Rescaling the Energy Function in Hopfield Networks

Xinchuan Zeng and Tony R. Martinez
Computer Science Department, Brigham Young University, Provo, Utah 84602
Email: zengx@axon.cs.byu.edu, martinez@cs.byu.edu

Abstract

In this paper we propose an approach that rescales the distance matrix of the energy function in the Hopfield network for solving optimization problems. We rescale the distance matrix by normalizing each row in the matrix and then adjusting the parameter for the distance term. This scheme has the capability of reducing the effects of clustering in data distributions, which is one of main reasons for the formation of invalid solutions. We evaluate this approach through a large number (20,000) simulations based on 200 randomly generated city distributions of the 10*-city traveling salesman problem. The result shows that, compared to those using the original Hopfield network, rescaling is capable of increasing the percentage of valid tours by 17.6%, reducing the error rate of tour length by 11.9%, and increasing the chance of finding optimal tours by 14.3%.*

Introduction

Hopfield and Tank [1] proposed that an optimal or a suboptimal solution of the *traveling salesman problem* (*TSP*), a classical example of combinatorial optimization problems, can be obtained by finding a minimum of an appropriate *energy function* which is implemented by a neural network (*Hopfield network*). For a N-city *TSP* problem, the network consists of $N \times N$ connected neurons, in which the linking strengths (weights) between neurons are set to encode the information about the constraints and the cost function of a particular city distribution of *TSP*. Each neuron updates its input value based on the information received from all other neurons. Hopfield and Tank showed that such a neural network can often find a near-optimal solution in a short time.

Since Hopfield and Tank's work, there has been growing interest in the Hopfield network due to its advantages over other approaches for solving optimization problems. The advantages include massive parallelism, convenient hardware implementation of the neural network architecture, and a common approach for solving various optimization problems. Much research has focused on analyzing and improving the original model in order to obtain a higher percentage of valid solutions and solutions with better quality.

The work by Wilson and Pawley [2] showed that the Hopfield network often failed to converge to valid solutions. When it converged, the obtained solution was often far from the optimal solution. Brandt et al. [3] and Aiyer et al. [4] showed that the convergence of the Hopfield network could be improved by modifying the energy function. The modified energy functions have a higher degree of constraints for the network to relax to valid solutions. Li [5] combined the Hopfield network with the "augmented Lagrange multipliers" algorithm from optimization theory, which has better convergence and quality of solutions while requiring many more iterations. Catania et al. [6] applied a fuzzy approach to automatically tune the parameters in the Hopfield network, and reported that the automatic parameter tuning performed better than heuristic and manual tuning. Although these approaches have demonstrated improvement, most of them were empirically tested only on a single or a small number of TSP city distributions, and general performance on a large number of city distributions were not reported.

Despite the improvement of the performance of the Hopfield network over the past decade, this model still has some basic problems [7, 8]. One of them is that the performance of the Hopfield network is inconsistent: The performance is good for some city distributions of *TSP* but poor for other distributions. The performance is usually better for city distributions with simple topology, but poor for those with complex topology and multiple clusters, in which cases the network often fails to find valid tours or finds one with poor quality.

Another problems is that the performance is sensitive to the choice of parameters in the energy function: a different parameter setting can lead to a significant difference in performance. In previous work, we proposed a new activation function to reduce effects of noise [9] and a new relaxation procedure to reach a smoother relaxation process [10]. Both approaches have been shown capable of significantly improving performance based on a large number of simulations.

In this work we propose an approach that is capable of reducing the negative impacts of distribution clusterings on the network performance by rescaling the distance matrix. In the energy function of the Hopfield network for solving *TSP*, the distance matrix reflects the city distribution and is applied to control the quality of obtained solutions. The matrix element d_{XY} represents the distance from city X to city Y. For a city distribution with multiple clusters, this type of representation can cause difficulty in finding valid tours due to a large imbalance among matrix elements due to clustering. The proposed approach reduces this effect by rescaling each row of the matrix in order to improve the balance. This approach has been tested through a large number $(20,000)$ of simulations based on 200 randomly generated city distributions of 10-city *TSP*. The result shows that it is capable of improving the performance significantly: 7.6% increase in the percentage of valid tours, 11.9% decrease in the error rate, and 14.3% increase in the chance of finding optimal tours.

Hopfield Network Basics

For an N-city *TSP*, the Hopfield network [1] has $N \times N$ fully connected neurons in the network, in which the row index represents the city and the column index represents the order of city in a tour. The constraints and the cost of the *TSP* are represented by an energy function, which is used to determine the connecting weights between neurons. Hopfield's original energy function for an N-city *TSP* is given by [1]:

$$E = \frac{A}{2}\sum_{X=1}^{N}\sum_{i=1}^{N}\sum_{j=1,j\neq i}^{N} V_{Xi}V_{Xj} + \frac{B}{2}\sum_{i=1}^{N}\sum_{X=1}^{N}\sum_{Y=1,Y\neq X}^{N} V_{Xi}V_{Yi} + \frac{C}{2}(\sum_{X=1}^{N}\sum_{i=1}^{N} V_{Xi} - N_0)^2$$
$$+\frac{D}{2}\sum_{X=1}^{N}\sum_{i=1}^{N}\sum_{Y=1,Y\neq X}^{N} d_{XY}V_{Xi}(V_{Y,i+1} + V_{Y,i-1}) \tag{1}$$

where X, Y are row indices, and i, j are column indices, V_{Xi} is the activation for neuron (X, i), and d_{XY} is the distance between cities X and Y. The first three terms enforce the constraints for a valid tour, and the last term represents the cost function for obtaining a short tour. The value of each parameter (A, B, C, and D) measures the importance of the corresponding term. Each neuron (X, i) has an input value U_{Xi} and an activation (output) value V_{Xi}. The connecting weight between neuron (X, i) and (Y, j) is set according to:

$$W_{Xi,Yj} = -A\delta_{XY}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{XY}) - C - Dd_{XY}(\delta_{j,i+1} + \delta_{j,i-1}) \tag{2}$$

where δ_{ij} is equal to 1 if $i = j$, and equal to 0 otherwise. Each neuron (X, i) is also connected to an external input current: $I_{Xi} = CN_0$. Before the relaxation of the network, the initial value of each U_{Xi} is set to be a constant value (determined by the condition: $\sum_{X=1}^{N}\sum_{i=1}^{N} V_{Xi} = N$) and is then perturbed with small random noise to break the symmetry of the network. During relaxation, each neuron updates its input and activation value based on the weighted activations of other neurons and its own value. The input value $U_{Xi}^{(n+1)}$ at iteration step *(n+1)* is given by:

$$U_{Xi}^{(n+1)} = U_{Xi}^{(n)} + \Delta U_{Xi} \tag{3}$$

where $U_{Xi}^{(n)}$ is the value at iteration step *(n)*. The value of ΔU_{Xi} is given by the following equation:

$$\Delta U_{Xi} = (-\frac{U_{Xi}}{\tau} + \sum_{Y=1}^{N}\sum_{j=1}^{N} W_{Xi,Yj}V_{Yj} + I_{Xi})^{(n)}\Delta t \tag{4}$$

where $\tau(= RC)$ is the time constant of an RC circuit and was set to be 1.0 by Hopfield and Tank [1]. The activation $V_{Xi}^{(n+1)}$ at iteration step *(n+1)* is then determined by $U_{Xi}^{(n+1)}$ through an *activation (output) function*. In the Hopfield network, the activation function is the hyperbolic tangent function:

$$V_{Xi}^{(n+1)} = \frac{1}{2}(1 + tanh(\frac{U_{Xi}^{(n+1)}}{u_0})) \tag{5}$$

where u_0 is the amplification parameter that reflects the steepness of the activation function.

Hopfield and Tank [1] showed that the network is guaranteed to converge to a local minimum in the case of symmetric ($W_{Xi,Yj} = W_{Yj,Xi}$) weights.

Rescaling the Distance Matrix

The distance matrix d_{XY} in the energy function plays an important role in controlling the quality of solutions. From Eq. (2) and (4), we can see that the second term in Eq. (4) for the activation change ΔU_{Xi} is proportional to the value: $-D\sum_{Y=1}^{N} d_{XY}V_{XY}$. Since each V_{XY} has a similar average level, that term is approximately proportional to the value: $-D\sum_{Y=1}^{N} d_{XY}$.

Now consider a case of two clusters C_1 and C_2, where N_1 and N_2 are the number of cities in C_1 and C_2 respectively. Further consider the case that cluster C_1 is much larger than C_2, i.e. $N_1 >> N_2$. For a city X_1 in C_1 and X_2 in C_2, d_{X_1Y} would be smaller than d_{X_2Y} for $(N_1 - 1)$ cities in C_1 while larger for N_2 cities in C_2. Since $N_1 >> N_2$, the value $\sum_{Y=1}^{N} d_{X_1Y}$ would be smaller than $\sum_{Y=1}^{N} d_{X_2Y}$.

The net effect is that the activations for the cities in cluster C_1 have overall higher values than those in C_2 (a smaller $\sum_{Y=1}^{N} d_{X_2Y}$ leads to a larger $-D\sum_{Y=1}^{N} d_{XY}$). A potential consequence is the formation of an invalid tour – a closed loop inside cluster C_1 without inclusion of those cities in C_2. We have observed such phenomenon for some city distributions where one or two cities are isolated from the main cluster.

Based on this analysis, we propose the following approach to rescale the distance matrix so that a smaller cluster has a similar overall activation level as a larger cluster.

For each row X ($X = 1, 2, ..., N$), we first calculate the summation: $S_X = \sum_{Y=1}^{N} d_{XY}$. Then each original matrix element d_{XY} is modified to be d'_{XY} using the formula: $d'_{XY} = \frac{d_{XY}}{S_X}$. Note that d'_{XY} is normalized for each row X, i.e. $\sum_{Y=1}^{N} d'_{XY} = 1$. Now with the modified matrix d'_{XY}, the two terms $\sum_{Y=1}^{N} d'_{X_1Y}V_{XY}$ and $\sum_{Y=1}^{N} d'_{X_2Y}V_{XY}$ would have a similar overall level even though they are located in two unbalanced clusters. Using the modified matrix has the capability of reducing uneven overall activation levels among network nodes caused by unbalanced distribution clusterings.

In order to maintain a similar overall magnitude for the last term in Eq. (1), we modify the parameter D to be D':

$$D' = \frac{D\sum_{X=1}^{N} S_X}{N} \tag{6}$$

Besides replacing d_{XY} and D by d'_{XY} and D', all the other parameters, and all the formulas and procedures described in the last section are kept exactly the same.

Simulation Results

We have evaluated the performance of the rescaling approach through simulations based on 200 randomly generated 10-city *TSP* city distributions, including wide varieties of topologies.

Many previous studies used only one (e.g., [1]) or a small number of city distributions (e.g., 10 distributions in [2]) in their simulations. This may lead to unreliable conclusions when comparing two algorithms, because the performance of an algorithm often depends on the topology in a city distribution, and different algorithms may favor different types of topologies. Using a large number of city distributions can reduce this effect and allows a better evaluation.

In the simulation, 100 runs are conducted for each of the 200 city distributions. For each of the 100 runs, different random noise is added to the initial values of the neurons.

For a fixed set of parameters (dt, u_0, etc), the quantities to be evaluated are first averaged over 100 runs for each city distribution, and then averaged over the entire 200 city distributions. Thus, 20,000 runs are needed to obtain the simulation results.

We experimented with different sets of 20,000 runs for a fixed set of parameters. The results show that the estimated quantities are fairly stable. Their values vary within a range of about 1% among different sets of 20,000 runs. This demonstrates that the number of runs in our simulation is large enough to make a reasonable estimation of the evaluated quantities.

The original energy function of the Hopfield network was used in the simulation, and the parameters in the energy function are those used by Hopfield and Tank [1]: $A = B = D = 500$, $C = 200$, $N_0 = 15$. The value of dt in Eq. (4) is set to be 10^{-5}, and the value of u_0 in Eq. (5) is set at 0.02.

The fraction of random noise in the initial values of neurons is set to be 0.001 in the simulation. We tried several different values for the fraction in the range from 0.0001 to 0.01. The performance of the network is only slightly sensitive to this parameter, and the result using 0.001 is slightly better than the others.

Table 1: Simulation results

Algorithm	*Valid(%)*	*Err(%)*	*Opt(%)*
Original	20.5	5.9	59.5
Rescaling	24.1	5.2	68.0
Difference	+17.6%	-11.9%	+14.3%

Table 1 shows the simulation results using the rescaling approach (labeled by **Rescaling**) and comparisons to those without it (labeled by **Original**). It also displays the precentage of differences obtained by **Rescaling** with respect to **Original** (labeled by **Difference**). *Valid(%)*, *Err(%)* and *Opt(%)* represent the percentage of valid tours, error rate, and chance of finding optimal tours respectively, which are defined in the following.

Valid(%) is the weighted average percentage of valid tours over $N_{CityDist}$ $(= 200)$ city distributions:

$$Valid = \frac{\sum_{i=1}^{N_{CityDist}} Valid_i}{N_{CityDist}} \tag{7}$$

where $Valid_i$ is the percentage of valid tours for city distribution i and is defined in the following.

For each city distribution i, there are a total of $N_{total,i}$ (=100) runs with different initial input values. The maximum number of iterations allowed for each run is set to be 500. If a valid tour can not be reached within 500 iterations, the network will stop and the tour is counted as invalid.

If $N_{valid,i}$ is the number of valid tours among the total of $N_{total,i}$ runs, then $Valid_i$ is defined by:

$$Valid_i = \frac{N_{valid,i}}{N_{total,i}} \tag{8}$$

Err(%) represents the error rate, which is averaged over 100 different runs for each city distribution and then averaged over the 200 different city distributions as defined in the following.

For city distribution i, the error of a valid tour j is defined by:

$$Err_{i,j} = \frac{d_{i,j} - d_{i,opt}}{d_{i,opt}} \tag{9}$$

where $d_{i,j}$ is the tour length of a valid tour j and $d_{i,opt}$ is the optimal (shortest) tour length of the city distribution i.

The error for city distribution i is the averaged error over all valid tours:

$$Err_i = \frac{\sum_{j=1}^{N_{valid,i}} Err_{i,j}}{N_{valid,i}} \tag{10}$$

The error shown in the table is the average error of valid tours in all city distributions and is weighted by the percentage of valid tours for each city distribution:

$$Err = \frac{\sum_{i=1}^{N_{CityDist}} (Valid_i Err_i)}{\sum_{i=1}^{N_{CityDist}} Valid_i} \tag{11}$$

Opt(%) is the chance of finding optimal tours of city distributions defined by:

$$Opt = \frac{N_{Optimal}}{N_{CityDist}} \tag{12}$$

where $N_{Optimal}$ is the number of the city distributions for each of which at least one of the 100 runs can reach its optimal tour. For example, $Opt = 68.0\%$ in the table (using rescaling) means that for each of $68.0\% * 200 = 136$ city distributions (among 200), at least one of the 100 runs for that distribution is capable of reaching the optimal tour.

From the result in Table 1, we can see that the rescaling approach performs better in all the categories. Rescaling can find more valid tours (by 17.6%), demonstrating that reducing clustering effects by rescaling the distance matrix has positive effects on the formation of valid tours. Rescaling is also capable of improving the quality of tours (error rate reduced by 11.9%). It can maintain a better balance in the activations of all nodes in the network, and thus allow a broader view of possible tours to choose a better one. This also explains why rescaling has a higher chance of finding optimal tours (by 14.3%).

Summary

In this paper, we have proposed an approach to rescale the distance matrix in the Hopfield network for solving *TSP*. The objective of rescaling is to reduce activation imbalances in the network caused by distribution clustering, so that the network is capable of obtaining more valid tours with higher qualities. The scheme we applied is normalizing each row in the distance matrix such that each city has an overall balanced distance-term, even when there exist unbalanced clusters in a city distribution.

We evaluate the rescaling approach based on a large number (20,000) of simulations using 200 randomly generated city distributions of the 10-city TSP. The result shows that it has the capability of obtaining a 17.6% increase in the percentage of valid tours, a 11.9% decrease in error rate, and a 14.3% increase in the chance of finding optimal tours.

In future we plan to extend this research in the following directions: (i) combine the rescaling approach with other approaches (e.g., different activation functions); (ii) evaluate its performance on other optimization problems in order to better evaluate its generality.

Acknowledgments

This research is funded in part by a grant from *fonix* Corp.

References

[1] Hopfield, J. J. and Tank, D. W., Neural Computations of Decisions in Optimization Problems. Biological Cybernetics, vol. 52, pp. 141-152, 1985.

[2] Wilson, G. V. and Pawley, G. S., On the Stability of the Traveling Salesman Problem Algorithm of Hopfield and Tank. Biological Cybernetics, vol. 58, pp. 63-70, 1988.

[3] Brandt, R. D., Wang, Y., Laub, A. J. and Mitra, S. K., Alternative Networks for Solving the Traveling Salesman Problem and the List-Matching Problem. Proceedings of IEEE International Conference on Neural Networks, San Diego, CA. II: 333-340, 1988.

[4] Aiyer, S. V. B., Niranjan, M. and Fallside, F., A Theoretical Investigation into the Performance of the Hopfield Model. IEEE Transactions on Neural Networks, vol. 1, no. 2, pp. 204-215, 1990.

[5] Li, S. Z., Improving Convergence and Solution Quality of Hopfield-Type Neural Networks with Augmented Lagrange Multipliers. IEEE Transactions On Neural Networks, vol. 7, no. 6, pp. 1507-1516, 1996.

[6] Catania, V., Cavalieri, S. and Russo, M., Tuning Hopfield Neural Network by a Fuzzy Approach. Proceedings of IEEE International Conference on Neural Networks, pp. 1067-1072, 1996.

[7] Cooper, B. S., Higher Order Neural Networks-Can they help us Optimise?. Proceedings of the Sixth Australian Conference on Neural Networks (ACNN'95), pp. 29-32, 1995.

[8] Van den Bout, D. E. and Miller, T. K., Improving the Performance of the Hopfield-Tank Neural Network Through Normalization and Annealing. Biological Cybernetics, vol. 62, pp. 129-139, 1989.

[9] Zeng, X. and Martinez, T. R., A New Activation Function in the Hopfield Network for Solving Optimization Problems. Fourth International Conference on Artificial Neural Networks and Genetic Algorithms, 1999.

[10] Zeng, X. and Martinez, T. R., A New Relaxation Procedure in the Hopfield Network for Solving Optimization Problems. Neural Processing Letters, vol. 10, pp. 1-12, 1999.