Solving Min-Max Multiple Traveling Salesman Problems by Chaotic Neural Network

Takafumi Matsuura and Kazumiti Numata

Department of Management Science, Faculty of Engineering, Tokyo University of Science
1-3 Kagurazaka, Shinjuku-ku, Tokyo 162-8601, Japan
Email: matsuura@ms.kagu.tus.ac.jp, numata@ms.kagu.tus.ac.jp

Abstract—In this paper, for solving min-max multiple traveling salesman problems (min-max mTSP), we propose two heuristic methods: a tabu search with CORSS-exchange and a chaotic search which controls execution of CROSS-exchange. In earlier studies, it has already been shown that the chaotic search shows better performances than the tabu search for $\mathcal{NP}$-hard combinatorial optimization problems. However, it is not clear that the chaotic search is better than the tabu search for min-max type problem. The simulation results for min-max mTSP show that the chaotic search exhibits also higher performance than the tabu search. In addition, the chaotic search shows better performances than the conventional methods: a team ant colony optimization algorithm and a method by using competition-based neural network.

1. Introduction

The traveling salesman problem (TSP) is one of the most famous and well-studied $\mathcal{NP}$-hard combinatorial optimization problems. In the TSP, a set of n cities and a distance matrix $D = (d_{ij})$ between cities i and j are given. A salesman starts from a city, must visit each city exactly once and come back to the starting city. The goal of the TSP is to find the shortest tour.

In the real world, a starting city corresponds to salesman's office and the other cities correspond to customers or clients. If the number of customers increases, all customers cannot be visited in a day (or within the time limit) by a single salesman. One of the simple idea for resolving the problem is that the customers are visited by multiple salesmen: the customers are divided into many groups and customers in each group are visited by a single salesman. This traveling problem has been formulated as a multiple traveling salesman problem (mTSP) [1, 2]. In the mTSP, m salesmen visit a set of n cities, and each salesman starts from a city called depot and goes back to the depot. In the mTSP, each city must be visited exactly once by only one salesman. Then, the goal of the mTSP is to minimize a total length of m tours.

The general objective of the mTSP minimizes the total length of m tours. However, it is important to consider not only the total cost but also a cost of each salesman, because, in a real situation, if length of one salesman is much longer than that of the other salesmen, he feels discontented. Therefore, França et al. proposed a min-max multiple traveling salesman problem (min-max mTSP) [3]. An objective of the min-max mTSP is to minimize a cost of the most expensive route among all salesmen. In this paper, we only deal with the min-max mTSP. Various algorithms for solving the min-max mTSP have been proposed [4, 5]. Somhom et al. proposed a method by using competition-based neural network [4] and Vallivaara have used an ant colony optimization algorithm [5].

As for heuristic algorithms for solving the combinatorial optimization problems, many algorithms are proposed, for example, a tabu search [6, 7] and a chaotic search [8–14]. In earlier studies, it has already been shown that the chaotic search shows better performances than the tabu search [9–14]. However, it is not clear that the chaotic search is better than the tabu search for the min-max type of problem. Therefore, in this paper, we propose a method by using the tabu search and a method by using a chaotic neural networks. As a result, the chaotic search method exhibits also higher performance than the tabu search. In addition, the chaotic search shows better performances than the conventional methods.

2. Min-Max Multiple Traveling Salesman Problem

In a min-max multiple traveling salesman problem (mTSP), a set of cities $V = \{1, 2, ..., n\}$ ($\{1\}$ is a depot), distance d_{ij} ($i, j \in V$) between cities i and j, and a set of salesmen $K = \{1, 2, .., m\}$ are given. Each city is visited exactly once by only one salesman. An objective of the min-max mTSP is to minimizes the length of the most expensive tour among all salesmen. x_{ijk}, y_{ik}, f_{ijk} are decision variables. When a salesman k visits a city j directly after a city i, x_{ijk} takes 1, otherwise 0. When a salesman k visits city i, y_{ik} takes 1, otherwise 0. f_{ijk} is non-negative integer decision variables to eliminate sub tours. Using these notations, a formulation of the min-max mTSP is described as follow:

$$\min \quad \max_k \left(\sum_{j=1}^{n} \sum_{j=1}^{n} d_{ij} x_{ijk} \right) \tag{1}$$

$$\text{s.t} \quad \sum_{k=1}^{m} \sum_{j=2}^{n} x_{1jk} = m \tag{2}$$

$$\sum_{k=1}^{m} \sum_{h=2}^{n} x_{h1k} = m \tag{3}$$

$$\sum_{k=1}^{m} y_{ik} = 1 \qquad \forall i \in V \tag{4}$$

$$\sum_{h=1}^{n} x_{hik} = y_{ik} \qquad \forall k \in K, \forall i \in V \tag{5}$$

$$\sum_{j=1}^{n} x_{ijk} = y_{ik} \qquad \forall k \in K, \forall i \in V \tag{6}$$

$$f_{ijk} \leq (n-1)x_{ijk} \qquad \forall i, j \in V, \forall k \in K \tag{7}$$

$$\sum_{k=1}^{m}\sum_{i=1}^{n} f_{i1k} = n-1 \tag{8}$$

$$f_{1jk} = 0 \qquad \forall k \in K, \forall i \in V \tag{9}$$

$$\sum_{j=1}^{n} f_{ijk} - \sum_{h=1}^{n} f_{hik} = y_{ik} \qquad \forall i \in V, \forall k \in K \tag{10}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i, j \in V, \forall k \in K \tag{11}$$

$$y_{ik} \in \{0,1\} \qquad \forall i \in V, \forall k \in K \tag{12}$$

$$f_{ijk} \geq 0 \qquad \forall i, j \in V, \forall k \in K \tag{13}$$

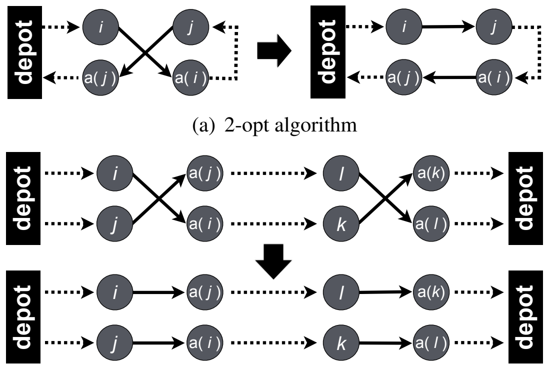
Eq.(1) represents the objective function that minimizes the maximum length of the tour. Eqs.(2) and (3) expresses that each salesman goes to any city from a depot and comes back to the depot from any city. Eq.(4) represents that each city is exactly visited by one salesman. Eqs.(5) and (6) represents that if a salesman k visits a city i, the city i is visited by the salesman k from exactly one city h, and the salesman goes to exactly one city j from the city i. Eqs. (7)~(10) are constraint to eliminate a sub-tour. Eqs. (11)~(13) are constraint of decision variables.

3. Proposed methods

3.1. Local Search Methods

To improve length of the longest tour, we used 2-opt algorithm and CROSS-exchange. The 2-opt algorithm exchanges two paths for other two paths in the same tour (Fig.1(a)). On the other hand, the CROSS-exchange exchanges a partial tour in one tour and a partial tour of the other tours (Fig.1(b)). Local search methods move from a current solution to a better solution in neighborhoods of the current solution until an optimal or a local optimal solution is found. In the case of the min-max mTSP, if length of all tours after applying the CROSS-exchange is shorter than that of the current longest tour, the current solution moves to a new solution. However, in general, local search methods cannot find optimal solutions due to the local minimum problem. To resolve the problem, we propose two methods: a method by using a tabu search [6, 7] and a method by using a chaotic dynamics [8].

To overcome local optimum solutions, a current solution must be moved to the best improved solution in the neighborhoods, even if the best improved solution is worse than the current solution. Here, $p_{\max}$ is the longest tour in the current solution. To improve the length of $p_{\max}$, a partial tour in $p_{\max}$ and a partial tour in other tour p are exchanged by the CROSS-exchange. Then, after applying the CROSS-exchange, $p_{\max}$ and p change to $\hat{p}_{\max}$ and $\hat{p}$, respectively. The neighborhood of the current solution satisfies the following condition: $f(p_{\max}) > f(\hat{p}_{\max})$, where $f(p_{\max})$ is the length of $p_{\max}$. The best improved solution is selected from the neighborhoods so that $\Delta = f(p_{\max}) - f(\hat{p})$ is the maximum.



(a) 2-opt algorithm

(b) CROSS-exchange

Figure 1: Graphical representation of local search methods. In this example, *a(i)* is the next city to *i*. In (a), two paths (*i*–*a(i)* and *j*–*a(j)*) are deleted from the current tour, then new two paths, *i*–*j* and *a(j)*–*a(i)*, are added. In (b), a partial tour $a(i) \cdots k$ in one tour and a partial tour $a(j) \cdots l$ in the others are exchanged.

3.2. Tabu Search Method

The tabu search has been proposed by F. Glover as a general combinatorial optimization technique [6, 7]. One of the essential idea of the tabu search is that a deterministic approach can avoid a local minimum by using a list of prohibited solutions known as a tabu list. A previous solution is added to the tabu list and is not allowed to move back to it for a certain temporal duration called a tabu tenure. In a proposed method by using the tabu search, the tabu list is constructed as follows: when the CROSS-exchange is executed, a pair of cities i and j (Fig.1(b)) is memorized in the tabu list.

The procedure of the tabu search with CROSS-exchange is described as follows:

1. An n-city min-max mTSP is given. An initial solution is randomly constructed and each tour is improved by the 2-opt algorithm.
2. To minimize the length of the longest tour, the CROSS-exchange is applied to the solution until no further improvement can be obtained.
3. A current solution is improved by the tabu search with CROSS-exchange.
   (a) The best improved solution is selected from neighborhoods of the current solution.
   (b) An operation of the CROSS-exchange corresponding the best improved solution is executed. The pair of cites i and j is memorized in the tabu list. The executions of the CROSS-exchange by using cities i and j are prohibited for τ iterations. Then, if the best solution is obtained, the

CROSS-exchange and 2-opt algorithm are applied for the best solution until no further improvement can be obtained.

4. The 2-opt algorithm is applied to each tour.
5. Finish one iteration, and repeated the steps 3 and 4 for sufficiently many times.

3.3. Chaotic Search Method

In a chaotic search method (CS), execution of the CROSS-exchange is controlled by chaotic dynamics. To realize chaotic dynamics, we use a chaotic neural network composed of chaotic neurons [8]. In the CS, ${}_{n+m}C_2$ neurons are needed to solve an n-city problem with m salesmen. Figure 2 shows how to construct the chaotic neural network. Each neuron corresponds to a selection of two cities in the CROSS exchange (Fig.1(b)). If a neuron fires, an operation of the CROSS exchange corresponding to the neuron is executed.

The output of the ijth chaotic neuron is defined by $x_{ij}(t) = f(y_{ij}(t))$, where $f(y) = 1/(1+\exp(-y/\epsilon))$, and $y_{ij}(t)$ is an internal state of the ijth chaotic neuron at time t. If $x_{ij}(t) > \frac{1}{2}$, the ijth chaotic neuron fires at the time t, otherwise resting. The internal state $y_{ij}(t)$ is decomposed into two parts, $\xi_{ij}(t)$ and $\zeta_{ij}(t)$. Each component represents a different factor to the dynamics of neurons, a gain effect and a refractory effect, respectively.

The gain effect is expressed as:

$$\xi_{ij}(t+1) = \max_{k,l}\{\beta(t) \times \Delta_{ijkl}(t)\} \qquad (14)$$

$$\beta(t+1) = \beta(t) + \frac{q}{\frac{1}{n}\sum_{i=1}^{n}|\Delta_{ijkl}(t)|}, \qquad (15)$$

where $\Delta_{ijkl}(t)$ is a difference between the length of a current longest tour and that of a new tour after applying the CROSS-exchange. $\beta(t)$ is a scaling parameter at time t and increases with time t. In this way, we can gradually restrict the search space in a similar way as the simulated annealing [15]. q is the scaling parameter of the annealing effect and n is the number of cities. To obtain the same range of $\xi_{ij}(t)$ for all instances, $\beta(t)$ is adjusted by using values of $\Delta_{ijkl}(t)$ because the range of $\Delta_{ijkl}(t)$ depends on the city distribution of each instance.

The refractory effect has a similar memory effect as the tabu search [6, 7]. The same selection of a solution can be avoided by the refractory effect. The refractory effect is expressed as:

$$\zeta_{ij}(t+1) = -\alpha \sum_{d=0}^{t} k_r^d x_{ij}(t-d) + \theta \qquad (16)$$

where α is a scaling parameter of the refractory effect, k_r is a decay parameter, and θ is a threshold value. If a chaotic neuron has fired in the past, Eq. (16) becomes negative. Therefore, the refractory effect inhibits the firing of the chaotic neuron in response to its past firing history.

The procedure of the chaotic search is described as follows:

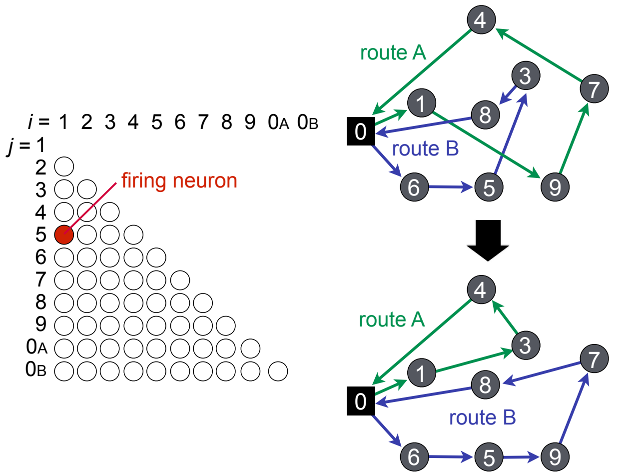


Figure 2: How to construct a chaotic neural network to an n-city mTSP ($m = 2$). In this figure, circles represent neurons. Each neuron corresponds to a selection of two cities i and j in the CROSS exchange (Fig.1)

1. An n-city mTSP is given. An initial solution is randomly constructed and each tour is improved by the 2-opt algorithm.
2. To minimize the length of the longest tour, the solution is improved by the CROSS-exchange until no further improvement can be obtained.
3. A current solution is improved by the CROSS-exchange driven by a chaotic dynamics.
   (a) Internal state of all neurons is updated.
   (b) The ijth neuron whose internal state is maximum is selected.
   (c) If the ijth neuron fires ($x_{ij}(t+1) > \frac{1}{2}$), corresponding CROSS-exchange is carried out. Then, if the best solution is obtained, the CROSS-exchange and 2-opt algorithm are applied for the tour until no further improvement can be obtained.
4. The 2-opt algorithm is applied to each tour.
5. Finish one iteration, and repeated the steps 3 and 4 for sufficiently many times.

4. Simulations and Results

To investigate the performances of the proposed methods we solved benchmark instances: eil51, eil76, eil101, kroA200, and fl417 [16]. These problems are used to investigate the performances of the conventional methods: a team ant colony optimization (TACO) [5] and a competition-based neural network (cNN) [4].

The tabu tenure of the tabu search (TS) is set to various values. The values of parameters in the chaotic search are set to various values. The values of parameter α in Eq.(16) are set to between from 0.1 to 1.5 by step size 0.1. The values of parameter k_r are set to between from 0.1 to 0.9 by step size 0.1. The values of parameters $\beta(0)$ and q are set to 0.0 and 0.00005 for all instances (Eq.(15)). The values of

Table 1: Computational results of chaotic search (CS), tabu search (TS), team ant colony optimization (TACO), and competition-based neural network (cNN).

instance	m	LS Best	LS Ave.	CS Best	CS Ave.	$^*(\alpha, k_r)$	TS Best	TS Ave.	$^{**}\tau$	TACO Best	TACO Ave.	cNN Best	cNN Ave.
eil51	2	244	269.9	**223**	224.9	(0.9,0.2)	**223**	224.1	7	224	224.7	247	248.7
	3	185	202.4	159	**161.0**	(1.2,0.4)	159	161.0	7	159	163.0	170	172.0
	4	156	171.6	130	131.5	(1.0,0.4)	130	**130.6**	3	130	131.6	136	137.3
eil76	2	320	344.9	**277**	283.1	(1.0,0.2)	**277**	281.9	5	278	**281.0**	289	292.0
	3	226	256.7	**193**	**196.0**	(1.1,0.2)	**193**	196.4	10	194	199.1	205	120.5
	4	181	212.8	**158**	160.6	(1.3,0.2)	**158**	**160.1**	4	161	163.6	159	162.8
eil101	2	374	403.1	330	336.7	(1.0,0.2)	**327**	336.0	20	**327**	**330.0**	340	344.7
	3	256	297.4	**225**	232.1	(1.1,0.1)	226	231.7	4	226	**227.8**	232	236.0
	4	214	244.4	**177**	181.7	(1.3,0.3)	**177**	**180.8**	6	178	181.0	187	189.7
kroA200	2	18518	19975.3	15454	15778.1	(0.9,0.2)	15503	15905.0	33	**15376**	**15499.3**	17353	11532.8
	3	13308	14958.8	**10771**	**11073.6**	(1.1,0.3)	10795	11180.9	44	10997	11186.5	11502	9276.3
	4	10760	12242.3	**8650**	**8869.8**	(0.8,0.5)	8674	8894.4	17	8917	9134.4	10433	7516.8
fl417	2	8285	9107.0	6742	**6898.7**	(0.5,0.6)	**6715**	7010.6	43	6804	6962.8	7207	7266.8
	3	6421	7460.5	**5134**	**5397.1**	(0.4,0.7)	5229	5462.0	36	5296	5470.0	5618	5902.5
	4	5777	6593.4	**4703**	**4855.5**	(1.0,0.5)	4716	4875.3	37	4844	5073.8	5032	5109.5

$*(\alpha, k_r)$: The values of parameters α and k_r when the best and the average solution are obtained by the CS.
$**\tau$: The tabu tenure when the best and the average solutions are obtained by the TS.

parameters ϵ and θ are set to 0.01 and 1.0, respectively. The proposed methods is applied for 1,000 iterations, namely, $1,000$ solutions are obtained in one trial. We compared the average length of the obtained longest tour with 30 different initial conditions and the results of conventional methods in Refs. [4] and [5].

Table 1 shows the results of each method. From Table 1, the performances of the local search is improved by the tabu search and the chaotic dynamics. The average lengths obtained by the proposed methods are smaller than that of the conventional methods for many instances. In addition, CS and TS find new best result for almost all instances.

However, when the number of salesman is $m = 2$, TACO obtains the best result for some instances. One of the possible reasons for the lower performance is that a method for improving the length of a single tour is different between proposed methods and TACO. In the proposed methods, the 2-opt algorithm is used. On the other hand, in TACO, the 3-opt algorithm, which is more powerful algorithm than the 2-opt algorithm, is used.

From Table 1, CS obtains better solutions than TS for many instances. These results indicate that the chaotic search is also effective than the tabu search. for min-max type of combinatorial optimization problem.

5. Conclusions

In this paper, for solving the min-max mTSP, we proposed two new methods : CROSS-exchange with tabu search and CROSS-exchange driven by chaotic neurodynamics. From the computational results, we confirmed that the proposed methods obtain better solutions than the conventional methods. In addition, the chaotic search shows better performances than the tabu search method. In the future work, to improve the proposed methods, we consider how to tune the values of parameters.

Acknowledgments

The research of T.M. is partially supported by Grant-in-Aid for Young Scientists (B) (No.25870770) from the JSPS.

References

[1] M. Bellmore and S. Hong (1974), *J. Assoc. Comput. Machinery*, **21**, 500-504.

[2] S. Hong and M.W. Padberg (1977), *Operations Research*, **25**, 871-874.

[3] P.M. França et al. (1995), *Transactions of Science*, **29**(3), 267–275.

[4] S. Somhom, A. Modares, and T. Enkawa (1999), *Computers & Operations Research*, **26**, 395–407.

[5] I. Vallivaara (2008), *Proc. of the 27th IASTED Int. Conf. on Modelling, Identification and Control*, 387–392.

[6] F. Glover (1989), *ORSA J. on Comput.*, **1**, 190–206.

[7] F. Glover (1990), *ORSA J. on Comput.*, **2**, 4–32.

[8] K. Aihara, T. Tanabe, and M. Toyoda (1990), *Phys. Lett. A*, **144**(6-7), 333–340.

[9] M. Hasegawa, T. Ikeguchi, and K. Aihara (1997), *PRL*, **79**, 2344–2347.

[10] M. Hasegawa, et al. (2002), *Neural Networks*, **15**, 271–283.

[11] M. Hasegawa, et al. (2002), *EJOR*, **139**, 543–556.

[12] T. Matsuura and T. Ikeguchi (2008), *Proc. of NOLTA 2008*, 77–80.

[13] T. Matsuura and T. Ikeguchi (2010), *Journal of NOLTA*, **1**(1), 207–220.

[14] T. Ikeguchi, S. Motohashi, and T. Matsuura (2010), *Proc. of NOLTA 2010*, 293–296.

[15] S. Kirkpatrick, et al. (1983): *Science*, **220**, 671–680.

[16] TSPLIB. http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/