# An Exact Algorithm for the Traveling Salesman Problem with Deliveries and Collections

**R. Baldacci**
*DISMI, University of Modena and Reggio E., V.le Allegri 15, 42100 Reggio E., Italy*

**E. Hadjiconstantinou**
*Imperial College, Management School, Exhibition Road, London SW7 2PG, United Kingdom*

**A. Mingozzi**
*Department of Mathematics, University of Bologna, Via Sacchi 3, 47023 Cesena, Italy*

**In this paper, we describe a new integer programming formulation for the Traveling Salesman Problem with mixed Deliveries and Collections (TSPDC) based on a two-commodity network flow approach. We present new lower bounds that are derived from the linear relaxation of the new formulation by adding valid inequalities, in a cutting-plane fashion. The resulting lower bounds are embedded in a branch-and-cut algorithm for the optimal solution of the TSPDC. Computational results on different classes of test problems taken from the literature indicate the effectiveness of the proposed method. © 2003 Wiley Periodicals, Inc.**

**Keywords:** traveling salesman problem; delivery and collection; valid inequalities; branch and cut

## 1. INTRODUCTION

The Traveling Salesman Problem with mixed Deliveries and Collections (TSPDC) is an extension of the well-known Traveling Salesman Problem (TSP) where the set of customers to be served is partitioned into two subsets, namely, delivery customers and collection customers. A vehicle with given capacity is stationed at a central depot and must be used to supply the delivery customers and to collect goods from the collection customers. In the TSPDC, the goods to be delivered are of a different type with respect to the goods to be collected. The vehicle leaves the depot with a load equal to the total demand of the delivery customers and returns to the depot with a load equal to the total demand of the collection customers. The TSPDC consists of determining a tour starting and ending at the depot, serving each customer exactly once, and having minimum length (the length of the tour is defined as the sum of the costs of the arcs composing it). The total load of the vehicle along the tour should never exceed the vehicle capacity. If, in any feasible solution, all delivery customers must precede the collection customers, then the problem is known as the Traveling Salesman Problem with Backhauls (TSPB).

Routing problems with deliveries and collections have many practical applications in the design and management of distribution systems (see [12]), such as the transportation of under-privileged children from home to vacation locations described in Mosheiov [19], which will be described briefly in Section 2.

Mosheiov [19] proposed a mathematical model for the TSPDC and heuristic algorithms based on the extension of methods for the standard TSP. Anily and Mosheiov [1] described a new heuristic with a worst-case performance ratio equal to 2 based on the solution of Shortest Spanning Trees. Gendreau et al. [11] proposed two heuristic algorithms for the TSPDC. The first was based on the optimal solution of a special case of TSPDC arising when the underlying graph is a cycle. In particular, they derived a linear time algorithm for the optimal solution of this special case and used it as a base for developing a heuristic for the general TSPDC. A further improvement is obtained by means of a second heuristic based on the tabu search approach that uses a neighborhood based on exchanges of two arcs. Gendreau et al. showed that their tabu search algorithm outperforms the heuristics of Mosheiov and of Anily and Mosheiov.

Heuristics for the TSPB were presented by Gendreau et al. [10] who presented the extension to TSPB of the GENIUS heuristic for the TSP.

The generalization of the TSPDC related to the Vehicle

Received July 2002; accepted March 2003
*Correspondence to*: R. Baldacci; e-mail: baldacci.roberto@unimore.it

Routing Problem (VRP), where several vehicles are available, was considered by Halse [14], who proposed a mathematical formulation and a heuristic algorithm based on a Lagrangian relaxation of the problem.

To our knowledge, no exact methods so far have been proposed in the literature for the optimal solution of the TSPDC.

In this paper, we investigate a new integer programming formulation for the TSPDC which is based on the two-commodity network flow formulation of the TSP described by Finke et al. [8]. This formulation is interesting in many different ways: It can be shown that its LP-relaxation satisfies a weak form of the subtour elimination inequalities. The formulation can also be modified to accommodate different constraints and, therefore, is capable of being extended to different routing problems. The two-commodity formulation was used by Lucena [18] to derive new lower bounds for the VRP and by Langevin et al. [17] for solving the TSP and the Makespan Problem with time windows. Recently, the two-commodity approach was used by Baldacci et al. [7] to derive a new mathematical formulation and a new branch-and-cut algorithm for the VRP. In this paper, we describe new lower bounds that are obtained from the linear relaxation of the new TSPDC formulation which are further strengthened by new valid inequalities and embedded in a branch-and-cut procedure to solve the problem optimally. The resulting branch-and-cut procedure was applied to a set of instances taken from the literature. The results indicate the effectiveness of the proposed method.

The paper is organized as follows: Section 2 gives a formal description of the TSPDC. In Section 3, the new two-commodity flow formulation of the TSPDC is presented. The computation of the lower bound for the TSPDC and the description of the valid inequalities are provided in Section 4. Section 5 describes the branch-and-cut algorithm for the exact solution of the TSPDC. Computational results are reported in Section 6.

## 2. PROBLEM DEFINITION

The TSPDC is defined as follows: Let $G = (V, E)$ be an undirected graph such that $V = \{0, n + 1\} \cup D \cup C$, where $D = \{1, \ldots, n_D\}$ corresponds to $n_D$ *delivery* customers, $C = \{n_D + 1, \ldots, n_D + n_C\}$ corresponds to $n_C$ *collection* customers, $n = n_D + n_C$, and the nodes 0 and $n + 1$ represent the depot. An integer quantity $q_i$ is associated with each customer $i \in V' = D \cup C$, where $q_i < 0$ if $i$ is a delivery customer and $q_i > 0$ if $i$ is a collection customer. We assume that $q_0 = q_{n+1} = 0$. Every edge $\{i, j\} \in E$ is assigned a nonnegative cost $c_{ij}$. It is assumed that $c_{0i} = c_{in+1}$, $\forall i \in V'$, and that $\{0, n + 1\} \notin E$. A vehicle of capacity $Q$ is located at depot 0 and must be used to supply the delivery customers and to collect goods from the collection customers.

A *feasible path* $P = (i_1, i_2, \ldots, i_{n+2})$ is a path in $G$ from node $i_1 = 0$ to node $i_{n+2} = n + 1$ such that each customer is visited exactly once and the vehicle load does not exceed $Q$ at any node of the path. For a given path $P$, $E(P)$ represents the set of edges of $G$ traversed. The cost of a feasible path corresponds to the sum of the costs of the edges forming the path. The TSPDC consists of determining the minimum-cost feasible path in $G$.

We denote by $Q_D = \sum_{i \in D} |q_i|$ and $Q_C = \sum_{i \in C} q_i$ the total quantity to be delivered to the delivery customers and the total quantity to be collected from the collection customers, respectively. To ensure the feasibility of the problem, we assume that $Q \geq \max[Q_D, Q_C]$. Notice that no feasible path $P$ exists where $i \in C$ with $Q_D + q_i > Q$ and $j \in D$ with $Q_C + |q_j| > Q$ are the first or the last customer visited, respectively. Thus, we assume that the edge set $E$ does not contain the following two edge subsets: $\{\{0, i\} : i \in C, Q_D + q_i > Q\}$ and $\{\{j, n + 1\} : j \in D, Q_C + |q_j| > Q\}$.

If $C = \emptyset$ and $q_i = 1$, $\forall i \in V'$, then the TSPDC reduce to the TSP, proving that the problem is NP-hard (see [9]).

Figure 1 shows an example of a TSPDC solution. The problem was generated by Mosheiov [19] for simulating the problem arising in the transportation of underprivileged children from home to vacation locations. In the specific application described by Mosheiov, a welfare organization must plan the transportation of underprivileged children from the organization site to vacation locations of volunteer families and back from the house of the families to the organization site. All transportations are performed by a bus and are confined to a limited number of dates, when some children end their vacation and others start theirs. The problem has 24 house locations, 13 of which are delivery locations (i.e., $|D| = 13$) and 11 are collection locations (i.e., $|C| = 11$). The bus capacity is equal to 45 seats (i.e., $Q = 45$). The total demand of the deliveries is equal to 45 and is equal to the total demand of the collections (i.e., $Q_D = Q_C = 45$). This example will also be used in Section 4.5 to illustrate our bounding procedure.

## 3. A NEW TWO-COMMODITY FLOW FORMULATION OF THE TSPDC

In this section, we describe a new integer programming formulation of the TSPDC based on a two-commodity network flow approach. This formulation uses two flow variables, $x_{ij}$ and $x_{ji}$, to represent an edge $\{i, j\} \in E$ of a feasible TSPDC solution. If the vehicle travels from $i$ to $j$, then flow $x_{ij}$ represents the load of the vehicle and flow $x_{ji}$ represents the empty space on the vehicle (i.e., $x_{ji} = Q - x_{ij}$). Conversely, if the vehicle travels from $j$ to $i$, the two flows $x_{ij}$ and $x_{ji}$ represent the empty space and the load of the vehicle, respectively.

The flow variables $x_{ij}$ and $x_{ji}$, $\forall \{i, j\} \in E$, define two flow paths for any feasible solution: One path from node 0 to node $n + 1$ is given by the flow variables representing the vehicle load, while the second path from node $n + 1$ to node 0 is defined by the flow variables representing the empty space on the vehicle.

Figure 2 shows an example of feasible TSPDC solution

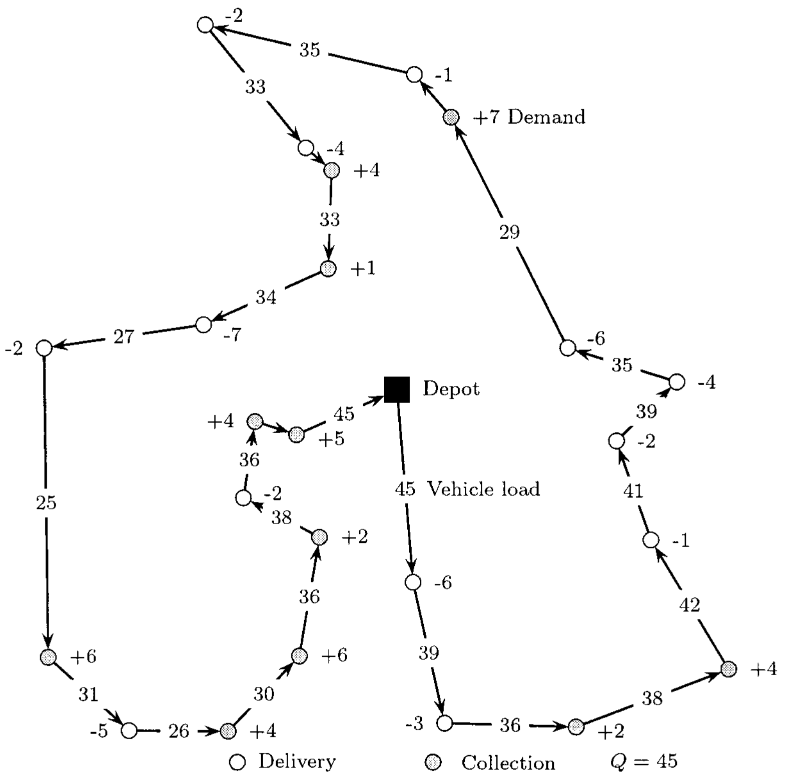


FIG. 1. Example of TSPDC solution (problem data obtained from Mosheiov [19]).

for a four-customer problem and a vehicle of capacity $Q = 10$, where the two paths $P^\alpha$ and $P^\beta$ are represented by the flow variables $\{x_{ij}\}$.

Path $P^\alpha$ is given by the variables representing the vehicle load: $\{x_{0\,1}, x_{1\,3}, x_{3\,2}, x_{2\,4}, x_{4\,5}\}$. For example, the flow $x_{01} = 9$ indicates that the vehicle leaves depot 0 with a load equal to the total demand of the two delivery customers (i.e., $Q_D = 9$). Path $P^\beta$ is defined by the variables representing the empty space on the vehicle: $\{x_{5\,4}, x_{4\,2}, x_{2\,3}, x_{3\,1}, x_{1\,0}\}$. For example, $x_{54} = 4$ indicates the empty space in

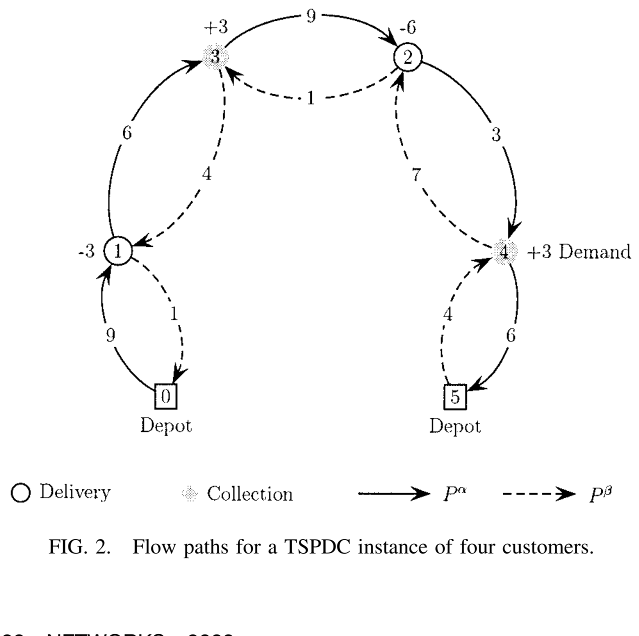


FIG. 2. Flow paths for a TSPDC instance of four customers.

the vehicle arriving at the depot. Note that for every edge $\{i, j\}$ of the path $x_{ij} + x_{ji} = Q$.

In the integer programming formulation, we will use the following notation: Let $\mathcal{L}$ be the family of all subsets of nodes in $V'$ such that $|S| \geq 3$, $\forall S \in \mathcal{L}$ (i.e., $\mathcal{L} = \{S : S \subseteq V'$ and $|S| \geq 3\}$). Let $q(S) = \sum_{i \in S} q_i$ denote the total net demand of the customers in $S$ and $\bar{S} = V \backslash S$. Let us define $\mathcal{L}' = \{S : S \in \mathcal{L}$ and $q(S) = 0\}$.

Let $\xi_{ij}$ be a 0–1 binary variable which is equal to 1 if and only if edge $\{i, j\} \in E$ is in the solution. Let $x_{ij}$ be the flow value of arc $(i, j)$ and $x_{ji}$ be the flow value of arc $(j, i)$, $\{i, j\} \in E$. The mathematical formulation of the TSPDC is as follows:

$$(F1) \quad z(F1) = \min \sum_{\{i,j\} \in E} c_{ij}\xi_{ij} \tag{1}$$

$$s.t. \sum_{j \in V} (x_{ij} - x_{ji}) = 2q_i, \quad \forall\, i \in V' \tag{2}$$

$$\sum_{j \in V'} x_{0j} = Q_D \tag{3}$$

$$\sum_{j \in V'} x_{jn+1} = Q_C \tag{4}$$

$$x_{ij} + x_{ji} = Q\xi_{ij}, \quad \forall\, \{i, j\} \in E \tag{5}$$

$$\sum_{\substack{j \in V \\ i<j}} \xi_{ij} + \sum_{\substack{j \in V \\ j<i}} \xi_{ji} = 2, \quad \forall\, i \in V' \tag{6}$$

$$\sum_{j \in V'} \xi_{0j} = 1 \tag{7}$$

$$\sum_{j \in V'} \xi_{jn+1} = 1 \tag{8}$$

$$\sum_{i \in S} \sum_{\substack{j \in \bar{S} \\ i<j}} \xi_{ij} + \sum_{i \in \bar{S}} \sum_{\substack{j \in S \\ i<j}} \xi_{ij} \geq 2, \ \forall\, S \in \mathscr{L}' \tag{9}$$

$$x_{ij} \geq 0,\, x_{ji} \geq 0, \quad \forall\, \{i, j\} \in E \tag{10}$$

$$\xi_{ij} \in \{0, 1\}, \quad \forall\, \{i, j\} \in E \tag{11}$$

Equations (2)–(4) and the nonnegativity constraints (10) define a feasible flow pattern from the source nodes 0 and the set of collection customers $C$ to the sink nodes $n + 1$ and $D$. The *outflow* at source node 0 [see Eq. (3)] is equal to the total demand of the delivery customers, while the *inflow* at sink node $n + 1$ [see Eq. (4)] corresponds to the total demand of the collection customers. Equations (2) state that the outflow minus the inflow at each customer $i \in V'$ is equal to $2q_i$. Constraints (5) define the edges of a feasible solution and constraints (6)–(8) force any feasible solution to contain two edges incident to each customer and one edge incident to nodes 0 and $n + 1$, respectively. Inequalities (9) are the subtour elimination inequalities for the family of subsets $\mathscr{L}'$. Notice that equations (6) avoid subtours formed by two nodes only.

The following lemma shows that formulation F1 implicitly satisfies subtour elimination inequalities for any set $S \in \mathscr{L}\backslash\mathscr{L}'$:

**Lemma 1.** *Given a feasible solution* ($\mathbf{x}$, $\boldsymbol{\xi}$) *of F*1, *the following inequality is satisfied by any set* $S \in \mathscr{L}\backslash\mathscr{L}'$:

$$\sum_{i \in S} \sum_{\substack{j \in \bar{S} \\ i<j}} \xi_{ij} + \sum_{i \in \bar{S}} \sum_{\substack{j \in S \\ i<j}} \xi_{ij} \geq 2. \tag{12}$$

**Proof.** Adding flow constraints (2) for customers in $S$, we have

$$\sum_{i \in S} \left( \sum_{j \in S} (x_{ij} - x_{ji}) + \sum_{j \in \bar{S}} (x_{ij} - x_{ji}) \right) = 2q(S). \tag{13}$$

We must consider two cases:

(a) $q(S) > 0$.

As $\sum_{i \in S} \sum_{j \in S} (x_{ij} - x_{ji}) = 0$ and $x_{ij} + x_{ji} \geq x_{ij} - x_{ji}$, $\forall \{i, j\} \in E$, from Eq. (13) we obtain the following inequality:

$$\sum_{i \in S} \sum_{j \in \bar{S}} (x_{ji} + x_{ij}) \geq 2q(S). \tag{14}$$

Using Eqs. (5), inequality (14) becomes

$$\sum_{i \in S} \sum_{\substack{j \in \bar{S} \\ i<j}} \xi_{ij} + \sum_{i \in \bar{S}} \sum_{\substack{j \in S \\ i<j}} \xi_{ij} \geq 2q(S)/Q. \tag{15}$$

Because of Eqs. (6) and the integrality constraints (11), the left-hand side of inequality (15) must not only be integer, but also even. Hence, from expression (15), we have

$$\sum_{i \in S} \sum_{\substack{j \in \bar{S} \\ i<j}} \xi_{ij} + \sum_{i \in \bar{S}} \sum_{\substack{j \in S \\ i<j}} \xi_{ij} \geq 2\lceil q(S)/Q \rceil, \tag{16}$$

and since $0 < q(S) \leq Q$, inequality (16) becomes inequality (12).

(b) $q(S) < 0$.

Similar to case (a), multiplying Eq. (13) by $-1$ and using $\sum_{i \in S} \sum_{j \in S} (-x_{ij} + x_{ji}) = 0$ and $x_{ij} + x_{ji} \geq -x_{ij} + x_{ji}$, $\forall \{i, j\} \in E$, we obtain the following inequality:

$$\sum_{i \in S} \sum_{j \in \bar{S}} (x_{ji} + x_{ij}) \geq -2q(S). \tag{17}$$

By means of the observations used above to show that inequality (14) implies inequality (12), it can be shown that inequality (17) implies inequality (12). ■

The validity of formulation F1 is shown by the following theorem:

**Theorem 1.** *The set of solutions to* (2)–(11) *corresponds to the set of solutions to TSPDC.*

**Proof.**

(A) Every TSPDC path $P = (i_1, i_2, \ldots, i_{n+2})$ provides a feasible solution to (2)–(11) by defining variables $\mathbf{x}$ and $\boldsymbol{\xi}$ as follows:

– $\xi_{ij}=1, \forall\{i, j\} \in E(P)$;

– $x_{i_1 i_2}=Q_D$ and $x_{i_k i_{k+1}} = x_{i_{k-1} i_k} + q_{i_k}, k = 2, \ldots, n + 1$;

$-x_{i_k i_{k-1}} = Q - x_{i_{k-1} i_k}, k = 2, \ldots, n + 1$;

$-\xi_{ij} = 0$ and $x_{ij} = x_{ji} = 0$ for each edge $\{i, j\} \in E\backslash E(P)$.

It is easy to verify that variables ($\mathbf{x}$, $\boldsymbol{\xi}$) as defined above are a feasible solution to (2)–(11).

(B) Every solution ($\mathbf{x}$, $\boldsymbol{\xi}$) to (2)–(11) is a TSPDC solution.

Consider the partial graph $G(F) = (V, F)$, where $F = \{\{i, j\} : \{i, j\} \in E \text{ and } \xi_{ij} = 1\}$. The degree of each node of $V'$ in $G(F)$ is 2 [see Eqs. (6)], and due to Eqs. (7) and (8), the degree of nodes 0 and $n + 1$ is equal to 1. Moreover, using Lemma 1, the subgraph $G(F)$ does not contains subtours involving customer subsets $S \in \mathcal{L}\backslash\mathcal{L}'$. Subtours involving customer subsets $S \in \mathcal{L}'$ are explicitly avoided by constraints (9). Since subtour elimination inequalities are satisfied for any subset $S \in \mathcal{L}$, then the subgraph $G(F)$ is connected and corresponds to a simple path from node 0 to node $n + 1$. Moreover, from Eqs. (5), the vehicle load is not exceeded at any node of the path. Thus, $G(F)$ corresponds to a feasible TSPDC solution. ■

## 4. A LOWER BOUND ON THE TSPDC

This section investigates a lower bound on the TSPDC obtained from the LP-relaxation of formulation F1 by removing constraints (9) and (11) and by adding valid inequalities for strengthening the lower bound value.

After removing constraints (9) and using Eqs. (5) to eliminate variables $\{\xi_{ij}\}$ from the objective function (1) and from Eqs. (6), (7), and (8), the LP-relaxation of formulation F1 can be written as follows:

$$(LF1) \quad z(LF1) = \min \frac{1}{Q} \sum_{\{i,j\} \in E} c_{ij}(x_{ij} + x_{ji}) \tag{18}$$

$s.t.$(2), (3), (4) and

$$x_{ij} + x_{ji} \le Q, \quad \forall \{i, j\} \in E \tag{19}$$

$$\sum_{j \in V} (x_{ij} + x_{ji}) = 2Q, \quad \forall\, i \in V' \tag{20}$$

$$\sum_{j \in V'} (x_{ij} + x_{ji}) = Q, \quad i = 0, n + 1 \tag{21}$$

$$x_{ij} \ge 0, x_{ji} \ge 0, \quad \forall \{i, j\} \in E. \tag{22}$$

The following lemma shows that formulation LF1 satisfies a weak form of the subtour elimination inequalities for every $S \in \mathcal{L}\backslash\mathcal{L}'$:

**Lemma 2.** *The variables* $\xi_{ij} = (x_{ij} + x_{ji})/Q$, $\forall\{i, j\} \in E$, *associated with a feasible solution* $\mathbf{x}$ *to* $LF1$, *satisfy the following inequalities*:

$$\sum_{i \in S} \sum_{\substack{j \in \bar{S} \\ i<j}} \xi_{ij} + \sum_{i \in \bar{S}} \sum_{\substack{j \in S \\ i<j}} \xi_{ij} \ge 2q(S)/Q, \quad \forall\, S \in \mathcal{L}\backslash\mathcal{L}'. \tag{23}$$

**Proof.** Similar to the proof of Lemma 1.

### 4.1. *Valid Inequalities to Improve the Lower Bound $z(LF1)$*

In this section, three classes of valid inequalities are described which have been used to improve the lower bound $z(LF1)$. These inequalities are satisfied by any feasible integer solution of formulation F1, but can be violated by an LF1 solution. Hence, the lower bound can be strengthened by adding to LF1, in a cutting plane fashion, those inequalities that are violated in the current solution.

It must be noted that, given a TSPDC instance, the optimal solution cost of the associated TSP instance is a valid lower bound on the optimal TSPDC solution cost. Therefore, the lower-bound LF1 can be improved by considering, in addition to the inequalities described in this section, all other inequalities known for the TSP (see Naddef and Pochet [20]). These latter inequalities are expressed in terms of variables $\{\xi_{ij}\}$ defined in F1 but can be added to LF1 once variables $\{\xi_{ij}\}$ are replaced with variables $\{x_{ij}\}$ using Eqs. (5). However, implementing these inequalities for the TSPDC not only requires a large amount of computational effort (see Applegate et al. [3]) but also makes little contribution to the relevant theory. Therefore, the main focus of this paper is to concentrate and investigate new valid inequalities specifically developed for the TSPDC.

**Flow Inequalities.** These inequalities are based on the following lemma:

**Lemma 3.** *Any feasible solution* $(\mathbf{x}, \boldsymbol{\xi})$ *of* $F1$ *satisfies the following inequalities*, *called flow inequalities*:

$$x_{ij}(Q - |q_j|) - |q_j| x_{ji} \ge 0, \quad \forall \{i, j\} \in E \quad or \quad \forall \{j, i\} \in E \quad with \quad j \in D, i \in V'\backslash\{j\}, \tag{24}$$

*and*

$$x_{ji}(Q - q_j) - q_j x_{ij} \ge 0, \quad \forall \{i, j\} \in E \quad or \quad \forall \{j, i\} \in E \quad with \quad j \in C, i \in V'\backslash\{j\}. \tag{25}$$

**Proof.** Inequalities (24) and (25) are satisfied by the two flow variables associated with edge $\{i, j\}$ such that $\xi_{ij} = 0$ since from Eq. (5) we have $x_{ij} = x_{ji} = 0$. Let $\{i, j\}$ and $\{j, k\}$ be the two edges incident to customer $j \in V'$ in a given F1 solution such that $\xi_{ij} = 1$ and $\xi_{jk} = 1$. We must consider two cases:

(a) $j \in D$. Using Eqs. (2) and (5),

$$-x_{ij} + x_{ji} - x_{kj} + x_{jk} = 2q_j, \tag{26}$$

$$x_{ij} + x_{ji} = Q, \tag{27}$$

$$x_{kj} + x_{jk} = Q. \tag{28}$$

Adding up these equations after having multiplied Eq. (26) by $-1$, we obtain

$$x_{ij} + x_{kj} = |q_j| + Q. \quad (29)$$

As $x_{kj} \leq Q$, from Eq. (29), we have $x_{ij} \geq |q_j|$. Hence, any feasible solution to F1 satisfies the following inequalities:

$$x_{ij} \geq |q_j|\xi_{ij}, \quad \forall \{i, j\} \in E \quad \text{with} \quad j \in D, i \in V' \quad (30)$$

and

$$x_{ij} \geq |q_j|\xi_{ji}, \quad \forall \{j, i\} \in E \quad \text{with} \quad j \in D, i \in V'. \quad (31)$$

Inequalities (24) follow directly from inequalities (30), (31), and Eqs. (5).

(b) $j \in C$. Using Eqs. (2), we obtain

$$-x_{ij} + x_{ji} - x_{kj} + x_{jk} = 2q_j, \quad (32)$$

while using Eqs. (5), we derive the two Eqs. (27) and (28). Adding up Eqs. (32), (27), and (28), we obtain

$$x_{ji} + x_{jk} = q_j + Q. \quad (33)$$

As $x_{jk} \leq Q$, from Eq. (33), we have $x_{ji} \geq q_j$. Hence, any feasible solution to F1 satisfies the following inequalities:

$$x_{ji} \geq q_j\xi_{ij}, \quad \forall \{i, j\} \in E \quad \text{with} \quad j \in C, i \in V' \quad (34)$$

and

$$x_{ji} \geq q_j\xi_{ji}, \quad \forall \{j, i\} \in E \quad \text{with} \quad j \in C, i \in V'. \quad (35)$$

Inequalities (25) follow directly from inequalities (34), (35), and Eqs. (5). ∎

**Subtour Elimination Inequalities.** Consider the following subtour elimination inequalities:

$$\sum_{i\in S}\sum_{\substack{j\in\bar{S}\\ i<j}} \xi_{ij} + \sum_{i\in\bar{S}}\sum_{\substack{j\in S\\ i<j}} \xi_{ij} \geq 2, \quad \forall S \in \mathcal{L}, \quad (36)$$

which are satisfied by any feasible solution to F1. Let **x** be a feasible solution to LF1. Then, the following inequalities

$$\sum_{i\in S}\sum_{j\in\bar{S}} (x_{ij} + x_{ji}) \geq 2Q, \quad \forall S \in \mathcal{L}, \quad (37)$$

which are obtained from (36) using Eqs. (5), can be violated by solution **x**. Therefore, inequalities (37) can be used to improve the value of lower bound $z(LF1)$.

**Capacity Inequalities.** These valid inequalities are based on the following observations:

Let $S \subset V'$ be such that $Q_D + q(S) > Q$. No feasible TSPDC solution exists in which the vehicle leaves depot 0 and visits all customers in $S$ before visiting any other customer in $V'\backslash S$, since, otherwise, the vehicle load would exceed the capacity $Q$. Therefore, any feasible TSPDC solution must satisfy the following inequalities, hereafter called *capacity inequalities:*

$$\sum_{i\in S\cup\{0\}}\sum_{\substack{j\in\bar{S}\\ i<j}} \xi_{ij} + \sum_{i\in\bar{S}}\sum_{\substack{j\in S\cup\{0\}\\ i<j}} \xi_{ij} \geq 3,$$
$$\forall S \subseteq V', Q_D + q(S) > Q. \quad (38)$$

Similarly, if we consider a subset $S \subset V'$ of customers such that $Q_C - q(S) > Q$, then the following inequality is also satisfied by any feasible F1 solution:

$$\sum_{i\in S\cup\{n+1\}}\sum_{\substack{j\in\bar{S}\\ i<j}} \xi_{ij} + \sum_{i\in\bar{S}}\sum_{\substack{j\in S\cup\{n+1\}\\ i<j}} \xi_{ij} \geq 3. \quad (39)$$

Similar to the subtour elimination inequalities, inequalities (38) and (39) can be written in term of variables $\{x_{ij}\}$ only and can be added to LF1 to improve the lower bound.

### 4.2. Variable Reduction

The number of variables of LF1 can be reduced by means of the following observations:

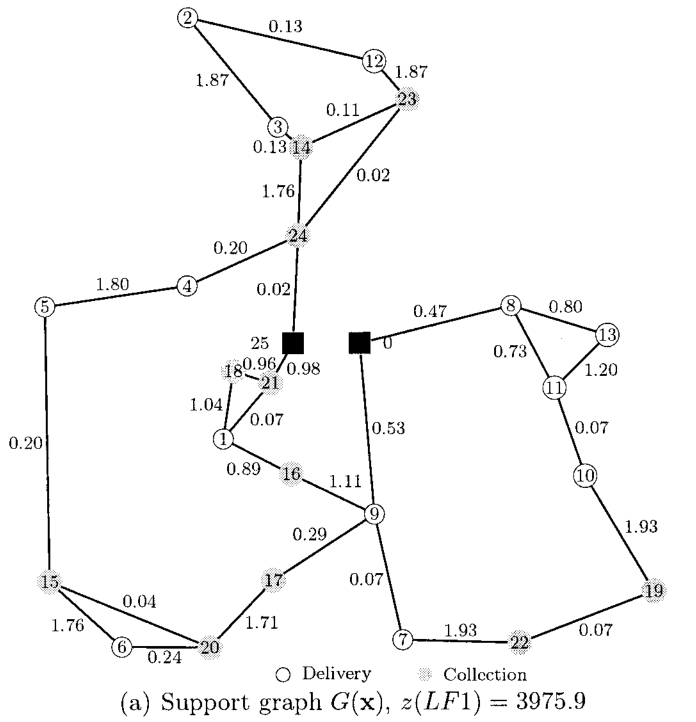
Let us denote by $\bar{z}(i, j)$ the optimal solution cost of LF1 imposing $x_{ij} + x_{ji} = Q$ (i.e., forcing edge $\{i, j\} \in E$ to be in the optimal TSPDC solution). The two flow variables $x_{ij}$ and $x_{ji}$ can be removed from problem LF1 and edge $\{i, j\}$ can be removed from $E$ if $\bar{z}(i, j) \geq z_{UB}$, where $z_{UB}$ is a known upper bound on the TSPDC. A possible way to compute a lower bound on $\bar{z}(i, j)$ is as follows:

Let $\{c'_{ij}\}$ be the reduced cost of variables $x_{ij}$ corresponding to the optimal solution of LF1. A lower bound on $\bar{z}(i, j)$ for fixing edge $\{i, j\}$ in the LF1 solution can be computed by solving the following LP:

$$\bar{z}(i, j) = z(LF1) + \frac{1}{Q} \min_{y_{min}\leq y\leq y_{max}} c'_{ij}y + c'_{ji}(Q - y), \quad (40)$$

where $y$ is the value of flow variable $x_{ij}$ and, consequently, $Q - y$ is the value of $x_{ji}$ satisfying the additional constraint $x_{ij} + x_{ji} = Q$. The values $y_{min}$ and $y_{max}$ represent the minimum and maximum values of variable $x_{ij}$ in any feasible TSPDC solution containing edge $\{i, j\}$, respectively. We have the following three cases:

(a) $i, j \in D$. Then, $x_{ij} \geq |q_j|$ and $x_{ji} \geq |q_i|$; therefore, $y_{min} = |q_j|$ and $y_{max} = Q - |q_i|$;
(b) $i, j \in C$. Then, $x_{ij} \geq |q_i|$ and $x_{ji} \geq |q_j|$; therefore, $y_{min} = |q_i|$ and $y_{max} = Q - |q_j|$;
(c) $i \in C$ and $j \in D$, or $i \in D$ and $j \in C$. In both cases, $x_{ij} \geq 0$ and $x_{ji} \geq 0$; therefore, $y_{min} = 0$ and $y_{max} = Q$.



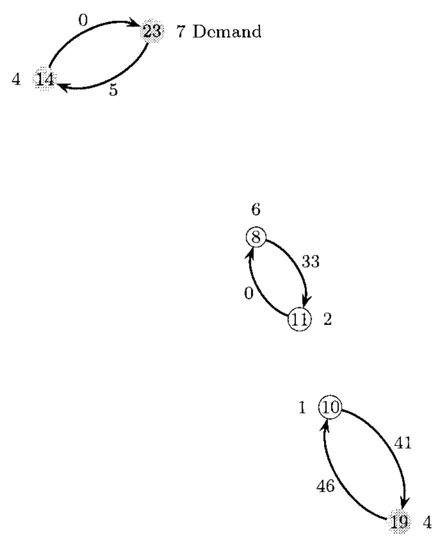


FIG. 3. Optimal LF1 solution before adding any valid inequalities.

### 4.3. Separation of Violated Inequalities

Let $\mathbf{x}$ be the optimal LF1 solution at any iteration of the cutting-plane procedure and let $G(\mathbf{x})$ be the weighted graph induced by the edges $\{i, j\} \in E$ with $(x_{ij} + x_{ji})/Q > 0$, which will be called the *support graph* of $\mathbf{x}$. Given the support graph $G(\mathbf{x})$, the separation problem for flow inequalities (24) and (25) can be easily solved by inspection.

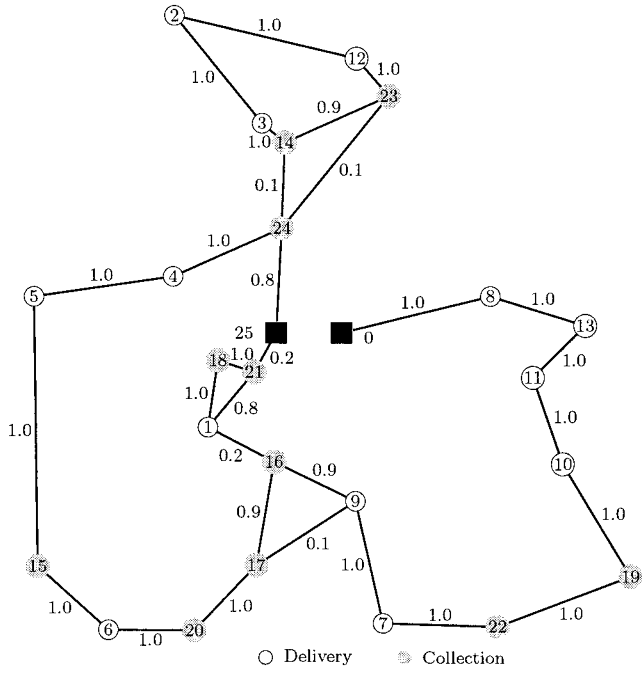


FIG. 4. Optimal LF1 solution after adding 13 trivial inequalities and 10 flow inequalities, $z(LF1) = 4312.9$.

**Separation of Subtour Elimination Inequalities.** A solution $\mathbf{x}$ of LF1 violates a subtour elimination inequality if and only if the minimum-weight cut in the supporting graph $G(\mathbf{x})$ has weight less than 2. Since the minimum-weight cut in a graph with nonnegative edge weights can be found in polynomial time using the algorithm proposed by Gomory and Hu [13], the separation problem for the subtour elimination inequalities can be solved in polynomial time. However, the procedure that we implemented for the identification of violated subtour elimination inequalities is based on the method proposed by Padberg and Rinaldi [21]. This algorithm has the same worst-case time bound as that of the Gomory–Hu algorithm, but it runs much faster in practice and it allows the execution of an exact separation algorithm at every iteration of a branch-and-cut procedure.

**Separation of Capacity Inequalities.** Capacity inequalities are identified using the heuristic method, called *greedy randomized algorithm,* proposed by Augerat et al. [5] to separate capacity constraints in the VRP.

The greedy randomized algorithm is an iterative procedure that is applied to a number of customer subsets generated *a priori.* Each subset $S \subseteq V'$, $S \neq \emptyset$, is expanded by the algorithm as follows: At each iteration, a customer $i^*$ $\in V' \backslash S$ such that

$$\sum_{j\in S} (x_{i^*j} + x_{ji^*}) = \max_{i\in V'\backslash S} \sum_{j\in S} (x_{ij} + x_{ji})$$

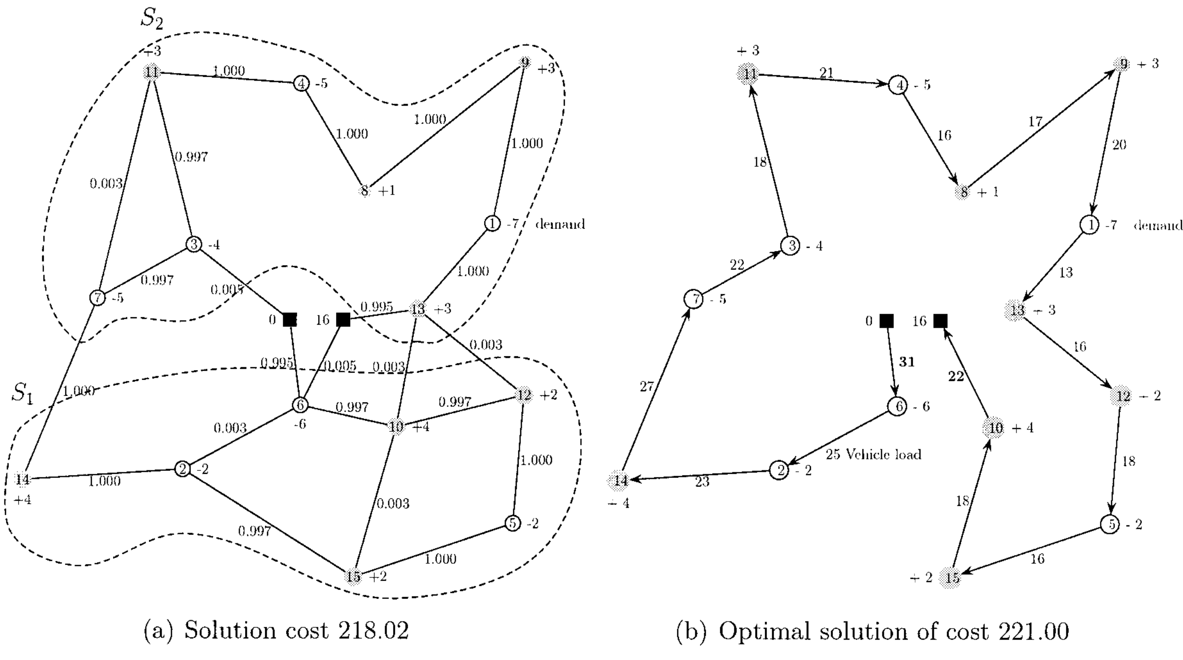

(a) Solution cost 218.02 (b) Optimal solution of cost 221.00

FIG. 5. Numerical example: capacity inequalities.

is added to $S$. Then, the set is checked for a possible violation of inequalities (38) and (39). The procedure is repeated until $S$ contains all customers in $V'$.

In our implementation, the initial customer subsets were build by randomly generating $10n$ subset of customers.

### 4.4. Computation of the Lower Bound

The lower bound on the TSPDC is computed using a cutting-plane-based linear programming procedure. In our implementation, it was found to be computationally efficient to remove from LF1 constraints (19) and to consider them as valid inequalities, called *trivial inequalities,* which are only added to the linear program when violated. Each iteration of the bounding procedure consists of three sets of inner iterations. First, a maximum number of iterations *ITERTF* is performed to identify violated trivial and flow inequalities and, second, a maximum number of iterations *ITERS* is performed to identify violated subtour inequalities. Finally, a maximum number of iterations *ITERC* is performed to identify violated capacity inequalities. Each one of these inner iterations solves the linear program resulting from LF1 and those inequalities identified during the previous iterations. The overall bounding procedure terminates when at least one of the following conditions is satisfied:

1. No further violated inequalities can be generated;
2. The lower bound did not increase during a certain number of iterations *ITERLB*;
3. A maximum number of iterations *ITERMAX* has been achieved.

In the numerical examples presented in the next section and in the computational results of Section 6, the following setting of parameters was used: $ITERTF = 10$, $ITERS = 10$, $ITERC = 10$, $ITERLB = 5$, and $ITERMAX = 20$.

### 4.5. Numerical Examples

This section presents two numerical examples to illustrate the procedure for the computation of the lower bound described in the previous section. The first one shows an example of the lower-bound computation, while the second one shows an example of violated capacity inequalities.

Consider the TSPDC instance described in Section 1. The optimal solution to this problem is given by $z(F1) = 4464$ and it is shown in Figure 1. The cost of the optimal TSP solution, which is a valid lower bound on the cost of the optimal solution cost of the problem, is 4430.

The first step of the bounding procedure involves solving problem LF1. The optimal solution value is found to be $z(LF1) = 3975.9$ and the corresponding support graph $G(\mathbf{x})$ is shown in Figure 3(a), where the weight $(x_{ij} + x_{ji})/Q$ is shown on each edge $\{i, j\}$ of $G(\mathbf{x})$.

Figure 3(b) shows that edge $\{10, 19\}$ violates the trivial inequalities as $x_{10\ 19} + x_{19\ 10} = 87$ while the edges $\{14, 23\}$ and $\{8, 11\}$ violate the corresponding flow inequalities as $x_{14\ 23}$ and $x_{11\ 8}$ are equal to 0.

Two iterations were subsequently performed to identify 13 trivial inequalities and 10 flow inequalities violated by the current LF1 solution. The optimal solution of the resulting linear program LF1 with cost $z(LF1) = 4312.9$ is shown in Figure 4.

TABLE 1. Class A: lower bounds.

| $n$ | $\beta$ | $\%z_{TSP}$ | $\%LB_0$ | $\%LB_1$ | $\%LB_2$ | $t_{LB_2}$ | $Triv$ | $Flow$ | $Sub$ | $Cap$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 15 | 0.05 | 100.0 | 93.1 | 100.0 | 100.0 | 1.3 | 9 | 1 | 0 | 0 |
| 15 | 0.10 | 98.6 | 91.9 | 98.6 | 99.5 | 0.4 | 11 | 10 | 1 | 4 |
| 15 | 0.20 | 98.6 | 91.9 | 98.6 | 99.5 | 0.4 | 11 | 10 | 2 | 4 |
| 25 | 0.05 | 100.0 | 78.1 | 99.7 | 99.7 | 0.4 | 13 | 2 | 2 | 0 |
| 25 | 0.10 | 99.4 | 77.6 | 99.0 | 99.0 | 0.4 | 16 | 5 | 2 | 0 |
| 25 | 0.20 | 97.8 | 76.4 | 97.4 | 97.8 | 0.4 | 15 | 11 | 2 | 18 |
| 35 | 0.05 | 100.0 | 91.5 | 99.7 | 99.7 | 0.5 | 19 | 8 | 5 | 0 |
| 35 | 0.10 | 100.0 | 91.5 | 99.7 | 99.7 | 0.5 | 21 | 12 | 8 | 0 |
| 35 | 0.20 | 100.0 | 91.5 | 99.7 | 99.7 | 0.5 | 22 | 17 | 7 | 0 |
| 44 | 0.05 | 100.0 | 64.5 | 100.0 | 100.0 | 0.8 | 28 | 33 | 16 | 27 |
| 44 | 0.10 | 100.0 | 64.6 | 100.0 | 100.0 | 0.7 | 26 | 41 | 16 | 16 |
| 44 | 0.20 | 100.0 | 64.8 | 100.0 | 100.0 | 1.0 | 29 | 48 | 14 | 71 |
| 50 | 0.05 | 100.0 | 88.5 | 99.3 | 99.3 | 0.6 | 25 | 8 | 6 | 0 |
| 50 | 0.10 | 100.0 | 88.5 | 99.3 | 99.3 | 0.6 | 28 | 20 | 4 | 0 |
| 50 | 0.20 | 100.0 | 88.5 | 99.3 | 99.5 | 0.7 | 33 | 30 | 5 | 16 |
| 75 | 0.05 | 100.0 | 90.1 | 100.0 | 100.0 | 0.8 | 43 | 18 | 3 | 0 |
| 75 | 0.10 | 99.8 | 90.0 | 99.8 | 99.8 | 1.2 | 50 | 32 | 5 | 0 |
| 75 | 0.20 | 99.8 | 90.0 | 99.8 | 99.8 | 1.2 | 49 | 47 | 8 | 0 |
| 100 | 0.05 | 100.0 | 65.3 | 99.2 | 99.2 | 4.3 | 60 | 56 | 30 | 22 |
| 100 | 0.10 | 100.0 | 65.3 | 99.2 | 99.2 | 4.6 | 60 | 99 | 28 | 49 |
| 100 | 0.20 | 99.8 | 65.1 | 99.4 | 99.4 | 5.4 | 65 | 94 | 24 | 142 |
| 120 | 0.05 | 100.0 | 65.4 | 99.4 | 99.4 | 4.8 | 70 | 35 | 32 | 0 |
| 120 | 0.10 | 100.0 | 65.4 | 99.4 | 99.4 | 5.0 | 81 | 52 | 24 | 15 |
| 120 | 0.20 | 100.0 | 65.2 | 99.4 | 99.4 | 5.1 | 73 | 83 | 24 | 52 |
| 150 | 0.05 | 100.0 | 82.1 | 99.7 | 99.7 | 5.7 | 89 | 25 | 13 | 0 |
| 150 | 0.10 | 99.9 | 82.0 | 99.6 | 99.7 | 8.3 | 99 | 57 | 15 | 9 |
| 150 | 0.20 | 99.4 | 81.6 | 99.0 | 99.4 | 7.2 | 104 | 71 | 13 | 30 |
| 199 | 0.05 | 100.0 | 82.7 | 99.9 | 99.9 | 20.5 | 114 | 85 | 34 | 7 |
| 199 | 0.10 | 100.0 | 82.7 | 99.6 | 99.9 | 24.3 | 127 | 130 | 36 | 37 |
| 199 | 0.20 | 99.9 | 82.5 | 99.7 | 99.7 | 30.5 | 139 | 186 | 32 | 99 |
| 261 | 0.05 | 99.8 | 80.7 | 98.9 | 98.9 | 82.6 | 153 | 144 | 53 | 338 |
| 261 | 0.10 | 99.5 | 80.5 | 98.7 | 98.7 | 250.8 | 177 | 197 | 56 | 1708 |
| 261 | 0.20 | 99.4 | 80.4 | 98.5 | 98.6 | 204.5 | 189 | 249 | 45 | 1778 |
| Avg. | | 99.7 | 80.0 | 99.4 | 99.5 | | | | | |

An example of violated subtour inequalities is given by the set $S = \{2, 3, 12, 14, 23\}$, where $\sum_{i \in S} \sum_{j \in \bar{S}} (x_{ij} + x_{ji}) = 0.2$. The lower bound is further improved by adding a set of seven violated subtour inequalities (generated in seven consecutive iterations) to problem LF1. The optimal solution value of the final linear program LF1 is $z(LF1) = 4464$, which corresponds to the optimal integer solution shown in Figure 1.

Figure 5 shows two examples of violated capacity inequalities. The problem has $n = 15$ customers. The vehicle capacity $Q$ is equal to 31, while $Q_D$ and $Q_C$ are equal to 31 and 22, respectively. Figure 5(a) shows the optimal LF1 solution of cost 218.02 obtained after adding to LF1 all violated trivial, flow, and subtour inequalities. Two violated capacity inequalities, represented in Figure 5(a) by the two sets $S_1$ and $S_2$, can be identified. For set $S_1$, we have $q(S_1) = 2$ and the left-hand side of inequality (38) is equal to 1.016, while for set $S_2$, we have $q(S_2) = -11$ and the left-hand side of inequality (39) is equal to 1.016. After adding the two violated capacity inequalities to LF1, the solution cost becomes 220.00 while the optimal solution, shown in Figure 5(b), has cost 221.

## 5. A BRANCH-AND-CUT ALGORITHM

The cutting-plane procedure for the computation of the lower bound of the TSPDC was embedded into a branch-and-cut scheme to solve the TSPDC to optimality. At each node of the branch-and-cut tree, the lower bound is computed by solving the LP program defined by problem LF1, the constraints derived from branching plus all valid inequalities found so far. A node of the branch-and-cut tree is fathomed if the lower-bound solution is a feasible solution or the lower bound obtained is not less than is the current upper bound. If a node is not fathomed, the incumbent solution is further divided in two subproblems by branching on a given inequality in the way described below. The subproblem chosen for expansion has the minimum lower bound among those not expanded. Two branching strategies have been implemented: branching on sets and branching on variables.

Branching on sets has been used in the branch-and-cut algorithm for the VRP by Augerat et al. [5]. This strategy consists of choosing a subset $S$, $S \subset V'$, $S \neq \emptyset$, for which $0 < p(S) < 2$, where $p(S) = 1/Q \sum_{i \in S} \sum_{j \in \bar{S}} (x_{ij} + x_{ji}) - 2$, and creating two subproblems: one adding constraint

TABLE 2. Class B: lower bounds.

| $n$ | $\beta$ | $\%z_{TSP}$ | $\%LB_0$ | $\%LB_1$ | $\%LB_2$ | $t_{LB_2}$ | *Triv* | *Flow* | *Sub* | *Cap* |
|---|---|---|---|---|---|---|---|---|---|---|
| 25 | 0.05 | 100.0 | 81.2 | 100.0 | 100.0 | 0.6 | 11 | 15 | 6 | 0 |
| 25 | 0.05 | 99.6 | 74.5 | 98.9 | 99.1 | 0.5 | 9 | 15 | 8 | 6 |
| 25 | 0.10 | 97.3 | 75.2 | 95.4 | 98.1 | 0.6 | 13 | 21 | 8 | 87 |
| 25 | 0.10 | 99.6 | 76.8 | 98.9 | 99.8 | 0.5 | 14 | 22 | 8 | 12 |
| 25 | 0.20 | 97.3 | 84.1 | 97.3 | 100.0 | 0.5 | 12 | 16 | 8 | 15 |
| 25 | 0.20 | 99.6 | 76.8 | 98.9 | 99.8 | 0.4 | 15 | 17 | 8 | 6 |
| 25 | ∞ | 99.6 | 85.7 | 97.3 | 99.0 | 0.5 | 14 | 22 | 3 | 44 |
| 25 | ∞ | 100.0 | 75.4 | 99.3 | 99.3 | 0.4 | 13 | 21 | 8 | 0 |
| 50 | 0.05 | 99.5 | 77.7 | 99.3 | 99.8 | 0.9 | 27 | 25 | 9 | 67 |
| 50 | 0.05 | 100.0 | 82.0 | 100.0 | 100.0 | 0.8 | 31 | 24 | 10 | 32 |
| 50 | 0.10 | 98.7 | 76.2 | 98.5 | 99.5 | 0.9 | 29 | 38 | 9 | 75 |
| 50 | 0.10 | 100.0 | 82.0 | 100.0 | 100.0 | 0.8 | 29 | 29 | 12 | 85 |
| 50 | 0.20 | 99.3 | 76.7 | 99.2 | 100.0 | 0.8 | 21 | 35 | 9 | 54 |
| 50 | 0.20 | 100.0 | 82.8 | 100.0 | 100.0 | 0.8 | 27 | 33 | 8 | 60 |
| 50 | ∞ | 97.9 | 81.4 | 99.3 | 99.3 | 0.9 | 23 | 48 | 11 | 81 |
| 50 | ∞ | 97.9 | 81.4 | 98.2 | 99.3 | 0.9 | 23 | 48 | 11 | 81 |
| 75 | 0.05 | 100.0 | 75.9 | 99.4 | 99.4 | 2.0 | 46 | 53 | 18 | 104 |
| 75 | 0.05 | 100.0 | 84.5 | 99.5 | 99.7 | 1.3 | 42 | 26 | 12 | 42 |
| 75 | 0.10 | 100.0 | 75.9 | 99.4 | 99.6 | 2.2 | 48 | 65 | 17 | 92 |
| 75 | 0.10 | 99.8 | 84.4 | 99.4 | 99.8 | 2.9 | 46 | 38 | 14 | 348 |
| 75 | 0.20 | 100.0 | 75.9 | 99.4 | 99.6 | 3.4 | 51 | 70 | 14 | 385 |
| 75 | 0.20 | 99.2 | 83.9 | 98.8 | 99.2 | 2.0 | 45 | 53 | 11 | 164 |
| 75 | ∞ | 98.2 | 76.2 | 98.5 | 98.8 | 2.2 | 54 | 89 | 17 | 86 |
| 75 | ∞ | 98.4 | 79.5 | 97.9 | 98.3 | 3.6 | 40 | 76 | 11 | 359 |
| 100 | 0.05 | 100.0 | 78.5 | 99.1 | 99.1 | 3.4 | 64 | 48 | 17 | 45 |
| 100 | 0.05 | 99.0 | 84.9 | 98.1 | 99.0 | 3.3 | 68 | 57 | 20 | 41 |
| 100 | 0.10 | 99.7 | 78.3 | 98.9 | 98.9 | 5.3 | 65 | 70 | 15 | 268 |
| 100 | 0.10 | 97.8 | 83.9 | 96.1 | 97.9 | 3.9 | 69 | 76 | 16 | 103 |
| 100 | 0.20 | 99.7 | 78.3 | 98.9 | 98.9 | 6.2 | 69 | 96 | 16 | 273 |
| 100 | 0.20 | 97.3 | 83.2 | 94.2 | 98.4 | 3.6 | 61 | 87 | 10 | 185 |
| 100 | ∞ | 98.6 | 77.7 | 97.8 | 97.8 | 4.3 | 70 | 87 | 12 | 276 |
| 100 | ∞ | 98.0 | 84.3 | 98.1 | 98.4 | 4.1 | 50 | 104 | 16 | 71 |
| 150 | 0.05 | 99.8 | 79.4 | 98.5 | 98.5 | 12.2 | 96 | 55 | 16 | 222 |
| 150 | 0.05 | 99.9 | 76.3 | 98.7 | 99.2 | 18.3 | 89 | 113 | 32 | 245 |
| 150 | 0.10 | 99.1 | 78.9 | 97.7 | 97.8 | 13.7 | 102 | 65 | 13 | 404 |
| 150 | 0.10 | 99.4 | 75.9 | 97.8 | 99.3 | 23.7 | 92 | 142 | 28 | 466 |
| 150 | 0.20 | 98.5 | 79.9 | 97.6 | 98.1 | 23.1 | 70 | 122 | 21 | 980 |
| 150 | 0.20 | 98.0 | 74.9 | 97.4 | 97.9 | 54.6 | 107 | 161 | 25 | 1401 |
| 150 | ∞ | 99.6 | 79.8 | 98.2 | 98.8 | 14.0 | 76 | 140 | 16 | 361 |
| 150 | ∞ | 100.0 | 77.5 | 99.3 | 99.3 | 24.3 | 97 | 189 | 31 | 472 |
| 200 | 0.05 | 99.6 | 81.5 | 98.8 | 98.8 | 27.8 | 118 | 150 | 47 | 106 |
| 200 | 0.05 | 99.5 | 80.9 | 99.1 | 99.2 | 39.6 | 141 | 139 | 53 | 218 |
| 200 | 0.10 | 99.7 | 81.6 | 98.9 | 99.0 | 44.6 | 124 | 164 | 46 | 384 |
| 200 | 0.10 | 99.1 | 80.5 | 98.6 | 98.7 | 41.8 | 145 | 173 | 38 | 400 |
| 200 | 0.20 | 99.5 | 81.4 | 98.7 | 98.9 | 45.4 | 127 | 193 | 37 | 612 |
| 200 | 0.20 | 99.2 | 80.6 | 98.7 | 98.8 | 58.2 | 157 | 189 | 34 | 1059 |
| 200 | ∞ | 98.9 | 80.3 | 98.0 | 98.1 | 36.2 | 143 | 191 | 24 | 467 |
| 200 | ∞ | 99.5 | 81.4 | 99.1 | 99.1 | 37.6 | 158 | 200 | 29 | 403 |
| Avg. | | 99.2 | 79.7 | 98.5 | 99.1 | | | | | |

$1/Q\ \sum_{i\in S} \sum_{j\in \bar{S}} (x_{ij} + x_{ji}) = 2$ and the other adding constraint $1/Q\ \sum_{i\in S} \sum_{j\in \bar{S}} (x_{ij} + x_{ji}) \geq 4$. The selection of subset $S$ is carried out in two steps: First, a candidate list of subsets is built heuristically and, second, one of them is selected from this list according to some criterion.

The list of candidate subsets is built using the greedy randomized algorithm described in Section 4.3. In this case, a set $S$ is added to the list of candidate subsets if $0 < p(S) < 2$.

From the candidate list, we select four subsets according to the following criteria:

- Select set $S_1$ with maximum demand;
- Select set $S_2$ which is farthest from the depot;
- Select set $S_3$ such that $1/Q\ \sum_{i\in S_3} \sum_{j\in V\setminus S_3} (x_{ij} + x_{ji})$ is as close as possible to 2.75;
- Select set $S_4$ such that $1/Q\ \sum_{i\in S_4} \sum_{j\in V\setminus S_4} (x_{ij} + x_{ji})$ is as close as possible to 3.

Then, one of the four subsets ($S_1$, $S_2$, $S_3$, $S_4$) is chosen for branching using the method described by Applegate et al. [2] for the TSP. For each subset $S \in \{S_1, S_2, S_3, S_4\}$, we solve the two corresponding subproblems and compute the

TABLE 3. Class C: lower bounds.

| $n$ | $\beta$ | $\%z_{TSP}$ | $\%LB_0$ | $\%LB_1$ | $\%LB_2$ | $t_{LB_2}$ | $Triv$ | $Flow$ | $Sub$ | $Cap$ |
|---|---|---|---|---|---|---|---|---|---|---|
| 25 | 0.05 | 100.0 | 90.9 | 100.0 | 100.0 | 0.4 | 13 | 10 | 2 | 0 |
| 25 | 0.05 | 100.0 | 77.7 | 100.0 | 100.0 | 0.4 | 18 | 11 | 1 | 0 |
| 25 | 0.10 | 100.0 | 90.9 | 100.0 | 100.0 | 0.4 | 13 | 15 | 2 | 0 |
| 25 | 0.10 | 100.0 | 77.7 | 100.0 | 100.0 | 0.4 | 18 | 9 | 1 | 0 |
| 25 | 0.20 | 100.0 | 90.9 | 100.0 | 100.0 | 0.4 | 14 | 16 | 2 | 0 |
| 25 | 0.20 | 97.1 | 75.8 | 97.1 | 97.9 | 0.5 | 17 | 27 | 2 | 78 |
| 25 | $\infty$ | 94.1 | 93.2 | 99.5 | 100.0 | 0.4 | 12 | 23 | 7 | 2 |
| 25 | $\infty$ | 100.0 | 79.8 | 100.0 | 100.0 | 0.4 | 10 | 15 | 0 | 0 |
| 50 | 0.05 | 100.0 | 77.1 | 100.0 | 100.0 | 0.7 | 25 | 38 | 10 | 0 |
| 50 | 0.05 | 100.0 | 80.0 | 99.3 | 99.3 | 1.3 | 28 | 64 | 16 | 47 |
| 50 | 0.10 | 100.0 | 77.1 | 100.0 | 100.0 | 0.7 | 29 | 30 | 7 | 0 |
| 50 | 0.10 | 100.0 | 80.7 | 99.3 | 99.3 | 1.4 | 35 | 71 | 16 | 64 |
| 50 | 0.20 | 100.0 | 77.6 | 100.0 | 100.0 | 0.7 | 29 | 41 | 8 | 0 |
| 50 | 0.20 | 100.0 | 80.7 | 99.3 | 99.3 | 1.3 | 30 | 72 | 16 | 48 |
| 50 | $\infty$ | 100.0 | 76.6 | 100.0 | 100.0 | 0.7 | 27 | 47 | 8 | 0 |
| 50 | $\infty$ | 100.0 | 80.7 | 99.3 | 99.3 | 1.3 | 33 | 77 | 16 | 73 |
| 75 | 0.05 | 100.0 | 75.0 | 100.0 | 100.0 | 1.9 | 40 | 80 | 13 | 87 |
| 75 | 0.05 | 100.0 | 89.3 | 100.0 | 100.0 | 1.4 | 37 | 46 | 8 | 65 |
| 75 | 0.10 | 100.0 | 75.0 | 100.0 | 100.0 | 2.1 | 48 | 108 | 9 | 67 |
| 75 | 0.10 | 100.0 | 89.3 | 100.0 | 100.0 | 1.4 | 38 | 61 | 8 | 54 |
| 75 | 0.20 | 99.2 | 77.7 | 100.0 | 100.0 | 2.3 | 57 | 107 | 8 | 177 |
| 75 | 0.20 | 100.0 | 89.3 | 100.0 | 100.0 | 1.8 | 52 | 79 | 8 | 109 |
| 75 | $\infty$ | 99.2 | 78.5 | 100.0 | 100.0 | 1.7 | 32 | 92 | 8 | 112 |
| 75 | $\infty$ | 100.0 | 90.0 | 100.0 | 100.0 | 1.8 | 52 | 120 | 10 | 26 |
| 100 | 0.05 | 100.0 | 83.0 | 100.0 | 100.0 | 2.2 | 55 | 87 | 3 | 0 |
| 100 | 0.05 | 100.0 | 78.6 | 100.0 | 100.0 | 3.9 | 55 | 110 | 15 | 12 |
| 100 | 0.10 | 100.0 | 83.0 | 100.0 | 100.0 | 2.6 | 58 | 94 | 6 | 0 |
| 100 | 0.10 | 100.0 | 78.6 | 100.0 | 100.0 | 3.7 | 60 | 113 | 11 | 33 |
| 100 | 0.20 | 100.0 | 83.0 | 100.0 | 100.0 | 2.3 | 41 | 98 | 5 | 0 |
| 100 | 0.20 | 100.0 | 79.2 | 100.0 | 100.0 | 4.7 | 72 | 160 | 14 | 47 |
| 100 | $\infty$ | 100.0 | 84.0 | 100.0 | 100.0 | 2.4 | 41 | 139 | 5 | 0 |
| 100 | $\infty$ | 100.0 | 80.3 | 100.0 | 100.0 | 6.2 | 79 | 204 | 12 | 197 |
| 150 | 0.05 | 100.0 | 85.7 | 100.0 | 100.0 | 9.0 | 50 | 126 | 8 | 0 |
| 150 | 0.05 | 100.0 | 69.5 | 100.0 | 100.0 | 12.6 | 59 | 150 | 10 | 53 |
| 150 | 0.10 | 100.0 | 85.7 | 100.0 | 100.0 | 10.2 | 64 | 157 | 6 | 0 |
| 150 | 0.10 | 100.0 | 69.5 | 100.0 | 100.0 | 17.0 | 71 | 207 | 13 | 201 |
| 150 | 0.20 | 100.0 | 85.7 | 100.0 | 100.0 | 12.7 | 59 | 180 | 8 | 0 |
| 150 | 0.20 | 100.0 | 69.5 | 100.0 | 100.0 | 34.7 | 99 | 306 | 9 | 514 |
| 150 | $\infty$ | 100.0 | 85.7 | 100.0 | 100.0 | 16.0 | 52 | 274 | 5 | 0 |
| 150 | $\infty$ | 100.0 | 69.5 | 100.0 | 100.0 | 65.1 | 154 | 675 | 21 | 223 |
| 200 | 0.05 | 100.0 | 84.0 | 100.0 | 100.0 | 30.9 | 58 | 211 | 8 | 0 |
| 200 | 0.05 | 100.0 | 88.5 | 100.0 | 100.0 | 60.6 | 61 | 322 | 9 | 36 |
| 200 | 0.10 | 100.0 | 84.0 | 100.0 | 100.0 | 23.5 | 61 | 255 | 2 | 0 |
| 200 | 0.10 | 100.0 | 88.5 | 100.0 | 100.0 | 41.1 | 46 | 335 | 9 | 23 |
| 200 | 0.20 | 100.0 | 84.0 | 100.0 | 100.0 | 33.8 | 50 | 284 | 6 | 0 |
| 200 | 0.20 | 100.0 | 88.5 | 100.0 | 100.0 | 33.5 | 32 | 276 | 8 | 0 |
| 200 | $\infty$ | 98.8 | 84.1 | 100.0 | 100.0 | 37.2 | 36 | 299 | 6 | 0 |
| 200 | $\infty$ | 100.0 | 88.5 | 100.0 | 100.0 | 39.5 | 33 | 353 | 8 | 0 |
| Avg. | | 99.8 | 81.9 | 99.9 | 99.9 | | | | | |

minimum of the increases in the lower bounds with respect to the lower bound of the current node. Then, we choose the subset which leads to the maximum of these minimum increases.

When a suitable set $S$ cannot be found using the greedy randomized algorithm, the branching on variables strategy is adopted. This involves the selection of an edge $\{i, j\}$ having a fractional value of $(x_{ij} + x_{ji})/Q$, and the generation of two subproblems: one by fixing $x_{ij} + x_{ji} = Q$ and the other by fixing $x_{ij} + x_{ji} = 0$. The edge $\{i, j\}$ is selected in such a way that is as close as possible to 0.5 and ties are broken by choosing the edge $\{i, j\}$ having maximum cost $c_{ij}$.

## 6. COMPUTATIONAL RESULTS

This section presents computational results for the branch-and-cut algorithm described in Section 5. The algorithm was coded in Fortran 77 and run on an IBM PC equipped with a Pentium III 900 MHz processor. CPLEX 6.5 [15] was used as the LP solver.

TABLE 4. Class A: branch-and-cut algorithm.

| $n$ | $\beta$ | $z_{UB}$ | $z^*$ | $\%LB_{BC}$ | *Nodes* | $t_{SEP}$ | $t_{LP}$ | $t_{TOT}$ |
|---|---|---|---|---|---|---|---|---|
| 15 | 0.05 | 218 | 218 | 100.0 | 0 | 0.0 | 0.5 | 1.3 |
| 15 | 0.10 | 221 | 221 | 100.0 | 2 | 0.0 | 0.0 | 0.4 |
| 15 | 0.20 | 221 | 221 | 100.0 | 2 | 0.0 | 0.0 | 0.4 |
| 25 | 0.05 | 306 | 306 | 100.0 | 2 | 0.0 | 0.1 | 0.5 |
| 25 | 0.10 | 308 | 308 | 100.0 | 2 | 0.1 | 0.0 | 0.4 |
| 25 | 0.20 | 313 | 313 | 100.0 | 12 | 0.2 | 0.4 | 1.0 |
| 35 | 0.05 | 351 | 351 | 100.0 | 6 | 0.1 | 0.2 | 0.6 |
| 35 | 0.10 | 351 | 351 | 100.0 | 6 | 0.1 | 0.1 | 0.6 |
| 35 | 0.20 | 351 | 351 | 100.0 | 6 | 0.0 | 0.2 | 0.6 |
| 44 | 0.05 | 619 | 619 | 100.0 | 0 | 0.1 | 0.3 | 0.8 |
| 44 | 0.10 | 619 | 619 | 100.0 | 0 | 0.2 | 0.1 | 0.7 |
| 44 | 0.20 | 619 | 619 | 100.0 | 0 | 0.1 | 0.5 | 1.0 |
| 50 | 0.05 | 426 | 426 | 100.0 | 48 | 0.9 | 2.0 | 3.7 |
| 50 | 0.10 | 426 | 426 | 100.0 | 28 | 0.8 | 1.5 | 2.7 |
| 50 | 0.20 | 426 | 426 | 100.0 | 30 | 1.0 | 1.7 | 3.1 |
| 75 | 0.05 | 538 | 538 | 100.0 | 0 | 0.1 | 0.2 | 0.8 |
| 75 | 0.10 | 539 | 539 | 100.0 | 2 | 0.3 | 1.5 | 2.3 |
| 75 | 0.20 | 539 | 539 | 100.0 | 2 | 0.2 | 2.4 | 3.2 |
| 100 | 0.05 | 501 | 501 | 100.0 | 58 | 10.0 | 22.9 | 34.5 |
| 100 | 0.10 | 501 | 501 | 100.0 | 60 | 10.6 | 27.4 | 39.5 |
| 100 | 0.20 | 502 | 502 | 100.0 | 24 | 6.9 | 19.8 | 27.6 |
| 120 | 0.05 | 535 | 535 | 100.0 | 202 | 36.5 | 65.1 | 107.8 |
| 120 | 0.10 | 535 | 535 | 100.0 | 174 | 36.2 | 84.3 | 126.6 |
| 120 | 0.20 | 535 | 535 | 100.0 | 108 | 27.4 | 74.9 | 106.3 |
| 150 | 0.05 | 698 | 698 | 100.0 | 10 | 5.5 | 5.7 | 12.3 |
| 150 | 0.10 | 699 | 699 | 100.0 | 22 | 11.8 | 12.3 | 25.8 |
| 150 | 0.20 | 703 | 702 | 100.0 | 414 | 173.4 | 1226.2 | 1423.9 |
| 199 | 0.05 | 761 | 761 | 100.0 | 30 | 58.3 | 35.2 | 97.4 |
| 199 | 0.10 | 763 | 761 | 100.0 | 178 | 195.0 | 381.4 | 596.1 |
| 199 | 0.20 | 762 | 762 | 100.0 | 62 | 109.2 | 176.2 | 293.3 |
| 261 | 0.05 | 2382 | 2382 | 99.2 | 194 | 393.8 | 3163.0 | * |
| 261 | 0.10 | 2389 | 2389 | 99.0 | 78 | 255.3 | 3326.0 | * |
| 261 | 0.20 | 2393 | 2393 | 98.8 | 68 | 235.3 | 3437.1 | * |

We considered three classes of test problems, called A, B, and C, which were proposed by Gendreau et al. [11].

Class A includes instances derived from symmetric VRPs found in the literature. For each VRP instance, a TSPDC instance was obtained where the customer set, the depot, and the cost matrix are the same as in the VRP one. For each customer $i$, two quantities were considered: a delivery demand $d_i$ and a pickup demand $p_i$. Let $r_i$ be the demand of customer $i$ in the VRP instance. The corresponding demands for the TSPDC were defined as follows:

$$d_i = r_i,\ p_i = \begin{cases} \lfloor(1-\beta)r_i\rfloor & \text{if } i \text{ is even} \\ \lfloor(1+\beta)r_i\rfloor & \text{if } i \text{ is odd} \end{cases} \quad i = 1, \ldots, n, \tag{41}$$

where $\beta$ is a real nonnegative parameter smaller than 1; the demand of customer $i$ was then defined as $q_i = d_i - p_i$. Finally, the vehicle capacity was defined as $Q = \max\{Q_D, Q_C\}$. For this class of problems, 11 symmetric VRP instances proposed in the literature were considered with $n$ ranging between 25 and 261; for each of them, three TSPDC instances were generated, corresponding to $\beta = 0.05$, 0.10, and 0.20.

Class B consists of Euclidean problems where for the depot and for each customer the coordinates are uniformly generated in the interval [0, 100] and the cost of each edge of graph $G$ is defined as the Euclidean distance between its endpoints.

Class C consists of symmetric problems where the cost of each edge of graph $G$ is uniformly and randomly generated in the interval [0, 100]. Then, the full cost matrix is derived by applying the Floyd–Warshall algorithm to graph $G$.

For classes B and C, test problems with $n = 25, 50, 75,$ 100, 150, and 200 were generated. The $d_i$ values were uniformly and randomly generated in [1, 100] and $q_i$ was defined according to (41) with the same $\beta$ values as in class A. Instances with uncorrelated values of $d_i$ and $p_i$, both uniformly and randomly generated in [1, 100], were generated as well; these instances are marked in the tables with $\beta = \infty$. For each class B and C and for each pair $n$ and $\beta$, two instances were generated.

All TSPDC instances generated were heuristically solved

TABLE 5. Class B: branch-and-cut algorithm.

| $n$ | $\beta$ | $z_{UB}$ | $z^*$ | $\%LB_{BC}$ | *Nodes* | $t_{SEP}$ | $t_{LP}$ | $t_{TOT}$ |
|---|---|---|---|---|---|---|---|---|
| 25 | 0.05 | 464 | 464 | 100.0 | 0 | 0.0 | 0.1 | 0.6 |
| 25 | 0.05 | 462 | 462 | 100.0 | 2 | 0.1 | 0.0 | 0.6 |
| 25 | 0.10 | 412 | 412 | 100.0 | 12 | 0.2 | 0.8 | 1.5 |
| 25 | 0.10 | 462 | 462 | 100.0 | 2 | 0.1 | 0.1 | 0.5 |
| 25 | 0.20 | 477 | 477 | 100.0 | 0 | 0.1 | 0.0 | 0.5 |
| 25 | 0.20 | 462 | 462 | 100.0 | 2 | 0.0 | 0.1 | 0.5 |
| 25 | $\infty$ | 482 | 482 | 100.0 | 4 | 0.2 | 0.0 | 0.6 |
| 25 | $\infty$ | 460 | 460 | 100.0 | 8 | 0.0 | 0.2 | 0.6 |
| 50 | 0.05 | 592 | 592 | 100.0 | 2 | 0.2 | 0.4 | 1.0 |
| 50 | 0.05 | 599 | 599 | 100.0 | 0 | 0.1 | 0.3 | 0.8 |
| 50 | 0.10 | 597 | 597 | 100.0 | 2 | 0.2 | 0.5 | 1.0 |
| 50 | 0.10 | 599 | 599 | 100.0 | 0 | 0.1 | 0.4 | 0.8 |
| 50 | 0.20 | 593 | 593 | 100.0 | 0 | 0.1 | 0.4 | 0.8 |
| 50 | 0.20 | 599 | 599 | 100.0 | 0 | 0.0 | 0.4 | 0.8 |
| 50 | $\infty$ | 618 | 612 | 100.0 | 18 | 0.8 | 1.4 | 2.6 |
| 50 | $\infty$ | 612 | 612 | 100.0 | 4 | 0.4 | 0.5 | 1.3 |
| 75 | 0.05 | 710 | 710 | 100.0 | 4 | 1.0 | 1.3 | 2.7 |
| 75 | 0.05 | 658 | 658 | 100.0 | 4 | 0.6 | 0.9 | 2.0 |
| 75 | 0.10 | 710 | 710 | 100.0 | 6 | 1.1 | 1.9 | 3.6 |
| 75 | 0.10 | 659 | 659 | 100.0 | 2 | 0.9 | 2.0 | 3.4 |
| 75 | 0.20 | 710 | 710 | 100.0 | 6 | 1.6 | 5.8 | 7.8 |
| 75 | 0.20 | 663 | 663 | 100.0 | 6 | 1.2 | 1.8 | 3.6 |
| 75 | $\infty$ | 723 | 723 | 100.0 | 120 | 9.8 | 87.8 | 99.8 |
| 75 | $\infty$ | 709 | 707 | 100.0 | 122 | 10.1 | 256.5 | 269.8 |
| 100 | 0.05 | 797 | 797 | 100.0 | 48 | 8.6 | 17.1 | 27.1 |
| 100 | 0.05 | 797 | 797 | 100.0 | 90 | 12.4 | 47.4 | 62.0 |
| 100 | 0.10 | 799 | 799 | 100.0 | 68 | 11.4 | 93.8 | 107.3 |
| 100 | 0.10 | 814 | 807 | 100.0 | 708 | 98.4 | 3183.1 | 3345.2 |
| 100 | 0.20 | 799 | 799 | 100.0 | 64 | 11.1 | 111.3 | 123.9 |
| 100 | 0.20 | 830 | 811 | 100.0 | 718 | 84.0 | 3323.9 | 3499.8 |
| 100 | $\infty$ | 808 | 808 | 100.0 | 508 | 56.6 | 3420.2 | 3500.9 |
| 100 | $\infty$ | 805 | 805 | 100.0 | 178 | 23.7 | 171.6 | 199.8 |
| 150 | 0.05 | 968 | 968 | 99.5 | 546 | 215.8 | 3357.4 | * |
| 150 | 0.05 | 907 | 902 | 100.0 | 576 | 222.9 | 2901.7 | 3164.4 |
| 150 | 0.10 | 975 | 975 | 99.0 | 248 | 140.1 | 3470.5 | * |
| 150 | 0.10 | 915 | 906 | 100.0 | 146 | 110.5 | 1411.7 | 1535.4 |
| 150 | 0.20 | 981 | 981 | 99.1 | 198 | 125.7 | 3486.6 | * |
| 150 | 0.20 | 919 | 919 | 98.7 | 144 | 105.3 | 3489.6 | * |
| 150 | $\infty$ | 970 | 970 | 99.7 | 328 | 153.1 | 3447.8 | * |
| 150 | $\infty$ | 901 | 901 | 100.0 | 152 | 78.2 | 870.3 | 959.1 |
| 200 | 0.05 | 1053 | 1053 | 99.4 | 622 | 529.5 | 3002.6 | * |
| 200 | 0.05 | 1105 | 1105 | 99.7 | 464 | 404.3 | 3148.4 | * |
| 200 | 0.10 | 1052 | 1052 | 99.9 | 340 | 349.2 | 3210.3 | * |
| 200 | 0.10 | 1110 | 1110 | 99.5 | 290 | 311.5 | 3256.5 | * |
| 200 | 0.20 | 1054 | 1054 | 99.6 | 250 | 282.5 | 3309.4 | * |
| 200 | 0.20 | 1109 | 1109 | 99.5 | 174 | 233.9 | 3379.7 | * |
| 200 | $\infty$ | 1061 | 1061 | 98.6 | 200 | 225.0 | 3377.7 | * |
| 200 | $\infty$ | 1106 | 1106 | 99.5 | 320 | 324.5 | 3235.5 | * |

using the Genetic algorithm of Baldacci and Mingozzi [6] and run on the same machine used to obtain the results of the branch-and-cut algorithm. Computational experience has shown that the Genetic algorithm produced results that were competitive with those obtained by the Tabu Search algorithm of Gendreau et al. [11]. The resulting upper bounds were used at the root node of the branch-and-cut algorithm for variable reduction as described in Section 4.2.

Lower bounds on the optimal TSPDC solutions were obtained by solving to optimality the corresponding TSP instances using the Concorde code of Applegate et al. [4].

The following notation is used in this section:

| | |
|---|---|
| $z_{UB}$ | cost of the solution obtained by the heuristic algorithm described in Baldacci and Mingozzi [6] |
| $z^*$ | cost of the optimal TSPDC solution or cost of the best solution found by the branch-and-cut algorithm or $z_{UB}$ |
| $z_{TSP}$ | cost of the optimal TSP solution |
| $LB_0$ | lower bound corresponding to the optimal solution cost of formulation LF1 |

TABLE 6. Class C: branch-and-cut algorithm.

| $n$ | $\beta$ | $z_{UB}$ | $z^*$ | $\%LB_{BC}$ | *Nodes* | $t_{SEP}$ | $t_{LP}$ | $t_{TOT}$ |
|---|---|---|---|---|---|---|---|---|
| 25 | 0.05 | 208 | 208 | 100.0 | 0 | 0.0 | 0.0 | 0.4 |
| 25 | 0.05 | 233 | 233 | 100.0 | 0 | 0.0 | 0.1 | 0.4 |
| 25 | 0.10 | 208 | 208 | 100.0 | 0 | 0.0 | 0.0 | 0.4 |
| 25 | 0.10 | 233 | 233 | 100.0 | 0 | 0.0 | 0.0 | 0.4 |
| 25 | 0.20 | 208 | 208 | 100.0 | 0 | 0.0 | 0.0 | 0.4 |
| 25 | 0.20 | 240 | 240 | 100.0 | 30 | 0.1 | 1.9 | 2.6 |
| 25 | $\infty$ | 221 | 221 | 100.0 | 0 | 0.0 | 0.1 | 0.4 |
| 25 | $\infty$ | 233 | 233 | 100.0 | 0 | 0.0 | 0.1 | 0.4 |
| 50 | 0.05 | 192 | 192 | 100.0 | 0 | 0.1 | 0.3 | 0.7 |
| 50 | 0.05 | 135 | 135 | 100.0 | 46 | 2.7 | 36.1 | 40.6 |
| 50 | 0.10 | 192 | 192 | 100.0 | 0 | 0.1 | 0.2 | 0.7 |
| 50 | 0.10 | 135 | 135 | 100.0 | 8 | 0.5 | 4.1 | 5.1 |
| 50 | 0.20 | 192 | 192 | 100.0 | 0 | 0.1 | 0.1 | 0.7 |
| 50 | 0.20 | 135 | 135 | 100.0 | 18 | 1.9 | 9.9 | 12.4 |
| 50 | $\infty$ | 192 | 192 | 100.0 | 0 | 0.1 | 0.2 | 0.7 |
| 50 | $\infty$ | 135 | 135 | 100.0 | 34 | 1.9 | 21.4 | 24.4 |
| 75 | 0.05 | 120 | 120 | 100.0 | 0 | 0.5 | 0.9 | 1.9 |
| 75 | 0.05 | 150 | 150 | 100.0 | 0 | 0.3 | 0.7 | 1.4 |
| 75 | 0.10 | 120 | 120 | 100.0 | 0 | 0.7 | 1.0 | 2.1 |
| 75 | 0.10 | 150 | 150 | 100.0 | 0 | 0.2 | 0.8 | 1.4 |
| 75 | 0.20 | 121 | 121 | 100.0 | 0 | 0.5 | 1.4 | 2.3 |
| 75 | 0.20 | 150 | 150 | 100.0 | 0 | 0.5 | 0.9 | 1.8 |
| 75 | $\infty$ | 121 | 121 | 100.0 | 0 | 0.4 | 0.8 | 1.7 |
| 75 | $\infty$ | 150 | 150 | 100.0 | 0 | 0.3 | 1.0 | 1.8 |
| 100 | 0.05 | 94 | 94 | 100.0 | 0 | 0.1 | 1.6 | 2.2 |
| 100 | 0.05 | 173 | 173 | 100.0 | 0 | 0.8 | 2.5 | 3.9 |
| 100 | 0.10 | 94 | 94 | 100.0 | 0 | 0.0 | 2.1 | 2.6 |
| 100 | 0.10 | 173 | 173 | 100.0 | 0 | 0.9 | 2.3 | 3.7 |
| 100 | 0.20 | 94 | 94 | 100.0 | 0 | 0.3 | 1.6 | 2.3 |
| 100 | 0.20 | 173 | 173 | 100.0 | 0 | 1.3 | 3.0 | 4.7 |
| 100 | $\infty$ | 94 | 94 | 100.0 | 0 | 0.2 | 1.8 | 2.4 |
| 100 | $\infty$ | 173 | 173 | 100.0 | 0 | 1.8 | 3.9 | 6.2 |
| 150 | 0.05 | 91 | 91 | 100.0 | 0 | 1.0 | 7.4 | 9.0 |
| 150 | 0.05 | 82 | 82 | 100.0 | 0 | 4.2 | 7.7 | 12.6 |
| 150 | 0.10 | 91 | 91 | 100.0 | 0 | 1.1 | 8.5 | 10.2 |
| 150 | 0.10 | 82 | 82 | 100.0 | 0 | 5.6 | 10.9 | 17.0 |
| 150 | 0.20 | 91 | 91 | 100.0 | 0 | 2.4 | 9.7 | 12.7 |
| 150 | 0.20 | 82 | 82 | 100.0 | 0 | 8.5 | 26.1 | 34.7 |
| 150 | $\infty$ | 91 | 91 | 100.0 | 0 | 2.8 | 12.6 | 16.0 |
| 150 | $\infty$ | 82 | 82 | 100.0 | 0 | 9.1 | 55.3 | 65.1 |
| 200 | 0.05 | 81 | 81 | 100.0 | 0 | 3.9 | 26.3 | 30.9 |
| 200 | 0.05 | 61 | 61 | 100.0 | 0 | 12.9 | 47.0 | 60.6 |
| 200 | 0.10 | 81 | 81 | 100.0 | 0 | 1.2 | 21.7 | 23.5 |
| 200 | 0.10 | 61 | 61 | 100.0 | 0 | 14.8 | 25.6 | 41.1 |
| 200 | 0.20 | 81 | 81 | 100.0 | 0 | 9.0 | 24.1 | 33.8 |
| 200 | 0.20 | 61 | 61 | 100.0 | 0 | 6.5 | 26.3 | 33.5 |
| 200 | $\infty$ | 82 | 82 | 100.0 | 0 | 17.0 | 19.7 | 37.2 |
| 200 | $\infty$ | 61 | 61 | 100.0 | 0 | 6.3 | 32.6 | 39.5 |

| | |
|---|---|
| $LB_1$ | lower bound obtained at the root node after adding trivial inequalities, flow inequalities, and the subtour elimination inequalities |
| $LB_2$ | lower bound obtained at the root node after adding trivial inequalities, flow inequalities, subtour elimination inequalities, and capacity inequalities |
| $LB_{BC}$ | lower bound of the branch-and-cut tree node with minimum lower bound at the end of the branch-and-cut algorithm |

The following notation refers to the computation of the lower bounds at the root node of the branch-and-cut algorithm (Tables 1–3):

TABLE 7. Average results for the genetic algorithm on classes A, B, and C.

| | Avg. $\%z_{UB}$ | Max $\%z_{UB}$ | Avg. $t_{UB}$ | Max $t_{UB}$ |
|---|---|---|---|---|
| Class A | 100.1 | 101.4 | 21.9 | 155.6 |
| Class B | 100.6 | 104.0 | 16.6 | 65.0 |
| Class C | 100.0 | 100.0 | 9.4 | 30.8 |

| | |
|---|---|
| $\%z_{TSP}$ | percentage ratio $z_{TSP}/z^*$ |
| $\%LB_0$ | percentage ratio $LB_0/z^*$ |
| $\%LB_1$ | percentage ratio $LB_1/z^*$ |
| $\%LB_2$ | percentage ratio $LB_2/z^*$ |
| $t_{LB_2}$ | computing time in seconds of lower bound $LB_2$ |
| $Triv$ | total number of trivial inequalities at the root node |
| $Flow$ | total number of flow inequalities at the root node |
| $Sub$ | total number of subtour elimination inequalities at the root node |
| $Cap$ | total number of capacity inequalities at the root node |

The following notation refers to the results of the branch-and-cut algorithm (Tables 4–6):

| | |
|---|---|
| $\%LB_{BC}$ | percentage ratio $LB_{BC}/z^*$ |
| $Nodes$ | number of nodes generated |
| $t_{SEP}$ | total time in seconds spent by the separation procedures |
| $t_{LP}$ | total time in seconds spent by the LP solver |
| $t_{TOT}$ | total computing time in seconds. We impose a time limit of 3600 seconds. If the time limit is reached, the instance is marked with an asterisk |

Tables 1–3 present details of the lower-bound computation at the root node. On all problem classes, Tables 1–3 show that lower bound $LB_2$ is tight, as shown by the average percentage errors 99.5, 99.1, and 99.9 for classes A, B, and C, respectively. Furthermore, the TSP lower bound is, on average, a tight lower bound, as it is, on average, better than $LB_2$ on classes A and B. However, lower bound $LB_2$ is slightly better than is the TSP bound on class C. The effectiveness of the capacity inequalities described in Section 4.1 can be seen by comparing columns $\%LB_1$ and $\%LB_2$. These inequalities are useful for problems of classes A and B, but they improve $LB_1$ only for two problems in class C.

Tables 4–6 report the results of the branch-and-cut algorithm. Table 4 shows that the branch-and-cut algorithm was able to find the optimal solution for all instances of class A involving up to 199 customers. The branch-and-cut algorithm was not completed within the imposed time limit for the last three test problems with 261 customers. Table 5 shows that the instances of class B are more difficult for the branch-and-cut algorithm than the class A instances of the same size. The branch-and-cut algorithm solved to optimality the first 35 instances involving up to 100 customers.

The results displayed in Table 6 suggest that problem instances in class C are easier than are those in classes A and B. Note that all class C instances, involving up to 200 customers, are solved to optimality by the branch-and-cut algorithm. These results are similar to those reported in the literature for the classical TSP, where randomly generated instances are generally easier than are Euclidean instances (see [16]).

Finally, Table 7 summarizes the performance of the Genetic algorithm described in Baldacci and Mingozzi [6] for classes A, B, and C. For each problem class, Table 7 reports the average (Avg. $\%z_{UB}$) and maximum (Max $\%z_{UB}$) upper-bound percentage error $\%z_{UB}$, which is computed for each instance as follows:

$$\%z_{UB} = \begin{cases} 100\ z_{UB}/z^*, & \text{if the instance has been solved to optimality;} \\ 100\ \mathrm{z_{UB}/LB_2}, & \text{otherwise.} \end{cases}$$

The average (Avg. $t_{UB}$) and maximum (Max $t_{UB}$) computational times in seconds for the Genetic algorithm are also given for each class.

## 7. CONCLUSIONS

In this paper, we considered the TSP with mixed deliveries and collections (TSPDC) in which a vehicle located at a central depot must be optimally used to serve two set of customers: a set of delivery and a set of collection customers. The vehicle capacity should not be exceeded along the tour and the total length of the tour must be minimized.

A new integer programming formulation of the TSPDC based on a two-commodity network flow approach was investigated. New lower bounds were obtained from the linear relaxation of this formulation which were further strengthened by valid inequalities and embedded in a branch-and-cut procedure to solve the problem optimally. The resulting cutting-plane algorithm was applied to three classes of test problems taken from the literature and involving problems with up to 261 customers. The computational results indicate the effectiveness of the proposed method.

## REFERENCES

[1] S. Anily and G. Mosheiov, The traveling salesman problem with delivery and backhauls, Oper Res Lett 16 (1994), 11–18.

[2] D. Applegate, R. Bixby, V. Chvátal, and W. Cook, Special session on tsp, 15th Int Symp on Mathematical Programming, University of Michigan, 1994.

[3] D. Applegate, R. Bixby, V. Chvátal, and W. Cook, Finding cuts in the tsp, Technical report 95-05, DIMACS, 1995.

[4] D. Applegate, R. Bixby, V. Chvátal, and W. Cook, Concorde: A code for solving traveling salesman problems, World Wide Web, http://www.math.princeton.edu/tsp/concorde.html, 2001.

[5] P. Augerat, J.M. Belenguer, E. Benavent, A. Corberan, D. Naddef, and G. Rinaldi, Computational results with a branch and cut code for the capacitated vehicle routing problem, Technical report 1 RR949-M, ARTEMIS-IMAG, Grenoble France, 1995.

[6] R. Baldacci and A. Mingozzi, New heuristic methods for the traveling salesman problem with mixed deliveries and collections, Technical report, Department of Mathematics, University of Bologna, Italy, 1999.

[7] R. Baldacci, A. Mingozzi, and E. Hadjiconstantinou, An exact algorithm for the capacitated vehicle routing problem based on a two-commodity network flow formulation, Technical report 16, Department of Mathematics, University of Bologna, Italy, 1999.

[8] G. Finke, A. Claus, and E. Gunn, A two-commodity network flow approach to the traveling salesman problem, Congr Numer 41 (1984), 167–178.

[9] M.R. Garey and D.S. Johnson, Computers and intractability: A guide to the theory of NP-completeness, W.H. Freeman, San Francisco, 1979.

[10] M. Gendreau, A. Hertz, and G. Laporte, The travelling salesman problem with backhauls, Comput Oper Res 23 (1996), 501–508.

[11] M. Gendreau, G. Laporte, and D. Vigo, Heuristics for the traveling salesman problem with pickup and delivery, Comput Oper Res 26 (1999), 699–714.

[12] B.L. Golden, A.A. Assad, and E.A. Wasil, "Routing vehicles in the real world: Applications in the solid waste, beverage, food, dairy and newspaper industries," The Vehicle Routing Problem, P. Toth and D. Vigo (Editors), SIAM, Philadelphia, 2002, pp. 245–286.

[13] R.E. Gomory and T.C. Hu, Multi-terminal network flows, SIAM J Appl Math 9 (1961), 551–570.

[14] K. Halse, Modelling and solving complex vehicle routing problems, Ph.D. thesis, IMSOR, Technical University of Denmark, 1992.

[15] ILOG, CPLEX 6.5 callable library, 2001.

[16] M. Jünger, G. Reinelt, and G. Rinaldi, The traveling salesman problem, Vol. 7, Network Models, Handbooks in Operations Research and Management Science, M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser (Editors), North-Holland, Amsterdam, 1995, pp. 225–330.

[17] A. Langevin, M. Desrochers, J. Desrochers, S. Glinas, and F. Soumis, A two-commodity flow formulation for the traveling salesman and the makespan problems with time windows, Networks 23 (1993), 631–640.

[18] A. Lucena, Exact solution approaches for the vehicle routing problem, Ph.D. thesis, Management Science Dept., Imperial College, London, 1986.

[19] G. Mosheiov, The travelling salesman problem with pick-up and delivery, Eur J Oper Res 79 (1994), 299–310.

[20] D. Naddef and Y. Pochet, The traveling salesman polytope revisited, Technical Report 98-42, Core, University of Louvain-la-Neuve, 1998.

[21] M.W. Padberg and G. Rinaldi, An efficient algorithm for the minimum capacity cut problem, Math Program 47 (1990), 19–36.