Journal of the Operations Research Society of Japan
Vol. 34, No. 2, June 1991

THE PRECEDENCE CONSTRAINED TRAVELING SALESMAN PROBLEM

Mikio Kubo Hiroshi Kasugai
Waseda University

(Received January 31, 1990; Revised October 2, 1990)

Abstract We consider a generalization of the classical traveling salesman problem (TSP) called the precedence constrained traveling salesman problem (PCTSP), i.e. given a directed complete graph $G(V,E)$, a distance D_{ij} on each arc $(i,j) \in E$, precedence constraints $\prec$ on V, we want to find a minimum distance tour that starts node $1 \in V$, visits all the nodes in $V - \{1\}$, and returns node 1 again so that node i is visited before node j when $i \prec j$. We present a branch and bound algorithm for the exact solutions to the PCTSP incorporating lower bounds computed from the Lagrangean relaxation. Our lower bounding procedure is a generalization of the restricted Lagrangean method that has been successfully adapted to the TSP by Balas and Christofides [2]. Our branch and bound algorithm also incorporates several heuristics and variable reduction tests. The computational results with up to 49 nodes show that our algorithm computes exact solutions to several classes of precedence constraints within acceptable computational requirements.

1. Introduction

Consider the following generalization of the traveling salesman problem (TSP).

(Precedence Constrained Traveling Salesman Problem : PCTSP)
Given a node set $V, n = |V|$, an arc set E, a directed complete graph $G(V,E)$, a distance D_{ij} on each arc $(i,j) \in E$, precedence constraints $\prec$ on V, we want to find a minimum distance tour that starts node $1 \in V$, visits all the nodes in $V - \{1\}$, and returns node 1 again so that node i is visited before node j when $i \prec j$.

Several applications of the vehicle routing problem such as the dial-a-ride routing problem (Jaw et al. [17], Psaraftis [27], [28], [29]) and the bus routing problem (Stein [31], Wren and Holiday [36]) includes precedence constraints between nodes. Although several variants of the vehicle routing problem have been considered in literature, researches for the PCTSP are few in spite of its practical importance. The objective of this paper is to develop an efficient branch and bound algorithm for the PCTSP incorporating the restricted Lagrangean method, that has been successfully applied to the TSP by Balas and Christofides [2].

The previous works for the PCTSP are summarized as follows.

Kalantari et al. [18] provided a branch and bound method based on the algorithm for the TSP developed by Little et al. [22], and could solve the 30-node problems with the asymmetric distance matrix. Suzuki and Nomura [33] also developed a branch and bound algorithm using bounds derived from the arborescence problem or the shortest path problem, and could solve 30-node problems with the sparse distance matrix. In their model, nodes are permitted to be visited more than twice and the starting and ending points are prespecified (this problem can be reduced to the ordinary PCTSP using a simple transformation).

Fischetti and Toth [11] proposed an algorithm for the PCTSP based on the additive bounding procedure. Their algorithm named *bound from variable decomposition* can be seen as the same algorithm as our bounding procedure 1. The only difference is that they use the shortest path problem as a subroutine instead of using the depth first search as in our algorithm. Since the ordinary shortest path algorithm requires $O(n^2)$ operations where n

is the number of nodes, and the depth first search runs in $O(E_0)$ where E_0 is an arc set of the admissible graph that will be defined later, our algorithm works much faster when the admissible graph is sparse, i.e. there exist fewer arcs in the graph compared to n^2. They derived another bounding procedure named *bound from disjuction* that can be easily observed to be identical to the third bounding procedure developed by Balas and Christofides [2].

A closely related and important special case of the PCTSP is the dial-a-ride routing problem (DARP) in which the precedence constraints are represented by pick-up and delivery points. Psaraftis [27] developed an exact solution method using a dynamic programming algorithm for the DARP whose time complexity is $O(n^2 3^n)$, and solved 20-node problems.

Several heuristic algorithms have been proposed for the DARP (Psaraftis [28], [29], Jaw et al. [17], Stein [31]). Psaraftis proposed the minimum spanning tree (MST) heuristic and the local search (k-opt) procedure specialized to the DARP. The MSP heuristic runs in $O(n^2)$, while the k-opt procedure runs in $O(n^k)$. Since the computational complexity of k-opt procedure grows in exponential order in k, he recommends $k = 2, 3$ in practice. Jaw et al. [17] provides an insertion heuristic and Stein [31],[32] provided several algorithms that run asymptotically well.

The organization of the paper is as follows: In section 2 several valid inequalities that represent the precedence constraints are derived. In section 3 these valid inequalities are taken into the Lagrangean function using specially designed heuristics called bounding procedures. In section 4 the branch and bound algorithm incorporating several lower bounding procedures, heuristics and variable reduction tests is described. Results of the numerical experiments are included in section 5. Section 6 contains conclusions.

2 Formulation

We give below an integer programming problem of the PCTSP.

(PCTSP : Formulation)

(1) $$\min \sum_{i\in V} \sum_{j\in V-\{i\}} D_{ij}X_{ij}$$

subject to

(2) $$\sum_{j\in V-\{i\}} X_{ij} = 1 \quad \forall i \in V,$$

(3) $$\sum_{j\in V-\{i\}} X_{ji} = 1 \quad \forall i \in V,$$

(4) $$\sum_{i\in S}\sum_{j\in S} X_{ij} \le |S| - 1 \quad \forall S \subset V, S \ne \emptyset,$$

(5) Tour satisfies all precedence relationship,

(6) $$X_{ij} \in \{0,1\} \quad \forall (i,j) \in E,$$

where

(7) $$X_{ij} = \begin{cases} 1 & \text{arc } (i,j) \in E \text{ is contained in the solution} \\ 0 & \text{otherwise.} \end{cases}$$

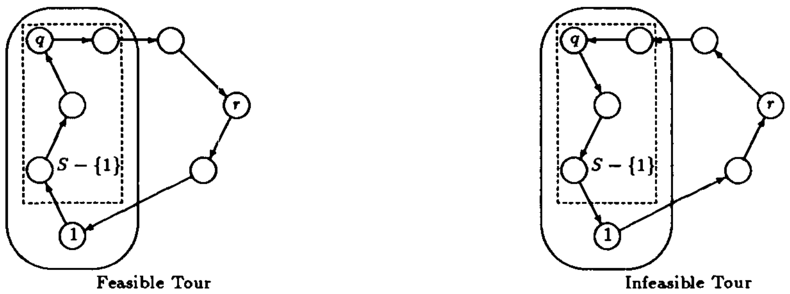

Figure 1: Derivation of Formulation 1

In the formulation above, equalities (2) and (3) are degree constraints. Constraints (4) prohibit two or more disjoint cycles. Constraints (5) represent precedence constraints between nodes.

We must rewrite the precedence constraints (5) using X_{ij} to complete the formulation. The precedence constraint $q \prec r$ can be represented as follows:

1. Set formulation For all $S \subseteq V$, $q \in S$, $1 \in S$, $r \notin S$, there exists an arc from $S - \{1\}$ to $V - S$ contained in the tour (see Figure 1).
2. Path formulation
   (a) Any path in the feasible tour with starting node r and ending node q must contain node 1.
   (b) Any path in the feasible tour with starting node 1 and ending node r must contain node q.
   (c) Any path in the feasible tour with starting node q and ending node 1 must contain node r.
3. Articulation node formulation
   (a) In the feasible tour the path with starting node 1 and ending node q and the path with starting node q and ending node r do not intersect each other.
   (b) In the feasible tour the path with starting node 1 and ending node q and the path with starting node $r$and ending node 1 do not intersect each other.
   (c) In the feasible tour the path with starting node q and ending node r and the path with starting node r and ending node 1 do not intersect each other.

The integer programming representation of the set formulation is

$$(8) \qquad \sum_{i \in S-\{1\}} \sum_{j \in V-S} X_{ij} \geq 1 \quad \forall S \subset V, 1 \in S, \forall q \in S, \forall r \notin S, q \prec r.$$

Theorem 1 *Constraint (8) is valid for the PCTSP.*

Proof : Assume that (8) is not satisfied. Then, the only arc emanating from S is from node 1; all paths from node q to node r visits node 1. This means the precedence constraint is not satisfied. Thus (8) is a valid inequality for the PCTSP. ∎

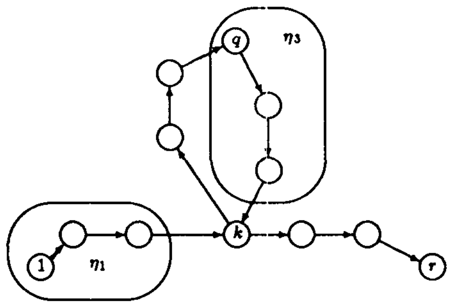

Figure 2: Derivation of Formulation 3.(a)

We derive the bounding procedure 1 that uses inequality (8) to improve the lower bound in subsection 3.2.

Using 2.(a), we can represent the precedence constraint as follows :

$$(9) \qquad \sum_{(i,j)\in A(\rho)} X_{ij} \leq\mid A(\rho)\mid -1 \quad \forall \rho \in P(r \mapsto q, 1), \forall q \in V, \forall r \in V, q \prec r,$$

where $P(a \mapsto b, c)$ is a set of all paths from node a to b that do not visit node c, and $A(\rho)$ denotes a set of arcs in path ρ.

Theorem 2 *Constraint (9) is valid for the PCTSP.*

Proof : We prove the theorem by showing the contraposition. Assume that (9) is not satisfied. Then we lead that there exists $\rho \in P(r \mapsto q, 1)$ such that

$$(10) \qquad \sum_{(i,j)\in A(\rho)} X_{ij} \geq\mid A(\rho)\mid,$$

which implies there exists a path from node r to node q that does not traverse node 1. Since any feasible tour must visit node 1 exactly once, there must be a path from node q to node r that contains node 1; this means that precedence constraint $q \prec r$ is not satisfied. Thus (9) is a valid inequality for the PCTSP. ∎

Similarly, 2.(b) and 2.(c) can be represented as inequalities using X_{ij}. The bounding procedure that takes this type of inequalities to the Lagrangean function is described in section 3.3.

We then represent 3.(a) as valid inequalities. For each $k \in V - \{1, q, r\}$, define the following four sets:

1. $\eta_1 \subset V$, $1 \in \eta_1, k, r \notin \eta_1$,
2. $\eta_2 \subset V$, $q \in \eta_2, k, r \notin \eta_2$,
3. $\eta_3 \subset V$, $q \in \eta_3, 1, k \notin \eta_3$,
4. $\eta_4 \subset V$, $r \in \eta_4, 1, k \notin \eta_4$.

Then the statement 3.(a) is true if and only if the following four events occurs simultaneously:

1. There exists at least one arc emanating from η_1 into $V-\eta_1-\{k,r\}$, or there exists at least one arc emanating from η_3 into $V-\eta_3-\{1,k\}$ (see Figure 2)
2. There exists at least one arc emanating from η_1 into $V-\eta_1-\{k,r\}$, or there exists at least one arc going into η_4 from $V-\eta_4-\{1,k\}$
3. There exists at least one arc going into η_2 from $V-\eta_2-\{k,r\}$ or there exists at least one arc emanating from η_3 into $V-\eta_3-\{1,k\}$
4. There exists at least one arc going into η_2 from $V-\eta_2-\{k,r\}$ or there exists at least one arc going into η_4 from $V-\eta_4-\{1,k\}$

Though we can derive four classes of inequalities corresponding to the four events above, for simplicity, we derive one class of inequalities corresponding to the first event.

$$\sum_{i\in\eta_1}\sum_{j\in V-\eta_1-\{k,r\}} X_{ij}+\sum_{i\in\eta_3}\sum_{j\in V-\eta_3-\{k,1\}} X_{ij}\geq 1 \tag{11}$$
$$\forall\eta_1\subset V, 1\in\eta_1, k\notin\eta_1, r\notin\eta_1, \forall\eta_3\subset V, q\in\eta_3, k\notin\eta_3, 1\notin\eta_3,$$
$$\forall q\in V, \forall r\in V, q\prec r, \forall k\in V-\{1,q,r\}$$

Before proving the validity of (11), we define the cutset that will be sometimes used in the following argument. For any $S_1, S_2\subseteq V, S_1\cap S_2=\emptyset$, the set of arcs $\{(i,j)\in E\mid i\in S_1, j\in S_2\}$ is called a directed cut set or briefly a cutset, and denoted by (S_1,S_2).

Theorem 3 *Constraint (11) is valid for the PCTSP.*

Proof : Assume that (11) is not satisfied. Then all the variables X_{ij} corresponding to arcs in cutsets $(\eta_1, V-\eta_1-\{k,r\})$ and $(\eta_3, V-\eta_3-\{k,1\})$ are 0. This implies that a path from node 1 to node q and a path from node q to node r intersect each other, i.e. contain the same node, say k. This means the tour is infeasible. Thus (11) is a valid inequality for the PCTSP. ∎

Validity of other types of inequalities can be proved similarly; hence omitted. We can derive eight types of inequalities from 3.(b) and 3.(c) using similar argument above, but they are too lengthy to describe here. The bounding procedure that uses the inequalities derived from 3.(a), 3.(b) and 3.(c) is described in section 3.4.

3 Lower Bounding Procedures

3.1 Restricted Lagrangean Relaxation

In this subsection, we briefly review the restricted Lagrangean approach introduced by Balas and Christofides [2].

Removing (4),(5) from the formulation of the PCTSP, we can get an assignment problem (AP) that can be solved using a primal-dual algorithm (Kuhn [20]) in $O(n^3)$ time.

Let us assume that relaxed constraints (4) and (5) are represented as

$$\sum_{i\in V}\sum_{j\in V} a^t_{ij}X_{ij}\geq a^t_0 \quad \forall t\in T. \tag{12}$$

Then we can get the following Lagrangean dual problem of the PCTSP.

(LDP : Lagrangean Dual Problem of the PCTSP)

$$\max\{L(w)\mid w\geq 0\}, \tag{13}$$

where w is a non-negative vector whose t-th component is a multiplier w_t corresponding to inequality (12), and

$$(14) \qquad L(w) = \min \sum_{i \in V} \sum_{j \in V} (D_{ij} - \sum_{t \in T} w_t a^t_{ij}) X_{ij} + \sum_{t \in T} w_t a^t_0$$

subject to (2),(3),(6).

Of course the Lagrangean dual (13) may be solved using the subgradient optimization technique (see for example [30]). In this case, however, the number of components of the vector w is an exponential number; such a procedure is computationally expensive. Therefore, we solve the restricted version of the Lagrangean dual problem.

Let us denote the optimal solution of the AP by $\hat{x}_{ij} = 1, \forall (i,j) \in E$, and the optimal multipliers by $u_i, \forall i \in V$ and $v_j, \forall j \in V$ corresponding to the constraints (2) and (3), respectively. Then we restrict the dual variables to satisfy the following two conditions :

Condition I (Dual Feasibility):

$$(15) \qquad D_{ij} \geq u_i + v_j + \sum_{t \in T} w_t a^t_{ij} \quad \forall i \in V, \forall j \in V,$$

Condition II (Complementary Slackness):

$$(16) \qquad \hat{x}_{ij} > 0 \quad \text{implies} \quad (i,j) \in E_0,$$

where

$$(17) \qquad E_0 = \{(i,j) \in E \mid D_{ij} = u_i + v_j + \sum_{t \in T} w_t a^t_{ij}\}.$$

Then the bound computed from the restricted Lagrangean dual is

$$(18) \qquad \sum_{i \in V} u_i + \sum_{j \in V} v_i + \sum_{t \in T} w_t a^t_{ij}$$

and the reduced cost $\bar{D}_{ij}$ is

$$(19) \qquad \bar{D}_{ij} = D_{ij} - u_i - v_j - \sum_{t \in T} w_t a^t_{ij}.$$

When we can add the inequality (12) to the objective function without violating the dual feasibility and complementary slackness conditions, and the corresponding multiplier w_t is positive, we say that w_t can be applied to (12). We also define the admissible graph $G_0(V, E_0)$ whose arc set E_0 is defined in (17).

The restricted version of the Lagrangean dual problem of the PCTSP can be solved using specialized adjustment heuristics called bounding procedures. Balas and Christofides [2] used three types of subtour elimination constraints to get tight lower bounds for the TSP, and derived several bounding procedures for the TSP. We use several forms of precedence constraints that can be added without disturbing dual feasibility and complementary slackness conditions, and give three bounding procedures that will be described in the following subsections.

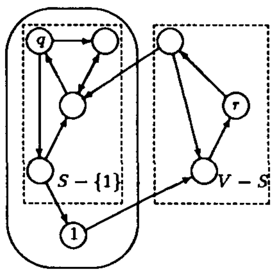

Figure 3: Cut Set $(S - \{1\}, V - S)$

3.2 Bounding Procedure 1

The first procedure, called the bounding procedure 1, uses the precedence constraints derived from the set formulation. The Lagrangean multiplier to the constraint (8) can be determined based on the following theorem.

Theorem 4 *A positive multiplier can be applied to the inequality (8) if and only if*

$$(S - \{1\}, V - S) \cap E_0 = \emptyset \tag{20}$$

is satisfied. Furthermore, if the condition above are satisfied, the multiplier

$$w = \min \{\bar{D}_{ij} \mid (i,j) \in (S - \{1\}, V - S)\} \tag{21}$$

can be applied to the inequality (8).

Proof : Suppose that there exists a set S satisfying (20). Then the positive multiplier defined by (21) can be applied to all arcs in the cutset $(S - \{1\}, V - S)$ without reducing any negative reduced costs, and obviously the conditions I and II of the restricted Lagrangean method are satisfied. Assume that $(S - \{1\}, V - S) \cap E_0 \neq \emptyset$. Then it is easily seen that a positive multiplier cannot be applied to the inequality (8). This proves the theorem. ∎

To improve the lower bound we must find a cutset $(S - \{1\}, V - S)$ that contains no arc in E_0 (see Figure 3). This can be done by the depth first search (see for example [34]) from node q on $G_0 - \{1\}$ (a subgraph of $G(V, E)$ deleting node 1 and all arcs connecting node 1). More precisely, the bounding procedure 1 can be described as follows. (in this paper, we adopt Pidgin-Algol [1] to describe the lower bounding procedures.)

Algorithm : Bounding Procedure 1 (BP1)
Input : node number n, reduced cost matrix $[\bar{D}_{ij}]$, admissible graph $G_0(V, E_0)$, precedence constraints $\prec$, node $1 \in V$.
Output : new reduced cost matrix $[\bar{D}_{ij}]$, $LB1$.

```
begin
set $LB1 := 0$;
for all $q \prec r$ do
    begin
        initialize the reachable set $S := \{q\}$;
        while $r \notin S$ do
            begin
```

```
                $G_1 := G_0 - \{1\}$;
                find a reachable set $S$ from node $q$ on the admissible graph
                $G_1$ using the depth first search;
                $w := \min\{\bar{D}_{ij} \mid (i,j) \in (S, V - S - \{1\})\}$;
                $\bar{D}_{ij} := \bar{D}_{ij} - w \quad \forall (i,j) \in (S, V - S - \{1\})$;
                $LB1 := LB1 + w$;
                add the arcs that attains minimum in (21) to $E_0$
            end
    end
end .
```

The number of operations of the above procedure is dominated by the order of the depth first search and the number of arcs added to G_0 in the course of the procedure. Since the number of precedence constraints is $O(n^2)$ and for each precedence constraint the depth first search runs in $O(n^2)$, the complexity of the former part is $O(n^4)$. Since the number of arcs added to G_0 must be less or equal to n^2 in the worst case and the operation for finding an arc with the minimum reduced cost requires $O(n)$, the complexity of the latter part is $O(n^3)$. Thus the overall complexity of the bounding procedure 1 is $O(n^4)$.

3.3 Bounding Procedure 2

We derive the second bounding procedure called to bounding procedure 2 using the precedence constraints derived from path formulation 2.(a), 2.(b) and 2.(c). Since the structure of inequalities are about the same, we describe how to adjust the multiplier to (9) only.

We rewrite (9) with a standard form of the restricted Lagrangean method.

$$(22) \qquad -\sum_{(i,j)\in A(\rho)} X_{ij} \geq 1 - \mid A(\rho) \mid \qquad \forall \rho \in P(r \mapsto q, 1), \forall q \in V, \forall r \in V, q \prec r$$

To describe the bounding procedure 2, we use several notations. We express our second bounding procedure in terms of the assignment tableau of the Hungarian algorithm. The tableau is defined as the ordered pair of V. A line is defined as a row or a column of the tableau, and a cell of the tableau is the intersection of a row and a column. Since the cells correspond to the arcs of $G(V, E)$, we denote them in the same way. We call that a set of lines L *covers* cell (i,j) when L contains column i or row j.

The bounding procedure 2 is based on the following theorem.

Theorem 5 *A positive multiplier can be applied to the inequality (22) if and only if there exists a set L of lines and a path $\rho \in P(r \mapsto q, 1)$ such that*

i *every cell $(i,j) \in A(\rho)$ is covered by L exactly once,*

ii *no cell (i,j) that satisfies $\bar{D}_{ij} = 0$ for all $(i,j) \notin A(\rho)$ is covered by any line in L.*

Proof : Suppose that there exists a line set L consisting of row set I and column set J which satisfies **i** and **ii** . A positive multiplier w can be applied to (22); w is added to all $(i,j) \in A(\rho)$. Then u_i and v_j is modified as follows :

$$u_i := u_i + w \quad \forall i \in I,$$
$$v_j := v_j + w \quad \forall j \in J.$$

It is easily seen that the conditions above guarantee the dual feasibility, i.e. all reduced costs are non-negative and complementary slackness condition, i.e. if $\hat{x}_{ij} = 1$ then $\bar{D}_{ij} = 0$. This proves 'if' part of the theorem.

Next, assume that a positive multiplier w can be applied to (22). Then we must increase $u_i + v_j$ by w to keep complementary slackness condition. This implies that **i** must be satisfied. Then suppose that **ii** is not satisfied, then no positive multiplier can be applied to (22). This proves 'only if' part of the theorem. ∎

Condition **ii** in the theorem above suggests that we can restrict our search on the arc set

$$E_p = \{(i,j) \in E_0 \mid \bar{D}_{ik} \neq 0, \forall k \in V, k \neq i, \bar{D}_{kj} \neq 0, \forall k \in V, k \neq j\} \tag{23}$$

when finding path ρ to improve the lower bound. A candidate path ρ is the shortest path from r to q on $G(V, E_p)$. We use the breadth first search [34] for finding such a path. In the sequel we refer this path to *the precedence violating path* .

Given a path ρ, lines are selected using the following heuristic procedure in which we select a line (row or column) that attains the maximum increase of the lower bound.

```
Procedure : Line Selection
Input : G(V,E), reduced cost matrix $[\bar{D}_{ij}]$, precedence violating path $\rho$.
Output : row set $I^*$, column set $J^*$.
begin
set $I^* := \emptyset$, $J^* := \emptyset$;
for all $(i,j) \in A(\rho)$ do
    begin
        $a := \min_{k \in V-\{j\}} \bar{D}_{ik}$;
        $b := \min_{k \in V-\{i\}} \bar{D}_{kj}$;
        if $a \geq b$ then $I^* := I^* \cup \{i\}$ else $J^* := J^* \cup \{j\}$
    end
end .
```

The lines selected by the procedure above may contain the same cell more than once. Therefore, we must determine the Lagrangean multiplier w as follows:

$$w = \min \left\{ \begin{array}{c} \min_{(i,j)\in(I^*,J^*)} \bar{D}_{ij}/2 \\ \min_{(i,j)\in(I^*,V-J^*)\cup(V-I^*,J^*)} \bar{D}_{ij} \end{array} \right\}. \tag{24}$$

The bounding procedure 2 can be described as follows :

```
Algorithm : Bounding Procedure 2 (BP2)
Input : node number $n$, reduced cost matrix $[\bar{D}_{ij}]$, admissible graph $G_0(V, E_0)$, precedence constraints $\prec$, node $1 \in V$ .
Output : new reduced cost matrix $[\bar{D}_{ij}]$ , $LB2$.
begin
set $LB2 := 0$;
for all $q \prec r$ do
    begin
        find a path $\rho$ from $r$ to $q$ on $G(V, E_p) - \{1\}$ using the breadth first search;
        find $I^*$ and $J^*$ using procedure Line Selection;
        compute $w$ using (24);
        change reduced costs as follows:
            $\bar{D}_{ij} := \bar{D}_{ij} - w \quad \forall (i,j) \in (I^*, V - J^*) \cup (V - I^*, J^*)$;
        $LB2 := LB2 + w$;
        add the arcs that attain minimum in (24) to $E_0$
        end
    end .
```

We can estimate the number of operation required as in the bounding procedure 1. Since the number of precedence constraints is $O(n^2)$, and both the breadth first search and procedure Line Selection run in $O(n^2)$, the overall complexity of bounding procedure 2 is $O(n^4)$.

3.4 Bounding Procedure 3

The third procedure, called the bounding procedure 3, uses the precedence constraints derived from 3.(a), 3.(b) and 3.(c). Since the structures of the inequalities are about the same, we describe how to adjust the multipliers to (11) to increase the lower bound. The Lagrangean multiplier to the constraint (11) can be determined based on the following theorem.

Theorem 6 *A positive multiplier can be applied to the inequality (11) if and only if $K_1 \cap E_0 = \emptyset$ and $K_2 \cap E_0 = \emptyset$, where*

$$K_1 = (\eta_1, V - \eta_1 - \{k, r\}) \tag{25}$$

and

$$K_2 = (\eta_3, V - \eta_3 - \{k, 1\}). \tag{26}$$

Proof : Suppose that $K_1 \cap E_0 = \emptyset$ and $K_2 \cap E_0 = \emptyset$ are satisfied. If we set $w = \min_{(i,j)\in K_1 \cup K_2} \bar{D}_{ij}$, then at least $w/2$ can be applied to (11). This proves 'if' part of the theorem.

Next assume that $K_1 \cap E_0 \neq \emptyset$ or $K_2 \cap E_0 \neq \emptyset$. Then it is easily seen that no positive multiplier can be applied to (11). This proves 'only if' part of the theorem. ∎

We first find the cutsets K_1 and K_2 using the same technique as in the bounding procedure 1. Then the multiplier is determined as follows:

$$w = \min \left\{ \begin{array}{l} \min_{(i,j)\in K_1 \cap K_2} \bar{D}_{ij}/2 \\ \min_{(i,j)\in K_1 \cup K_2 - K_1 \cap K_2} \bar{D}_{ij} \end{array} \right\}. \tag{27}$$

The bounding procedure 3 can be described as follows :

Algorithm : Bounding Procedure 3 (BP3)
Input : node number n, reduced cost matrix $[\bar{D}_{ij}]$, admissible graph $G_0(V, E_0)$, precedence constraints $\prec$, node $1 \in V$.
Output : new reduced cost matrix $[\bar{D}_{ij}]$, $LB3$.

```
begin
set $LB3 := 0$;
for all $q \prec r$ do
    begin
        for all $k \in V - \{1, q, r\}$ do
            begin
                find a reachable set $\eta_1$ from node 1 on $G_0 - \{r, k\}$;
                find a reachable set $\eta_3$ from node $q$ on $G_0 - \{1, k\}$;
                Let $K_1 = (\eta_1, V - \eta_1 - \{k, r\})$;
                Let $K_2 = (\eta_3, V - \eta_3 - \{k, 1\})$;
                compute $w$ using (27);
                change reduced costs as follows:
                $\bar{D}_{ij} := \bar{D}_{ij} - w \quad \forall (i,j) \in (I^*, V - J^*) \cup (V - I^*, J^*)$;
                $LB3 := LB3 + w$;
                add the arcs for which the minimum in (27) is attained
                to $E_0$
            end
    end
end .
```

We can estimate the number of operations as in the bounding procedure 1. Since the number of precedence constraints is $O(n^2)$ and for each precedence constraint and for each k the depth first search runs in $O(n^2)$, the overall complexity of bounding procedure 3 is $O(n^5)$.

3.5 Numerical Example

To illustrate the bounding procedures described above we give a small example. Consider the 10 node PCTSP with distance matrix $[D_{ij}]$ shown in Table 1, and precedence constraints $5 \prec 6$, $4 \prec 7$, $4 \prec 8$, $1 \prec j$ for all $j \in V - \{1\}$, $i \prec 10$ for all $V - \{10\}$.

The optimal assignment is $< 1,4,3,6,8,5,2,10,1 >$ and $< 7,9,7 >$ and its solution value is 172. The reduced cost matrix after solving the assignment problem is shown in Table 2 and the corresponding admissible graph in Figure 4.

Table 1: Distance Matrix$[D_{ij}]$

	1	2	3	4	5	6	7	8	9	10
1	-	97	83	21	63	-	-	-	39	-
2	-	-	80	73	12	25	73	8	67	27
3	-	89	-	79	47	7	59	90	78	85
4	-	45	13	-	78	90	87	96	30	-
5	-	22	36	53	-	60	35	39	24	-
6	-	82	97	58	-	-	10	15	94	74
7	-	72	81	-	44	12	-	60	13	86
8	-	77	59	-	14	75	48	-	44	65
9	-	72	97	24	92	26	40	84	-	96
10	0	-	-	-	-	-	-	-	-	-

Table 2: Reduced Cost Matrix $[\bar{D}_{ij}]$ (after solving the assignment problem)

	1	2	3	4	5	6	7	8	9	10
1	-	55	41	0	21	-	-	-	0	-
2	-	-	72	86	4	36	70	0	62	0
3	-	63	-	74	21	0	38	64	55	40
4	-	32	0	-	65	96	79	83	20	-
5	-	0	14	52	-	57	18	17	5	-
6	-	67	82	64	-	-	0	0	82	40
7	-	56	65	-	28	15	-	44	0	51
8	-	63	45	-	0	80	39	-	33	32
9	-	27	52	0	47	0	0	39	-	32
10	0	-	-	-	-	-	-	-	-	-

Bounding procedure 1 (BP1)

For $5 \prec 6$, we can find reachable set $S = \{1, 2, 5, 8, 10\}$. Thus for cutset $(S - \{1\}, V - S) = (\{2, 5, 8, 10\}, \{3, 4, 6, 7, 9\})$ we can apply the multiplier $w = 5 (= \bar{D}_{59})$ to (8); arc $(5, 9)$ is added to G_0. The lower bound after the bounding procedure 1 becomes 177, the reduced cost matrix is shown in Table 3 and the corresponding admissible graph in Figure 5.

Bounding procedure 2 (BP2)

For $5 \prec 6$, the breadth first search from node 5 on $G(V, E_p)$ finds the precedence violating path $\rho = (5, (5, 2), 2, (2, 10), 10, (10, 1), 1)$; a positive multiplier can be applied to the valid inequality of type 2.(c) with $P(5 \mapsto 1, 6)$. The cells $(5, 2)$, $(2, 10)$, $(10, 1)$ corresponding to arcs in ρ are covered by row set $I^* = \{10\}$ and column set $J^* = \{2, 10\}$. The multiplier $w = 27 (= \bar{D}_{92})$ can be applied to (22). The lower bound after the bounding procedure 2 becomes 204, the reduced cost matrix is shown in Table 4 and the corresponding admissible graph in Figure 6.

Bounding procedure 3 (BP3)

For $5 \prec 6$ and $k = 2$, we can get $\eta_1 = \{1, 3, 4, 7, 9\}$ and $\eta_3 = \{3, 4, 5, 6, 7, 8, 9\}$; cutsets become $K_1 = (\{1, 3, 4, 7, 9\}, \{5, 8\})$ and $K_2 = (\{3, 4, 5, 6, 7, 8, 9\}, \{1, 10\})$. We can apply $w = 5 (= \bar{D}_{8,10})$ to (11). Similarly, for $5 \prec 6$ and $k = 9$, we can get cutsets $K_1 = (\{1, 3, 4\}, \{2, 5, 7, 8, 10\})$ and $K_2 = (\{2, 5, 8, 10\}, \{3, 4, 6, 7\})$. The multiplier $w = 5 (= \bar{D}_{42})$ can be applied to (11). The lower bound after the bounding procedure 3 becomes 214, the reduced cost matrix is shown in Table 5 and the corresponding admissible graph in Figure 7.

Table 3: Reduced Cost Matrix $[\bar{D}_{ij}]$ (after BP1)

	1	2	3	4	5	6	7	8	9	10
1	-	55	41	0	21	-	-	-	0	-
2	-	-	67	81	4	31	65	0	57	0
3	-	63	-	74	21	0	38	64	55	40
4	-	32	0	-	65	96	79	83	20	-
5	-	0	9	47	-	52	13	17	0	-
6	-	67	82	64	-	-	0	0	82	40
7	-	56	65	-	28	15	-	44	0	51
8	-	63	40	-	0	75	34	-	28	32
9	-	27	52	0	47	0	0	39	-	32
10	0	-	-	-	-	-	-	-	-	-

Table 4: Reduced Cost Matrix $[\bar{D}_{ij}]$ (after BP2)

	1	2	3	4	5	6	7	8	9	10
1	-	28	41	0	21	-	-	-	0	-
2	-	-	67	81	4	31	65	0	57	0
3	-	36	-	74	21	0	38	64	55	13
4	-	5	0	-	65	96	79	83	20	-
5	-	0	9	47	-	52	13	17	0	-
6	-	40	82	64	-	-	0	0	82	13
7	-	29	65	-	28	15	-	44	0	24
8	-	36	40	-	0	75	34	-	28	5
9	-	0	52	0	47	0	0	39	-	5
10	0	-	-	-	-	-	-	-	-	-

Table 5: Reduced Cost Matrix $[\bar{D}_{ij}]$ (after BP3)

	1	2	3	4	5	6	7	8	9	10
1	-	23	41	0	11	-	-	-	0	-
2	-	-	62	76	4	26	65	0	57	0
3	-	31	-	74	11	0	38	54	55	8
4	-	0	0	-	55	96	79	73	20	-
5	-	0	4	42	-	47	13	17	0	-
6	-	40	82	64	-	-	0	0	82	8
7	-	29	65	-	23	15	-	39	0	19
8	-	36	35	-	0	70	34	-	28	0
9	-	0	52	0	42	0	0	34	-	0
10	0	-	-	-	-	-	-	-	-	-

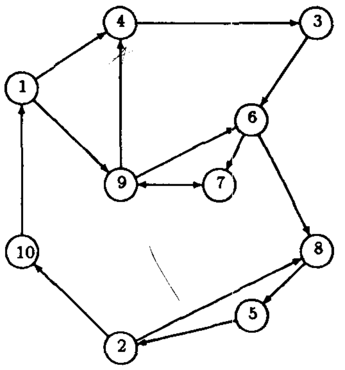
Figure 4: Admissible Graph (before bounding procedures)

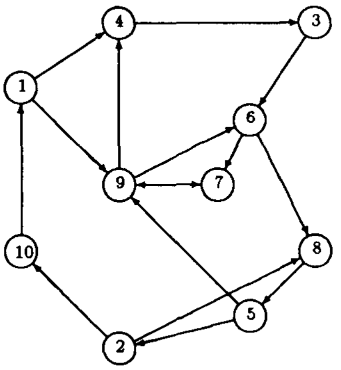
Figure 5: Admissible Graph (after BP1)

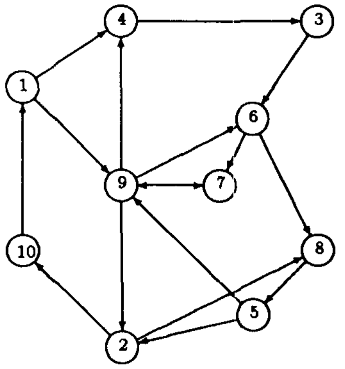
Figure 6: Admissible Graph (after BP2)

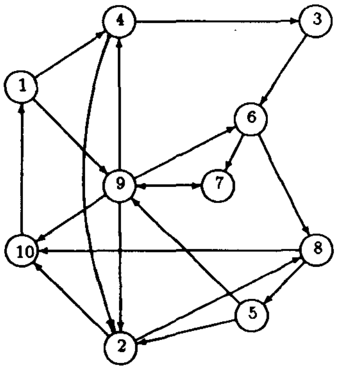
Figure 7: Admissible Graph (after BP3)

4 Branch and Bound Method

4.1 Outline of the Branch and Bound Method

The essential ingredients of our branch and bound procedure are:

1. A branching rule: We derive new branching scheme that uses the precedence constraints, and then it is combined other branching rules for the TSP.

2. A subproblem selection rule: We adopt the depth first search for selecting the subproblem in the branching tree.

3. Logical and variable fixing tests: We use a logical test using the precedence constraints. Furthermore, several variables are fixed using the reduced costs. Details are described in subsection 4.3.

4. Approximate algorithms: To get an upper bounds we use the insertion and local search algorithms. The insertion method constructs a feasible tour by inserting nodes one by one so as to satisfy the precedence constraints, and the local search methods are based on Or-opt procedure for the TSP [21]. These approximate algorithms incorporate several features designed to provide the efficiency and the performance. Details are described in the companion paper [19].

5. Finding of Hamiltonian cycles on the admissible graph: We find a Hamiltonian cycle on G_0 to get an upper bound and a precedence violating path that is used in the branching rule. Although finding a Hamiltonian cycle on any given graph is known to be NP-complete [13], we can find a Hamiltonian cycle using the depth first search with the limited searches on a sparse graph in modest computational requirement. In our experiment we set the upper limit of searches equal to $5n$.

Using the above ingredients, we can construct the branch and bound method. The outline of our branch and bound can be described as follows:

```
Algorithm : Branch and Bound for the PCTSP
Input : node number $n$, distance matrix $[D_{ij}]$, precedence constraints $\prec$, node $1 \in V$ .
Output : an optimal solution and its solution value $z^*$.
begin
(Initialization)
    put PCTSP on the list of active subproblems;
    set initial upper bound $UB := \infty$;
while list is not empty do
    begin
        (Subproblem Selection)
            choose a subproblem according to the subproblem selection rule
            and remove it from the list;
        (Logical Test)
            do a logical test using precedence constraints;
        (Finding Lower Bound)
            compute lower bound $LB$ using the assignment relaxation;
            improve $LB$ using subtour elimination constraints (this can be
            done using bounding procedures that are described in Balas and
            Christofides [2]);
            improve $LB$ using precedence constraints (this can be done using
            our bounding procedures that are described in section 3);
        (Finding Upper Bound)
            use approximate algorithms to find a feasible tour;
            find a Hamiltonian cycle on admissible graph $G_0$;
            if a better solution than $UB$ is found then store it and update
            $UB$;
            (Variable Fixing Test)
                do a variable fixing test using the reduced costs;
            (Subproblem Generation)
                if $LB < UB$ then generate new subproblems, place them on the
                list
        end
    end .
```

Several features of the branch and bound method are described in the following subsections.

4.2 Branching Rules

To describe the branching rules, we introduce several notations. Let us denote the subproblem in the branching tree by (N^0, N^1) where

$$N^0 = \{(i,j) \in E \mid X_{ij} = 0\} \tag{28}$$

and

$$N^1 = \{(i,j) \in E \mid X_{ij} = 1\}. \tag{29}$$

We denote the descendants of (N^0, N^1) by (N^0_r, N^1_r) for $r = 1, 2 \cdots, R$.

We describe three branching rules that are used intermittenly. The first one is derived from the precedence violating path. The second one, which is known to be the most efficient rule for the TSP, is derived from the subtour elimination inequality. The third one is based on the classical branching rule proposed by Little et al. [22] for the TSP.

4.2.1 The First Branching Rule

The first branching rule uses the precedence constraint $q \prec r$ for branching. For each $q \prec r$, the precedence violating paths are three types: $P(r \to q, 1), P(1 \to r, q), P(q \to 1, r)$ (see subsection 3.3). If we denote the precedence violating path by

$$(i_1, (i_1, i_2), i_2, \cdots, i_\lambda, (i_\lambda, i_{\lambda+1}), i_{\lambda+1}, \cdots, (i_{\Lambda-1}, i_\Lambda), i_\Lambda),$$

then $\Lambda - 1$ subproblems generated from (N^0, N^1) are

$$\begin{cases} N^0_\lambda = N^0 \cup (i_\lambda, i_{\lambda+1}) \\ N^1_\lambda = N^1 \cup \{(i_1, i_2), \cdots, (i_{\lambda-1}, i_\lambda)\} \end{cases} \quad \lambda = 1, 2 \cdots, \Lambda - 1. \tag{30}$$

The subproblems are mutually exclusive.

4.2.2 The Second Branching Rule

Murty [25] proposed the branching rule for the TSP based on the subtour elimination constraint. If we are given a subtour

$$(i_1, (i_1, i_2), \cdots, i_r, (i_r, i_{r+1}), i_{r+1}, \cdots, (i_{R-1}, i_R), i_R),$$

then generated $R - 1$ subproblems are:

$$\begin{cases} N^0_r = N^0 \cup (i_r, i_{r+1}) \\ N^1_r = N^1 \cup \{(i_1, i_2), \cdots, (i_{r-1}, i_r)\} \end{cases} \quad r = 1, 2, \cdots, R - 1. \tag{31}$$

4.2.3 The Third Branching Rule

Kalantari et al. [18] used the following branching rules based on the classical branching rule for the TSP proposed by Little et al. [22]. Let

$$(\hat{\imath},\hat{\jmath}) = \max_{(i,j)\in E} \varphi_{ij} \tag{32}$$

where

$$\varphi_{ij} = \min\{\bar{D}_{kj} \mid \forall k \in V, k \neq j\} + \min\{\bar{D}_{ik} \mid \forall k \in V, k \neq i\}. \tag{33}$$

Then two subproblems are generated as follows:

$$\begin{cases} N_1^0 = N^0 \cup (\hat{\imath},\hat{\jmath}) \\ N_1^1 = N^1 \end{cases} \tag{34}$$

and

$$\begin{cases} N_2^0 = N^0 \\ N_2^1 = N^1 \cup (\hat{\imath},\hat{\jmath}). \end{cases} \tag{35}$$

4.2.4 Selection of Branching Rule

We use three branching rules intermittenly according to the following rule. Let us denote the set of all precedence violating paths on G_0 defined in subsection 3.3 by All_p and the set of all subtours by All_s. If

$$\min \left\{ \begin{array}{l} \{\mid P \setminus \{N_k^0 \cup N_k^1\} \mid, \forall P \in All_p\} \\ \{\mid S \setminus \{N_k^0 \cup N_k^1\} \mid, \forall S \in All_s\} \end{array} \right\} \tag{36}$$

is greater than θn, then we use the third branching rule. Otherwise we use the first or second branching rule that attains minimum in (36). The parameter θ is set equal to 1/3 in the numerical experiments.

4.3 Tests

We use two tests: one is the logical test that uses the precedence constraints; another is the variable fixing test that uses the reduced costs. The logical test is used before the bounding procedures, while the variable fixing test is used after the bounding procedures.

4.3.1 Logical Test

The unnecessary variables X_{ij} can be fixed to 0 (or set $D_{ij} = \infty$) based on the following theorem.

Theorem 7 *If $j \prec i$ or $i \prec k \prec j (\exists k \in V)$, then we may set*

$$D_{ij} = \infty \tag{37}$$

without loss of optimality.

Proof : Obviously any feasible tour does not contain arc (i,j) when $j \prec i$. If $i \prec k \prec j$, then all feasible paths from i to j must pass k; any feasible tour can not contain arc (i,j). ∎

4.3.2 Variable Fixing Test

Let us denote the lower bound, the upper bound, the reduced cost of the subproblem by LB, UB, $[\bar{D}_{ij}]$, respectively. The variable fixing test is based on the following theorem.

Theorem 8 (Balas and Christofides [2]) *If the condition*

$$UB \leq LB + \bar{D}_{ij} \tag{38}$$

is satisfied, we may set $X_{ij} = 0$ *without losing optimality.*

From the theorem above, we may set variables that satisfy (38) equal to 0 for the current subproblem and its descendants.

5 Numerical Experiments

Our algorithm is implemented by BASIC and run on micro-computer PC9801VM (NEC). The speed of N88BASIC compiler is almost 1/3600 of VS-FORTRAN on IBM3081.

Distance matrix $[D_{ij}]$ is generated according to two types of instances below:

pure asymmetric : uniform integer over $[1, 100]$,

two dimensional Euclidean : nodes are distributed on 100×100 grids and then distances are calculated by the Euclidean metric between nodes.

The precedence constraints are generated according to the following two methods:

dial-a-ride precedence constraints : Given a node number n (odd number), we set $i \prec i + \lfloor n/2 \rfloor$, for $i = 2, 3, \cdots, \lfloor n/2 \rfloor$.

general precedence constraints : Given a parameter $\wp(0 \leq \wp \leq 1)$, we set $i \prec j$ with probability $\wp$ independently for all $(i, j) \in E, i < j$, and then find a transitive closure using Warshall-Floyd algorithm [12], [35].

In the tables, we use the following abbreviations.

CPU : Computational time represented by Hour : Minute : Second. The value in the parenthesis denotes CPU time after scaling that adjusts CPU time so that $n = 7$ is equal to 1. This value is used as the measure of the growth of the computational time when the problem size increases.

No. of Branches : Number of subproblems generated in the branch and bound method.

No. of FA : Number of fixed arcs in the first subproblem.

UB : The solution value calculated by the approximate algorithm.

Z : The optimal solution value.

Density : Density of the precedence constraints :

$$\frac{\text{number of precedence constraints}}{n(n-1)/2}. \tag{39}$$

We solve one randomly generated instance for each class of the problem. The results for the dial-a-ride precedence constraints are summarized in Table 6 and Table 7. The results for the general precedence constraints are summarized in Table 8 and Table 9.

The following conclusions are obtained:

Table 6: Comparison with the Algorithm Proposed by Kalantari et al. The results of Kalantari et al. is the computer time in seconds on Cray 1 at the University of Minnesota. (Asymmetric Distance Matrix, Dial-A-Ride Precedence Constraints)

	Our Algorithm					Kalantari et al.	
n	CPU		No. of Branches	No. of FA	UB/Z	CPU	
7	0:03	(1)	1	-	1	0.00152	(1)
13	3:48	(3.44)	17	65	1	0.0373	(1.11)
19	16:49	(2.28)	22	235	1	0.903	(4.06)
25	1:14:57	(2.57)	55	105	1	11.3	(12.79)
31	4:39:36	(3.28)	92	138	1	106.0	(40.93)
37	6:33:49	(1.91)	90	159	1	-	
43	14:24:57	(1.98)	130	186	1	-	
49	38:22:01	(2.73)	100	213	1	-	

Table 7: Result of Our Branch and Bound Algorithm with Various Problem Sizes. (Symmetric Distance Matrix and Dial-A-Ride Precedence Constraints)

n	CPU		No. of Branches	No. of FA	UB/Z
7	0:24	(1)	7	27	1
13	5:25	(0.61)	19	46	1
19	36:48	(0.62)	51	76	1
25	2:08:14	(0.55)	77	111	1
31	3:57:39	(0.34)	79	103	1
37	10:31:23	(0.38)	119	124	1
43	24:26:37	(0.41)	171	145	1
49	52:17:16	(0.47)	173	166	1

Table 8: Result of Our Branch and Bound Algorithm with $n = 31$. (Asymmetric Distance Matrix and General Precedence Constraints)

$\wp$	Density	CPU	No. of Branches	No. of FA	UB/Z
0.1	0.33	45:21	34	161	1
0.2	0.53	1:05:23	38	251	1
0.3	0.68	1:41:29	41	300	1
0.4	0.83	2:08:33	45	382	1
0.5	0.89	1:14:41	33	467	1
0.6	0.93	0:50:47	20	529	1

Table 9: Result of Our Branch and Bound Algorithm with $n = 31$. (Symmetric Distance Matrix and General Precedence Constraints)

$\wp$	Density	CPU	No. of Branches	No. of FA	UB/Z
0.1	0.24	2:55:17	85	98	1
0.2	0.44	3:33:11	69	173	1
0.3	0.79	1:50:09	33	320	1
0.4	0.81	2:56:48	49	339	1
0.5	0.94	1:57:19	31	508	1
0.6	0.96	1:24:40	31	550	1

1. Our algorithm performs better than the algorithm proposed by Kalantari et al. with respect to the growth of the computational time; we could solve relatively large instances.

2. For the general precedence constraints, CPU time of our algorithm is relatively small when $\wp$ is small and the distance matrix is asymmetric. We observe that when $\wp$ is small the PCTSP can be reduced to the ordinal TSP that can be solved by the assignment based branch and bound method.

3. For the general precedence constraints, our algorithm performs well when $\wp$ is large. This is due to the fact that when $\wp$ is large the logical test eliminates large number of unnecessary variables; the structure of the PCTSP becomes easier.

6 Conclusion

We considered the precedence constrained traveling salesman problem that has many practical applications in the area in vehicle routing and scheduling problems. We derived three bounding procedures, and developed a branch and bound method by incorporating these bounds. We could solve problems with up to 49 nodes exactly using our branch and bound method.

Acknowledgments

We wish to thank the anonymous referees for their valuable comments. This work was partially supported by the Waseda University Grant for Special Research Projects: 63A-42 and 89A-74.

References

[1] A. V. Aho, J. E. Hopcroft and J. D. Ulman , "The Design and Analysis of Computer Algorithms, " Addison-Wesley (1974).

[2] E. Balas and N. Christofides ,"A Restricted Lagrangean Approach to the Traveling Salesman Problem," Mathematical Programming, Vol. 21 (1981) pp. 19-46.

[3] M. Bellmore and J. C. Malone ," Pathology of Traveling Salesman Subtour Elimination Algorithm," Operations Research, Vol. 19 (1971) pp. 278-307.

[4] M. Bellmore and G. I. Nemhauser ,"The Traveling Salesman Problem: A Survey," Operations Research, Vol. 16 (1968) pp. 538-558.

[5] L. Bodin and B. Golden, "Classification in Vehicle Routing and Scheduling," Networks, Vol. 11 (1981) pp. 97-108.

[6] G. Carpaneto and P. Toth , "Some New Branching and Bound Criteria for the Asymmetric Traveling Salesman Problem," Management Science, Vol. 26 (1980) pp. 736-743.

[7] P. L. Cassidy and H. S. Bennett , "TRAMP - A Multi-Depot Vehicle Scheduling System," Operations Research Quarterly, Vol. 23 (1972) pp. 151-163.

[8] H. Crowder and M. W. Padberg ,"Solving Large-Scale Symmetric Traveling Salesman Problems to Optimality," Management Science, Vol. 26 (1980) pp. 495-509.

[9] G. B. Dantzig, D. R. Fulkerson and S. Johnson ,"Solution of a Large Scale Traveling Salesman Problem," Operations Research, Vol. 2 (1954) pp. 393-410.

[10] M. A. Efroymson and T. L. Ray, "A Branch and Bound Algorithm for Plant Location," Operations Research, Vol. 14 (1966) pp. 361-368.

[11] M. Fischetti and P. Toth," An Additive Bounding Procedure for Combinatorial Optimization Problems," Operations Research, Vol. 37 (1989) pp. 319-328.

[12] R. M. Floyd ," Algorithm 97, Shortest Path," Communication of the ACM, Vol. 5 (1962) pp. 345.

[13] M. R. Garey and D. S. Johnson ,"Computers and Intractability : A Guide to the Theory of NP-Completeness," Freeman (1979).

[14] W. V. Gehrlein ,"On Methods for Generating Random Partial Order," Operations Research Letters, Vol. 5 (1986) pp. 285-291.

[15] B. Golden, L. Bodin, T. Doyle and W. Stewart JR. ,"Approximate Traveling Salesman Algorithm," Operations Research, Vol. 28 (1980) pp. 694-711.

[16] M. Held and R. M. Karp, "A Dynamic Programming Approach to Sequencing Problems, " SIAM Journal of Applied Mathematics, Vol. 10 (1962) pp. 196-210.

[17] J. Jaw, A. R. Odoni, H. N. Psaraftis and N. H. M. Wilson ,"A Heuristic algorithm for the Multi-Vehicle Advance Request Dial-A-Ride Problem with Time Windows," Transportation Research, Vol. 20B (1986) pp. 243-257.

[18] B. Kalantari, A. V. Hill and S. R. Arora , " An Algorithm for the Traveling Salesman Problem with Pickup and Delivery Customer," European Journal of Operations Research, Vol. 22 (1985) pp. 377-386.

[19] M. Kubo and H. Kasugai," Heuristic Algorithms for the Dial-A-Ride Routing Problem," Journal of Operations Research Society of Japan, Vol. 33, No.4 (1990) pp. 335-353.

[20] H. W. Kuhn ," The Hungarian Method for the Assignment Problem," Naval Research Logistics Quarterly, Vol. 2 (1955) pp. 83-97.

[21] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys ,"The Traveling Salesman Problem," John Wiley and Sons (1985).

[22] J. D. C. Little, K. G. Murty, D. M. Sweeney and C. Karel ,"An Algorithm for the Traveling Salesman Problem," Operations Research, Vol. 11 (1963) pp. 972-989.

[23] T. L. Magnanti ,"Combinatorial Optimization and Vehicle Fleet Planning: Perspectives and Prospects," Networks, Vol. 11 (1981) pp. 121-132.

[24] S. Martello ,"ALGORITHM 595 An Enumerative Algorithm for Finding Hamiltonian Circuits in a Directed Graph," ACM Transactions on Mathematical Software, Vol. 9 (1983) pp. 131-138.

[25] K. G. Murty," An Algorithm for Ranking all the Assignments in Order of Increasing Cost," Operations Research, Vol. 16 (1968) pp. 682-687.

[26] M. W. Padberg and S. Hong," On the Symmetric Traveling Salesman Problem: A Computational Study," Mathematical Programming Vol. 7 (1980) pp. 78-107.

[27] H. Psaraftis ,"A Dynamic Programming Solution to the Single Vehicle Many-to-Many

Intermediate Request Dial-A-Ride Problem," Transportation Science, Vol. 2 (1980) pp. 130-154.

[28] H. Psaraftis ,"Analysis of an $O(N^2)$ Heuristic for the Single Vehicle Many-to-Many Euclidean Dial-A-Ride Problem," Transportation Science, Vol. 117B (1983) pp. 133-145.

[29] H. Psaraftis ,"K-interchange Procedures for Local Search in a Precedence Constrained Routing Problem," European Journal of Operations Research, Vol. 13 (1983) pp. 391-402.

[30] N. Z. Shor ,"Minimization Methods for Nondifferentiable Functions," Springer-Verlag (1985).

[31] D. Stein ,"Scheduling Dial-A-Ride Transportation System," Transportation Research, Vol. 12 (1978) pp. 232-249.

[32] D. Stein ,"An Asymptotic, Probabilistic Analysis of a Routing Problem," Mathematics of Operations Research, Vol. 3 (1978) pp. 89-101

[33] H. Suzuki and S. Nomura ,"A Shortest Path Problem with Visiting Order Constraints," (in Japanese) Proceedings of the 7th Mathematical Programming Symposium Japan (1986) pp. 127-142.

[34] R. E. Tarjan ,"Data Structures and Network Algorithms," SIAM Publications (1983).

[35] S. Warshall ,"A Theorem on Boolean Matrices," Journal of the ACM, Vol. 9 (1962) pp. 11-12.

[36] A. Wren and A. Holiday ,"Computer Scheduling of Vehicle from One or More Depot to a Number of Delivery Point," Operations Research Quarterly, Vol. 23 (1972) pp. 333-344.

Mikio KUBO : Department of
Industrial Engineering and
Management,
Waseda University,
3-4-1, Okubo Shinjuku,
Tokyo 169, Japan