters for a Hopfield type neural network, the coefficients of the energy function is chosen as shown in Table I. In equation (17), in order to allow the dynamics of neurons to wander freely in their state space, the value of λ_{ij} is chosen as 1 and for simplicity it is assumed that $g_{ij}=g$, all independent of the subscript $(i,\ j)$. Originally, a Hopfield type neural network is implemented by the electrical circuit in the form of Fig. 1. However, as a matter of convenience the time evolution of the state of neurons can be simulated by numerically solving equation (18) and (19). This corresponds to solving a system of $N(N-1)/2$ nonlinear differential equations, where the variables are the neuron's output voltages V_{ij}'s. Accordingly, the simulation consists of observing and updating the neuron's output voltages at incremental time steps δt, where δt is *0.01 ms* in this paper. In addition, the time constant τ of each neuron is set to 1 without any loss of generality in equation (18) and (19). The initial value of U_{ij} is assigned between 0 and 1 randomly. The simulation result for Fig. 4 is illustrated in Table III. In Table III, the values of 0, 1, $\cdots$, 4 in rows mean the number of node i and the values in columns mean the number of node j similarly.

As compared with Fig. 4, the vector- represented network shown in Fig. 5 has a different configuration; a directed path $P^{04}=(0,\ 1,\ 4)$ is the shortest path, where the coordinates of nodes are identical with the case in Fig. 4 and the cost value C_{ij} is the distance between node i and node j. The simulation result for Fig. 5 is shown in Table IV, using the above parameters.

These results show that the proposed algorithm operates well. When the number of nodes becomes larger, the shortest path can be obtained through the proper tuning of the weight coefficient μ_i.

V. CONCLUSION

The shortest path finding algorithm on the vector-represented network was proposed, using a Hopfield type neural network. To design a Hopfield type neural network, the energy function, which represents the condition for the shortest path, was defined. Using this energy function, the parameters for a Hopfield type neural network could be obtained. Through the simulation, the effectiveness of the proposed algorithm was presented. The local minima problem for a Hopfield type neural network was discussed. It was shown that the proposed algorithm could be applied to many cases by proper tuning the coefficients of the energy function.

REFERENCES

[1] Robert Sedgewick, "*Algorithms*," Addison Wesley, 1984.

[2] Mustafa K. Mehmet Ali and Faouzi Kamoun, "Neural Networks for Shortest Path Computa- tion and Routing in Computer Networks," *IEEE Trans. on Neural Networks*, Vol. 4, No. 6, Nov., 1993.

[3] David W. Tank and John J. Hopfield, "Simple "Neural" Optimization Networks : An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit," *IEEE Trans. on Circuits and Systems*, Vol. CAS-33, No. 5, May 1986.

Table. I. Coefficients of energy function

μ_1	μ_2	μ_3	μ_4	μ_5	μ_6	μ_7
800	2000	5000	4000	4000	4000	500

Table. II. Coordinates of nodes

0	1	2	3	4
(1, 6)	(4, 10)	(6, 4)	(10, 10)	(13, 7)

Table. III. Result for Fig. 4

	0	1	2	3	4
0	X	0	0	0.5	0.4
1	X	X	0	0	0
2	X	X	X	0	0
3	X	X	X	X	0
4	X	X	X	X	X

(Initial Condition)

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

(Iterations = 1000)

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

(Iterations = 2000)

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

(Iterations = 3000)

Table. IV. Result for Fig. 5

	0	1	2	3	4
0	X	0	0.5	0.5	0.4
1	X	X	0.1	0	0
2	X	X	X	0	0
3	X	X	X	X	1
4	X	X	X	X	X

(Initial Condition)

	0	1	2	3	4
0	X	0.9	0	0	0.1
1	X	X	0	0.8	1
2	X	X	X	0	0.1
3	X	X	X	X	0
4	X	X	X	X	X

(Iterations = 1000)

	0	1	2	3	4
0	X	0.9	0	0	0
1	X	X	0	0.1	1
2	X	X	X	0	0
3	X	X	X	X	0
4	X	X	X	X	X

(Iterations = 2000)

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

(Iterations = 3000)

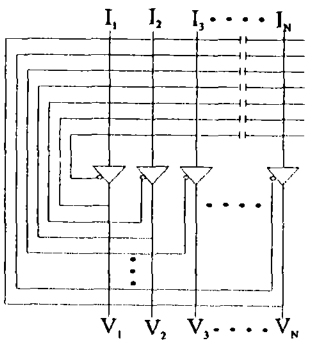

Fig. 1. Hopfield type neural network circuit

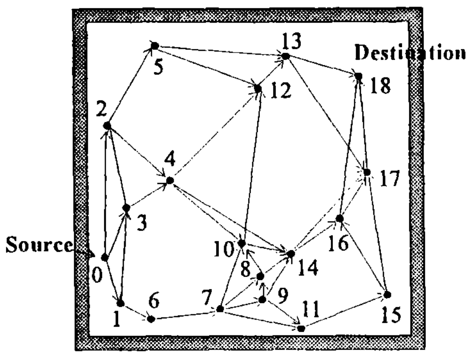

Fig. 2. Vector-represented network

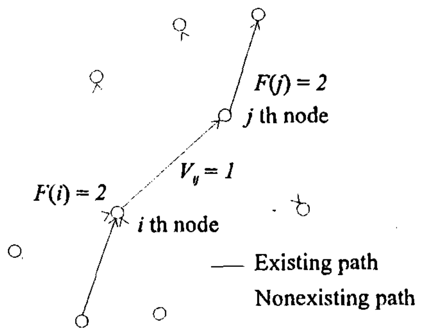

Fig. 3. When $F(i)F(j)V_{ij}=4$

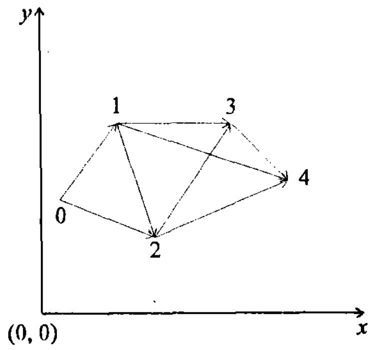

Fig. 4. Vector-represented network (0-2-4)

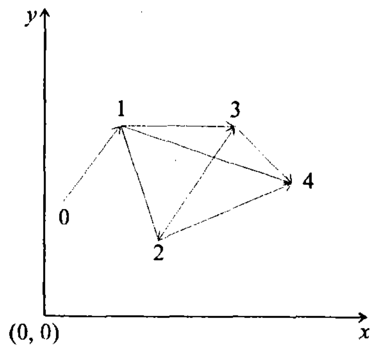

Fig. 5. Vector-represented network (0-1-4)