Combination of Chaotic Neurodynamics with the 2-opt Algorithm to Solve Traveling Salesman Problems

M. Hasegawa and T. Ikeguchi
Faculty of Industrial Science and Technology, Science University of Tokyo, 2641 Yamazaki, Noda-shi 278, Japan

K. Aihara
Faculty of Engineering, The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo 113, Japan
(Received 7 April 1997)

We propose a novel approach for combinatorial optimization problems. For solving the traveling salesman problems, we combine chaotic neurodynamics with heuristic algorithm. We select the heuristic algorithm of 2-opt as a basic part, because it is well understood that this simple algorithm is very effective for the traveling salesman problems. Although the conventional approaches with chaotic neurodynamics were only applied to such very small problems as 10 cities, our method exhibits higher performance for larger size problems with the order of 10^2. [S0031-9007(97)04059-3]

PACS numbers: 82.20.Mj, 02.60.Pn, 05.45.+b, 87.10.+e

Various methods are proposed for solving the traveling salesman problem (TSP) which is one of the typical NP (nondeterministic polynomial)-hard combinatorial optimization problems. One of the new approaches, or modern heuristics, is based upon artificial neural networks. The basic concept of this approach was proposed by Hopfield and Tank [1], who applied an artificial neural network with symmetric mutual connections to the TSP, which has a kind of gradient descent dynamics, namely, a decreasing property of the computational energy function. Although this approach is very attractive from the viewpoint of an application of artificial neural networks, the Hopfield-Tank neural network has a notorious local minimum problem. In order to solve such a difficult problem, a new approach using the chaotic neural network [2,3] has been proposed [4–6]. The chaotic dynamics has several particular properties. One of them, self-similarity, is that attractors of chaotic dynamical systems usually have fractal structures. Therefore, chaotic search is expected to be efficient because the chaotic dynamics searches solutions of the TSP only along such a fractal structure with zero Lebesgue measure in the state space, if the optimum solution is located in the searching region.

However, these methods based on the recurrent neural networks have two serious problems. First, the Hopfield-Tank neural network, which provides the basic framework of the approach with chaotic dynamics, requires $n \times n$ mutual connections where n is the number of neurons. In the case of solving an N-city TSP, the number of neurons n is N^2 [1]. Therefore, the number of mutual connections becomes $n^2 = N^4$. If the number of cities N increases, the number of mutual connections becomes huge, consequently calculation gets difficult. Second, constructing a closed feasible tour (which is a constraint in the case of solving the TSP and translated as starting from a city, visiting each city exactly once, and returning to the starting city) is not easy in this approach because a closed tour is realized only by firing patterns of the neural networks that satisfy the constraints, namely, the number of firing neurons must be 1 in each row and each column. If the state of neural networks does not satisfy the constraint term, it cannot form even a closed feasible tour.

As another conventional approach for solving the TSP, several heuristic methods have been proposed by, e.g., Croes [7], Lin [8], and Lin and Kernighan [9]. It has been shown that these methods are effective to obtain near-optimal solutions of the TSP and applicable to larger problems than the methods based on the Hopfield-Tank neural networks, because the constraint of the TSP is intrinsicly included in those heuristic methods. Namely, every solution realizes a closed feasible tour, and the complexity is smaller than that of the Hopfield-Tank neural networks. However, since these heuristic methods also utilize a kind of gradient descent dynamics, they share the local minimum problem.

As an approach for overcoming such a local minimum problem, simulated annealing is known to be useful, but it utilized only stochastic dynamics. Since there are some results implying that chaotic neurodynamics is more effective for combinatorial optimization than stochastic dynamics in the framework of the Hopfield-Tank neural networks [5], it is natural to expect that chaotic neurodynamics would also be effective for the local minimum problems appearing in heuristic methods.

In this Letter, we apply chaotic neurodynamics to the local minimum problem of the heuristic algorithm. In our method, there are no constraint terms which must be satisfied for forming a closed feasible tour because they are already included in the basic heuristic algorithm itself. Namely, our method always produces closed feasible tours.

We select the 2-opt as the basic algorithm among the heuristic methods, because it is well known that a very good near-optimum solution can be obtained with

 0031-9007/97/79(12)/2344(4)$10.00

this simple algorithm. The 2-opt algorithm searches the minimum length tour by changing the visiting order of cities.

The schematic explanation of the 2-opt is described in Fig. 1. We define "a link *X*-*Y*" as a connection between a city *X* and a city *Y*, and "a tour" as the total visiting order. The 2-opt algorithm is as follows:

1. At first, cities are arranged in a random order. This is an initial state.
2. The total length of the present tour D_1 is calculated. In Fig. 1, a present tour is *A-B-H-I-J-C-D-E-F-G-A*.
3. A link *A-B* and another link *C-D* are selected and virtually interexchanged, then a new tour *A-C-J-I-H-B-D-E-F-G-A* is formed. The total length of the new tour is D_2.
4. If $D_1 > D_2$, these links *A-B* and *C-D* are actually interexchanged. Then the shorter length tour with new links *A-C* and *B-D* is obtained by this interexchange.
5. Such a procedure is repeated for all links between each two cities while shorter length tours are found.
6. If there is no more decrease of the total tour length, the procedure is terminated.

Although this algorithm provides near-optimum solutions, there exists the local minimum problem in this algorithm, because it is based upon a kind of gradient descent dynamics. Namely, there are many undesirable local minima where the 2-opt algorithm gets stuck.

Therefore, chaotic dynamics realized by the chaotic neural network model [2,3] is introduced for avoiding such a local minimum problem. In order to implement the 2-opt into the chaotic neural network, a neural network model which behaves as the conventional 2-opt is constructed as the first step. In the case of solving *N*-city TSP, there are $N \times N$ ways for constructing new links in the 2-opt, so $N \times N$ neurons are prepared and they are arranged on an $N \times N$ grid. In this neural network, the (i, j)th neuron corresponds to a link *i*-*j* that is a link between cities *i* and *j*, which means a city *j* is visited next to a city *i*. Then, the following equation is defined as a firing rate of the (i, j)th neuron:

$$x_{ij}(t+1) = D_1(t) - D_{ij}(t), \tag{1}$$

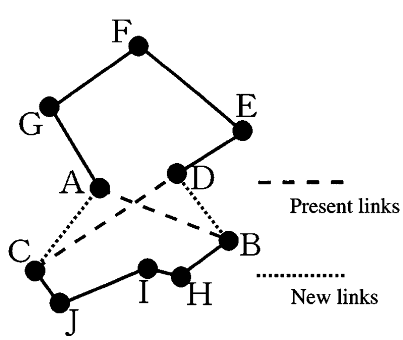

FIG. 1. The schematic representation of the 2-opt. *A*, *B*, ..., *I*, and *J* represent cities. Dashed lines describe present links, and dotted lines describe new links.

$$\begin{aligned} &x_{ij}(t) \geq 0 \quad \text{a link } i\text{-}j \text{ should be connected}, \\ &x_{ij}(t) < 0 \quad \text{no change}, \end{aligned} \tag{2}$$

where $D_1(t)$ is a tour length at time *t*, $D_{ij}(t)$ is the new tour length obtained by applying the 2-opt to make the city *j* visited next to the city *i*. This neural network changes its state asynchronously. Let us assume, for example, that a city *i* corresponds to the city *A*, and a city *j* to *C* in Fig. 1. In the 2-opt algorithm, when the link *A-C* is connected, then the links *A-B* and *C-D* must be cut and the link *B-D* must be connected. Then, the visiting order between the cities *C* and *B* is reversed. In order to realize such an order-exchanging process, if the (i, j)th neuron fires at time $t+1$, a city that had been visited next to a city *j* at time *t* (corresponding to the city *D* in Fig. 1) must be visited next to the city that had been visited next to a city *i* at time *t* (corresponding to the city *B* in Fig. 1), and the visiting order between the city that was visited next to the city *i* at time *t* and the city *j* (corresponding to *B-H-I-J-C* in Fig. 1) must be reversed.

Then, we modify the neural network version of the 2-opt described in Eqs. (1) and (2) to the chaotic neural network version in order to realize chaotic escape from local minima. Here, chaotic neurodynamics is applied to order changes. In the previous methods with chaotic dynamics [4], only the mutual interactions between neurons have information for minimizing an energy function. In the proposed method, we apply external inputs to chaotic neurons in order to control firings for minimizing a tour length. Namely, $D_1(t) - D_{ij}(t)$ in Eq. (1) can be used as the external input for the internal states $\xi_{ij}(t)$ of the chaotic neurons [2,3].

Next, in order to control firings of chaotic neurons, we adopt some connections with negative constant weights. Because only a single city must be visited next to a city, only a single neuron in each row or each column in an $N \times N$ grid is expected to fire. Frequent firings of neurons on the same column or row would lead to many executions of the 2-opt algorithm from the same city. Then a constraint for controlling such firings is described as follows:

$$E_m = \sum_{i=1}^{N}\left(\sum_{k=1}^{N} x_{ik} - R\right)^2 + \sum_{k=1}^{N}\left(\sum_{i=1}^{N} x_{ik} - R\right)^2, \tag{3}$$

where *R* is a parameter for controlling a firing rate. It should be noted that although Eq. (3) is prepared as a constraint term, it does not necessarily need to be satisfied in a strict sense, because the essential constraint for forming a closed feasible tour is already included in the basic part of our network, or the 2-opt algorithm itself. Namely, this constraint is only introduced for a more efficient search. From Eq. (3), the value of the connection weights W^A_{ijkl} between the (i, j)th and the (k, l)th neurons and the value of the threshold a_{ij} of the (i, j)th neuron are set as follows:

$$W^A_{ijkl} = -C[\delta_{ik}(1 - \delta_{jl}) + \delta_{jl}(1 - \delta_{ik})], \quad (4)$$

$$a_{ij} = CR, \quad (5)$$

where C is a positive constant. These connections are sparse and take the same value which gives lower computational complexity.

Moreover, we also introduce another kind of connections in order to obtain higher solving abilities. We make negative connections between the (i, j)th and the (j, i)th neurons. These neurons are corresponding to the links between cities i and j. If both neurons fire, the link between i and j is connected repeatedly. It means that neural network searches the same state, and it is not efficient. In order to avoid such a situation, the following connections are introduced:

$$W^B_{ijkl} = -B\delta_{il}\delta_{jk}, \quad (6)$$

where B is a positive constant.

From the chaotic neural network model [2,3] and Eqs. (4)–(6), the novel order changing process which we propose is described as follows:

$$\xi_{ij}(t + 1) = k_s\xi_{ij}(t) + h(D_1(t) - D_{ij}(t)), \quad (7)$$

$$\eta_{ij}(t + 1) = k_m\eta_{ij}(t) - C\sum_{l=1,l\neq i,j}^{N} x_{il}(t) - C\sum_{l=1,l\neq i,j}^{N} x_{lj}(t) - Bx_{ji}(t), \quad (8)$$

$$\zeta_{ij}(t + 1) = k_r\zeta_{ij}(t) - \alpha x_{ij}(t) + CR, \quad (9)$$

$$x_{ij}(t + 1) = f[\xi_{ij}(t + 1) + \eta_{ij}(t + 1) + \zeta_{ij}(t + 1)], \quad (10)$$

where $\xi_{ij}(t)$, $\eta_{ij}(t)$, and $\zeta_{ij}(t)$ are internal states of externally applied inputs, feedback inputs, refractoriness of the (i, j)th neuron, respectively, k_s, k_m, and k_r are the decay parameters of each internal state, α is a scaling parameter for the refractory effect, $x_{ij}(t)$ is the output of the (i, j)th neuron, and f is a sigmoidal function, namely, $f(z) = 1/[1 + \exp(-z/\epsilon)]$. If $x_{ij}(t)$ is larger than a threshold θ, it is defined that the (i, j)th neuron fires, and the tour order is changed so as to satisfy that a city j is visited next to a city i.

By Eq. (7), if $D_{ij}(t)$ is shorter than $D_1(t)$, the (i, j)th neuron becomes easy to fire, which enables a total tour length to be decreased. If the behavior of this neural network is chaotic, the internal states would never take the same values fundamentally, because the chaotic dynamics has nonperiodicity in its own property.

The result of our method applied to a 105-city problem, Lin105 in Ref. [10], is shown in Fig. 2, with $k_m = 0.0$, $k_s = 0.0$, $R = 1.75$, $\epsilon = 0.001$, $C = 0.001\,25$, $B = \alpha/2$, $h(z) = z$, $\theta = \frac{1}{2}$, and changing the values of k_r and α. Those results are the average values of the best solutions found in 10 000 iterations for each run with a different initial condition. They are compared with the results of stochastic dynamics which are realized by replacing Eq. (9) by the following random neuron model [5]:

$$\zeta_{ij}(t + 1) = -\alpha Z(t) + CR, \quad (11)$$

where $Z(t)$ is Gaussian distributed random numbers with the zero mean and the unit variance. From Fig. 2, it is clear that the chaotic dynamics leads to higher solving abilities at least than the simple stochastic dynamics. It should be noted that although the stochastic dynamics cannot search the optimum solution, 14 379, within 10 000 iterations on each run, the optimum solution can be obtained 100% by our novel method with several parameter values. Moreover, the average iteration for obtaining the optimum solution using the novel method with $k_r = 0.95$ and $\alpha = 0.015$ is only 463.5.

In order to confirm the effectiveness and the robustness of our method to TSP, we applied our method to five 100-city problems, KroA100, KroB100, KroC100, KroD100, and KroE100 in Ref. [10]. Results are shown in Table I. They are also average solutions obtained by changing the initial condition in each run, with $k_r = 0.955$, $k_m = 0.0$, $k_s = 0.0$, $R = 1.95$, $\epsilon = 0.000\,75$, $\alpha = 0.0115$, $C = 0.001\,15$, $B = 0.005\,75$, $h(z) = 1.1z$, and $\theta = \frac{1}{2}$. From Table I, it is clear that the method with chaotic dynamics is more effective than stochastic dynamics with the random neuron model. The 100% solving ability is obtained for the problem KroD100 by the chaotic method. By these experiments, it is confirmed that our method is efficient for various TSPs.

Our method is also applied to a 318-city problem, Lin318 [10], and the tour length of 42 196 is obtained with $k_r = 0.875$, $k_m = 0.2$, $k_s = 0.0$, $R = 0.85$, $\epsilon = 0.003$, $\alpha = 0.09$, $C = 0.09$, $h(z) = 0.05z$, and $B = 0.045$.

In the case of using chaotic dynamics for combinatorial optimization, the network state keeps fluctuating without

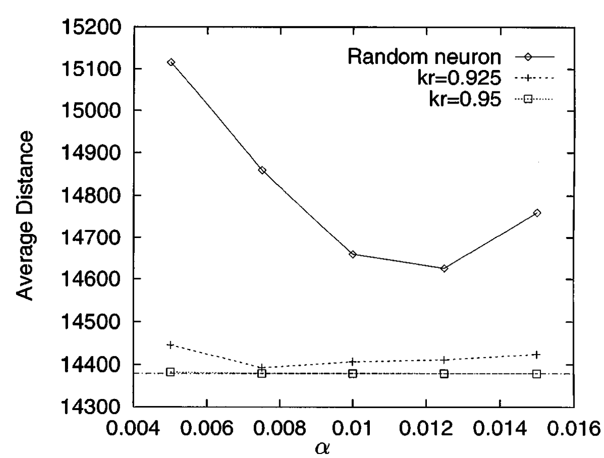

FIG. 2. Average solutions of a 105-city problem (Lin105) in the cases of using chaotic neurons and random neurons. The optimum solution, 14 379, is also shown by a dash-dotted line.

TABLE I. Average solutions of five different 100-city problems, compared with the results of the random neuron model. A value of 21 294.0 for KroD100 means the optimum solution is obtained for all runs of that problem.

	The optimum	Chaotic neuron	Random neuron
KroA100	21 282	21 285.8	21 953.0
KroB100	22 141	22 150.7	22 510.7
KroC100	20 749	20 749.7	21 365.3
KroD100	21 294	21 294.0	21 587.3
KroE100	22 068	22 078.7	22 407.3

stopping even at global minimum. In order to treat such a problem peculiar to the chaotic search, the original 2-opt is applied to the network state at each iteration time obtained by our novel method. Then, for Lin318, the result is improved to be 42 112 with $k_r = 0.875$, $k_m = 0.2$, $k_s = 0.0$, $R = 0.9$, $\epsilon = 0.003$, $\alpha = 0.09$, $h(z) = 0.125z$, $B = 0.045$, and $C = 0.09$, with 10 000 cutoff time.

In this Letter, we combine the advantages of novel modem heuristic approaches, namely, the chaotic neurodynamics for avoiding undesirable local minima, and the 2-opt method, which is applicable to large problems. Our neural network model has sparse mutual connections, therefore the complexity of the neural network is small. Furthermore, our neural network can always construct a closed feasible tour because the constraint for forming a closed feasible tour is already included in the heuristic algorithm which is the basic part of our method.

In the previous approaches using chaotic dynamics [4–6], high solving abilities were obtained only in toy problems with the order of 10 cities. The largest problem was a 48-city TSP; however, only 5% solving ability was reported [6]. Our method exhibits higher solving abilities than the previous ones even in larger problems.

As the other advantage, our model can make heuristic methods chaotic. The chaotic dynamics can be easily applied to other optimization problems, because our neural network requires only a single evaluation parameter of the state, for example, a total tour length in the TSP, which is applied to the chaotic neurons as external inputs. We have already applied this type of the chaotic neural network to the local minimum problem of image segmentation and obtained good performances [11].

Although we experimentally decided the parameter values of chaotic neural networks in this study, it is an important future problem to develop an effective algorithm for deciding the parameter values to obtain the better performances on the basis of theoretical and numerical consideration on dynamical properties of the chaotic neural networks [12,13].

The authors would like to thank Professor T. Matozaki for his encouragement and Professor T. Fukao for his valuable comments. This research was partly supported by Research Fellowships of the Japan Society for the Promotion of Science for Young Scientists to M. H.

[1] J. J. Hopfield and D. W. Tank, Biol. Cybernet. **52**, 144 (1985).
[2] K. Aihara, T. Takabe, and M. Toyoda, Phys. Lett. A **144**, 333 (1990).
[3] K. Aihara, in *Bifurcation Phenomena in Nonlinear Systems and Theory of Dynamical Systems,* edited by H. Kawakami (World Scientific, Singapore, 1990), p. 143.
[4] H. Nozawa, Chaos **2**, 377 (1992).
[5] T. Yamada and K. Aihara, J. Intell. Fuzzy Syst. **5**, 53 (1997).
[6] L. Chen and K. Aihara, Neural Netw. **8**, 915 (1995).
[7] A. Croes, Oper. Res. **5**, 791 (1958).
[8] S. Lin, Bell Syst. Tech. J. **44**, 2245 (1965).
[9] S. Lin and B. W. Kernighan, Oper. Res. **21**, 498 (1973).
[10] G. Reinelt, ORSA J. Comput. **3**, 4 (1991); **3**, 376 (1991).
[11] M. Hasegawa, T. Ikeguchi, T. Matozaki, and K. Aihara, Trans. IEICE Fundamentals **E79-A**, 1630 (1996).
[12] M. Adachi and K. Aihara, Neural Netw. **10**, 83 (1997).
[13] L. Chen and K. Aihara, Physica (Amsterdam) **104D**, 286 (1997).