A New Solution for the Symmetric Traveling Salesman Problem

Hiroshi KUBO*
Norio OKINO*

(Received July 31, 1974)

Abstract

Although the branch and bound technique is successful in obtaining an exact solution of traveling salesman problem, it is well known that the method which makes use of this technique can not be effectively applied to cases where the cost matrix is symmetric. A new method is and presented here which makes use of the branch and bound technique, differs essentially from the existing methods in the following two points: 1) to retain the symmetric property of the cost matrix during the reducing operations, and 2) to treat city pairs disregarding their directions. Our method reduced the computing time by about one-fifth of the existing solutions for 30 city problems, and it is expected that the larger the number of cities, the smaller this ratio would become. Further it is expected that our method can be easily extended to nonsymmetric problems.

1. Introduction

Exact solutions for the traveling salesman problem hitherto proposed are, in almost all cases, obtained by using the branch and bound technique. The branch and bound technique is often useful for solving difficult problems such as combinatorial problems. It consists of the following two fundamental features. 1) Dividing any problem into plural problems more constrained so that solving them all is equivalent to solving the original one (Branching). 2) At each problem evaluating the lower bound that is always smaller than the optimal solution of the problem (Bounding).

The general procedure for solving any provided problem by using the branch and bound technique is as follows: 1) Repeat dividing the problems under a certain strategy and to evaluate the lower bound of each problem until a feasible solution is obtained. 2) Regard this feasible solution as the upper bound and reject all problems whose lower bound is greater than this upper bound. 3) If there are any problems whose lower bound is smaller than the upper bound, then repeat the division of these problems, and continue to apply the same procedure. If this is not the case, the upper bound becomes the optimal solution of the original problem. All methods using the branch and bound technique

* Department of Precision Engineering, Faculty of Engineering, Hokkaido University, Sapporo, Japan

are classified by algorithms of branching and bounding respectively. Existing solutions for the traveling salesman problem are classified into the following large two groups by branching algorithms: One is the tour building method proposed by Little et al.[1] whose algorithm of branching is determined by selecting a certain city pair or prohibiting it. Another is the subtour elimination method proposed by Shapiro[2] and others whose algorithms are determined by prohibiting a subtour of the solution obtained by solving the assignment problem.

In the case of a symmetric cost matrix, however, these solutions require much longer computing time and much larger working memory than in the case of a nonsymmetric one. The new solution proposed in this paper can effectively solve the symmetric traveling salesman problem and comes under the tour building method. For preparations this paper begins by describing the formulation of the traveling salesman problem and Little's solution. The new solution and its computational experiments are also mentioned in the following section.

2. Traveling Salesman Problem

The traveling salesman problem is formulated as follows: The number of cities for a salesman to visit is written by n (nonnegative integer) and the cost matrix is given as $n \times n$ matrix $D=(d_{ij})$, where d_{ij} (nonnegative integer) is the distance measure from city i to city j. Let t be a tour (1) which is a closed path where a salesman can visit each of all n cities only once, with the ordered pair (i, j) representing the direct path from city i to city j. (i, j) is an element of the tour (1).

$$(1) \qquad t=\left\{(i_1, i_2), (i_2, i_3), \cdots, (i_n, i_1)\right\}$$

Let T be a set of all feasible tours t which are not contradictory to D. The traveling salesman problem is asking for the tour t^* and the cost c^* so that the cost of a tour (2) is minimized over T as shown in (3).

$$(2) \qquad c(t)=\sum_{(i,j)\in t} d_{ij}$$

$$(3) \qquad c^*=c(t^*)=\min_{t\in T} c(t)$$

The algorithm of the branching in Little's method is determined by selecting a certain city pair or prohibiting it. In a certain problem X, hereafter we shall refer to it as the node X, let S_X be a set of city pairs already selected and let R_X be a set of city pairs already prohibited or that of prohibiting subtours made by S_X. Then the algorithm of Little's branching divides the node X into the node Y selecting a certain city pair $(\bar{i}, \bar{j})$ and the node $\bar{Y}$ prohibiting it (see Fig. 1). The means of deciding the city pair $(\bar{i}, \bar{j})$ will be mentioned later. S_Y, R_Y, $S_{\bar{Y}}$ and $R_{\bar{Y}}$ are shown in Fig. 1, where $(\bar{k}, \bar{l})$ is to prohibit the new subtour which is made by selecting city pair $(\bar{i}, \bar{j})$.

When branching starts from the node X in Fig. 1, the operations begin by making the reduced matrix D_X as follows: Elements corresponding to R_X are

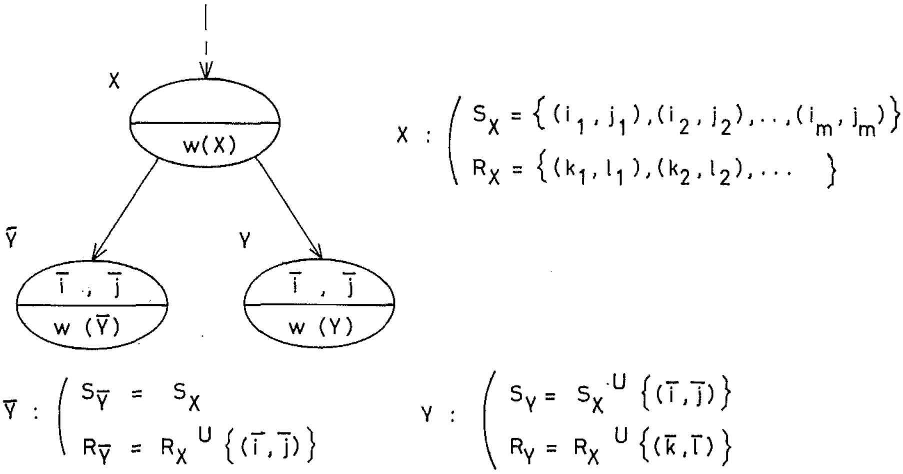

Fig. 1. Explanation of Branching

inhibited and the others are the same as the initial cost matrix D as shown in (4). For all rows not contained in I_X the minimum value of each row is subtracted from all elements of its row as shown in (5), and so are those for all columns not contained in J_X as shown in (6), where I_X is a set of rows which are eliminated by S_X, while J_X is a set of columns such as I_X. The lower bound $w(X)$ of the node X is given by (7).

(4) $\quad D_X'' = (d_{ij}'') \qquad d_{ij}'' = \infty \,(\text{if}\,(i,j) \in R_X) \quad \text{or } d_{ij}(\text{otherwise})$

(5) $\quad D_X' = (d_{ij}') \qquad d_{ij}' = d_{ij}'' - \min\limits_{p \notin J_X} d_{ip}'' \qquad \text{for every } i \notin I_X$

where $\quad I_X = \{i | (i,j)\, S_X\}, \qquad J_X = \{j | (i,j)\, S_X\}$

(6) $\quad D_X = (d_{ij}^X) \qquad d_{ij}^X = d_{ij}' - \min\limits_{p \notin I_X} d_{pj}' \qquad \text{for every } j \notin J_X$

(7) $\quad w(X) = \sum\limits_{(i,j) \in S_X} d_{ij} + \sum\limits_{i \notin I_X} \min\limits_{p \notin J_X} d_{ij}'' + \sum\limits_{j \notin Z_X} \min\limits_{p \notin I_X} d_{pj}'$

There is at least one zero element for each row and each column, not contained in I_X and J_X, respectively. The city pair used by branching from the node X is decided so that $(\bar{i}, \bar{j})$ maximizes $\theta(i,j)$ over these zero elements as shown in (8).

(8) $\quad \theta(i,j) = \min\limits_{p \notin I_X \cup \{i\}} d_{pj}^X + \min\limits_{p \notin J_X \cup \{j\}} d_{ip}^X \qquad \text{for every } d_{ij}^X = 0$

The lower bound of the node $\bar{Y}$ is evaluated by (9), and that of the node Y is evaluated by (11) after reducing the matrix D_Y from D_X as shown in (10), (5) and (6).

(9) $\quad w(\bar{Y}) = w(X) + \theta(i,j)$

(10) $\quad D_Y'' = (d_{ij}'') \qquad d_{ij}'' = \infty \,(\text{if}\,(i,j) = (\bar{k}, \bar{l})) \qquad \text{or } d_{ij}^X \text{ (otherwise)}$

(11) $\quad w(Y) = w(X) + \sum\limits_{i \notin I_Y} \min\limits_{j \notin J_Y} d_{ij}'' + \sum\limits_{j \notin J_Y} \min\limits_{i \notin J_Y} d_{ij}'$

It is clearly shown that solving the node X is equivalent to solving the node Y and the node $\bar{Y}$ in Fig. 1, and that if the number of city pairs in S_X is $n-2$, the node X certainly possesses a feasible solution. The strategy of branching is to always divide the node whose lower bound is minimal. Another strategy is to always branch from the node made immediately by selecting a certain city pair until the lower bound of the node becomes greater than the upper bound or the number of selected city pairs becomes $n-2$. In both strategies, the lower bound is evaluated by (5) when the branching starts from the node possessing the smallest lower bound anew; while the lower bound is evaluated by (11) when the branching starts from the node made immediately.

3. New Solution for The Symmetric Cost Matrix

In the enumeration method for the traveling salesman problem, when the cost matrix is symmetric ($d_{ij}=d_{ji}$) it is solved easier than the nonsymmetric problem because of not taking the city pairs into account with regard to their directions. But in the existing solutions using the branch and bound technique, it is much harder to solve the symmetric problem than the nonsymmetric one in contrast to the enumeration method. As for Little's method, it is reported that the city pair $(\bar{j}, \bar{i})$ is selected frequently at the node in which the reverse city pair $(\bar{i}, \bar{j})$ is already prohibited. As a result of selecting a reverse city pair, it frequently happens that the number of nodes is too large and the computing time is too long to solve the problem.

The new solution proposed in this paper differs from these existing solutions in the following two points: 1) Not taking the city pairs into account with repect to their directions as in the enumeration method. 2) Keeping the symmetric property of the cost matrix during reducing operations.

3.1 Branching

It is sufficiently clear that only the upper triangular elements of the cost matrix is operated when the cost matrix is symmetric at any time and is treated as disregarding their directions. Therefore the line Li, $i=1, 2, \cdots, n$ is defined on the matrix (see Fig. 2) and only the city pairs which are the elements on these lines are treated hereafter. The elements on the line are represented by (12).

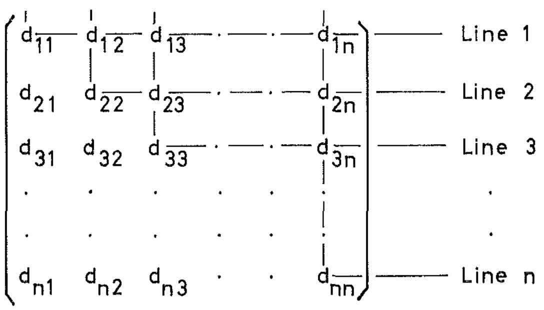

Fig. 2. Definiton of Line

$$d_{Li(j)} = d_{Lj(i)} = d_{ij} \qquad (\text{if } i \leq j) \text{ or } d_{ji} \text{ (otherwise)} \tag{12}$$

Let S_X (13) be a set of the city pairs selected from the elements on the lines at a certain node X, where m is their number. And let P_X (14) be a set of the partial paths which are made by the use of all city pairs in S_X only once regardless of their directions, where k_p and l_p are the edge cities of the same partial path and satisfy (15).

$$S_X = \{(i_1, j_1), (i_2, j_2), \cdots, (i_m, j_m)\} \qquad i_p < j_p \tag{13}$$

$$P_X = \{\langle i_1, j_1 \rangle, \langle i_2, j_2 \rangle, \cdots\} \qquad i_p < j_p \tag{14}$$

$$i_p \neq i_q, \quad i_p \neq j_q, \quad j_p \neq j_q \qquad \langle i_p, j_p \rangle, \langle i_q, j_q \rangle \in P_X \tag{15}$$

The city pairs selected in node X of the proposed method are taken to be all the pairs which are not contradictory to the partial paths in P_X regardless of their directions. An example of S_X, P_X and the selected city pairs of node X, is shown in Fig. 3. As for the selection of city pairs, it is evident that four nodes in Little's method are reduced to one node by the proposed method in the case of Fig. 3.

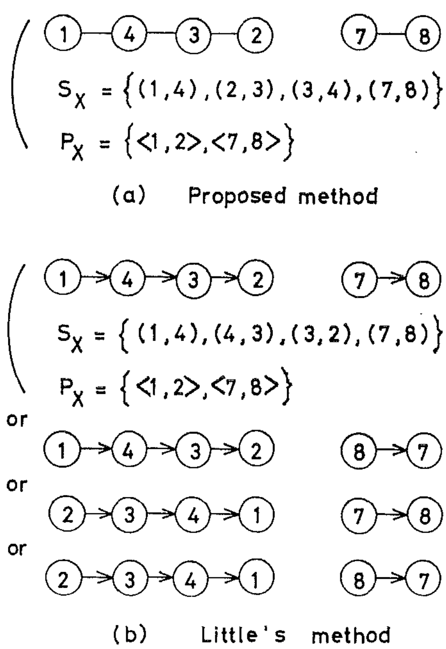

Fig. 3. Selected city pairs S_X and partial ipathes P_X in node X

Let R'_X be a set of city pairs inhibited already on lines. Then the city pairs in R'_X and their reverses are all inhibited in node X of the proposed method. Under these definitions of selecting and inhibiting the city pairs, it is proved easily that the branching in the proposed method remains optimal in the following two operations at each node: 1) Eliminating the lines from which two city pairs are selected. 2) Inhibiting the city pairs in R_X (16) on lines.

$$R_X = R'_X \cup \{(k_1, l_1), (k_2, l_2), \cdots\} \qquad \langle k_p, l_p \rangle \in P_X \tag{16}$$

When the number in S_X becomes $n-2$, then the node X is possessed of two feasible solutions as well as Little's method. The algorithm of the branching in the proposed method is determined either by selecting a certain city pair (i, j) on a line or the reverse (j, i), or by prohibiting both of them. That is, if S_X and R_X in a node X is not contradictory to a tour, then it is clearly shown that the node Y and the node $\bar{Y}$, selecting and prohibiting a city pair respectively which is not on any eliminated line and is not prohibited, are equivalent to the node X. City pairs in R_X and not in R'_X are to prohibit the subtours made by S_X.

3.2 Lower Bound

When branching is started from a certain node X with S_X and R_X in the proposed solution, the reduced matrix D_X is made as follows: Elements on lines corresponding to R_X are all prohibited and other elements on lines not eliminated are the same as the initial cost matrix D as shown in (17). Secondly the minimum value of each line is subtracted from all elements of its line for all lines not contained in I_X, as was shown in (18). Then the lower bound $v(X)$ of the node X is evaluated by (19), where I_X in (20) is a set of lines already eliminated for two city pairs to be selected and J_X in (21) is a set for only one city pair to be selected already.

(17) $$D'_X=(d'_{Li(j)}) \qquad d'_{Li(j)}=\infty \quad (\text{if } (Li(j))\in R_X)$$
$$= d_{Li(j)} \quad (\text{otherwise})$$

(18) $$D_X=(d^X_{Li(j)}) \qquad d^X_{Li(j)}=d_{Li(j)}-\min_{p\notin I_X} d_{Li(p)}$$
$$\text{for every } i\notin I_X$$

(19) $$v(X)=\sum_{(i,j)\in S_X} d_{ij}+\sum_{i\notin I_X} a_i{}^*\min_{j\notin I_X} d'_{Li(j)}$$

where $a_i=1$ (if $i\in J_X$) or 2 (otherwise)

(20) $$I_X=\{i|(Li(j))\in S_X \quad \text{and} \quad \langle Li(j)\rangle\notin P_X\}$$

(21) $$J_X=\{j|\langle Li(j)\rangle\in P_X\}$$

It is easily proved that $v(X)$ in (19) is the lower bound of the node X, by the aid of the fact that the difference between $v(X)$ and $w(X)$ in (7) is only the difference of the ways of subtracting the matrix in (5), (6) and (18). It will be explained by the use of the example shown in Fig. 4. When S_X is assumed to

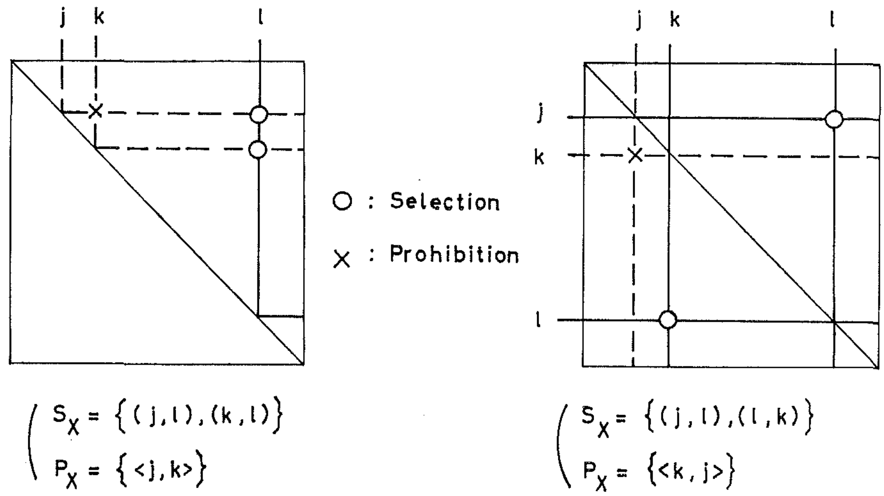

Fig. 4.(a) Proposed method **Fig. 4.(b)** Little's method

be a set of city pairs selected in the node X of the proposed method, P_X is obtained as shown in Fig. 4(a). This node corresponds to either the node of Little's method shown in Fig. 4(b) or the node given by reversing the order of S_X. The discussion of the reverse case in Fig. 4(b) is omissible evidently because of dealing with row and column alternatively. As for R_X, it is omissible because the city pairs prohibited in (a) and their reverse ones are both prohibited in (b). Both row and column corresponding to line l eliminated in (a) are eliminated in (b). Then their lines are not related to (17) as well as Little's method. Either row or column corresponding to line j or k selected only one city pair in (a) is eliminated in (b). Then the minimum values of their lines are added in (19) as well as in (7). Neither row nor column corresponding to line i not selected in any city pair in (a) are eliminated in (b), then the minimum values of their lines are added twice in (19) for row i and column i as well as in (7). That is, the difference between the second member of (19) and the second and third member of (7) is that of the sequence subtracting the minimum values. In the proposed method the sequence is at the same time for row and column, while in Little's method it is in order as all columns after all rows. The symmetry of the cost matrix is retained in the proposed method. Similarly the same is proved in the case where there are plural partial paths.

There are at least one zero element for each line not contained in I_X. The city pair used by branching from the node X is decided so that $(\bar{i}, \bar{j})$ maximizes $\phi(i,j)$ over these zero elements as shown in (22). The lower bound of the node $\bar{Y}$ prohibiting $(\bar{i}, \bar{j})$ is evaluated by (23). That of node Y selecting it is evaluated by (25) after reducing the matrix D_Y from D_X, as was shown in (24) and (18), where $(L\bar{k}(\bar{l}))$ is the city pair prohibiting the subtour which is made immediately by selecting city pair $(\bar{i}, \bar{j})$.

(22) $$\phi(i,j) = a_i{}^* \min_{p \in I_X{}^U\{j\}} d^X_{Li(p)} + a_j{}^* \min_{p \in I_X{}^U\{i\}} d^X_{Lj(p)} \qquad \text{for every } d^X_{Li(j)} = 0$$

(23) $$v(\bar{Y}) = v(X) + \phi(\bar{i}, \bar{j})$$

(24) $$D'_Y = (d'_{Li(j)}) \qquad d'_{Li(j)} = \infty \quad (\text{if } Li(j) = Lk(l))$$
$$= d^X_{Li(j)} \quad (\text{otherwise})$$

(25) $$v(Y) = v(X) + \sum_{i \in I_Y} a_i{}^* \min_{j \in I_Y} d'_{Li(j)}$$

It is proved in the same way as $v(X)$ in (19) that both $v(Y)$ and $v(\bar{Y})$ are the lower bounds of the node Y and $\bar{Y}$, respectively. When the branching is started from the node Y made immediately, the lower bound is evaluated by (25) after reducing the matrix by (24) repeatedly. And when the branching is started from the node finding the minimum value, it is evaluated by (19).

3.3 Algorithm

The flow-chart of the proposed solution is shown in Fig. 5. The working of the algorithm will be explained by tracking through the flow-chart.

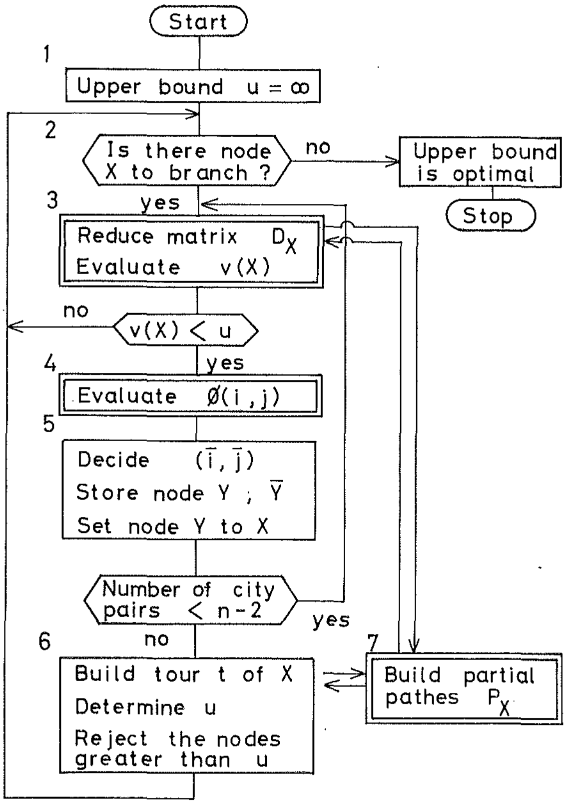

Fig. 5. Flow-chart of the proposed method

Box 1 starts the calculation by setting the upper bound u to infinity.

Box 2 decides the node X from which branching is started. If there is no node to be branched, then the algorithm is finished and the upper bound becomes the optimal solution.

Box 3 evaluates the lower bound $v(X)$ after reducing the matrix D_X. If the lower bound $v(X)$ is not smaller than the upper bound u then go to Box 2, else go to Box 4.

Box 4 evaluates the measure of selecting the city pair $\phi(i,j)$

Box 5 decides the city pair $(\bar{i}, \bar{j})$ used by branching from the node X stores the new nodes Y and $\bar{Y}$, and sets the node X to the node Y. If the number of city pairs selected becomes $n-2$, then go to Box 6, else go to Box 3.

Box 6 builds a tour by using of S_X, determines the upper bound u, and eliminates the nodes whose lower bound are greater than u.

Box 7 makes the partial paths and is called from Boxes 3 and 6.

The algorithm of the proposed method is different in only double boxes from that of the method which is used for the comparison with the proposed method in the next section. In each method, the strategy of branching is always to branch from the node made immediately, and by the use of an index table the time for seeking the minimum value for each row and column or each line is reduced as much as possible. The existing method improved by us as well as the proposed method makes it possible to obtain the value more precisely in less computing time. The method reduces the computing time to one-half of the time required by Little's method. Well shall call this the 'improved method' in the next section.

4. Computational Experiments

The proposed method is examined and will be compared with the improved method in relation to the computing time and relation to the number of nodes by solving many examples. The computer used in the experiments is FACOM 230-60 (Hokkido University Computing Center) and the program is coded by FORTRAN. The cost matrixes of the examples are made by the use of random numbers generated by the computer. An example of 10 city problemsis shown

Table 1. An example of cost matrixes

i \ j	1	2	3	4	5	6	7	8	9	10
1	∞	8	3	4	9	17	16	18	10	6
2	8	∞	2	8	16	18	19	2	9	10
3	3	2	∞	14	3	7	10	10	7	15
4	4	8	14	∞	19	1	20	16	10	14
5	9	16	3	19	∞	1	1	11	8	5
6	17	18	7	1	1	∞	17	18	15	14
7	16	19	10	20	1	17	∞	0	4	18
8	18	2	10	16	11	18	0	∞	8	3
9	10	9	7	10	8	15	4	8	∞	12
10	6	10	15	14	5	14	18	3	12	∞

Table 2. Computing time (10 ms)

Number of cities (Number of examples)		10 (10)	20 (10)	30 (2)
Proposed method	minimum	12	72	3509
	average	31	676	8477
	maximum	48	1822	13444
Improved method	minimum	24	682	27617
	average	67	2521	59193
	maximum	119	7626	90769
Nonsymmetric cases		24	370	2160

in Table 1. In the experiments, ten examples for 10 and 20 city problems respectively and two examples for 30 city problems are solved.

Table 2 arranges the res ults of experiments for the computing time. The proposed method makes it possible to solve 30 city problems in 85 sec. on an average. This time is reduced to one-fifth compared with that of the improved method, and it is expected that the ratio becomes smaller when the number of cities increases.

Table 3. Number of nodes (1/2)

Number of cities (Number of examples)		10 (10)	20 (10)	30 (2)
Proposed method	minimum	9	28	645
	average	25	220	1275
	maximum	40	573	1905
Improved method	minimum	12	162	2590
	average	36	500	4508
	maximum	68	1318	6425
Nonsymmetric cases		13	96	356

Table 3 arranges the results for the number of nodes generated torough the solution. The number doubled in the table corresponds to that of the existing solutions. It is shown in these resclts that the number of nodes in the proposed method is greatly reduced compared with that of the improved method.

In order to compare the symmetric problem with the nonsymmetric problem, the computing time and the number of nodes are shown in the same tables 2 and 3, by means of the improved method for nonsymmetric problems. It is readily seen from the table that the symmetric problems are rather difficult in these cases.

5. Conclusion

A new algorithm solving for the symmetric traveling salesman problem is proposed. The method retains the symmetry of the cost matrix through reducing operations and deals with city pairs disregarding their directions. The concept of these technique will be applied to other problems. In conclusion we shall mention here that the method proposed in our present paper can treat and successfully solve the symmetric traveling salesman problems by using an electronic computer.

References

1) Little, J. D. C., K. G. Murty, D. W. Sweeney and C. Karel: "Algorithm for the Traveling Salesman Problem", Opns. Res. **11** (1963).

2) Shapiro, D.: "Algorithms for the Solution of the Optimal Cost Traveling Salesman Problem", Sc. D. Thesis, Washington University, St. Louis (1966).

3) Lawler, E. L. and D. E. Wood: "Branch-and-Bound Method: a Survey", Opens. Res. **14** (1966).

4) Bellmore, M. and G. L. Nemhauser: The Traveling Salesman Problem: a Survey", Opns. Res. 16 (1968).

5) Sleckhan, H.: "A Theorem on Symmetric Traveling Salesman Problems", Opns. Res. **18** (1970).

6) Roberts, S. M. and Flores: An Engineering Approach to the Traveling Salesman Problem", Management Sci. 13 (1966).

7) Karg, R. L. and G. L. Thompson: "A Heuristic Approach to Solving Traveling Salesman Problems", Management Sci. **10** (1964).