2010 International Symposium on Nonlinear Theory and its Applications
NOLTA2010, Krakow, Poland, September 5-8, 2010

Double-Assignment Method Driven by Chaotic Neurodynamics for Quadratic Assignment Problems

Kazuaki Shibata† and Yoshihiko Horio†

†Graduate School of Engineering, Tokyo Denki University
Chiyoda-ku, Tokyo, 101-8457, Japan
Email: 09kme26@ms.dendai.ac.jp

Abstract—A quadratic assignment problem (QAP) is one of the NP-hard combinatorial optimization problems. Local search methods such as the 2-opt method are used to solve the QAPs. However, they are usually trapped in the local minima. In order to solve this problem, heuristic methods such as the tabu search have been proposed. In particular, the exponential chaotic tabu search shows exellent performance. On the other hand, we proposed a double-assignment method which searches sub-optimal solutions through a solution space including infeasible solutions. We have shown that the proposed method is superior to the 2-opt algorithm in solving the QAPs. In the double-assignment method, an infeasible solution is first constructed from the initial solution by assigning two elements to one index. Then, a feasible solution is composed from the infeasible solution.

In this paper, we improve the double-assignment method for the QAPs by introducing chaotic neurodynamics. We show numerical simulation results comparing the performance of the improved method and the original double-assignment method.

1. Introduction

A quadratic assignment problem (QAP) [1] is one of the NP-hard combinatorial optimization problems. Local search methods such as the 2-opt method are used to solve the QAPs. However, they are usually trapped in the local minima. In order to solve this problem, heuristic methods such as the tabu search have been proposed [2]. In particular, the exponential chaotic tabu search [3] shows exellent performance. In the exponential chaotic tabu search, the chaotic neurodynamics from a chaotic neural network drive the 2-opt algorithm. Our final goal is to apply the Lin-Kernighan algorithm [4], which is an effective local search method for traveling salesman problems (TSPs) [5], to the exponential chaotic tabu search for the QAPs instead of the 2-opt algorithm. To this end, we have proposed a double-assignment method based on the Lin-Kernighan algorithm [6]. In addition, we have shown that the double-assignment method has higher performance than the 2-opt method in solving the QAPs [6]. On the other hand, the Lin-Kernighan algorithm was driven by the chaotic dynamics to have good performance in solving the TSPs [7].

In this paper, we improve the double-assignment method for the QAPs by introducing the chaotic dynamics from a chaotic neural network. Numerical simulation results for the improved method are shown.

2. Quadratic Assignment Problem

We breifly explain the QAP with a plant location problem of size N as an example. The purpose of the plant location problem is to find the location of the plants that minimizes the cost given by Eq. (1) when the all plants are assigned to different cities.

$$F(\boldsymbol{p}) = \sum_{i=1}^{N}\sum_{j=1}^{N} a_{ij} b_{p(i)p(j)} \tag{1}$$

We introduce a permutation $\boldsymbol{p}$ which gives the combinations of the cities and plants as in Eq. (2).

$$\begin{array}{lccccccccc} \textit{index}: & 1 & & 2 & \ldots & i & \ldots & j & \ldots & N \\ \boldsymbol{p}: & \{\ p(1), & & p(2), & \ldots, & p(i), & \ldots, & p(j), & \ldots & p(N)\} \end{array} \tag{2}$$

The *index* of $\boldsymbol{p}$ expresses one of the cities, and the element of $\boldsymbol{p}$, $p(i)$, shows the plant assigned to the city i. In Eq. (1), a_{ij} gives the distance between the city i to the city j, and b_{ij} is the flow between the plant i and the plant j. The distance matrix $\boldsymbol{A}$, and the flow matrix $\boldsymbol{B}$ are given as in Eqs. (3) and (4), respectively.

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{N1} & \cdots & a_{NN} \end{bmatrix} \tag{3}$$

$$B = \begin{bmatrix} b_{11} & \cdots & b_{1N} \\ \vdots & \ddots & \vdots \\ b_{N1} & \cdots & b_{NN} \end{bmatrix} \tag{4}$$

3. Double-Assignment Method

The double-assignment method [6] searches sub-optimal solutions through a solution space including infeasible solutions. First, this method constructs an infeasible solution by assigning two plants to one city. Then, we reconstruct a feasible solution by reassigning one of the two plants to an empty city. The procedure of the double-assignment method is as follows.

Step 0 : Let $m = 0$ and $n = 0$, where m is the total number of runs in Step 1 and Step 2, and n is the number of runs in Step 3.

Step 1 : This step is shown in Fig. 1. We first choose a plant i at random. Then, we choose a city j from the cities to which the plant i is not assigned. Next, we assign the plant i to the city j. After that, we make $m = 1$. Moreover, the city to which the city i was originally assigned is denoted as y_1.

Step 2 : We increment m by 1, that is, $m = m + 1$. We choose a city y_m among the cities which have only one plant. We choose the city x_m that gives the minimum increase in the cost from the cities excluding the cities x_1 to x_{m-1} and y_1 to y_m. Then, we reassign the plant in the city y_m to the city x_m. If $m < N/2$, we repeat **Step 2**, otherwise we proceed to **Step 3**. **Step 2** is illustrated in Fig. 2.

Step 3 : We increment n by 1 as $n = n + 1$. We choose a plant from the plants assigned to the cities x_1 to x_m, which gives the minimum increase in the cost when it is assigned to the city y_n. Then, we assign the chosen plant to the city y_n. Moreover, we denote the city where the chosen city resided as z_n. If $n = m$, we complete the algorithm, otherwise we repeat **Step 3**.

Changes of the cost in **Step 1** and **Step 2** can be written by Eqs. (5) and (6), respectively. In Eqs. (5) and (6), ΔF^1_{ij} is the increase of the cost when the plant in the city i is assigned to the city j in **Step 1**, and ΔF^2_{ij} is that in **Step 2**, respectively. In these equations, $p(i)$ and $p(j)$ are the plants which are assigned to the city i and city j, respectively. Moreover, $p_1(i)$ is the plant assigned to the city i at the beginning, and $p_2(i)$ is the plant newly assigned to the city i.

$$\begin{aligned}\Delta F^1_{ij} &= -\left(a_{ij}b_{p(i)p(j)} + a_{ji}b_{p(j)p(i)}\right) \\ &+ \sum_{k=1,k\neq i,j}^{N}\left(b_{p_1(i)p_1(k)}\right)\left(a_{kj} - a_{ki}\right) \\ &+ \sum_{k=1,k\neq i,j}^{N}\left(b_{p_1(k)p_1(i)}\right)\left(a_{jk} - a_{ik}\right) \qquad (5)\end{aligned}$$

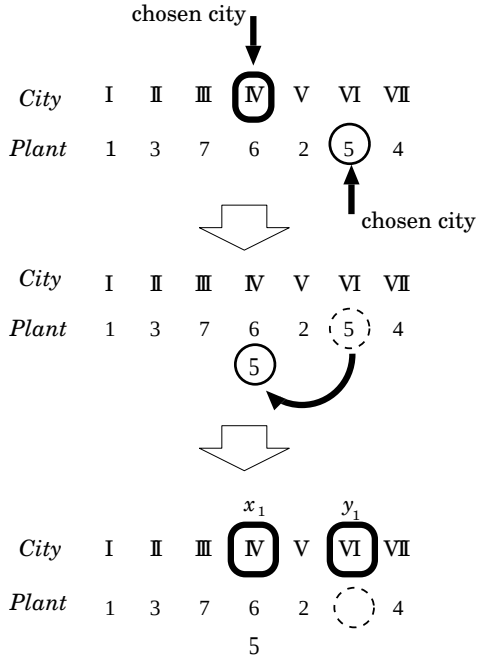

Figure 1: **Step 1** of the proposed method (size-7 QAP). If the randomly chosen city is IV, and the chosen plant is 5, we assign the plant 5 to the city IV. However, we do not assign the plant 6, which is already assigned to city IV, to the city VI.

$$\begin{aligned}\Delta F^2_{ij} &= -\left(a_{ij}b_{p(i)p(j)} + a_{ji}b_{p(j)p(i)}\right) \\ &+ \sum_{k=1,k\neq i,j}^{N}\left(b_{p_1(i)p_1(k)} + b_{p_1(i)p_2(k)}\right)\left(a_{kj} - a_{ki}\right) \\ &+ \sum_{k=1,k\neq i,j}^{N}\left(b_{p_1(k)p_1(i)} + b_{p_2(k)p_1(i)}\right)\left(a_{jk} - a_{ik}\right) \qquad (6)\end{aligned}$$

Furthermore, the increase of the cost in **Step 3** is given in Eq. (7). In Eq. (7), ΔF^3_{ij} represents the change in the cost when the plant i is assigned to the city j in **Step 3**. Moreover, l is the city to which the plant i was originally assigned. Moreover, $p(j)$ and $p(l)$ are the plants assigned to the city j and the city l, respectively.

$$\begin{aligned}\Delta F^3_{ij} &= a_{lj}b_{ip(l)} + a_{jl}b_{p(l)i} \\ &+ \sum_{k=1,k\neq i,j}^{N}\left(b_{ip_1(k)} + b_{ip_2(k)}\right)\left(a_{kj} - a_{kl}\right) \\ &+ \sum_{k=1,k\neq i,j}^{N}\left(b_{p_1(k)i} + b_{p_2(k)i}\right)\left(a_{jk} - a_{lk}\right) \qquad (7)\end{aligned}$$

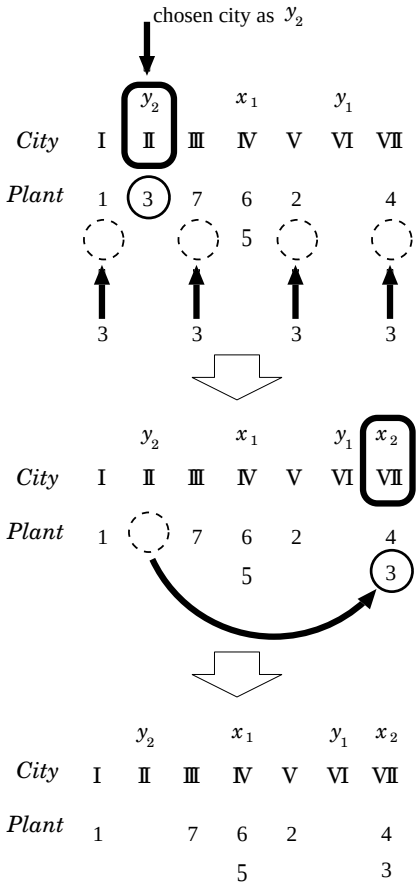

Figure 2: **Step 2** of the proposed method (size-7 QAP, $m = 1$). If the city Ⅱ is randomly chosen, we calculate the increases in the cost when the plant 3, which is assigned to the city Ⅱ, is temporarily assigned to the city Ⅰ, Ⅲ, Ⅴ and Ⅶ, respectively. If the increase in the cost is the lowest when the plant 3 is temporarily assigned to the city Ⅶ, we eventually assign the plant 3 to the city Ⅶ. We do not assign the plant 4 to the city Ⅱ which is already assigned to city Ⅶ, as in **Step 1**.

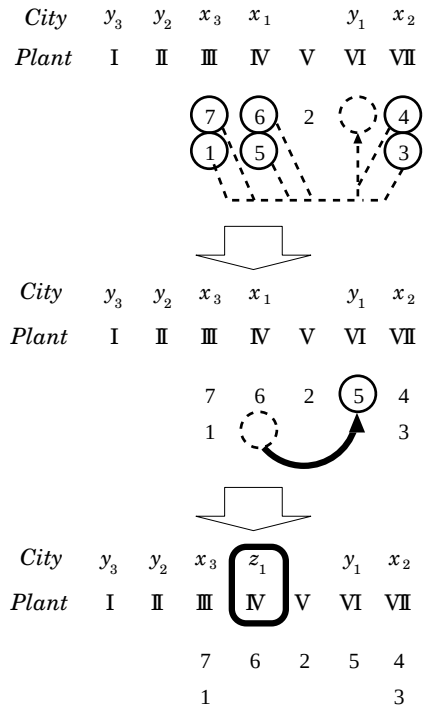

Figure 3: **Step 3** of the proposed method (size-7 QAP, $n = 1$). In the case when the randomly selected city is the city Ⅱ, we evaluate the increases in the cost when the plant 3, which is currently assigned to the city Ⅱ, is temporarily assigned to the cities Ⅲ, Ⅳ, Ⅵ and Ⅶ, respectively, which have only one city assigned. If the increase in the cost is the lowest when the plant 5 is temporarily assigned to the city Ⅳ, we eventually assign the plant 3 to the city Ⅶ.

4. Double-Assignment Method Driven by Chaotic Neurodynamics

In this section, we propose the double-assignment method driven by the chaotic neurodynamics. We use a chaotic neural network constructed by the chaotic neurons given by

$$\zeta_{ij}(t+1) = k_r\zeta_{ij}(t) - \alpha x_{ij}(t) + R \quad (8)$$

$$\xi_{ij}(t+1) = \beta(F_1^P(t) - F_{ij}^P(t)) \quad (9)$$

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

where, $\zeta_{ij}(t+1)$ is the refractory effect, $\xi_{ij}(t+1)$ is the gain effect, $x_{ij}(t+1)$ is the output of the (i, j)th neuron, β is a scaling parameter of the gain effect, k_r is a decay parameter of the refractory effect ($0 < k_r < 1$), α is a scaling parameter of the refactory effect ($\alpha > 0$), R is an external bias, and $f(y)$ is a sigmoidal output function of the neuron ($f(y) = 1/(1 + e^{-y/\epsilon})$). $F_1^P(t)$ represents the cost at time t, and $F_{ij}^P(t)$ is the cost of the assignment of the plant i to the city j through the double-assignment method.

In the proposed method, we asynchronously update the neural network as follows. To update the (i, j)th neuron, $\zeta_{ij}(t+1)$ and $\xi_{ij}(t+1)$ are calculated. To obtain $\xi_{ij}(t+1)$, the double-assignment method which assigns the plant i to the city j is executed. This temporal solution is denoted as P'. Then, if $x_{ij}(t+1) > \frac{1}{2}$, we update the current solution to P'. Here, $F_1^P(t) - F_{ij}^P(t)$ is normalized by $a_M b_M$, where $a_M = \max\{a_{ij}\}$ and $b_M = \max\{b_{ij}\}$. The order of neuron updates are $(1, 1) \longrightarrow (1, 2) \longrightarrow \cdots \longrightarrow (1, N) \longrightarrow (2, 1) \longrightarrow (2, 2) \longrightarrow \cdots \longrightarrow (2, N) \longrightarrow \cdots \longrightarrow (N, N-2) \longrightarrow (N, N-1) \longrightarrow (N, N)$, for the size N problem. When the (N, N)th neuron was updated, one iteration is completed.

5. Simulation Results

We compare the performance of the proposed method with the chaotic neurodynamics and the original double-assignment method through numerical simulations. In the original double-assignment method without chaotic dynamics, we first try all the combinations of the indices and elements for assignments. Then, we finally update the cur-

rent solution using the combination that gives the minimum cost. We define the above one update as one iteration with the original double-assignment method. Moreover, we use 5,000 iterations for one trial, and 30 trials are done for both methods.

Tables 1 and 2 show the average gaps from the optimal solutions, and the frequency of the optimam solution (OS), respectively. Table 3 shows the parameters for the chaotic neural network used in the proposed method.

From Table 1, the proposed method with the chaotic dynamics obtains better solutions for size-20 problems than the original double-assignment method except for Tai20b. However, Table 2 shows the original method is better than the proposed method for size-30 problems except for Lipa30a and Lipa30b.

6. Conclusions

We have proposed the double-assignment method driven by the chaotic neurodynamics. The proposed method was shown to have better performance than the conventional method for the QAPs of size of 20. However, the improvements have not been obtained for size-30 QAPs except for Lipa30a and Lipa30b. Therefore, we will decipher the reason for the above results, and further improve the proposed method.

Table 1: The average gap from the optimum solution, and the number of the optimum solution, (OS), obtained during 30 trials for the size-20 QAPs

	Original Method		Proposed Method	
Instance	Average Gap	OS	Average Gap	OS
Nug20	1.08949	1	0.21530	5
Tai20a	1.75716	0	0.93555	1
Tai20b	0.32451	12	0.63842	0
Had20	0.49889	1	0.27930	4
Rou20	1.42614	0	0.90682	0
Scr20	8.68429	0	1.60926	2
Chr20a	8.75000	0	1.17701	6
Chr20b	6.54482	1	2.18741	6
Chr20c	9.26319	2	0.15981	27
Lipa20a	1.01819	15	0.00000	30
Lipa20b	0.87076	28	0.00000	30

Table 2: The average gap from the optimum solution, and the number of the optimum solution, (OS), obtained during 30 trials for the size-30 QAPs

	Original Method		Proposed Method	
Instance	Average Gap	OS	Average Gap	OS
Nug30	1.59482	0	1.68408	0
Tai30a	2.62372	0	3.41508	0
Tai30b	0.30766	0	1.30766	0
Lipa30a	1.11575	0	0.98750	19
Lipa30b	5.36006	0	1.83971	0

Table 3: Parameter values for the chaotic neuron.

Instance	α	β	k_r	R
Nug20	0.7	0.7	0.4	0.1
Tai20a	1.0	0.7	0.7	0.1
Tai20b	1.0	1.0	0.1	0.1
Had20	0.1	0.7	0.1	0.1
Rou20	0.1	1.0	0.7	0.1
Scr20	1.0	1.0	0.1	0.1
Chr20a	1.0	0.7	0.1	0.1
Chr20b	1.0	0.7	0.1	0.1
Chr20c	1.0	0.7	0.1	0.1
Lipa20a	1.0	1.0	0.7	0.1
Lipa20b	1.0	1.0	0.7	0.1
Nug30	0.7	0.7	0.4	0.1
Tai30a	1.0	0.7	0.7	0.1
Tai30b	1.0	1.0	0.1	0.1
Lipa30a	1.0	1.0	0.7	0.1
Lipa30b	1.0	1.0	0.7	0.1

Acknowledgement

This work was supported by Kakenhi (20300085).

References

[1] R. E. Brekard et al, "QAPLIB - A quadratic assignment problem library,"
URL: http://www.opt.math.tu-graz.ac.at/qaplib/

[2] E. D. Taillard, "Robust tabu search for the quadratic assingnment problem," *Parallel Computing*, vol. 17, pp. 443–455, 1991.

[3] M. Hasegawa, T. Ikeguchi and K. Aihara, "A novel chaotic search for quadratic assignment problems," *European Journal of Operational Research*, vol. 139, pp. 543–556, 2002.

[4] S. Lin and B. Kernighan, "An effective heuristic algorithm for the traveling salesman problem," *Operational Research*, vol. 21. pp. 498–516, 1973.

[5] "The Traveling Salesman Problem," E. L. Lawer, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, eds., John Wiley and Sons, 1985.

[6] K. Shibata and Y. Horio, "An application of Lin-Kernighan algorithm to quadratic assignment problems," *IEICE Technical Report*, vol. 109, NLP2009-164, pp. 37–42, 2010.

[7] S. Motohashi, T. Matsuura and T. Ikeguchi, "Chaotic search method using the Lin-Kernighan algorithm for traveling salesman problems," in *Proceedings of International Symposium on Nonlinear Theory (NOLTA 2008)*, pp. 144–147, 2008.