In this paper, we propose a novel approach for solving large scale TSPs by the chaotic neural networks.

While we have already shown effectiveness of the chaotic searches including tabu search on the Quadratic Assignment Problems (Hasegawa et al., 2000), we apply such a method to much larger problems of TSPs.

We realize two novel methods that have different coding structures (Table 1).

In both methods, the 2-opt exchange is utilized for the basic updating of torus.

The first method is an extension of a tabu search which memorizes paths that have been connected.

Since each path should be represented by two cities which are linked with each other, this requires memorizes of the order of n² for the n-city TSP.

The second one is an exten-sion of a tabu search which memorizes cities that have appeared on 2-opt updating, namely this requires memorizes paths previously connected, seems better to memorize states of the tour and has larger information of previous searches.

However, it requires a larger amount of computer memory because the required number of neurons is n².

On the other hand, the latter method can be realized by small-scale neural networks, and only a very small computer memory is required.

From this point of view, we expect that the latter method is applicable to very large TSPs.

Moreover, we also apply a reducing method of computational workload of the above chaotic neural networks, in order to make it possible to apply our method to much larger TSPs.

We evaluate the solvable performance of these methods; the conventional ordinary tabu search, the exponential tabu search (Hasegawa et al., 2000), which reduces tabu effects exponentially, and the conventional stochastic search.

Furthermore, we also propose a controlling method of our chaotic neural network for easy applications to various problems.

The chaotic neural network model has many parameters and the solvable performance for combinatorial optimization problems depends on those values.

Although it is possible to make effective search by appropriate parameter values, manual finding of such good parameter value for each different problem is quite hard when we have to solve various problems, and therefore it is difficult to apply our method to practical problems.

Then, we construct a method to automatically tune those parameter values for realizing better search that suits for each individual problem.

As a result, we show that our method exhibits good performance even on very large TSPs up to an 85,900-city problem.

2. Novel chaotic searches for the TSP

2.1. The method using n² neurons for an n-city problem (two-dimensional method)

2.1.1. Tabu search memorizing paths

We utilize the 2-opt method for updating solutions of the TSP.

As shown in Fig. 2, the 2-opt algorithm inter-exchanges two paths, i-a(i) and j-a(j), with another two paths, i-j and a(i)-a(j), where a(i) and a(j) are cities next to I and j, respectively.

By this 2-opt updating, the solution is always preserved to be feasible.

It should be noted again that the way for updating a solution is completely different from the conventional chaotic dyna-mical approach (Chen & Aihara, 1995; Hasegawa et al., 1995; Nozawa, 1992; Yamada & Aihara, 1997) which is based on the Hopfield-Tank neural network and which often offers non-feasible solutions/

Fist, the basic tabu search of the proposed chaotic method is realized using a tabu list which memorizes paths that has been connected as follows: when the 2-opt exchange is done that links the cities i and j as shown in Fig.2, a pair of (i,j) is memorized in the tabu list.

It should be noted that the path ((a(i), a(j)) is also connected by this 2-opt exchange but this does not become tabu in the tabu in the tabu list introduced here. At each 2-opt updating, only a single interexchange is chosen, which offers the largest gain (most shortens or least lengthens the total tour) in all non-tabu moves whose corresponding pairs (i,j) and ((a(i), a(j)) are not stored in the tabu list. This tabu search requires the memories of the two-dimensional elements (i, j), because each path should be represented by two cities.

2.1.2. Tabu search neural network based on paths

Then, we implement the above tabu search on a neural network model using refractory effects. In the case of select-ing a path, there are n(n -1)/2 ways at each 2-opt updating.

Consequently, n(n-1)/2 kinds of two-dimensional ele-ments have possibilities to be memorized in the tabu list. In order to define the tabu effects of all these elements, n(n-1)/2 neurons are prepared and labeled by (i,j), (i = 1,…,n, j = i + 1,…n) as shown in Fig. 2.

Each (i,j)th neuron corresponds to the path between the cities i and j (in this paper, (i, j) and (j, i) are the same). If the (i, j)th neuron fires, the corresponding paths, i – j and a(i) – a(j) are connected with the 2-opt exchange as shown in Fig. 2. In the case of the tabu search, this firing, where s is the tabu list size.

Then, a neural network, which behaves the same as the conventional ordinary tabu search, can be realized by the following equations with a synchronous updating:

Where Δij(t) is the gain of the objective function value (the tour length) offered by the 2-opt exchange which links cities i and j, namely, Δij(t) = D0(t) -Dij(t), where D0(t) and Dij(t) are the length of the tour at time t and that with the 2-opt exchange which links cities i and j (Fig. 2), β, the scaling parameter of the gain effect, kr, the decay parameter of the tabu effect, α, the scaling parameter of the tabu effect, xij(t) are the internal states of the (i, j)th neuron at time t corresponding to the again effect and the tabu effect of the path between I and j, respectively.

If the summation of internal states,

{εij(t + 1) + ηa(i)a(j)(t+1) + ηij(t + 1)}

If the largest in all neurons, this (i, j)th neuron fires and the path between cities I and j is connected by the 2-opt exchange. For memorizing the connection of this path, xij(t + 1) is set to 1, and the outputs of al other neurons xkl(t + 1), (k,l) ≠ (i, j) and (k, l) ≠ (a(i), a(j)) are set to 0.

In this neural network, if the (i, j)th neuron has fired within previous s iterations, a firing of this (i, j)th neuron is avoided with a sufficiently large positive α in Eq. (2).

Similarly, a previous firing of the (a(i), a(j))th neuron also depresses a firing of the (i, j)th neuron εa(i)a(j)(t + 1) in Eq. (3).

The gain effect is externally applied to each neuron through εij(t), and firing of a neuron with a larger gain becomes easier by Eq. (1).

This neural network includes various methods that have different tabu effects, depending on the parameter values.