2009 International Conference on Communications and Mobile Computing

Solving the Shortest Path Routing Problem Using Noisy Hopfield Neural Networks

Wen Liu and Lipo Wang
School of Electrical and Electronic Engineering
Nanyang Technological University
Block S1, 50 Nanyang Avenue, Singapore 639798

Abstract

To improve neural network algorithms for the shortest path routing problem (SPRP), we propose a solution using a noisy Hopfield neural network (NHNN), i.e., by adding decaying stochastic noise to the continuous Hopfield neural network (HNN). We also modify the energy function for the SPRP. Simulation results show that our approach achieves better route optimality compared to other algorithms that employ the HNN.

1 Introduction

For the shortest path routing problem (SPRP) [2–4, 9], Ali and Kamoun [3] formulated the problem onto a Hopfield neural network (HNN) and simulated the algorithm on a 5-node network with satisfactory results. Park and Choi [9] introduced a new term into the energy function proposed by Ali and Kamoun [3] so that the flow is one-directional to the destination, but at the same time this term made connection weights of the neural network dependent on the topology of the communication network. Araujo et al [4] proposed a new formulation and extended the HNN to a two-layer architecture. They enhanced reliability but impaired the ability to reach optima. Ahn et al [2] added two new terms into Park-Choi energy function to avoid possible loops. Although the simulation showed improvements over other approaches, its energy function has seven terms and it is hard to tune the associated seven parameters.

A number of authors have explored adding noise to the recurrent neural network in order to improve network performance. Jim et al [7] presented a detailed investigation on the effects of injecting synaptic noise to recurrent neural networks and analyzed several injection methods. They added different kinds of noise onto synaptic connection weights of the neural network and showed several benefits, such as improvements on generalization and convergence. Das and Olurotimi [5, 8] analyzed noisy recurrent neural networks in both continuous and discrete cases. They showed that knowledge about the behavior of neural networks in the presence of noise would be helpful in design and parameter selection. Wang et al [10, 11] proposed a noisy chaotic neural network and successfully applied the new model to the traveling salesman problem and the channel assignment problem. In this letter we use a noisy Hopfield neural network (NHNN) to improve HNN's ability to reach the global optimal solution, while retaining the merits of Ali and Kamoun's model, i.e., model simplicity and the ability to adapt to changes in the communication network topology, the source or the destination node, and link costs. This analysis is necessary as thermal noise is inherent in analog implementations, especially in very large scale integration (VLSI) implementations.

2 The shortest path routing problem

2.1 Problem formulation

The SPRP aims to find a path in a communication network from source node s to destination node d such that the total cost on the path is minimum. Here, we use the formulation proposed in [3].

Considering a communication network with n nodes, the neural network is arranged on an $n \times n$ matrix, with all diagonal elements removed and every element of the matrix is treated as a neuron. The neuron of row x and column i describes the link from node x to node i in the communication network. P_{xi} characterizes the connection topology of the communication network: if the arc from node x to node i does not exist, $P_{xi} = 1$; otherwise, $P_{xi} = 0$.

The routing solution is described by the final output V_{xi} of neuron (x, i) as follows:

$$V_{xi} = \begin{cases} 1, & \text{if the arc from node } x \text{ to node } i \text{ is on the final path;} \\ 0, & \text{otherwise.} \end{cases}$$

978-0-7695-3501-2/09 $25.00 © 2009 IEEE
DOI 10.1109/CMC.2009.366

The cost of an arc from node x to node i is denoted by C_{xi}, which is a real non-negative number. For non-existing arcs, $C_{xi} = 0$.

2.2 Energy function

We modify the energy function based on the one proposed by [3] and [9].

$$E = \frac{\mu_1}{2}\sum_{x=1}^{n}\sum_{i=1,i\neq x}^{n} C_{xi}V_{xi} + \frac{\mu_2}{2}[\sum_{x=1}^{n}\left(\sum_{i=1,i\neq x}^{n} V_{xi} - \sum_{i=1,i\neq x}^{n} V_{ix} - \gamma_x\right)^2 \quad (1)$$

$$+ \sum_{x=1}^{n}\sum_{i=1,i\neq x}^{n} P_{xi}V_{xi}] \quad (2)$$

where $\gamma_x = 1$, if $x = s$; $\gamma_x = -1$, if $x = d$; otherwise, $\gamma_x = 0$. $\{\mu_i, i = 1, 2\}$ are the weighting constants. The μ_1 term aims at minimizing the cost on the route. The μ_2 term is the constraints with two parts. The first part is used to satisfy the requirement that for each single node except the source and the destination, the number of incoming links is equal to the number of outgoing links. The second part penalizes neurons that represent non-existing links in the communication network. There are only two parameters needed to be balanced in this model, which makes tuning of parameters more efficient.

We have discarded the fourth term of the energy function used in [3] which forces the outputs of the neural network to approach either 0 or 1. We feel that this term is not necessary as other constraint terms can achieve the same objective. The third and fifth terms in [3] are combined here as in [9], and furthermore, we have combined all the constraint terms so that there are only two parameters in the energy function to tune.

3 A noisy Hopfield neural network

A noisy Hopfield neural network (NHNN) is obtained by adding decaying stochastic noise into the continuous HNN [6]:

$$\frac{dU_{xi}(t)}{dt} = -\frac{1}{\tau}U_{xi}(t) \quad (3)$$

$$+ \sum_{y=1}^{N}\sum_{j=1,j\neq y}^{N} w_{(yj->xi)}V_{yj}(t) \quad (4)$$

$$+ I_{xi} + n(t) \quad (5)$$

$$V_{xi} = f_{xi}(U_{xi}) \quad (6)$$

$$\frac{1}{1 + e^{-U_{xi}/\epsilon_{xi}}} \quad (7)$$

where τ is a circuit time constant. I_{xi} is the input bias of neuron (x, i). ϵ_{xi} ($\epsilon \geq 0$) is the steepness parameter of the neuronal output function. $n(t)$ is the random noise.

$$-\frac{\partial E}{\partial V_{xi}} = \sum_{y=1}^{N}\sum_{j=1,j\neq y}^{N} w_{(yj->xi)}V_{yj}(t) + I_{xi} \quad (8)$$

In this paper, we consider random noise $n(t)$ with uniform and Gaussian distribution. For uniform distribution, $n(t)$ is uniformly chosen from the range $(-A[n], A[n])$, where $A[n]$ is the noise amplitude. For exponential decaying, $A[n(t+1)] = (1-\beta)A[n(t)]$, β ($0 \leq \beta \leq 1$) is the damping factor. For linear decaying, $A[n(t)] = A[n(0)] - \kappa t$, where κ is the slope of linear decaying. As for Gaussian noise, $n(t)$ follows Gaussian distribution with mean 0 and initial variance $\sigma^2(0) = 1$. We reduce the variance by $\sigma^2(t+1) = (1-\beta)\sigma^2(t)$. Hence as time goes on, the chance to have a large noise value decreases.

The connection weights and bias terms are derived by substituting the energy function in Eqn. 8 with Eqn. 1:

$$w_{(yj->xi)} = -\mu_2[\delta_{xy} - \delta_{xj} + \delta_{ij} - \delta_{iy}] \quad (9)$$

$$I_{xi} = -\frac{\mu_1}{2}C_{xi}(1 - \delta_{xm}\delta_{is}) + \frac{\mu_2}{2}\delta_{xm}\delta_{is} \quad (10)$$

$$- \frac{\mu_2}{2}P_{xi}(1 - \delta_{xm}\delta_{is}) \quad (11)$$

where δ_{ij} is the Kronecker delta. No terms in the connection matrix depend on the cost of links or the topology of the communication network. Thus the connection matrix is independent of any changes in the communication network. The cost term is mapped into the bias. The benefit is immense because there is no need to adjust internal parameters of the neural network to adapt to changes in the environment, therefore the hardware implement of the neural network is a general one [3].

4 Simulation Results

Firstly, we run the NHNN with exponentially decaying uniform noise on 10000 randomly generated 20-node network topologies with a Linux cluster (16-node dual Xeon 3.06 GHz, Intel IA32). The cost of an arc from node x to node i , i.e., C_{xi}, is randomly generated and normalized.

As in [1], the parameters in our model are chosen as follows: $\epsilon_{xi} = \epsilon = 1$, independent of neuron location (x, i). $\tau = 1$, $\Delta t = 10^{-4}$, and $\beta = 0.001$. Initial inputs of the neural network $U_{xi}(0)$ are randomly generated between $[-0.001, 0.001]$. At the end of each iteration, we set each neuron on or off according to its output value. If $V_{xi} \geq 0.5$, the neuron is on, i.e., $V_{xi} = 1$, which means the link from x to i is chosen in the final optimal tree, and vice versa.

We compare the performance of NHNNs with different initial noise amplitudes (from 0.0 to 5.0) in Table 1. According to [2,3] and our experience on the routing problem,

Table 1. Performance comparison of different noise levels.

Noise level	0.0	0.5	1.0	2.0	3.0	5.0
Route opt %	71.46	82.64	89.42	92.73	90.89	84.25

Table 2. Performance comparison of different μ_1 with μ_2 fixed at 2500

μ_1	100	150	200	250	300
Route opt %	88.02	94.49	95.32	93.68	92.73

the weighting constants are initially set to $\mu_1 = 300$ and $\mu_2 = 2500$. Here, the route optimal rate denotes the ratio at which the algorithm reached the optimal solution in 10000 runs, in comparison with Dijkstra's results. Adding noise helps the neural network reaching the global optima: without noise, the optimal rate is 71.46%; when the initial noise amplitude $A[n(0)] = 2.0$, the optimal rate is 92.73%, with more than 20% improvement. However, if the noise amplitude is too high, the route optimal rate starts to decrease. The algorithm fails to find a valid solution if the algorithm does not converge to the optimal solution, as we are using soft constraints, where penalty terms are added to the objective function directly.

We also investigate the sensitivity of the model to the weighting coefficients in Table 2. According to [1, 9], we varies μ_1 from 100 to 300 while fixing μ_2 at 2500. The initial noise amplitude is fixed at $A[n(0)] = 2.0$. The setting of those weighting coefficients is problem-specific and is carried out by trial-and-error. We found that when μ_2 is fixed at 2500 the model performs the best at $\mu_1 = 200$ and the route optimal rate reaches 95.32%. As there are only two weighting constants, it is not difficult to find the proper weighting constants by trial-and-error.

Similarly, we also analyze the effect of weighting coefficient μ_2 in Table 3 by varying μ_2 from 1500 to 3500 while fixing μ_1 at 200. From Tables 2 and 3, we can see that the performance of the NHNN is not sensitive to the setting of connection weights. Therefore the tuning of these two parameters for a good performance is not difficult.

We further investigate NHNNs for the SPRP on networks with 30, 50, and 80 nodes. The Performance of the proposed NHNN and those of Park and Choi [9] and Ahn [1, 2] are compared in Table 4. The NHNN reaches global optima more effectively.

To know more about the effect of the additive noise in the HNN, for the uniformly distributed noise, we have done a comparison of performance brought by noise with different decaying laws (exponential and linear) and different decaying rates. Results are shown in Table 5. μ_1 and μ_2 are set to 200 and 3000, respectively. It is evident that exponential decay works better than linear decay in terms of the probability to reach optima and parameter sensitivity.

Also, we simulate NHNNs using random noise with Gaussian distribution. When using Gaussian noise, we gradually decrease the variance of the distribution, which is different from the case with uniform noise distribution. Hence the probability to have a small value noise increases as iteration goes on. As shown in Table 6 and Table 5, Gaussian noise is comparable to uniform noise in performances.

Table 3. Performance comparison of different μ_2 with μ_1 fixed at 200.

μ_2	1500	2000	2500	3000	3500
Route opt %	89.12	92.81	95.32	96.18	95.04

Table 4. Performance comparison between Park and Choi, Ahn and our proposed approach.

Network size	20	30	50	80
Park and Choi [9]	72.14	67.35	50.18	41.12
Ahn [2]	78.72	71.16	62.40	50.77
NHNN	96.18	88.37	79.91	62.90

5 Conclusion

We proposed a solution using *noisy Hopfield neural networks* for the shortest path routing problem. We simplify

Table 5. Performance (Route opt %) comparison of different noise amplitude decreasing laws and rates.

Network size			20	30	50	80
uniform	exponential	$\beta = 0.01$	91.17	83.87	75.42	54.76
		$\beta = 0.001$	96.18	88.37	79.91	62.90
		$\beta = 0.0001$	71.73	63.90	54.93	48.55
	linear	$\kappa = 0.0001$	69.08	58.74	43.79	36.77
		$\kappa = 0.0002$	86.35	79.70	62.41	51.35
		$\kappa = 0.0005$	72.27	66.38	51.30	42.76

Table 6. Performance (Route opt %) comparison of different kinds Gaussian additive noise.

Network size	20	30	50	80
$\beta = 0.1$	82.08	72.75	57.30	51.26
$\beta = 0.01$	88.16	83.52	64.85	59.47
$\beta = 0.001$	90.08	87.12	67.10	62.80
$\beta = 0.0001$	62.52	52.06	34.18	30.25

the energy function for the problem so there are only two weighting constants to tune. We also evaluated effects of different noise distributions and decaying rates, as well as the sensitivity of the neural network model to the values of weighting constants. Adding the decaying noise enhances the route optimal rate for the SPRP, compared with other heuristics based on the HNN.

Acknowledgment

We sincerely thank the editors and reviewers for carefully reading the paper and for many constructive comments that have helped to significantly improve the paper.

References

[1] C. W. Ahn, R. S. Ramakrishna, A genetic algorithm for shortest path routing problem and the sizing of populations, IEEE Transactions on Evolutionar computation 6 (6) (2002) 566–579.

[2] C. W. Ahn, R. S. Ramakrishna, C. G. Kang, I. C. Choi, Shortest path routing algorithm using hopfield neural network, Electronics Letters 37 (19) (2001) 1176–1178.

[3] M. K. M. Ali, F. Kamoun, Neural networks for shortest path computation and routing in computer networks, IEEE Transactions on Neural Networks 4 (6) (1993) 941–954.

[4] F. Araujo, B. Ribeiro, L. Rodrigues, A neural network for shortest path computation, IEEE Transactions on Neural Networks 12 (5) (2001) 1067–1073.

[5] S. Das, O. Olurotimi, Noisy recurrent neural network: The continuous-time case, IEEE Transactions on neural networks 9 (5) (1998) 913–936.

[6] J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, Proc. Nat. Academy Science 79 (1982) 2554–2558.

[7] K. Jim, C. L. Giles, B. Horne, An analysis of noise in recurrent neural networks: Convergence and generalization, IEEE Transactions on neural networks 7 (6) (1996) 1424–1438.

[8] O. Olurotimi, S. Das, Noisy recurrent neural network: The discrete-time case, IEEE Transactions on neural networks 9 (5) (1998) 937–946.

[9] D. C. Park, S. E. Choi, A neural network based multi-destination routing algorithm for communication network, in: Proc. of 1998 IEEE Int. Joint Conf. on Neural Networks, 1998.

[10] L. P. Wang, S. Li, F. Y. Tian, X. J. Fu, A noisy chaotic neural network for solving combinatorial optimization problems: stochastic chaotic simulated annealing, IEEE Transactions on System, Man, and Cybernetics-Part B: Cybernetics 34 (5) (2004) 2119–2125.

[11] L. P. Wang, H. Shi, A gradual noisy chaotic neural network for solving the broadcast scheduling problem in packet radio networks, IEEE Transactions on neural networks 17 (4) (2006) 989 – 1000.