Paper

Learning Method of Hopfield Neural Network and Its Application to Traveling Salesman Problem

Rong Long Wang* Student Member
Zheng Tang* Non-member
Qi Ping Cao** Non-member
Hiroki Tamura* Member
Masahiro Ishii* Non-member

A learning method of the Hopfield neural network is presented for efficiently solving combinatorial optimization problems. The learning method adjusts the balance between the constraint term and the cost term of the energy function so as to keep the Hopfield network updating in a gradient descent direction of energy. This paper describes and analyzes the learning method and shows its application to the traveling salesman problem (TSP). The performance is evaluated through simulating 100 randomly generated instances of the 10-city traveling salesman problem and some TSPLIB benchmark problems. The simulation results show that the performance of the proposed learning method on these test problems is very satisfactory in terms of both solution quality and running time. The proposed learning method finds the optimal solutions on the test TSPLIB benchmark problems in very short computation time.

Keywords: Combinatorial Optimization Problems, Hopfield neural network, Learning, Traveling Salesman Problem

1. Introduction

The idea of using neural networks to provide solutions to difficult NP-hard optimization problems [1], [2] originated in 1985 when Hopfield and Tank demonstrated that the traveling salesman problem (TSP) could be solved using a Hopfield neural network [3], [4]. Since Hopfield and Tank's work, there has been growing interest in the Hopfield network because of 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. The greatest advantage of this method is that a physical device for solving the problem can be constructed, generating a solution. As the device is in the form of a complex multiple-feedback circuit, it is possible to implement it in silicon [5]. To test the possible device all that is necessary is to simulate on a computer the response of the circuit in actual working conditions. Although it may seem a disadvantage that the particular information of the problem is encoded in the circuit it must be appreciated that the success of the basic method opens the way to improvements in which it is the general information of the problem which becomes encoded in the circuit. Such circuits could then be used routinely to solve repeatedly occurring problems in almost a reflexive manner. Without this advantage the method would be simply a competitor for the solution of the Travelling Salesman Problem, and as such would not compare with analogue algorithms such as that of Durbin and Willshaw [6] or other optimization models [7][8][9]. Unfortunately, the technique, which requires minimization of an energy function containing several terms and parameters, was shown that it often failed to converge to the valid solutions and when it converged, the obtained solution was often far from the optimal solution by Wilson and Pawley [5] in 1988. Since then, various modifications have been proposed to improve the convergence of the Hopfield network.

Focusing on the energy function, Brandt [10], Bout [11] and Aiyer [12] showed that the convergence of the Hopfield network could be improved by modifying the energy function. Protzel et al. [13] studied Brandt et al.'s formulation with different parameters. They found that despite the improved convergence of the modified versions of the Hopfield network, the network might not converge to the solutions with good quality. Takefuji et al. showed that the decay term in the Hopfield neural network increases the energy function under some conditions [14]. They modified the motion equation in order to guarantee the local minimum convergence. However, with the Hopfield neural network, the state of system is forced to converge to a local minimum. In other words, the neural network cannot always find the optimum solution. Therefore, several neuron models and heuristics such as hysteresis binary neuron model [15], neuron filter [16], hill-climbing term and omega function [17], La-

*Faculty of Engineering, Toyama University.
Toyama-shi, Japan 930-8555
**Tateyama Systems Institute.
Toyama-shi, Japan 930-0001

grange relaxation [18] and pots spin [19] have been proposed to improve the convergence of the networks. In addition, some researchers tried to modify the internal dynamics of the network to improve the search quality of solution of the Hopfield network [20] [21]. Smith [22] summarized the works and demonstrated the potential of the Hopfield neural networks for solving combinatorial optimization problems.

Despite the improvement of the performance of the Hopfield network over more than a decade, this model still has some basic problems [23], [24], [25]. One of the problems is that the performance of the Hopfield network is very sensitive to the weights (parameters) of the constraint term and the cost term in the energy function. There is no rule of thumb to find out an appropriate parameter set. Instead, trial-and-error has to be applied.

In this paper, we present a Hopfield network learning method for efficiently solving combinatorial optimization problems. The learning method adjusts the balance between the constraint term and the cost term of the energy function so as to keep the Hopfield network updating in a gradient descent direction of energy. This learning method provides a mechanism for shifting the local minimum by adjusting the parameter set. We analyze the learning method theoretically and evaluate the performance experimentally through simulating the TSP. The simulation results based on 100 randomly generated instances of the 10-city traveling salesman problem and some TSPLIB benchmark problems [26] show that the proposed learning method can find one hundred percent valid solutions which are optimal or near optimal solutions of these problems.

2. Hopfield Neural Network for Combinatorial Optimization

The Hopfield neural network consists of many neurons that are nonlinear elements. In the Hopfield neural network approach, a problem is represented by a Liapunov energy function including cost and constraint terms that reflect the objective of the solution. The objective of the constraint term is to find the valid solution. The objective of the cost term is to find the best solution. Thus, in general the energy function of the Hopfield network with N neurons can be described as following:

$$E(V_1, V_2, ..., V_n) = A \cdot E_1(V_1, V_2, ..., V_n) - B \cdot E_2(V_1, V_2, ..., V_n) \quad \cdots (1)$$

where E_1 is the constraint term and E_2 is the cost term, $V_i (i = 1, \cdots, N)$ is the output of neuron.

The gradient descent method seeks the local minimum of the energy function E by using the motion equations of neurons. Hopfield et al. proposed that the motion equation is composed of the partial derivation term of the energy function as the gradient descent method, and the decay term with a time constant τ [3].

$$\frac{dU_i(t)}{dt} = \frac{\partial E(V_1, V_2, ..., V_n)}{\partial V_i} - \frac{U_i}{\tau} \quad \cdots\cdots\cdots (2)$$

where $U_i(t)$ is the internal state of neuron $\sharp i$

Takefuji et al. showed that the decay term increases the energy function under some conditions [14]. They modified the motion equation in order to guarantee the local minimum convergence.

$$\frac{dU_i(t)}{dt} = \frac{\partial E(V_1, V_2, ..., V_n)}{\partial V_i} \quad \cdots\cdots\cdots\cdots (3)$$

Each neuron updates its input value U_i based on the motion equation. Specifically, the value $U_i(t+1)$ at iteration step $(t+1)$ is given by:

$$U_i(t+1) = U_i(t) + \frac{dU_i(t)}{dt} \cdot \Delta t \quad \cdots\cdots\cdots\cdots (4)$$

The output is updated from U_i using a non-linear function called neuron model. In the binary Hopfield neural network, the following McCulloch-Pitts [27] binary neuron model has been used as the input/output function:

$$V_i = \begin{cases} 1 & if\ U_i > 0 \\ 0 & if\ U_i \leq 0 \end{cases} \quad \cdots\cdots\cdots\cdots\cdots\cdots (5)$$

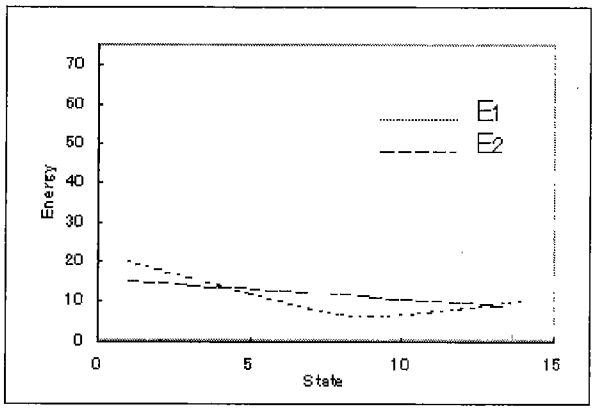
Each neuron updates its input potential according to the updating rule (Eq.(4)) and sends its output in response to the input according to the input/output function (Eq.(5)). All neurons operate in parallel and each adjusts its own state to the states of all the others; in consequence, the network converges to a final configuration. In this way, we can find the solution to the problem simply by observing the final configuration that the network converged.

3. Learning of Hopfield Neural Network

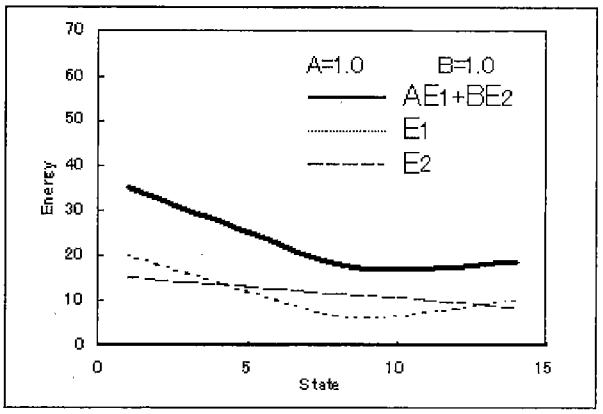
Although Hopfield and Tank's simulation [3] on a 10-city TSP instance showed that the Hopfield neural network was able to obtain a good solution in a reasonable amount of time, this model has some practical limitations: performance is not always good, and performance becomes poorer with large problem, typically larger than 10 cities [5]. Furthermore, in the Hopfield network, there is no way to reach the global minimum from a local minimum. As can be seen from the equation of the energy function (Eq.(1)), the parameter A is used as the weight for the constraint to ensure the feasible solution, and B as weight for the optimization criterion. If the value of A is chosen too small compared with B, the network will emphasize optimizing the cost term ($E2$) without giving enough consideration to the solution feasibility. A high A does the opposite, may lead to a valid solution, but it may be far from optimal.

We now propose a learning method that adjusts the balance between the constraint term and the cost term of the energy function by modifying the parameters of these terms so as to keep the network updating in a gradient descent of energy.

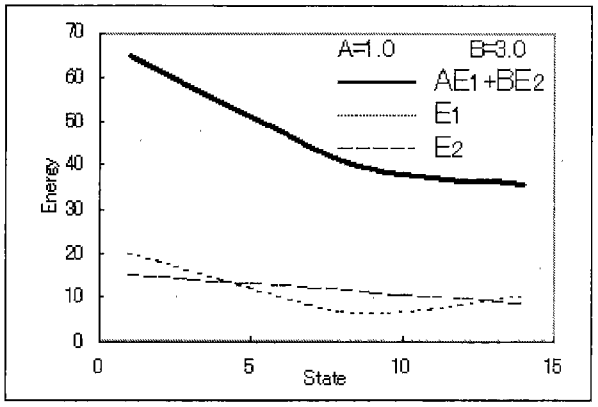
For a energy function consisted from the constraint term (E_1) and the cost term (E_2) as described in Eq.(1), the terrain of the energy function is determined by the terrain of the constraint and cost terms and the weights for the constraint and cost terms. We use a two-dimensional conceptual graph (Fig.1) to describe a



(a)



(b)



(c)

Fig. 1. A two-dimensional conceptual graph of a part of terrain of energy function.

part of the terrain of the energy function. The value of the constraint, cost terms, and the total energy function are reflected in the height of the graph. Abscissa of the graph corresponds to the possible states of the network. We consider the case that the constraint term has a local minimum valley and the terrain of cost function is a descent slope in neighborhood of the local minimum of the constraint term as shown in Fig.1 (a). When we set A and B to 1.0, the terrain of total energy can be expressed as Fig.1 (b). We can see that the terrain of the total energy has a local minimum engendered by the constraint term. However, after we increase the value of B to 3.0, the local minimum of the total energy vanishes as shown in Fig1.(c). It will be the same for the case that the cost term has a local minimum valley and the terrain of the constraint function shows a descent slope. Although for simplicity we assumed two-dimensional cases, it will work in a space of any dimension. Unless the minimum of the total energy function is not only the minimum of the constraint term but also the minimum of the cost term, it is easy to find the direction like Fig.1 of the minimum. Thus, we can design a learning method for eliminating the local minima by modifying the parameter set of the energy function.

First, we analyze the mathematics characteristic of the binary Hopfield neural network at a minimum. It is well known that at a minimum, the energy function always satisfy:

$$\Delta E_i \geq 0 \quad for \quad i = 1, 2, ...N \cdots\cdots\cdots\cdots (6)$$

The variation of energy of the network with the state change of neuron $\sharp i$ $(i = 1, 2, \cdots, N)$ can be written as:

$$\Delta E_i = \frac{\partial E(V_1, V_2, ..., V_N)}{\partial V_i} \cdot \Delta V_i \cdots\cdots\cdots\cdots (7)$$

where $V_1, V_2, \cdots, V_N$ are the states of neurons at the minimum. Using Eq.(1) we have:

$$\Delta E_i = (A \cdot \frac{\partial E_1(V_1, V_2, ..., V_N)}{\partial V_i} + B \cdot \frac{\partial E_2(V_1, V_2, ..., V_N)}{\partial V_i}) \cdot \Delta V_i \quad for \quad i = 1, 2, ..., N \cdots\cdots\cdots\cdots (8)$$

Because the aim of the learning is to make the energy value of the Hopfield network decrease with the state change of at least one neuron, we can modify the parameters A and B to make one of Eq.(8) smaller than zero. To do this, we establish the following learning rule from Eq.(8):

1. Select one neuron that satisfies the following equation:

$$\frac{\partial E_1(V_1, V_2, ..., V_N)}{\partial V_i} \cdot \frac{\partial E_2(V_1, V_2, ..., V_N)}{\partial V_i}) < 0 \quad (9)$$

Eq.(9) is equivalent to assuming that the variation of the constraint and cost terms with the state change of neuron $\sharp i$ has different direction like Fig.1. Thus it is possible to make Eq.(8) of neuron $\sharp i$ smaller than zero by modifying the parameters A and B. It is easy to find the neuron with the relation of Eq.(9) at a general local minimum. As mentioned above, there is a kind of minima in the total energy function that it is not only the minimum of the constraint term but also the minimum of the cost term. In this kind of minima, there is no neuron with the relation of Eq.(9) and the local minimum cannot be eliminated using the proposed learning method. However, this kind of minima is always near to global minimum. It can be used as the condition of terminating the learning.

2.Learning rule

Under Eq.(6) and Eq.(9), we can have two and only two cases about the values of the two terms of Eq.(8).

Case 1: $\frac{\partial E_1(V_1,V_2,...,V_N)}{\partial V_i} \cdot \Delta V_i < 0$ and $\frac{\partial E_2(V_1,V_2,...,V_N)}{\partial V_i} \cdot \Delta V_i > 0$. In this case, learning is performed by modifying the parameter A according the following learning rule:

$$A_{new} = -B \cdot \frac{\partial E_2(V_1,...,V_N)}{\partial V_i} / \frac{\partial E_1(V_1,...,V_N)}{\partial V_i} + \delta \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \quad (10)$$

Case 2: $\frac{\partial E_2(V_1,V_2,...,V_N)}{\partial V_i} \cdot \Delta V_i < 0$ and $\frac{\partial E_1(V_1,V_2,...,V_N)}{\partial V_i} \cdot \Delta V_i > 0$. In this case, . In this case, the parameter B is modified according the following learning rule:

$$B_{new} = -A \cdot \frac{\partial E_1(V_1,...,V_N)}{\partial V_i} / \frac{\partial E_2(V_1,...,V_N)}{\partial V_i} + \delta \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \quad (11)$$

where δ is a small positive constant that controls the learning speed. Using Eq.(9) we can see that the modified parameter (A_{new} and B_{new}) will be positive. In case 1, after learning, the parameter set becomes A_{new}, B, and with the state change of neuron $\sharp i$, the variation of the energy of the network can be described as the following formula by applying Eq.(10) into Eq.(8).

$$\Delta E_i = \delta \cdot \frac{\partial E_1(V_1,V_2,...,V_N)}{\partial V_i} \cdot \Delta V_i \quad \cdots\cdots\cdots \quad (12)$$

Using the condition of the case 1, we can see that Eq.(12) is smaller than zero.

$$\Delta E_i = \delta \cdot \frac{\partial E_1(V_1,V_2,...,V_N)}{\partial V_i} \cdot \Delta V_i < 0 \quad \cdots\cdots \quad (13)$$

Similarly, in the case 2 the modified parameter set (A, B_{new}) can lead the following inequation.

$$\Delta E_i = \delta \cdot \frac{\partial E_2(V_1,V_2,...,V_N)}{\partial V_i} \cdot \Delta V_i < 0 \quad \cdots\cdots \quad (14)$$

From Eq.(13) and Eq.(14), we can see that energy of the network decreases with the state change of neuron $\sharp i$ under the above learning rule. Furthermore, the modified parameter set can also lead energy of the network decrease with the state change of some other neurons. Thus, the learning eliminates the local minimum that the network falls into.

Although, we described the proposed learning method on the energy function with the constraint term and the cost term, other kind of problem with only the constraint term in the energy function (for example the N-Queens problem) can also be mapped onto the proposed learning method by dividing the energy function into two parts.

4. Parallel Algorithm

The following procedure describes the algorithm procedure for solving combinatorial optimization problems using the proposed learning method in synchronous parallel mode. For the practical problem, in order to reduce the computation time we always set the maximum number of the iteration step. When the iteration step exceeds the maximum one, the network is considered as falling into a convergence state. This rule is usually used in the binary Hopfield neural networks. If the *targ_cost* is the target total cost set by the user as expected one, we have:

(1) Set the learning parameter δ, and initial parameters A, B. Set *targ_cost* and other constants.

(2) Initialize the state of the network.

(3) Update the Hopfield network in synchronous parallel mode with the current parameter set until the network converges a stable state or the iteration step exceeds the maximum one.

(4) Check the network, if constraint condition and *targ_cost* are reached, then terminates this procedure.

(5) Check Eq.(9) for $i = 1, \cdots, N$

If there is no neuron with satisfying Eq.(9), terminates this procedure. This means that the network falls into a minimum that is not only the minimum of the constraint term but also the minimum of the cost term, and is a near global minimum.

(6) Select (randomly) one neuron with satisfying Eq.(9)

(7) Use learning rule (Eq. (10) or Eq.(11)) to compute the new parameter set.

(8) Go to the step 3.

5. Application to the Traveling Salesman Problem

The traveling salesman problem (TSP) is one of the most famous combinatorial optimization problems. Nowadays, it plays a very important role in the development and testing of new optimization techniques. In this section, we apply the proposed learning method to the TSP.

The TSP consist of finding the shortest closed path by which every city out of a set of N cities is visited once and only once. In the Hopfield neural network approach, a TSP instance is represented by an energy function including cost and constraint terms that reflect the objective of the solution. The objective of the constrain is to find a valid tour, which requires that each city must be visited once and only once. The objective of the cost is to find the shortest valid tour. For an N-city TSP problem, the network consists of $N \times N$ neurons and the neurons are fully inter-connected. The row index for neuron represents the city. The column index represents the order of the city in the tour. Therefore, the constraint term of the energy function can be described as following:

$$E_1 = \sum_{i=1}^{N}(\sum_{j=1}^{N} V_{ij} - 1)^2 + \sum_{l=1}^{N}(\sum_{k=1}^{N} V_{kl} - 1)^2 \quad \cdot \quad (15)$$

where i and k are the row indices; j and l are the column indices; V_{ij} is the state of neuron $\sharp ij$. The first term of Eq.(15) enforce the constraint that no city can be visited more than once and the second term of Eq.(15) does not allow the salesman to visit two different cities at the same time.

The cost term of the energy function is given by:

$$E_2 = \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1,k\neq i}^{N} d_{ik}V_{ij}(V_{k,i+1} + V_{k,i-1}) \cdot (16)$$

where d_{ik} is the measure of the distance between cities i and k. Thus, the total energy function of the TSP can be described as following:

$$\begin{aligned} E &= A \cdot E_1 + B \cdot E_2 \\ &= A\sum_{i=1}^{N}(\sum_{j=1}^{N} V_{ij} - 1)^2 + A\sum_{l=1}^{N}(\sum_{k=1}^{N} V_{kl} - 1)^2 \\ &+ B\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1,k\neq i}^{N} d_{ik}V_{ij}(V_{k,i+1} + V_{k,i-1}) \quad (17) \end{aligned}$$

where A, B are parameters which are used as the weight for the constraint to ensure feasible solution and the weight for the optimization criterion, respectively.

The motion equation for neuron $\sharp ij$ is given by:

$$\begin{aligned} \frac{dU_{ij}}{dt} &= -2A(\sum_{m=1}^{N} V_{im} - 1) - 2A(\sum_{n=1}^{N} V_{nj} - 1) \\ &\quad -B\sum_{k=1,k\neq i}^{N} d_{ik}(V_{k,i+1} + V_{k,i-1}) \quad \cdots \cdot (18) \end{aligned}$$

Applying Eq.(15) and Eq.(16) into Eq.(10) and Eq.(11), the learning rule Eq.(10) (case 1) and Eq.(11) (case 2) become:

$$\begin{aligned} A_{new} &= -B \cdot \frac{\partial E_2(V_1,...,V_N)}{\partial V_i} \Big/ \frac{\partial E_1(V_1,...,V_N)}{\partial V_i} \\ &\quad +\delta \\ &= -B\sum_{k=1,k\neq i}^{N} d_{ik}(V_{k,i+1} + V_{k,i-1}) \\ &\quad /2[(\sum_{m=1}^{N} V_{im} - 1) - (\sum_{n=1}^{N} V_{nj} - 1)] + \delta \quad (19) \end{aligned}$$

$$\begin{aligned} B_{new} &= -A \cdot \frac{\partial E_1(V_1,...,V_N)}{\partial V_i} \Big/ \frac{\partial E_2(V_1,...,V_N)}{\partial V_i} \\ &\quad +\delta \\ &= -2A[(\sum_{m=1}^{N} V_{im} - 1) - (\sum_{n=1}^{N} V_{nj} - 1)] \\ &\quad / \sum_{k=1,k\neq i}^{N} d_{ik}(V_{k,i+1} + V_{k,i-1}) + \delta \quad \cdots (20) \end{aligned}$$

where δ is a small positive constant. According to the algorithm procedure described in Section 4, we simulate the TSP. The simulation results are given in the next section.

6. Simulation Results on TSP

In order to assess the effectiveness of the proposed learning method, extensive simulations were carried out over randomly generated instances of the 10-city traveling salesman problem and some TSPLIB benchmark problems on PC Station (PentiumIII 800MHz). Simulations referred to initial parameter set at A=2.0 B=1.0. In the experiments, δ was selected to 0.2. In order to reduce the computation time, we set maximum number of iteration step to 800, when the iteration step exceeds the maximum iteration; the network is considered as falling into a convergence state.

The first TSP instance that we tested was a randomly generated 10-city traveling salesman problem. Figure 2 showed the location distribution of the cities of this instance. Figure.3 showed the results of a simulation on the instance that illustrates a typical progressive intermediate solution during the learning. Initially the Hopfield network converged to a stable state (Fig.3(a)). From this stable state, we found that in $4th$ visiting, the salesman did not visit any city, and in $6th$ and $7th$ visiting, the salesman visited the same city (city $\sharp 4$). City $\sharp 2$ and $\sharp 9$ have not been visited. It was obviously not a valid solution. After the $18th$ learning, the Hopfield network found the path of Fig.3(b) with a closed path of 31.196. In this problem, the network performed totally 29 learnings and finally found the shortest closed path of 28.067 as shown in Fig.3(c). The computation time of the run on this problem was 4.02 seconds.

Many previous studies used only one data set (the one used by Hopfield and Tank in their original paper [3]) or a small number of data sets (for example, 10 data set in [5]) in their simulations. This may result in an unreliable conclusion when used in evaluating an algorithm. The reason is that the performance of an algorithm often depends on the location distribution of the cities in a data set. In order to reduce this effect and exactly evaluate the proposed learning method, we randomly generated 100 data sets of 10-city problems. For each data set, 100 runs with different initial input value of neuron were performed. In these 10000 runs, we found that the proposed learning method can find one hundred percent valid solutions in short computation time, and the solution quality is very good. The average learning epoch and computation time of these 10000 runs were 24 times and 5.80 seconds, respectively.

Although we generated the very good solutions for 10-city problem, we could not identify if the solution found by the proposed learning method is optimal solution. In

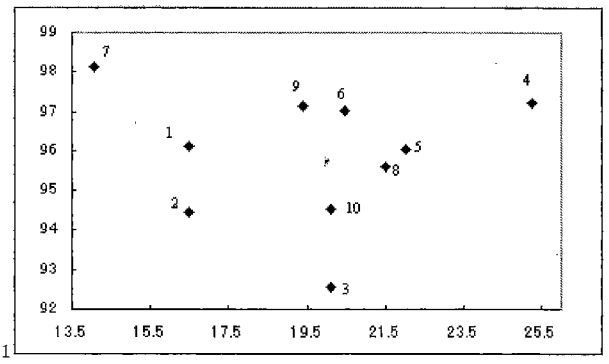


Fig. 2. Location distribution of the cities of 10-city TSP instance.

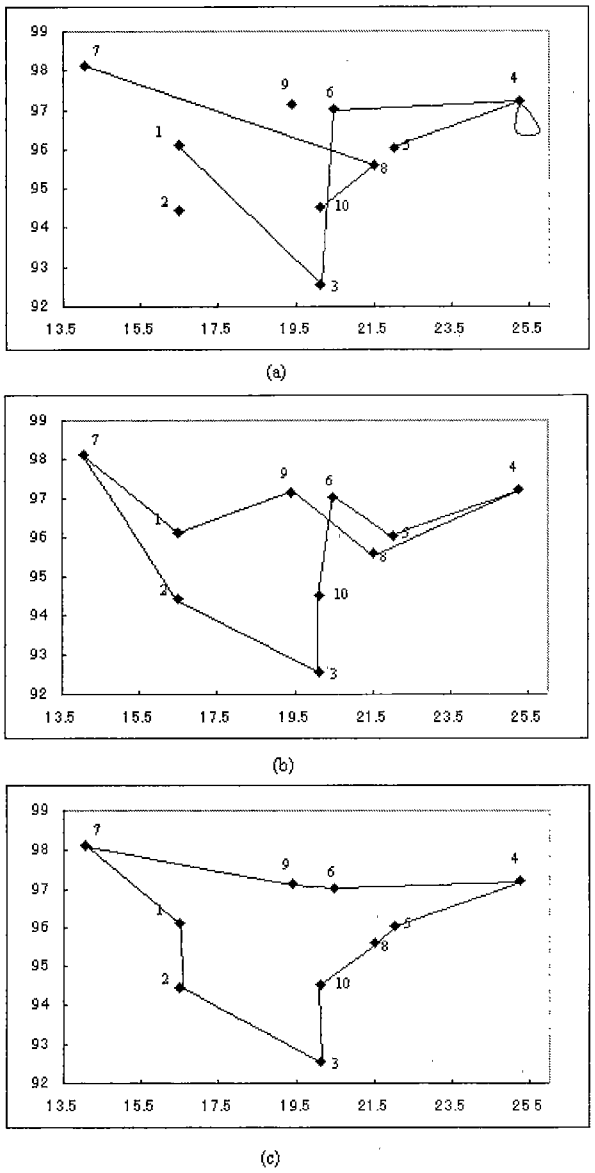


Fig. 3. Results of a simulation on 10-city TSP instance.

order to see if the proposed learning method can find optimal solution, we also tested the proposed learning algorithm on some TSPLIB benchmark problems [26]. We selected ulysses22 (22-city) and eil51 (51-city) to test the proposed learning method. The shortest path of ulysses22 and eil51 generated by the proposed learning algorithm were 7013 and 426, respectively which were the same as the best paths proclaimed at TSPLIB [26] and have been proven to be global optimality. Furthermore, for each of the two instances, 100 simulation runs with different initial input values of neuron were performed. We found that the proposed learning method could find one hundred percent valid solutions and the rates of optimal solution were 63

It is worth to note that our simulations were performed on a series computer for generating optimal or near-optimal solution to TSP, and naturally result in large CPU times that were uncompetitive with alternative techniques. We adhered to the philosophy that the model being tested should be always having a possible "silicon implementation" [5][22]. Thus on a parallel computation device the computation time of the proposed method will become very short. Furthermore, because of the simplicity of the proposed learning method, it is easy to implement the proposed method on electronic circuits.

7. Conclusions

A Hopfield network learning method for efficiently solving combinatorial optimization problems was proposed in this paper. The learning method adjusts the balance between the constraint term and the cost term of the energy function so as to keep the Hopfield network updating in a gradient descent direction of energy. The learning method was analyzed theoretically and evaluated experimentally through simulating the TSP. The simulation results based on 100 randomly generated instances of the 10-city traveling salesman problems and some TSPLIB benchmark problems showed that the proposed learning method could find hundred percent valid solutions that are optimal solutions or near optimal solutions. The proposed learning method is not limited for solving the local minimum problem of Hopfield network; it can be also used to solve the local minimum problem of quadratic polynomial function in [0,1] topological space.

Acknowledgment

This work was supported in part by the grant-in-aid for Science Research of the Ministry of Education, Science and Culture of Japan under Grant (C)(2)12680392.

(Manuscript received March 14, 2002, revised September 6, 2002)

References

(1) R. Garey and S. Johnson: "Computers and Intractability, a guide to the theory of NP-completeness", Freeman and Company, (1991)

(2) G. L. Nemhauser and L. A. Wolsey: "Integer and combinatorial optimization", John Wiley & Sons, New York, (1988)

(3) J. J. Hopfield and D.W.Tank: " 'Neural' computation of decisions in optimization problems", *Biol. Cybern.*, No.52, pp.141-152 (1985)

(4) J. J. Hopfield and D.W.Tank: "Computing with neural circuits: a model", *Science*, No.233, pp.625-633 (1986)

(5) G. V. Wilson and G. S. Pawley "On the stability of the traveling salesman problem algorithm of Hopfield and Tank", *Boil. Cybern.*, Vol.58, pp.63-70 (1988)

(6) R. Durbin and D. Willshaw: "An analogue approach to the traveling salesman problem using an elastic net method", *Nature*, Vol.326, pp.689-691 (1987)

(7) S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi: "Optimisation by Simulated Annealing", *Science*, Vol.220, pp.671-680 (1983)

(8) D. E. Goldberg: "Genetic algorithm in search, optimization and mechine learning", *Addison Wesley, Rrading*, MA, (1989)

(9) J. L. Bentley: "Fast algorithm for geometric traveling salesman problems", *ORSA Journal on Computing*, Vol.4, pp.387-411 (1992)

(10) R. D. Brandt, Y. Wang, A. J. Laub, and S. K. Mitra: "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: pp.333-340 (1988)

(11) D. E. Van den Bout and T. K. Miller III: "A traveling salesman objective function that works", in Proc. Int. Conf. Neural Networks, Vol.II pp.299-303 (1988)
(12) S. V. B. Aiyer, M. Niranjan, and F. Fallside: "A theoretical investigation into the performance of the Hopfield network", *IEEE Trans. Neural Networks*, Vol.1, No.2, pp.204-215 (1990)
(13) P.W. Protzel, D. L. Plumbo, and M. K. Arras: "Performance and fault-tolerance of neural network for optimization", *IEEE Trans. Neural Networks*, Vol.4, pp.600-614 (1993)
(14) Y. Takefuji and K. C. Lee: "Artificial neural networks for four-coloring map problems and K-colorability problems", *IEEE Trans. Circuits Syst.*, Vol.38, No. 3, pp.326-333 Mar. (1991)
(15) Y. Takefuji and K. C. Lee: "An artificial hysteresis binary neuron: A model suppressing the oscillatory behaviors of neural dynamics", *Biol. Cybern.*, Vol.64, pp.353-356 (1991)
(16) Y. Takenaka, N. Funabiki, and T. Higashino: "A proposal of neuron filter: A constraint resolution scheme of neural networks for combinatorial optimization problems", *IEICE Trans. Fundamentals*, Vol.E83-A, No.9, pp.1815-1823 (2000)
(17) Y. Takefuji and K. C. Lee: "A parallel algorithm for tiling problems", *IEEE Trans. Neural networks*, Vol.1, No.1, pp.143-145 Mar. (1990)
(18) Z. L. Stan: "Improving convergence and solution quality of Hopfield-type neural networks with augmented Lagrange multipliers", *IEEE Trans. Neural Networks*, Vol.7, No.6, pp.1507-1516 (1996)
(19) S. Ishii and S. Masaaki: "Chaotic potts spin model for combinatorial optimization problems", *Neural Networks*, Vol.10, pp.941-963 (1997)
(20) K. Smith, M. Palaniswami, and M. Krishnamoorthy: "Neural Techniques for Combinatorial Optimization with Applications", *IEEE Trans. Neural Networks*, Vol.9, No.6 pp.1301-1318 (1998)
(21) X. Zeng and T. Martinez: "A new relaxation procedure in the Hopfield network for solving optimization problem", *Neural Processing Letters*, Vol.10, pp.211-222 (1999)
(22) K. A. Smith: "Neural Networks for combinatorial optimization: A Review of more than a decade of research", *INFORMS Journal on Computing*, Vol.11, No.1, pp.15-34 (1999)
(23) B. S. Cooper: Higher order neural networks-can they help us optimize? In Proceedings of the Sixth Australian Conference on neural networks (ACNN'95) pp.29-32 (1995)
(24) D. E. Bout and T. K. Miller: "Improving the performance of the Hopfield-Tank neural network through normalization and annealing", *Biol. Cybern.*, Vol.62, pp.129-139 (1989)
(25) J. Arabas: "A genetic approach to the Hopfield neural network in the combinatorial optimization problems", *Bulleting Polish Academy of Science*, Vol.42, pp.59-66 (1994)
(26) G.Reinelt: "A Traveling Salesman Problem Library", *ORSA Journal on computer*, Vol.3, pp.376-384 (1991)
(27) W. S. McCulloch and W. H. Pitts: "A logical calculus of ideas immanent in nervous activity", *Bull. Math. Biophys*, Vol.5 pp.115-133 (1943)

Rong Long Wang (Student Member) received a B.S. degree from Hangzhe teacher's college, Zhejiang, China and an M.S. degree from Liaoning University, Liaoning, China in 1987 and 1990, respectively. Since 1990 he has been an Instructor in Benxi University, Liaoning, China. Now he is working toward the Ph.D degree at Toyama University, Japan. His main research interests are neural networks and optimization problems.

Zheng Tang (Non-member) received a B.S. degree from Zhejiang University, Zhejiang, China in 1982 and an M.S. degree and a D.E. degree from Tsinghua University, Beijing, China in 1984 and 1988, respectively. From 1988 to 1989, he was an Instructor in the Institute of Microelectronics at Tsinhua University. From 1990 to 1999, he was an Associate Professor in the Department of Electrical and Electronic Engineering, Miyazaki University, Miyazaki, Japan. In 2000, he joined Toyama University, Toyama, Japan, where he is currently a Professor in the Department of Intellectual Information Systems. His current research interests include intellectual information technology, neural networks, and optimizations.

Qi Ping Cao (Non-member) received a B.S. degree from Zhejiang University, Zhejiang, China and an M.S. degree from Beijing University of Posts and Telecommunications, Beijing, China in 1983 and 1986, respectively. From 1987 to 1989, she was an Assistant Professor in the Institute of Microelectronics at Shanghai Jiaotong University, Shanghai, China. From 1989 to 1990, she was a Research Student in Nagoya University, Nagoya, Japan. From 1990 to 2000, she was a Senior Engineer in Sanwa Newtech Inc., Japan. In 2000, she joined Tateyama Systems Institute, Japan. Her current research interests include multiple-valued logic, neural networks, and optimizations.

Hiroki Tamura (Member) received the B.E and M.E degrees from Miyazaki University in 1998 and 2000. From 2000 to 2001, he was a Engineer in Asahi Kasei Corporation, Japan. In 2001, he joined Toyama University, Toyama, Japan, where he is currently a Technical Official in Department of Intellectual Information Systems. His main research interests are neural networks and optimization problems.

Masahiro Ishii (Non-member) obtained a Diploma in Electronic Engineering at Akita University in 1990 and a Doctorate from Tokyo Institute if Technology in 1995. Between 1995 and 1997 he was a Postdoctoral Fellow at the Centre for Vision Research in York University of Canada. He was a Research Associate at Tokyo Institute of Technology between 1997 and 2000. In 2000 he moved to Toyama University, where he is now a lecturer of computer Engineering.