# Branch-and-Cut for the Pickup and Delivery Traveling Salesman Problem with FIFO Loading

Jean-François Cordeau* Mauro Dell'Amico† Manuel Iori†

February 24, 2009

**Abstract**

This paper introduces a branch-and-cut algorithm for a variant of the pickup and delivery traveling salesman problem in which pickups and deliveries must obey the first-in-first-out policy. We propose a new mathematical formulation of the problem and several families of valid inequalities which are used within the branch-and-cut algorithm. Computational experiments on instances from the literature show that this algorithm outperforms existing exact algorithms, and that some instances with up to 25 requests (50 nodes) can be solved in reasonable computing time.

Keywords: Traveling salesman problem, pickup and delivery, FIFO loading, branch-and-cut.

## 1 Introduction

In the classical *Traveling Salesman Problem with Pickup and Delivery* (TSPPD) each customer request is represented by an origin location and a destination location, and the problem is to find a minimum-cost Hamiltonian cycle visiting all locations with the additional constraint that the origin location of each request is visited before the corresponding destination location. The TSPPD has applications in several areas such as urban courier services, less-than-truckload transportation, and dial-a-ride systems (Cordeau et al., 2007).

The *TSPPD with FIFO Loading* (TSPPDF) is a variant of the TSPPD where the pickup and delivery operations must be done in a first-in-first-out fashion: if the

*Canada Research Chair in Logistics and Transportation and CIRRELT, HEC Montréal, 3000 chemin de la Côte-Sainte-Catherine, Montréal, H3T 2A7 Canada.

†DISMI, University of Modena and Reggio Emilia, Via Amendola 2, Padiglione Buccola, 42100, Reggio Emilia, Italy.

pickup of a request $i$ is performed before that of a request $j$, then the delivery of request $i$ must be performed before that of request $j$. The TSPPDF was recently introduced by Erdogan et al. (2009) who mention that the problem arises in dial-a-ride transportation when fairness is a major concern and passengers resent another passenger being pickup up after them but dropped off before them. The problem may also arise in the routing of some automatic guided vehicles that load items on one end and unload them at the other end.

Erdogan et al. (2009) have proposed an integer linear programming formulation of the problem with a polynomial number of variables and constraints. Using the branch-and-bound algorithm of CPLEX, the authors were able to solve instances with 12 requests to optimality within four hours of computing time. For instances with 37 or more requests, however, even the LP relaxation of the problem could not be solved within the four hour time limit. Local search heuristics were thus proposed and results were reported on instances with up to 375 requests. The authors have also shown that the TSPPDF is $\mathcal{NP}$–hard. In another recent paper, Carrabs et al. (2007a) have used an additive branch-and-bound algorithm to solve the TSPPDF. Using lower bounds from the assignment problem and shortest spanning $r$-arborescence problem, they were able to solve most instances with up to 13 requests and some instances with 15, 17 and 19 requests. Their algorithm also strongly benefits from the application of elimination rules that gradually remove from the graph arcs that are incompatible with prior branching decisions.

The TSPPD has received far more attention in the literature. Several heuristic algorithms have been proposed for this problem (see, e.g., Van der Bruggen et al., 1993; Healy and Moll, 1995; Renaud et al., 2000, 2002), while exact algorithms were developed, among others, by Kalantari et al. (1985), Ruland and Rodin (1997) and Dumitrescu et al. (2009). Another closely related problem is the *TSPPD with LIFO Loading* (TSPPDL) in which pickup and delivery operations must obey the last-in-first-out policy (see, e.g., Xu et al., 2003). Algorithms for the TSPPDL have been suggested, among others, by Pacheco (1995, 1997), Cassani (2004), Carrabs et al. (2007a,b), and Cordeau et al. (2009). The best exact algorithm for the TSPPDL is currently that of Cordeau et al. (2009) which is able to solve most instances with up to 17 requests as well as some instances with 21 and 25 requests.

In this article, we introduce a new formulation of the TSPPDF as well as several families of valid inequalities. These inequalities are used within a branch-and-cut algorithm which is capable of solving some instances with up to 25 requests.

The remainder of the article is organized as follows. In the next section, we provide a formal definition of the problem and introduce a mathematical formulation. Section 3 then describes the valid inequalities which are used within the branch-and-cut algorithm introduced in Section 4. Finally, computational experiments are presented in Section 5, and this is followed by the conclusions.

## 2 Problem Definition and Mathematical Formulation

Let $n$ denote the number of customer requests. One can define the TSPPDF on a complete directed graph $G = (N, A)$ where $N = \{0, \ldots, 2n+1\}$ is the node set and $A$ is the arc set. Nodes 0 and $2n+1$ represent the origin and destination depots, while subsets $P = \{1, \ldots, n\}$ and $D = \{n+1, \ldots, 2n\}$ represent pickup and delivery nodes, respectively. Each request $i$ is associated with a pickup node $i$ and a delivery node $n+i$. Each arc $(i, j)$ has a non-negative routing cost $c_{ij}$.

The TSPPD consists in finding the minimum cost route starting from the origin depot 0, visiting every node in $P \cup D$ exactly once, and finishing at the destination depot $2n+1$, while satisfying the precedence constraints which require that the pickup node of any given request be visited before the corresponding delivery node. The TSPPDF requires, in addition, that the FIFO constraints be satisfied: if the pickup node $i \in P$ is visited before the pickup node $j \in P$, then the delivery node $n+i \in D$ must be visited before the delivery node $n+j \in D$.

As in the work of Erdogan et al. (2009) and Carrabs et al. (2007a), we focus here on the uncapacitated case.

To formulate the TSPPD, we associate to each arc $(i, j) \in A$ a binary variable $x_{ij}$ taking value 1 if and only if node $j$ is visited immediately after node $i$. As usual, we define $\bar{S} = N \setminus S$, $x(S) = \sum_{i,j \in S} x_{ij}$, $x(\delta^+(S)) = \sum_{i \in S, j \notin S} x_{ij}$ and $x(\delta^-(S)) = \sum_{i \notin S, j \in S} x_{ij}$. For any node $i \in N$, let also $x(i, S) = \sum_{j \in S} x_{ij}$ and $x(S, i) = \sum_{j \in S} x_{ji}$. We also define the collection $\mathcal{S}$ of all node subsets $S \subset N$ such that $0 \in S, 2n+1 \notin S$ and there exists a node $i$ such that $i \notin S$ and $n+i \in S$.

Using this notation, the TSPPD can then be formulated as the following integer program:

$$\text{(TSPPD)} \qquad \text{Minimize} \sum_{(i,j) \in A} c_{ij} x_{ij} \tag{1}$$

subject to

$$x(\delta^+(i)) = 1 \qquad \forall i \in P \cup D \cup \{0\} \tag{2}$$

$$x(\delta^-(i)) = 1 \qquad \forall i \in P \cup D \cup \{2n+1\} \tag{3}$$

$$x(S) \leq |S| - 1 \qquad \forall S \subseteq P \cup D, |S| \geq 2 \tag{4}$$

$$x(S) \leq |S| - 2 \qquad \forall S \in \mathcal{S} \tag{5}$$

$$x_{ij} \in \{0, 1\} \qquad \forall (i, j) \in A. \tag{6}$$

The objective function (1) minimizes the total routing cost. Constraints (2) and (3) ensure that each pickup and delivery node is visited exactly once. Constraints (4)

ensure the connectivity of the solution while constraints (5) impose the precedence relationships between pickups and deliveries. The latter were introduced by Balas et al. (1995) in the context of the asymmetric TSP with precedence constraints, and by Ruland and Rodin (1997) in the context of the TSPPD. The idea behind these constraints is that if a set $S$ contains the origin depot 0 and a pickup node $i$ but not the corresponding delivery node $n+i$, then there cannot be a path starting from node 0 and visiting every other node in $S$ before leaving the set. As a consequence, the maximum flow on the arcs in $S$ cannot exceed $|S|-2$. Since the TSPPDF is a restriction of the TSPPD, these constraints are also valid for the former problem.

We now define the collection $\Omega$ of all subsets $S \subset P \cup D$ for which there is at least one request $j$ such that $j, n+j \in S$. The TSPPDF can be formulated by introducing an additional set of constraints in model (1)-(6).

**Proposition 1** *The FIFO policy can be imposed through the following constraints:*

$$x(i,S) + x(S) + x(S,n+i) \leq |S| \quad \forall S \in \Omega, \forall i \in P : i, n+i \notin S. \tag{7}$$

**Proof.** Suppose that the FIFO policy is satisfied but that one of these constraints is violated. Since $x(S) \leq |S|-1$ by constraints (4), this violation implies that $x(i,S) = x(S,n+i) = 1$ for a given set $S \in \Omega$ and node pair $i, n+i$. As a result, there is a path starting from $i$, visiting every node in $S$ and reaching $n+i$. But this is impossible because this path would violate the FIFO policy. Suppose now that all constraints are satisfied but the FIFO policy is not. This implies that between a given node pair $i, n+i$, the route visits a pickup node $j$ as well as the delivery node $n+j$. Hence, a constraint is violated for the set $S$ containing all nodes visited between $i$ and $n+i$, a contradiction. □

One can easily check that constraints (7) can be lifted as follows:

$$x(i,S) + x(S) + x(S,n+i) + x_{i,n+i} \leq |S| \quad \forall S \in \Omega, \forall i \in P : i, n+i \notin S. \tag{8}$$

Indeed, if $x_{i,n+i}$ takes value 1, then $x(i,S) + x(S,n+i) = 0$ and the inequality reduces to (4). The resulting inequality is represented in Figure 1

Figure 1: FIFO inequality.

# 3 Valid Inequalities

In this section we describe the valid inequalities that are added to model (1)-(7) to strengthen its linear programming relaxation within the branch-and-cut algorithm. We first describe some known inequalities for the TSPPD and we then introduce new inequalities that rely on the structure of the TSPPDF.

For general references on branch-and-cut algorithms and their application in the context of vehicle routing problems we refer the reader to Caprara and Fischetti (1997) and Naddef and Rinaldi (2002).

## 3.1 Inequalities from the TSPPD literature

All valid inequalities for the TSPPD are also valid for the TSPPDF. We thus include in our model some of the most effective inequalities for the TSPPD: strengthened subtour elimination constraints and generalized order constraints.

Balas et al. (1995) proposed the following inequalities to model the precedence constraints:

$$x(n+i,S) + x(S) + x(S,i) \leq |S| \quad \forall S \subset N, \forall i \in P : i, n+i \notin S. \tag{9}$$

These constraints can be explained as follows. For any node subset $S \subset N$, we know from subtour elimination constraints that $x(S) \leq |S| - 1$. If $x(n+i,S) = x(S,i) = 1$ there exists a path going from node $n+i$ to node $i$, which contradicts the precedence relationships and violates one of the constraints (9).

For any given subset $S \subseteq P \cup D$, let $\pi(S) = \{i \in P : n+i \in S\}$ and $\sigma(S) = \{n+i \in D : i \in S\}$ denote the sets of *predecessors* and *successors* of $S$, respectively. Balas et al. (1995) also introduced the following *predecessor* and *successor* inequalities for the precedence-constrained asymmetric TSP which generalizes the TSPPD:

$$x(S) + x(S, \bar{S} \cap \pi(S)) + x(S \cap \pi(S), \bar{S} \setminus \pi(S)) \leq |S| - 1 \tag{10}$$

$$x(S) + x(\bar{S} \cap \sigma(S), S) + x(\bar{S} \setminus \sigma(S), S \cap \sigma(S)) \leq |S| - 1. \tag{11}$$

Figure 2 illustrates the predecessor inequality (10) with the sets $S = \{j, k, n+i, n+j\}$, $\bar{S} \cap \pi(S) = \{i\}$, $S \cap \pi(S) = \{j\}$ and $\bar{S} \setminus \pi(S) = \{l\}$.

Figure 2: An example of a predecessor inequality (10).

For a given ordered set $O = \{i_1, i_2, \ldots, i_k\} \subseteq N$, with $k \geq 3$, Grötschel and Padberg (1985) proposed two generalizations of cycle inequalities for the asymmetric

TSP, called $D_k^+$ and $D_k^-$. These were later strengthened by Cordeau (2006), in the context of the pickup and delivery problem, to obtain the following *lifted* $D_k^+$ and $D_k^-$ inequalities:

$$\sum_{h=1}^{k} x_{i_h i_{h+1}} + 2\sum_{h=2}^{k-1} x_{i_h i_1} + \sum_{h=3}^{k-1}\sum_{l=2}^{h-1} x_{i_h i_l} + \sum_{n+j\in \bar{S}\cap\sigma(S)} x_{n+j,i_1} \leq k-1 \qquad (12)$$

$$\sum_{h=1}^{k} x_{i_h i_{h+1}} + 2\sum_{h=3}^{k} x_{i_1 i_h} + \sum_{h=4}^{k}\sum_{l=3}^{h-1} x_{i_h i_l} + \sum_{j\in \bar{S}\cap\pi(S)} x_{i_1 j} \leq k-1, \qquad (13)$$

where $i_{k+1} = i_1$.

Figure 3 illustrates inequality (12) for the set $O = \{i_1, i_2, i_3, i_4\}$.

Figure 3: An example of a lifted $D_k^+$ inequality (12)

.

Finally, let $U_1, \ldots, U_k \subset P \cup D$ be mutually disjoint subsets such that $i_1, \ldots, i_k \in P$ are requests for which $i_l, n+i_{l+1} \in U_l$ for $l = 1, \ldots, k$ (where $i_{k+1} = i_1$). The following *generalized order constraints* were introduced by Ruland and Rodin (1997) for the TSPPD:

$$\sum_{l=1}^{k} x(U_l) \leq \sum_{l=1}^{k} |U_l| - k - 1. \qquad (14)$$

Because of subtour elimination constraints, we know that $x(U_l) \leq |U_l| - 1$ for every set $l$. However, one can check that if $x(U_l) = |U_l| - 1$ for every set $l = 1, \ldots, k$, then there will be a path violating the precedence constraint for one of the requests $i_l$. As a result, at least one arc from this path must not be used.

## 3.2 New TSPPDF inequalities

We now introduce new families of valid inequalities for the TSPPDF.

### 3.2.1 Simple inequalities

The first four inequalities follow directly from the application of the FIFO policy.

**Proposition 2** *For any pair $i, j \in P$ the following inequality holds for the* TSPPDF*:*

$$x_{ij} + x_{n+i,j} + x_{n+i,2n+1} + \sum_{h\in D: h\neq n+i, n+j} x_{n+i,h} \leq 1. \qquad (15)$$

**Proposition 3** *For any pair $i, j \in P$ the following inequality holds for the* TSPPDF*:*

$$x_{ij} + x_{j,n+j} + x_{i,n+j} + \sum_{h \in D: h \neq n+i, n+j} x_{h,n+j} \leq 1. \tag{16}$$

**Proposition 4** *For any pair $i, j \in P$ the following inequality holds for the* TSPPDF*:*

$$x_{n+i,n+j} + x_{i,n+i} + x_{i,n+j} + \sum_{h \in P: h \neq i,j} x_{ih} \leq 1. \tag{17}$$

**Proposition 5** *For any pair $i, j \in P$ the following inequality holds for the* TSPPDF*:*

$$x_{n+i,n+j} + x_{n+i,j} + x_{0j} + \sum_{h \in P: h \neq i,j} x_{hj} \leq 1. \tag{18}$$

Figure 4 provides an example of a violated inequality of type (15) where the flow on each of the dashed arcs is equal to 0.5.

Figure 4: An example of a violated inequality (15)

.

The next three inequalities come from the fact that whenever an arc $(j, n+j)$ is used, it follows from the FIFO policy that the vehicle arrives empty at $j$ and leaves empty from $n+j$.

**Proposition 6** *For any node $j \in P$ and any set $H \subseteq P \setminus \{j\}$ the following inequality holds for the* TSPPDF*:*

$$\sum_{i \in H} x_{ij} + x_{j,n+j} + \sum_{n+h \in \sigma(P \setminus H)} x_{n+h,n+j} \leq 1. \tag{19}$$

**Proposition 7** *For any node $j \in P$ and any set $H \subseteq P \setminus \{j\}$ the following inequality holds for the* TSPPDF*:*

$$\sum_{i \in H} x_{ji} + x_{j,n+j} + \sum_{n+h \in \sigma(P \setminus H)} x_{n+j,n+h} \leq 1. \tag{20}$$

**Proposition 8** *For any pair $i, j \in P$ the following inequality holds for the* TSPPDF*:*

$$x_{ij} + x_{j,n+j} + x_{n+j,n+i} \leq 1. \tag{21}$$

Figure 5: An example of a violated inequality (19)

.

Figure 5 provides an example of a violated inequality of type (19) where the flow on each of the dashed arcs is equal to 0.5.

One can also check by case enumeration that the following two inequalities follow from the FIFO policy.

**Proposition 9** *For any pair $i, j \in P$ and any node $k \in P \setminus \{i, j\}$ the following inequality holds for the* TSPPDF*:*

$$x_{ij} + x_{n+i,j} + x_{n+i,k} + x_{n+i,2n+1} + x_{k,n+i} + \sum_{n+h \in D: h \neq i,j} (x_{n+i,n+h} + x_{k,n+h}) \leq 2. \quad (22)$$

**Proposition 10** *For any pair $i, j \in P$ and any node $n + k \in D \setminus \{n + i, n + j\}$ the following inequality holds for the* TSPPDF*:*

$$x_{n+i,n+j} + x_{i,n+i} + x_{i,n+j} + x_{i,n+k} + x_{n+k,i} + \sum_{h \in P: h \neq i,j} (x_{ih} + x_{n+k,h}) \leq 2. \quad (23)$$

Figure 6 provides an example of a violated inequality of type (22) where the flow on each of the dashed arcs is equal to 0.5.

Figure 6: An example of a violated inequality (22)

.

### 3.2.2 Alternative FIFO inequalities

The next set of inequalities can be used interchangeably with (8) to impose the FIFO policy in the TSPPD.

**Proposition 11** *Consider a node subset $S \subset P \cup D$ and two requests $i, j$ such that $i, n+j \notin S$ and $|S \cap \{j, n+i\}| = 1$. The following inequality is valid for the TSPPDF.*

$$x(i, S) + x(S) + x(S, n + j) + x_{i,n+j} \leq |S|. \quad (24)$$

**Proof.** Observe that if $x_{i,n+j} = 1$ in a feasible solution, then $x(i, S) = x(S, n+j) = 0$ and (24) reduces to (4).

Suppose that FIFO policy holds, but that (24) is violated for a set $S$ and a pair $i, j$. The left-hand-side of (24) is greater than $|S|$ if $x(i, S) = x(S, n + j) = 1$ and $x(S) = |S| - 1$. This implies that there is a path from $i$ to $n + j$. If $j \in S$ and $n + i \notin S$, then $n + j$ is visited before $n + i$ and the FIFO policy is not satisfied for request $i$, which contradicts the hypothesis. If instead $j \notin S$ and $n + i \in S$, then $j$ is visited before $i$ and the FIFO policy is not satisfied for $j$, which again contradicts the hypothesis.

Suppose now that constraints (24) are satisfied, but the FIFO policy is not. This implies that there exists a pair of requests $i, j$ whose pickup nodes are visited in this order, while the delivery nodes are visited in the reverse order. In this case $x_{i,n+j} = 0$ since the vehicle must visit at least node $j$ after node $i$. More specifically there is a path which starts at $i$, terminates at $n + j$ and includes $j$ but not $n + i$. By selecting a set $S$ containing all the vertices in the path, except $i$ and $n + j$, one can check that the corresponding constraint (24) is not satisfied, a contradiction. □

Figure 24 illustrates the alternative FIFO inequalities. In case (a), the set $S$ contains node $j$, while in case (b) this set contains node $n + i$.

Figure 7: Alternative FIFO inequalities (24)

### 3.2.3 Shutter inequalities

Consider a pair of requests $i$ and $j$. The FIFO policy implies that at most one arc between $(i, j)$ and $(n + j, n + i)$ may belong to a feasible solution, while the degree constraints (2) and (3) make the arc $(n + j, i)$ incompatible with each of the previous ones. Hence,

$$x_{ij} + x_{n+j,n+i} + x_{n+j,i} \leq 1. \tag{25}$$

Similar reasoning leads to the following inequality:

$$x_{ij} + x_{n+j,n+i} + x_{n+i,j} \leq 1. \tag{26}$$

Inequalities (25) and (26) can in fact be merged into the following inequality:

$$x_{ij} + x_{n+j,n+i} + x_{n+j,i} + x_{n+i,j} \leq 1. \tag{27}$$

Inequality (25) can also be generalized by considering a sequence of three or more requests, thus obtaining the following *shutter inequality* which is depicted in Figure 8.

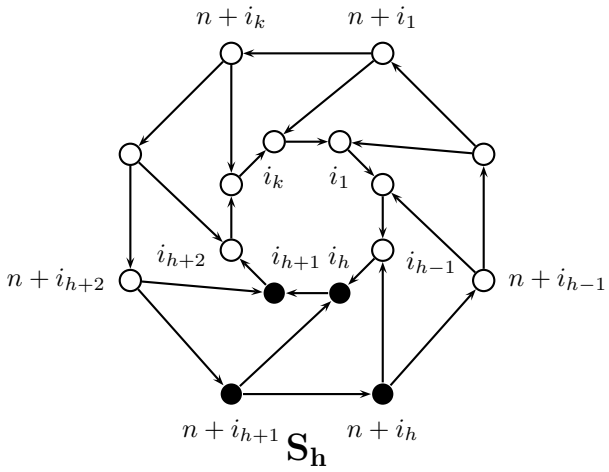


Figure 8: Shutter inequality.

**Proposition 12** *Consider a sequence of $k \geq 3$ requests $i_1, i_2, \ldots, i_k$ and let $i_{k+1} = i_1$ then the following inequality holds for the* TSPPDF*:*

$$\sum_{h=1}^{k} (x_{i_h, i_{h+1}} + x_{n+i_{h+1}, i_h} + x_{n+i_{h+1}, n+i_h}) \leq k - 1. \tag{28}$$

Before proving the above proposition we introduce some notation and observation that will be used in the following. First observe that we extend the sequence of requests in a circular way, i.e., request indices $i_h$ should be read modulo $k$. Given a request $i_h$, with $h = 1, \ldots, k$, we call *zig-zag set* the set $S_h = \{(i_h, i_{h+1}), (n+i_{h+1}, i_h), (n+i_{h+1}, n+i_h)\}$, distinguished by the shaded vertices in Figure 8. Moreover, we call *clockwise* the arcs starting from each pickup node (internal circuit) and *counter clockwise* the arcs starting from each delivery node (external circuit). From (25) we know that only one of the three arcs of $S_h$ may belong to a feasible integer solution. Two cases may arise:

(a) If the clockwise arc $(i_h, i_{h+1})$ belongs to a feasible solution, then, the two counter clockwise arcs $(n + i_{h+2}, i_{h+1})$, $(n + i_{h+2}, n + i_{h+1}) \in S_{h+1}$ cannot belong to the same feasible solution, due to the degree constraint or to the FIFO policy. Hence, either $S_{h+1}$ is empty or the arc $(i_{h+1}, i_{h+2})$ is in the solution.

(b) If one of the two counter clockwise arcs starting from node $n + i_{h+1}$ belongs to a feasible solution, then the degree constraint or the FIFO policy impose that $(i_{h-1}, i_h)$ cannot belong to this solution. Hence, either $S_{h-1}$ is empty or exactly one of the arcs $(n + i_h, i_{h-1}), (n + i_h, n + i_{h-1})$ belongs to the solution.

Applying (a) to sets $S_h, S_{h+1}, \ldots$, in turn, or (b) to sets $S_h, S_{h-1}, \ldots$ one can prove the following observation.

**Observation 1** *If a solution uses a clockwise arc, then there is a path starting with this arc that contains only clockwise arcs, until an empty zig-zag set is reached. If a solution uses a counter clockwise arc starting from a node $n+i_h$, then there is a path that visits $n+i_h, n+i_{h-1}, n+i_{h-2}, \ldots$ using counter clockwise arcs and possibly arcs not in* (28)*, until an empty zig-zag set is reached.*

Using the above observation, we are now in a position to prove Proposition 12.

**Proof.** The left hand side of inequality (28) is the sum of the $x$ variables associated with the $k$ zig-zag sets $S_1, \ldots, S_k$. From inequality (25) we know that at most one arc for each set may belong to a feasible solution, so we can prove (28) by showing that at least one of the $k$ zig-zag sets is empty. Without loss of generality suppose that set $S_h$ is not empty. For any choice of the arcs in $S_h$, Observation 1 shows that there is an empty zig-zag set, otherwise there exists a path (clockwise or counter clockwise) which returns to a node of $S_h$. □

### 3.2.4 Lifted shutter inequalities

The shutter inequalities can be lifted by adding arcs between pairs of pickup vertices.

**Proposition 13** *Consider a sequence of $k > 3$ requests $i_1, i_2, \ldots, i_k$ and let $i_{k+1} = i_1$ then the following inequality holds for the* TSPPDF*:*

$$\sum_{h=1}^{k} \left( x_{i_h,i_{h+1}} + x_{n+i_{h+1},i_h} + x_{n+i_{h+1},n+i_h} + \sum_{l=h+2}^{h+k-2} x_{i_h,i_l} \right) \leq k-1. \tag{29}$$

**Proof.** Consider a feasible solution. If no lifted arc (inner summation) is used then the proof follows from Proposition 12. Otherwise, let $p$ be the number of lifted arcs used in the solution. Suppose that the solution has at least $p+1$ empty zig-zag sets. In this case, recalling that at most one arc from each zig-zag set is used, we know that the total number of arcs from (29) is at most $k-p-1$ (from the zig-zag sets) plus $p$ (from the lifted arcs) and the proposition holds.

To conclude the proof it remains to show that there are more than $p$ empty zig-zag sets. Let $r$ denote the number of different head/tail vertices of the $p$ lifted arcs used, and note that $p < r \leq 2p$. The different head/tail vertices divide the circuit of the clockwise arcs into $r$ parts which define $r$ corresponding sectors of the shutter. Consider any sector and let $i_h$ and $i_{h+j}$ be the two head/tail vertices which define the sector (i.e., $i_h$ and $i_{h+j}$ are consecutive along the circuit). From Observation 1 we know that one of the following cases arises: *i)* there is an empty zig-zag set in this sector; *ii)* there is a clockwise path from $i_h$ to $i_{h+j}$; *iii)* there is a path from $n+i_{h+j}$ to

$n+i_h$. In the latter case, since the precedence constraints hold for a feasible solution, we also know that there is a counter clockwise path from $i_{h+j}$ to $i_h$. Now suppose that the $r$ sectors contain $q \leq p$ empty zig-zag sets. The total number of arcs and paths among the $r$ head/tail vertices is $p+(r-q) \geq r$, but, in this case, at least one subtour arises. We have thus shown that more than $p$ empty zig-zag sets exist, which concludes the proof. □

By applying the same ideas used to generalize (25), we may also generalize (26), which results in the following inequality.

**Proposition 14** *Consider a sequence of $k > 3$ requests $i_1, i_2, \ldots, i_k$ and let $i_{k+1} = i_1$ then the following inequality holds for the* TSPPDF*:*

$$\sum_{h=1}^{k} \left( x_{i_h,i_{h+1}} + x_{n+i_h,i_{h+1}} + x_{n+i_{h+1},n+i_h} + \sum_{l=h+2}^{h+k-2} x_{n+i_h,n+i_l} \right) \leq k-1. \qquad (30)$$

# 4 Branch-and-Cut

The formulation introduced in Section 2 along with the valid inequalities described in Section 3 are used within a branch-and-cut algorithm which we now describe.

## 4.1 Starting model

At the root node we initialize our model with constraints (2), (3), and all subtour elimination constraints (4) with $|S| = 2$. We also add the following set of inequalities:

- the simple predecessor inequalities (10) obtained by setting $S$ equal to $\{i, j\}$, $\{i, n+j\}$ and $\{i, n+i, j\}$, and the successor inequalities (11) with $S$ equal to $\{n+i, n+j\}$, $\{i, n+j\}$ and $\{i, n+i, n+j\}$;
- simple cases of the strengthened $D_k^+$ inequalities (12) and $D_k^-$ inequalities (13) with $k = 3$: $x_{n+i,j} + x_{ji} + x_{i,n+i} + x_{n+j,n+i} + 2x_{j,n+i} \leq 2$ and $x_{i,n+i} + x_{n+i,n+j} + x_{n+j,i} + x_{ij} + 2x_{i,n+j} \leq 2$;
- the constraints (15), (16), (17), (18), (21) and (27) whose cardinality is $n^2$.

## 4.2 Separation procedures

The separation of the inequalities from the literature, namely (4)-(5) and (9)-(14) is done in the way described by Cordeau et al. (2009). Inequalities (4), (5) and

(9) are separated exactly by performing $O(n)$ maximum flow computations, whereas (10)-(14) are separated by fast heuristic procedures.

Constraints (8) are separated exactly with a procedure similar to the one used by Cordeau et al. (2009) for the LIFO constraints. Using (2), (3) and $x(S)+x(\delta^+(S)) = |S|$ we can rewrite (8) as

$$x(\delta^+(S)) + x(i, \bar{S}) + x(\bar{S}, n+i) \quad \geq \quad 2 + x_{i,n+i}. \tag{31}$$

For each node $i \in P$ we need to check all sets $S \in \Omega$ such that $i, n+i \notin S$ (recall that $\Omega = \{S \subset P \cup D : \exists j \in P$ such that $j, n+j \in S\}$). This check is performed by considering, in turn, one vertex $j \in P$ at a time. Let $x^*$ denote the current fractional optimal solution. We start from the standard *support digraph* $G^* = (N, A^*)$, that is obtained from $G$ by selecting each arc $(i,j) \in A$ with $x^*_{ij} > 0$ and associating with it a capacity of value $x^*_{ij}$. Then we apply the following transformations:

(i) we increase the capacity of the arcs $(j,k)$ by the value $x^*_{ik} + x^*_{k,n+i}$ $(k \in P \cup D)$;

(ii) we set to a large value (e.g., 3) the capacity of arcs $(0,i)$, $(i,n+i)$ and $(j,n+j)$.

We then solve the maxflow problem from $j$ to $i$ and we define $S^* \ni j$ as the set which identifies the minimum cut separating $j$ from $i$. Note that due to the capacity setting (ii), for any minimum cut the vertices $0, i$ and $n+i$ belong to one shore of the cut and $j, n+j$ to the other, so $S^* \in \Omega$ and is a valid set for computing (31). If the maximum flow value is smaller than $2 + x^*_{i,n+i}$, then $S^*$ defines a violated constraint (31). We can show the correctness of this claim as follows. Let $f^*$ denote the optimal flow and recall that the maximum flow value, say $v^*$, is equal to the sum of the flows on the arcs in $\delta^+(S^*)$. Then $v^* = f^*(\delta^+(S^*)) = f^*(j, \bar{S^*}) + f^*(S^* \setminus \{j\}, \bar{S^*})$. Since all the outgoing arcs in the minimum cut are saturated we can rewrite $v^* = x^*(j, \bar{S^*}) + x^*(i, \bar{S^*}) + x^*(\bar{S^*}, n+i) + x^*(S^* \setminus \{j\}, \bar{S^*}) = x^*(\delta^+(S^*)) + x^*(i, \bar{S^*}) + x^*(\bar{S^*}, n+i)$, which proves the validity of the procedure.

We separate constraints (19) and (20) in $O(n^2)$ by observing that, given an arc $(j, n+j) \in A$, for any $H \in P \setminus \{j\}$ exactly one of the two arcs $(h,j)$ and $(n+h, n+j)$ is used in the constraints, independently of the choice of set $H \subseteq P$. It follows that we can determine the most violated constraint (19), for arc $(j, n+j)$, by scanning all requests in $P \setminus \{j\}$ and adding the current request $i$ to set $H$ if $x_{ij} \geq x_{n+i,n+j}$. The most violated constraints (20) are computed similarly.

The separation of constraints (22) is carried out in a straightforward way by considering each arc $(i,j) \in A$, each $k \in P \setminus \{i,j\}$, and evaluating the constraint. This method leads to an $O(n^4)$ time procedure. We can reduce the computing time

by rewriting (22) as

$$x_{ij} + x_{n+i,j} + x_{n+i,k} + x_{n+i,2n+1} + \sum_{n+h \in D} x_{n+i,n+h} + \sum_{n+h \in D} x_{k,n+h} - x_{k,n+j} - x_{n+i,n+j} \leq 2. \tag{32}$$

We can compute the two summations in (32), for each $\ell \in P \cup D$, through a preprocessing phase running in $O(n^2)$ time. Using these values each evaluation of (32) is done in $O(1)$ and the overall separations requires $O(n^3)$ time.

A similar procedure is used to separate inequalities (23).

The separation of inequalities (24) is done with a procedure similar to the one adopted for (8). We discuss the case $j \in S, n+i \notin S$: the second case $j \notin S, n+i \in S$ can be easily derived from this one. We can rewrite (24) as

$$x(i, \bar{S}) + x(\delta^+(S)) + x(\bar{S}, n+j) \geq 2 + x_{i,n+j}. \tag{33}$$

We consider the support digraph $G^*$ and apply the following capacity transformations:

(i) the capacity of each arc $(j,k)$ $(k \in P \cup D \setminus \{j\})$ is increased by $x^*_{ik} + x^*_{k,n+j}$;

(ii) we set to a large value (e.g., 3) the capacity of arcs $(i, n+j)$, $(n+j, i)$, $(i, n+i)$ and $(0, i)$.

Transformation (ii) imposes that in any minimum cut the vertices 0, $i$, $n+i$ and $n+j$ belong to the same shore of the cut. We compute a maximum flow from source $j$ to sink $i$ on the resulting digraph, and we define $S^* \ni j$ as the set which identifies the minimum cut separating $j$ from $i$. If the maximum flow value is greater than $2 + x^*_{i,n+j}$, then $S^*$ defines a violated constraint (33). The separation of (24) may thus require $O(n^2)$ maximum flow computations.

The shutter inequalities (28) can be separated exactly in polynomial time with the following procedure. Let us define new arc costs $\overline{x}^*_{ij} = 1 - x^*_{ij} + x^*_{n+j,i} + x^*_{n+j,n+i}$ and rewrite (28) as

$$\sum_{h=1}^{k} \overline{x}^*_{i_h, i_{h+1}} \geq 1, \tag{34}$$

with $i_{k+1} = i_1$. The separation of (34) is equivalent to finding a circuit of $k$ vertices in the subgraph $G_P$ induced by $P$, with total cost strictly smaller than one. We can apply the well-known $O(n^3)$ procedure of Floyd (1962) to compute the all-pairs shortest paths in $G_P$. Since negative costs may exist, negative length circuits can be found in $G_P$. In this case the procedure terminates with one such circuit, which induces a violated inequality. Otherwise, we have computed all shortest paths and it is then sufficient to consider one arc $(i,j)$ at a time to identify the shortest circuit using $(i,j)$ and, finally, the shortest circuit of $G_P$. If the cost of this circuit is smaller

than one we have found the most violated shutter inequality. Otherwise, we have proved that no inequality is violated.

Unfortunately the procedure outlined above cannot be applied to the lifted shutter inequalities (29) and (30). We thus separate them by means of a tabu search heuristic similar to the one used by Cordeau et al. (2009) to separate the so called "hamburger" inequalities. We start with a sequence $\sigma$ of three randomly selected requests. Than, we compute all the moves obtained by deleting one of the requests in $\sigma$, switching the positions of two requests in $\sigma$, and inserting a request in a given position in $\sigma$. We choose the move leading to the maximum value of the left-hand side of the inequality being separated minus $|\sigma|$. The tabu list is implemented as a set of couples $\langle i, j \rangle$, where $i = -1$ means that request $j$ is inserted in $\sigma$, $j = -1$ means that $i$ is removed from $\sigma$, and $i, j \geq 0$ implies that $i$ and $j$ are exchanged in $\sigma$. When a move $\langle i, j \rangle$ is performed, the opposite move $\langle j, i \rangle$ is forbidden for $\max(5, 2n/10)$ iterations. The algorithm is halted after 50 iterations or after 10 violated cuts have been added to the model. In practice, in the worst case the algorithm requires 50 iterations, each having complexity $O(n^2)$.

## 4.3 Branching strategy

Following the branching rules introduced by Carrabs et al. (2007a), we branch by extending a feasible path from the origin depot. We start by identifying the arc with largest flow among those leaving the depot, and we create two branches corresponding to this variable taking value 0 or 1. At each node, we extend the path by branching on the outgoing arc with the largest flow. As a result, we may sometimes branch on variables taking value 1. Every time a variable is set to 1 in a branch, we apply the *filters* used by Carrabs et al. (2007a) to eliminate incompatible arcs from the graph.

## 4.4 Separation strategy

At a given node of the branch-and-bound tree, we first call the heuristic separation procedures for FIFO constraints, and then the heuristic procedures for the classical TSPPD constraints. After these two steps, we create the support digraph based on the (fractional) $x$ values and call the exact separation procedures for FIFO constraints, followed by the exact procedures for the TSPPD constraints. This process stops as soon as $\theta$ inequalities have been added to the model, where $\theta$ is a parameter (see Section 5). To reduce computing times, we do not call the separation procedures at nodes where the the number of variables already fixed to 1 by the branching is greater than or equal to $n/2$.

Although every valid inequality that is added during the branching is a global cut, in the sense that it is valid for any portion of the search tree, the strength of these

inequalities depends on the variables that have been set to 1. As a result, some cuts are redundant in some subtrees and their presence slows down the simplex algorithm. To alleviate this problem, we add all cuts as local cuts.

To avoid re-generating the same local cut at several nodes in the tree, however, we use a pool of cuts. Every time we process a node, we first check if the pool contains violated cuts. In that case, we add to the model the $\theta$ most violated cuts.

# 5 Computational Results

We coded the branch-and-cut algorithm in C++ and used CPLEX 9 as Integer Linear Programming solver. We ran the algorithm on a Pentium IV at 3 GHz, and tested its behavior on the same test bed used by Carrabs et al. (2007a) and Erdogan et al. (2009). This test bed was generated from nine TSP instances of the TSPLIB: a280, att532, brd14051, d15112, d18512, fnl4461, nrw1379, pr1002 and ts225. From each TSP instance, nine TSPPDF instances were generated by selecting 19, 23, 27, 31, 35, 39, 43, 47 and 51 vertices, respectively. The smallest instances (19 vertices) were constructed by performing random matchings between the selected vertices to obtain $n = 9$ pairings between pickups and deliveries (i.e., nine requests). The unmatched vertex represents both the origin and the destination depot. Each larger instance was in turn constructed from the previous one by considering four more vertices from the original TSP file, and randomly matching them into two requests. This procedure ensures that the optimal solution value of a larger instance is not lower than that of a smaller one. The costs associated with the arcs were obtained by rounding down to the next integer the Euclidean distances between the vertices. Our branch-and-cut algorithm was given a limit of 3 hours of CPU time on each instance. It takes as an initial upper bound the value obtained by the heuristic algorithm of Erdogan et al. (2009), which runs in a few seconds for the instance size considered here.

We first focus on the results obtained by solving the LP relaxation of the problem. In Table 1 we report the results obtained by the plain formulation (model (1)–(7)), and by this formulation enlarged by considering one family of valid inequalities at a time. For each row, the column labeled *Inst.* gives the name of the original TSP instance, and the column labeled $n$ gives the number of requests in the TSPPDF instance. The other columns in the table give the percentage gaps between the LP relaxation of the given formulation and the upper bound. The gaps are computed as $100(UB-LB)/UB$, where $UB$ is the upper bound provided by the heuristic algorithm of Erdogan et al. (2009). For these columns we use as labels the equation number of the family of inequalities. The last row reports, for each column, the average over all instances. The last columns are grouped by inequalities from the TSPPD literature (Section 3.1) and new FIFO inequalities (Section 3.2).

One can observe that the plain formulation is rather weak and that the integrality gaps are large: around 16% on average, and more than 32% in the worst case (instance ts225 with $n = 15$). This performance is improved by the successive families of inequalities. Among the ones from the TSPPD literature, inequalities (10) and (11) give the best improvement, as they reduce the gap by 1.7% on average, and by almost 8% in the best case (instance a280 with $n = 11$). Among the simple inequalities, (15) and (16) based on arc $(i, j)$, and inequalities (17) and (18) based on arc $(n+i, n+j)$ are the most effective ones, as they reduce the average gap by roughly 1%. Inequalities (19), (20) and (21) based on arc $(j, n + j)$ have a slightly worse behavior then the previous ones. It is worth recalling that (15), (16), (17), (18) and (21) are all added directly at the root node, whereas (19) and (20) are separated also at the children nodes and can lead to further improvements. Inequalities (22) and (23) have a weak performance. Their use is nonetheless very important during the branching as they are effective in improving the lower bounds and thus reducing the number of nodes in the tree. Lifting the FIFO inequalities by means of (24) gives a small but systematic improvement in the gap. The lifted shutter inequalities (29) and (30) lead to an average improvement of roughly 0.7%. Again, these inequalities become effective during the exploration of the branch-and-bound tree.

In Table 2 we focus on the marginal contribution of each family of inequalities. In column labeled *full* we present the percentage gap of the linear relaxation of our full formulation (i.e., the one containing all the inequalities). In the next columns we give the gaps obtained by the full formulation **minus** the family of inequalities whose number appears as label of the column. The use of all the inequalities leads to an important improvement of more than 5% with respect to the plain formulation. In some cases this improvement can be very high, as for instances a280 with $n \leq 13$, where it is 12% and more. Removing a single inequality at the time does not affect significantly this improvement, with the exception of inequalities (10) and (11), that, if removed, produce a worsening of roughly 1.4% in the average gap. In some cases deleting an inequality from the model can lead to an improvement in the gap. This is a known phenomenon, due to the implicit heuristic separation procedures of CPLEX.

In Table 3 we finally report the results obtained by running our branch-and-cut algorithm with a CPU time limit of 3 hours on each instance. Columns labeled *Inst.* and $n$ have the same meaning as in the two previous tables, whereas column labeled $UB$ reports the upper bound value. It is worth noting that this upper bound coincides with the optimal value for all the instances where a proof of optimality is given. Our formulation is compared with the two other exact algorithms developed for the TSPPDF: the additive branch-and-bound algorithm of Carrabs et al. (2007a), and the linear integer programming formulation with a polynomial number of variables and constraints of Erdogan et al. (2009). These two procedures were run on a 2.4 GHz AMD Opteron 250 processor, with a 3-hour CPU time limit. For the additive

branch-and-bound, the column labeled *nodes* reports the number of nodes explored by the enumeration tree, and the column labeled *sec* indicates the number of seconds required to run the algorithm. For the linear integer programming formulation only the CPU time information is known. For our branch-and-cut algorithm we also report the lower bound obtained ($LB$), the percentage gap ($\%gap = 100(UB - LB)/UB$) and the number of local cuts stored in the pool (*pool*).

The formulation we use is the full one, with the exception of the generalized order constraints (14) since they proved to worsen the average performance of the complete branch-and-cut algorithm. All other families of inequalities are useful. On the basis of our computational evidence with some trial values of the parameters, we set the maximum number of cuts to be added at each node as $\theta = 10$ (better than 5 or 20) and the maximum size of the pool of local cuts to 20000 cuts (better than 10000 or 30000). As mentioned in the previous section, we stop separating inequalities when the number of variables already fixed to 1 by the branching is greater than or equal to $n/2$ (better than $n$ or $n/3$).

All instances with up to 13 requests (i.e., 27 vertices) were solved to optimality in usually less than one hour. Instance d15112 is solved in one hour and three minutes and is the one requiring the most time. Among the instances with 15 requests, four out of nine are solved to optimality within the given time limit. For the unsolved instances, the percentage gap can be quite large. The worst case is instance nrw1379, where the gap is larger than 10%. The easiest groups of instances are the ones derived from a280 and pr1002, for which the branch-and-cut solves all instances in the test bed (up to 51 vertices) to optimality. Both the number of nodes and the number of local cuts in the pool grow from the instances with small size to the instances with large size. For 8 cases out of 49 the number of local cuts stored in the pool reaches the maximum value allowed.

The branch-and-cut algorithm outperforms the two other exact algorithms for the TSPPDF. The compact formulation of Erdogan et al. (2009) solves consistently to optimality only the instances with 9 requests. Moreover, this is achieved with very large CPU times. No instance with 13 requests or more was solved. The additive branch-and-bound algorithm of Carrabs et al. (2007a) has a better performance, but is nevertheless outperformed by branch-and-cut. The smallest unsolved instance has 13 requests (versus 15 for the branch-and-cut) while the largest solved instance has 19 requests (versus 25 for the branch-and-cut). Moreover the branch-and-cut algorithm was able to solve eight more instances. Although comparing computing times between the two algorithms does not lead to any meaningful conclusion, one can observe that the CPU time consumption of the branch-and-bound algorithm grows consistently with the size of the instance, while it is less predictable for branch-and-cut. Nevertheless, the number of nodes explored by the branch-and-cut is much smaller, often by several orders of magnitude. Since the branching schemes used by

both algorithms are very similar, the main difference comes from the tighter lower bounds provided by the formulation and inequalities introduced in this paper.

# 6 Conclusions

This paper has introduced a branch-and-cut algorithm for the TSPPDF. The FIFO policy affects the combinatorial structure of the problem and makes it very difficult to solve. Branch-and-cut techniques can solve instances of the symmetric TSPPD with up to 35 requests (see Dumitrescu et al., 2009). For the problem addressed in this paper, instances with only 15 requests remain unsolved after being tackled with three different exact techniques. The TSPPDF also appears to be more difficult to solve than the TSPPD with LIFO loading. On a very similar set of test instances, the smallest unsolved instances have 23 requests (see Cordeau et al., 2009). The combinatorial structures of the two problems are very different and so are the valid inequalities introduced for each of them.

The results presented in this paper were obtained with a formulation based on classical arc variables with an exponential number of constraints separated within a branch-and-cut framework. This approach outperforms previous exact algorithms for the same problem. The use of valid inequalities to strengthen the formulation is very important, but it is not the only ingredient that contributes to the success of the branch-and-cut algorithm. Significant improvements were obtained by using a tailored branching strategy enriched with fathoming criteria, and by using local cuts instead of global ones. It is an interesting topic for future research to evaluate the use of these ideas in branch-and-cut techniques for other variants of the traveling salesman problem.

# Acknowledgements

We thank the Italian Ministero dell'Istruzione, dell'Università e della Ricerca (MIUR) and the Canadian Natural Sciences and Engineering Research Council (grant 227837-04). This support is gratefully acknowledged.

# References

E. Balas, M. Fischetti, and W.R. Pulleyblank. The precedence-constrained asymmetric traveling salesman polytope. *Mathematical Programming*, 68:241–265, 1995.

A. Caprara and M. Fischetti. Branch-and-cut algorithms. In M. Dell'Amico, F. Maffioli, and S. Martello, editors, *Annotated Bibliographies in Combinatorial Optimization*, pages 45–63. Wiley, New York, 1997.

F. Carrabs, R. Cerulli, and J.-F. Cordeau. An additive branch-and-bound algorithm for the pickup and delivery traveling salesman problem with LIFO or FIFO loading. *INFOR*, 45:223–238, 2007a.

F. Carrabs, J.-F. Cordeau, and G. Laporte. Variable neighbourhood search for the pickup and delivery traveling salesman problem with LIFO loading. *INFORMS Journal on Computing*, 19:618–632, 2007b.

L. Cassani. Algoritmi euristici per il TSP with rear-loading. Master's thesis, Università di Milano, http://www.crema.unimi.it/˜righini/Papers/Cassani.pdf, 2004.

J.-F. Cordeau. A branch-and-cut algorithm for the dial-a-ride problem. *Operations Research*, 54:573–586, 2006.

J.-F. Cordeau, G. Laporte, J.-Y. Potvin, and M.W.P. Savelsbergh. Transportation on demand. In C. Barnhart and G. Laporte, editors, *Transportation*, Handbooks in Operations Research and Management Science 14, pages 429–466. Elsevier, Amsterdam, 2007.

J.-F. Cordeau, M. Iori, G. Laporte, and J.J. Salazar-Gonzalez. Branch-and-cut for the pickup and delivery traveling salesman problem with LIFO loading. *Networks*, 2009. DOI: 10.1002/net.20312.

I. Dumitrescu, S. Ropke, J.-F. Cordeau, and G. Laporte. The traveling salesman problem with pickup and delivery: Polyhedral results and a branch-and-cut algorithm. *Mathematical Programming*, 2009. DOI: 10.1007/s10107-008-0234-9.

G. Erdogan, J.-F. Cordeau, and G. Laporte. The pickup and delivery traveling salesman problem with first-in-first-out loading. *Computers & Operations Research*, 36:1800–1808, 2009.

R.W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5:345, 1962.

M. Grötschel and M.W. Padberg. Polyhedral theory. In E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, editors, *The Traveling Salesman Problem*, pages 251–305. Wiley, New York, 1985.

P. Healy and R. Moll. A new extension of local search applied to the dial-a-ride problem. *European Journal of Operational Research*, 83:83–104, 1995.

B. Kalantari, A.V. Hill, and S.R. Arora. An algorithm for the traveling salesman problem with pickup and delivery customers. *European Journal of Operational Research*, 22:377–386, 1985.

D. Naddef and G. Rinaldi. Branch-and-cut algorithms for the capacitated VRP. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*, pages 225–242. SIAM, Philadelphia, 2002.

J.A. Pacheco. Heuristico para los problemas de ruta con carga y descarga en sistemas LIFO. *SORT, Statistics and Operations Research Transactions*, 21: 153–175, 1997.

J.A. Pacheco. Problemas de rutas con carga y descarga en sistemas LIFO: soluciones exactas. *Estudios de Economía Aplicada*, 3:69–86, 1995.

J. Renaud, F.F. Boctor, and J. Ouenniche. A heuristic for the pickup and delivery traveling salesman problem. *Computers & Operations Research*, 27:905–916, 2000.

J. Renaud, F.F. Boctor, and G. Laporte. Perturbation heuristics for the pickup and delivery travelling salesman problem. *Computers & Operations Research*, 29: 1129–1141, 2002.

K.S. Ruland and E.Y. Rodin. The pickup and delivery problem: Faces and branch-and-cut algorithm. *Computers & Mathematics with Applications*, 33(12): 1–13, 1997.

L.J.J. Van der Bruggen, J.K. Lenstra, and P.C. Schuur. Variable-depth search for the single-vehicle pickup and delivery problem with time windows. *Transportation Science*, 27:298–311, 1993.

H. Xu, Z.-L. Chen, S. Rajagopal, and S. Arunapuram. Solving a practical pickup and delivery problem. *Transportation Science*, 37:347–364, 2003.

Table 1: Integrality gaps for the plain formulation plus one family of valid inequalities at a time.

| *Inst.* | *n* | *plain* | *TSPPD literature* | | | | *new FIFO inequalities* | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | (9) | (10),(11) | (12), (13) | (14) | (15) | (16) | (17) | (18) | (19) | (20) | (21) | (22) | (23) | (24) | (29) | (30) |
| a280 | 9 | 12.94 | 12.94 | 9.14 | 12.18 | 11.68 | 10.91 | 9.90 | 10.91 | 11.68 | 11.17 | 12.18 | 12.94 | 12.69 | 12.69 | 12.94 | 11.42 | 11.42 |
| | 11 | 18.12 | 17.06 | 10.45 | 15.99 | 16.42 | 15.14 | 14.29 | 15.14 | 15.99 | 15.57 | 16.42 | 18.12 | 16.84 | 17.91 | 16.63 | 14.93 | 15.14 |
| | 13 | 23.18 | 23.18 | 18.32 | 21.87 | 22.80 | 19.44 | 19.63 | 20.00 | 19.44 | 20.93 | 20.56 | 21.68 | 22.24 | 22.62 | 22.43 | 20.93 | 21.31 |
| | 15 | 23.01 | 23.01 | 18.71 | 21.19 | 21.85 | 20.20 | 19.70 | 19.70 | 19.87 | 21.03 | 20.86 | 21.69 | 22.35 | 22.52 | 22.68 | 20.36 | 20.70 |
| | 17 | 20.26 | 20.26 | 16.76 | 19.24 | 19.24 | 17.93 | 17.20 | 17.64 | 17.64 | 19.24 | 18.51 | 19.53 | 19.97 | 19.97 | 20.12 | 18.66 | 18.80 |
| | 19 | 20.33 | 19.92 | 15.93 | 18.82 | 18.41 | 18.27 | 18.54 | 17.58 | 17.99 | 18.96 | 18.41 | 18.96 | 19.51 | 19.78 | 19.92 | 17.86 | 17.99 |
| | 21 | 19.95 | 19.82 | 16.16 | 18.43 | 18.81 | 18.06 | 17.55 | 17.55 | 17.42 | 18.81 | 18.18 | 19.07 | 19.57 | 19.44 | 19.57 | 17.93 | 18.18 |
| | 23 | 20.69 | 19.98 | 15.58 | 18.43 | 17.95 | 19.26 | 18.55 | 19.02 | 18.67 | 19.50 | 19.02 | 19.74 | 20.45 | 20.21 | 19.74 | 17.12 | 17.36 |
| | 25 | 21.51 | 21.17 | 16.44 | 18.69 | 19.03 | 20.05 | 19.03 | 19.71 | 19.26 | 20.16 | 19.93 | 20.50 | 21.17 | 21.28 | 20.95 | 18.02 | 18.13 |
| att532 | 9 | 11.28 | 9.70 | 8.64 | 10.40 | 8.89 | 10.79 | 9.56 | 9.63 | 11.01 | 10.05 | 11.21 | 11.28 | 11.28 | 11.28 | 11.28 | 11.19 | 11.23 |
| | 11 | 9.08 | 8.84 | 8.60 | 8.99 | 8.80 | 8.84 | 8.40 | 8.40 | 8.69 | 8.16 | 8.84 | 9.08 | 9.08 | 9.08 | 9.08 | 8.91 | 8.99 |
| | 13 | 14.36 | 14.04 | 12.61 | 13.63 | 13.31 | 13.86 | 13.13 | 12.93 | 13.52 | 13.31 | 13.70 | 13.72 | 13.73 | 14.04 | 14.04 | 13.75 | 13.81 |
| | 15 | 17.13 | 17.12 | 15.79 | 16.83 | 15.68 | 16.93 | 16.36 | 16.58 | 16.97 | 16.50 | 17.03 | 17.12 | 17.08 | 16.93 | 17.08 | 16.73 | 16.68 |
| | 17 | 16.80 | 16.77 | 16.01 | 16.56 | 15.66 | 16.77 | 16.04 | 16.24 | 16.85 | 16.28 | 16.70 | 16.82 | 16.66 | 16.56 | 16.70 | 16.30 | 16.30 |
| | 19 | 18.51 | 18.50 | 17.41 | 18.30 | 18.10 | 17.98 | 17.43 | 17.05 | 17.82 | 17.52 | 17.91 | 17.83 | 18.54 | 18.24 | 18.42 | 18.10 | 18.09 |
| brd14051 | 9 | 3.65 | 3.65 | 3.53 | 3.60 | 3.65 | 3.65 | 3.65 | 3.63 | 3.58 | 3.63 | 3.63 | 3.58 | 3.65 | 3.65 | 3.63 | 3.60 | 3.60 |
| | 11 | 3.89 | 3.89 | 3.70 | 3.84 | 3.89 | 3.86 | 3.86 | 3.86 | 3.89 | 3.89 | 3.89 | 3.89 | 3.89 | 3.89 | 3.89 | 3.86 | 3.86 |
| | 13 | 12.59 | 12.59 | 12.42 | 12.54 | 12.56 | 12.56 | 12.54 | 12.56 | 12.56 | 12.59 | 12.59 | 12.59 | 12.59 | 12.56 | 12.59 | 12.54 | 12.54 |
| | 15 | 17.78 | 17.78 | 17.65 | 17.76 | 17.76 | 17.78 | 17.78 | 17.78 | 17.78 | 17.78 | 17.78 | 17.78 | 17.78 | 17.76 | 17.78 | 17.76 | 17.76 |
| d15112 | 9 | 16.35 | 16.35 | 13.19 | 16.48 | 16.62 | 15.92 | 15.93 | 16.32 | 16.48 | 16.35 | 16.51 | 16.62 | 16.18 | 16.35 | 16.62 | 16.45 | 16.48 |
| | 11 | 19.86 | 19.82 | 18.46 | 19.75 | 19.47 | 19.06 | 19.06 | 19.13 | 19.01 | 19.13 | 19.12 | 19.52 | 18.88 | 19.82 | 19.79 | 19.82 | 19.79 |
| | 13 | 25.72 | 25.72 | 23.96 | 24.82 | 25.38 | 24.12 | 23.96 | 24.20 | 24.27 | 24.21 | 24.27 | 25.44 | 23.96 | 25.53 | 25.72 | 25.31 | 25.05 |
| | 15 | 27.26 | 27.26 | 24.34 | 26.57 | 26.21 | 25.93 | 25.46 | 26.03 | 26.19 | 26.07 | 26.26 | 27.05 | 26.31 | 27.10 | 27.26 | 26.69 | 26.52 |
| d18512 | 9 | 4.54 | 4.54 | 4.45 | 4.52 | 4.52 | 4.50 | 4.52 | 4.52 | 4.50 | 4.52 | 4.50 | 4.50 | 4.52 | 4.50 | 4.54 | 4.54 | 4.54 |
| | 11 | 9.30 | 9.28 | 9.11 | 9.24 | 9.30 | 9.24 | 9.26 | 9.26 | 9.24 | 9.26 | 9.24 | 9.28 | 9.30 | 9.26 | 9.28 | 9.26 | 9.24 |
| | 13 | 10.19 | 10.17 | 10.02 | 10.10 | 10.17 | 10.02 | 10.04 | 10.02 | 10.02 | 10.04 | 10.04 | 10.15 | 10.17 | 10.15 | 10.17 | 10.12 | 10.10 |
| | 15 | 17.52 | 17.50 | 17.40 | 17.44 | 17.52 | 17.08 | 17.25 | 17.29 | 17.21 | 17.21 | 17.21 | 17.50 | 17.50 | 17.44 | 17.50 | 17.50 | 17.42 |
| fnl4461 | 9 | 14.42 | 14.28 | 13.52 | 14.42 | 14.23 | 13.33 | 13.43 | 13.43 | 13.61 | 13.47 | 13.66 | 13.47 | 13.00 | 13.57 | 13.24 | 13.90 | 13.80 |
| | 11 | 19.48 | 19.48 | 19.39 | 19.48 | 19.48 | 18.71 | 19.26 | 18.84 | 18.79 | 18.88 | 18.84 | 19.09 | 18.66 | 19.26 | 18.41 | 19.48 | 19.48 |
| | 13 | 21.33 | 21.33 | 21.22 | 21.33 | 21.10 | 20.79 | 20.29 | 20.06 | 19.75 | 20.37 | 19.98 | 20.06 | 20.64 | 20.91 | 20.45 | 20.99 | 21.10 |
| | 15 | 22.14 | 22.14 | 21.96 | 22.10 | 22.14 | 20.80 | 21.04 | 20.37 | 20.90 | 20.62 | 20.48 | 22.00 | 21.82 | 22.07 | 22.00 | 21.89 | 21.75 |
| | 17 | 24.82 | 24.82 | 24.58 | 24.68 | 24.45 | 24.18 | 24.22 | 23.62 | 23.72 | 23.62 | 23.62 | 24.22 | 24.68 | 24.68 | 24.78 | 24.68 | 24.65 |
| nrw1379 | 9 | 8.53 | 8.53 | 8.46 | 8.53 | 8.49 | 8.53 | 8.38 | 8.35 | 8.27 | 8.35 | 8.27 | 8.27 | 8.53 | 8.53 | 8.20 | 8.49 | 8.53 |
| | 11 | 8.13 | 8.13 | 7.99 | 8.06 | 8.13 | 8.13 | 7.99 | 7.89 | 8.03 | 7.96 | 8.03 | 7.89 | 8.13 | 8.13 | 8.10 | 8.10 | 8.10 |
| | 13 | 17.21 | 17.09 | 16.15 | 16.97 | 16.82 | 17.09 | 16.51 | 16.24 | 16.55 | 16.48 | 16.64 | 16.51 | 17.03 | 17.12 | 16.76 | 17.12 | 17.15 |
| | 15 | 21.19 | 21.19 | 20.70 | 21.04 | 20.76 | 21.13 | 20.56 | 20.70 | 20.76 | 20.76 | 20.99 | 20.79 | 20.99 | 20.90 | 21.07 | 21.10 | 21.07 |
| pr1002 | 9 | 6.52 | 6.52 | 6.47 | 6.72 | 6.71 | 5.58 | 6.21 | 6.30 | 5.45 | 6.43 | 5.58 | 6.67 | 5.59 | 6.79 | 6.83 | 6.70 | 6.64 |
| | 11 | 9.07 | 9.01 | 7.90 | 8.67 | 9.05 | 7.48 | 7.57 | 7.38 | 6.53 | 7.61 | 6.69 | 8.15 | 7.04 | 8.99 | 8.32 | 8.65 | 8.65 |
| | 13 | 11.91 | 11.76 | 9.48 | 11.03 | 10.52 | 10.28 | 10.95 | 10.33 | 9.79 | 10.86 | 9.87 | 11.32 | 9.85 | 11.69 | 11.81 | 10.80 | 10.81 |
| | 15 | 11.42 | 11.40 | 9.69 | 11.01 | 11.27 | 10.02 | 10.42 | 10.42 | 9.64 | 10.33 | 9.79 | 10.84 | 10.10 | 10.80 | 11.30 | 10.99 | 11.01 |
| | 17 | 11.92 | 11.87 | 10.43 | 11.35 | 11.07 | 10.82 | 11.12 | 11.02 | 10.92 | 11.08 | 10.59 | 11.73 | 10.76 | 11.56 | 11.89 | 10.90 | 10.83 |
| | 19 | 12.78 | 12.78 | 10.91 | 12.18 | 12.64 | 11.72 | 12.17 | 11.94 | 11.50 | 12.15 | 11.38 | 12.26 | 11.71 | 12.36 | 12.70 | 11.98 | 11.96 |
| | 21 | 12.44 | 12.14 | 10.13 | 11.86 | 12.19 | 11.41 | 11.28 | 11.28 | 10.50 | 11.31 | 10.54 | 11.64 | 11.00 | 11.79 | 11.87 | 11.91 | 11.92 |
| | 23 | 11.57 | 11.42 | 9.84 | 11.27 | 11.36 | 10.93 | 10.72 | 10.61 | 10.36 | 10.70 | 10.13 | 11.10 | 10.30 | 11.04 | 11.20 | 11.10 | 11.10 |
| | 25 | 13.54 | 13.40 | 11.62 | 12.83 | 13.23 | 12.89 | 12.80 | 12.76 | 12.40 | 12.75 | 11.95 | 13.09 | 12.47 | 13.01 | 13.36 | 12.76 | 12.69 |
| ts225 | 9 | 21.73 | 21.73 | 21.71 | 21.73 | 21.72 | 21.71 | 21.71 | 21.72 | 20.55 | 21.72 | 20.87 | 21.30 | 21.73 | 21.43 | 21.30 | 21.73 | 21.71 |
| | 11 | 21.41 | 21.41 | 20.51 | 21.41 | 21.42 | 21.40 | 21.41 | 21.40 | 20.66 | 21.41 | 20.71 | 20.83 | 21.41 | 21.17 | 20.66 | 21.42 | 21.42 |
| | 13 | 23.72 | 23.72 | 22.69 | 23.66 | 23.62 | 23.68 | 23.72 | 23.62 | 23.60 | 23.67 | 23.68 | 23.60 | 23.72 | 23.72 | 23.64 | 23.66 | 23.66 |
| | 15 | 32.30 | 32.30 | 29.75 | 31.73 | 31.78 | 31.71 | 32.24 | 30.75 | 31.22 | 31.70 | 31.83 | 30.70 | 32.30 | 32.30 | 31.87 | 31.12 | 31.16 |
| Averages | | 16.19 | 16.07 | 14.49 | 15.68 | 15.63 | 15.32 | 15.16 | 15.14 | 15.12 | 15.39 | 15.28 | 15.74 | 15.66 | 15.97 | 15.92 | 15.49 | 15.52 |

Table 2: Integrality gaps for the full formulation minus one family of valid inequalities at a time.

| *Inst.* | *n* | *plain* | *full* | *TSPPD literature* | | | | *New FIFO Inequalities* | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | (9) | (10),(11) | (12), (13) | (14) | (15) | (16) | (17) | (18) | (19) | (20) | (21) | (22) | (23) | (24) | (29) | (30) |
| a280 | 9 | 12.94 | 3.05 | 3.05 | 4.82 | 1.78 | 3.30 | 3.05 | 4.31 | 1.27 | 3.05 | 2.03 | 3.05 | 2.79 | 2.03 | 3.05 | 3.81 | 2.79 | 3.05 |
| | 11 | 18.12 | 4.05 | 4.26 | 8.96 | 3.84 | 3.41 | 4.05 | 7.46 | 4.05 | 3.84 | 3.62 | 4.05 | 3.84 | 4.26 | 4.05 | 4.48 | 4.05 | 4.26 |
| | 13 | 23.18 | 9.53 | 9.53 | 12.71 | 9.53 | 8.79 | 8.41 | 10.84 | 8.22 | 8.22 | 9.53 | 9.53 | 9.53 | 8.60 | 8.22 | 9.53 | 10.84 | 9.72 |
| | 15 | 23.01 | 10.60 | 10.60 | 12.91 | 12.09 | 11.59 | 9.11 | 12.75 | 10.43 | 10.60 | 10.60 | 10.76 | 10.76 | 10.93 | 10.76 | 10.76 | 10.43 | 11.92 |
| | 17 | 20.26 | 10.50 | 10.20 | 12.83 | 10.20 | 9.62 | 9.18 | 11.08 | 9.18 | 9.18 | 10.64 | 10.64 | 10.50 | 9.33 | 10.50 | 10.50 | 10.64 | 10.35 |
| | 19 | 20.33 | 10.03 | 10.03 | 12.09 | 11.81 | 9.75 | 9.48 | 12.36 | 8.79 | 10.16 | 10.03 | 11.26 | 8.65 | 9.34 | 8.79 | 10.03 | 11.95 | 10.44 |
| | 21 | 19.95 | 9.72 | 9.60 | 12.37 | 10.86 | 9.22 | 8.59 | 11.74 | 9.60 | 9.60 | 9.60 | 9.60 | 11.11 | 9.85 | 11.11 | 9.85 | 9.60 | 9.85 |
| | 23 | 20.69 | 10.58 | 10.58 | 11.77 | 10.94 | 10.82 | 9.27 | 11.18 | 9.27 | 9.16 | 10.94 | 10.34 | 9.39 | 9.27 | 11.06 | 10.46 | 10.94 | 10.46 |
| | 25 | 21.51 | 10.47 | 10.59 | 12.39 | 11.60 | 11.26 | 11.26 | 12.05 | 10.70 | 10.59 | 10.36 | 10.47 | 11.82 | 10.59 | 11.49 | 10.36 | 10.36 | 10.47 |
| att532 | 9 | 11.28 | 4.10 | 4.10 | 5.11 | 4.30 | 5.53 | 4.15 | 4.15 | 4.17 | 4.15 | 4.07 | 4.15 | 4.15 | 4.22 | 4.15 | 4.37 | 4.35 | 4.15 |
| | 11 | 9.08 | 4.77 | 4.77 | 6.24 | 4.88 | 4.64 | 4.86 | 4.77 | 4.75 | 4.82 | 4.77 | 4.77 | 4.77 | 4.79 | 4.99 | 5.26 | 4.77 | 4.77 |
| | 13 | 14.36 | 9.68 | 9.68 | 10.78 | 9.16 | 9.39 | 9.43 | 9.96 | 9.61 | 10.50 | 9.23 | 9.94 | 9.39 | 9.71 | 9.80 | 10.00 | 9.78 | 9.50 |
| | 15 | 17.13 | 12.68 | 12.68 | 13.85 | 13.59 | 13.64 | 13.22 | 13.23 | 13.15 | 12.25 | 13.47 | 12.41 | 12.48 | 13.32 | 13.40 | 13.32 | 13.50 | 12.36 |
| | 17 | 16.80 | 13.22 | 13.22 | 13.32 | 12.91 | 13.82 | 13.15 | 13.04 | 13.14 | 13.06 | 13.14 | 13.15 | 12.98 | 13.04 | 13.41 | 13.12 | 13.15 | 13.11 |
| | 19 | 18.51 | 14.39 | 14.39 | 15.21 | 14.52 | 14.87 | 14.81 | 14.66 | 14.64 | 14.63 | 14.39 | 14.39 | 14.90 | 14.64 | 15.38 | 14.39 | 14.51 | 14.45 |
| brd14051 | 9 | 3.65 | 3.35 | 3.31 | 3.37 | 3.35 | 3.33 | 3.35 | 3.33 | 3.33 | 3.33 | 3.35 | 3.33 | 3.35 | 3.35 | 3.33 | 3.51 | 3.31 | 3.35 |
| | 11 | 3.89 | 3.39 | 3.43 | 3.75 | 3.43 | 3.36 | 3.45 | 3.45 | 3.39 | 3.41 | 3.43 | 3.39 | 3.34 | 3.48 | 3.41 | 3.61 | 3.43 | 3.41 |
| | 13 | 12.59 | 11.95 | 11.95 | 12.38 | 11.92 | 11.95 | 12.01 | 11.97 | 12.01 | 11.90 | 11.97 | 11.92 | 11.92 | 11.97 | 11.86 | 12.30 | 11.95 | 11.95 |
| | 15 | 17.78 | 17.23 | 17.40 | 17.57 | 17.40 | 17.44 | 17.23 | 17.46 | 17.25 | 17.40 | 17.40 | 17.40 | 17.25 | 17.28 | 17.27 | 17.46 | 17.27 | 17.40 |
| d15112 | 9 | 16.35 | 11.48 | 11.48 | 14.24 | 11.95 | 12.09 | 11.82 | 11.74 | 11.48 | 11.46 | 11.50 | 11.48 | 11.51 | 11.70 | 11.50 | 11.41 | 11.49 | 11.38 |
| | 11 | 19.86 | 14.91 | 14.91 | 16.59 | 14.77 | 14.85 | 15.06 | 15.77 | 14.68 | 15.48 | 14.89 | 14.77 | 14.88 | 15.02 | 14.93 | 14.95 | 14.86 | 15.09 |
| | 13 | 25.72 | 18.06 | 18.06 | 19.78 | 17.84 | 18.26 | 17.46 | 19.14 | 18.08 | 18.05 | 17.63 | 18.03 | 17.62 | 19.03 | 18.87 | 18.18 | 18.03 | 17.64 |
| | 15 | 27.26 | 19.58 | 19.58 | 20.90 | 19.75 | 20.45 | 19.70 | 20.74 | 19.73 | 19.77 | 19.17 | 19.48 | 19.94 | 20.63 | 20.11 | 19.67 | 19.77 | 19.77 |
| d18512 | 9 | 4.54 | 4.16 | 4.16 | 4.41 | 4.16 | 4.16 | 4.16 | 4.18 | 4.16 | 4.16 | 4.16 | 4.16 | 4.16 | 4.16 | 4.18 | 4.16 | 4.16 | 4.16 |
| | 11 | 9.30 | 8.06 | 8.51 | 9.06 | 8.12 | 8.08 | 8.12 | 8.34 | 8.14 | 7.91 | 8.40 | 8.04 | 7.95 | 8.31 | 8.25 | 8.23 | 7.95 | 8.06 |
| | 13 | 10.19 | 8.18 | 8.20 | 9.53 | 8.41 | 8.39 | 8.45 | 8.73 | 8.20 | 8.39 | 8.39 | 9.04 | 8.39 | 8.90 | 9.11 | 8.52 | 8.41 | 8.39 |
| | 15 | 17.52 | 15.49 | 15.83 | 16.70 | 15.81 | 15.83 | 15.87 | 16.03 | 15.78 | 15.78 | 15.91 | 15.74 | 15.83 | 15.83 | 16.37 | 16.03 | 15.81 | 15.81 |
| fnl4461 | 9 | 14.42 | 8.87 | 8.87 | 11.67 | 9.25 | 8.87 | 10.53 | 9.06 | 10.15 | 8.97 | 10.01 | 8.92 | 9.20 | 9.87 | 10.20 | 9.25 | 9.20 | 8.87 |
| | 11 | 19.48 | 15.75 | 15.71 | 17.21 | 16.31 | 16.27 | 15.67 | 16.40 | 15.71 | 16.05 | 16.22 | 16.01 | 16.27 | 16.48 | 16.18 | 16.78 | 16.44 | 16.01 |
| | 13 | 21.33 | 17.71 | 17.71 | 18.44 | 18.13 | 17.63 | 17.82 | 18.06 | 17.71 | 17.79 | 17.86 | 17.79 | 17.55 | 17.44 | 17.75 | 17.86 | 17.71 | 17.44 |
| | 15 | 22.14 | 19.03 | 19.03 | 19.10 | 18.89 | 18.05 | 19.00 | 19.10 | 18.05 | 18.93 | 19.07 | 19.03 | 19.10 | 18.05 | 19.07 | 18.61 | 19.03 | 18.61 |
| | 17 | 24.82 | 22.29 | 22.32 | 22.32 | 22.29 | 21.92 | 22.39 | 22.32 | 22.29 | 22.25 | 22.29 | 21.29 | 22.32 | 21.59 | 22.09 | 22.32 | 22.29 | 22.29 |
| nrw1379 | 9 | 8.53 | 7.24 | 7.24 | 8.01 | 7.24 | 7.16 | 7.16 | 7.20 | 7.24 | 7.16 | 7.16 | 7.20 | 7.16 | 7.83 | 7.20 | 7.20 | 7.16 | 7.16 |
| | 11 | 8.13 | 6.88 | 7.19 | 7.29 | 6.95 | 7.12 | 6.95 | 6.95 | 6.95 | 6.95 | 6.95 | 7.02 | 6.95 | 7.16 | 6.95 | 7.02 | 6.95 | 6.95 |
| | 13 | 17.21 | 14.39 | 14.42 | 14.75 | 14.48 | 14.51 | 14.51 | 14.36 | 14.45 | 14.39 | 14.51 | 14.36 | 14.36 | 14.36 | 14.39 | 14.21 | 14.51 | 14.51 |
| | 15 | 21.19 | 18.77 | 18.77 | 19.23 | 18.77 | 18.94 | 18.86 | 18.83 | 18.92 | 18.83 | 18.80 | 18.86 | 18.83 | 19.09 | 18.94 | 18.80 | 18.80 | 18.77 |
| pr1002 | 9 | 6.52 | 2.43 | 2.43 | 2.71 | 1.97 | 2.11 | 1.16 | 0.67 | 1.99 | 2.14 | 1.34 | 1.97 | 2.13 | 3.00 | 1.02 | 1.81 | 1.85 | 2.07 |
| | 11 | 9.07 | 2.52 | 2.52 | 4.25 | 2.87 | 2.01 | 2.10 | 2.22 | 1.97 | 2.04 | 2.04 | 2.56 | 2.72 | 3.08 | 2.81 | 2.73 | 2.22 | 2.66 |
| | 13 | 11.91 | 4.50 | 4.50 | 6.45 | 4.11 | 4.68 | 4.43 | 4.64 | 4.44 | 4.19 | 4.51 | 4.32 | 4.10 | 4.83 | 4.57 | 4.25 | 4.47 | 4.45 |
| | 15 | 11.42 | 4.91 | 4.91 | 6.91 | 6.48 | 4.87 | 6.26 | 5.34 | 5.10 | 5.10 | 4.82 | 4.81 | 5.13 | 5.62 | 5.41 | 6.28 | 4.98 | 5.08 |
| | 17 | 11.92 | 6.35 | 6.73 | 6.97 | 6.74 | 6.48 | 6.76 | 6.57 | 6.41 | 6.35 | 6.32 | 6.39 | 6.47 | 6.82 | 6.84 | 7.05 | 6.40 | 6.78 |
| | 19 | 12.78 | 7.02 | 7.02 | 8.22 | 7.15 | 7.22 | 7.01 | 7.18 | 6.99 | 7.09 | 7.03 | 7.00 | 7.04 | 7.34 | 7.18 | 7.45 | 7.02 | 7.01 |
| | 21 | 12.44 | 6.58 | 6.53 | 7.73 | 6.70 | 7.54 | 6.58 | 7.38 | 6.35 | 6.75 | 6.34 | 6.41 | 6.32 | 6.75 | 6.57 | 7.04 | 6.44 | 6.34 |
| | 23 | 11.57 | 6.97 | 7.17 | 7.62 | 6.77 | 6.39 | 7.02 | 7.11 | 6.25 | 7.07 | 6.72 | 6.70 | 6.69 | 7.17 | 6.94 | 7.47 | 6.71 | 6.78 |
| | 25 | 13.54 | 8.51 | 8.36 | 9.26 | 8.46 | 8.63 | 8.47 | 8.62 | 8.44 | 8.53 | 8.47 | 8.36 | 8.37 | 8.95 | 8.51 | 9.12 | 8.37 | 8.39 |
| ts225 | 9 | 21.73 | 18.74 | 18.32 | 19.54 | 18.77 | 17.39 | 17.39 | 18.53 | 17.06 | 18.37 | 17.07 | 18.32 | 17.15 | 17.03 | 17.05 | 18.43 | 18.63 | 18.37 |
| | 11 | 21.41 | 16.72 | 16.06 | 19.26 | 16.72 | 16.44 | 16.04 | 15.93 | 15.56 | 16.01 | 15.53 | 15.89 | 15.94 | 15.63 | 14.43 | 15.81 | 16.06 | 15.41 |
| | 13 | 23.72 | 19.75 | 19.76 | 23.30 | 19.90 | 19.96 | 19.86 | 20.16 | 19.35 | 19.89 | 20.05 | 21.36 | 20.07 | 20.11 | 20.18 | 21.81 | 19.79 | 19.88 |
| | 15 | 32.30 | 25.89 | 25.97 | 28.88 | 26.50 | 26.31 | 26.50 | 25.80 | 26.24 | 28.10 | 25.92 | 25.62 | 27.76 | 26.42 | 26.35 | 25.95 | 26.36 | 27.98 |
| Averages | | 16.19 | 10.92 | 10.93 | 12.30 | 11.11 | 10.99 | 10.84 | 11.36 | 10.71 | 10.89 | 10.85 | 10.93 | 10.92 | 10.98 | 11.02 | 11.13 | 11.01 | 10.96 |

Table 3: Comparisons with exact algorithms from the literature.

| | | | Carrabs et al. | | Erdogan et al. | Branch-and-cut | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| *Inst.* | $n$ | $UB$ | *nodes* | *sec* | *sec* | $LB$ | *%gap* | *sec* | *nodes* | *pool* |
| a280 | 9 | 394 | 16404 | 0.2 | 284.0 | 394 | 0.00 | 1.5 | 11 | 136 |
| | 11 | 469 | 67000 | 1.1 | 5130.0 | 469 | 0.00 | 3.5 | 13 | 202 |
| | 13 | 535 | 1060629 | 25.9 | n.d. | 535 | 0.00 | 17.8 | 224 | 642 |
| | 15 | 604 | 7153930 | 244.0 | n.d. | 604 | 0.00 | 44.6 | 637 | 1144 |
| | 17 | 686 | 18407525 | 897.8 | n.d. | 686 | 0.00 | 109.8 | 903 | 1872 |
| | 19 | 728 | 125794822 | 7863.3 | n.d. | 728 | 0.00 | 164.9 | 934 | 2128 |
| | 21 | 792 | n.d. | n.d. | n.d. | 792 | 0.00 | 425.4 | 2009 | 3430 |
| | 23 | 841 | n.d. | n.d. | n.d. | 841 | 0.00 | 616.3 | 2163 | 3957 |
| | 25 | 888 | n.d. | n.d. | n.d. | 888 | 0.00 | 1396.1 | 3166 | 5631 |
| att532 | 9 | 4050 | 44058 | 0.3 | 1103.1 | 4050 | 0.00 | 3.0 | 29 | 256 |
| | 11 | 4548 | 145537 | 2.0 | 1896.1 | 4548 | 0.00 | 6.4 | 46 | 404 |
| | 13 | 5621 | 4028821 | 88.9 | n.d. | 5621 | 0.00 | 112.4 | 1531 | 3114 |
| | 15 | 5977 | 20321304 | 683.5 | n.d. | 5977 | 0.00 | 1114.6 | 10965 | 11070 |
| | 17 | 6196 | 169472851 | 7313.2 | n.d. | 6196 | 0.00 | 3160.0 | 20578 | 16499 |
| | 19 | 6645 | n.d. | n.d. | n.d. | 6368 | 4.17 | 10800.1 | 42300 | 20000 |
| brd14051 | 9 | 4357 | 941600 | 5.0 | 46.3 | 4357 | 0.00 | 3.3 | 10 | 432 |
| | 11 | 4400 | 5618833 | 52.7 | 652.9 | 4400 | 0.00 | 5.2 | 13 | 503 |
| | 13 | 4847 | 268489184 | 3998.8 | n.d. | 4847 | 0.00 | 1754.2 | 22278 | 12425 |
| | 15 | 5236 | n.d. | n.d. | n.d. | 4951 | 5.44 | 10800.0 | 84153 | 20000 |
| d15112 | 9 | 78016 | 115896 | 1.0 | 6329.9 | 78016 | 0.00 | 25.4 | 947 | 1533 |
| | 11 | 87166 | 1321367 | 18.9 | n.d. | 87166 | 0.00 | 112.2 | 2171 | 3167 |
| | 13 | 97921 | 62744515 | 1259.8 | n.d. | 97921 | 0.00 | 3779.7 | 58948 | 20000 |
| | 15 | 113268 | n.d. | n.d. | n.d. | 106775 | 5.73 | 10800.1 | 101259 | 20000 |
| d18512 | 9 | 4401 | 158974 | 1.2 | 376.6 | 4401 | 0.00 | 3.4 | 58 | 299 |
| | 11 | 4667 | 4941047 | 45.2 | n.d. | 4667 | 0.00 | 45.6 | 2285 | 1540 |
| | 13 | 4721 | 68754702 | 961.2 | n.d. | 4721 | 0.00 | 146.8 | 4589 | 2457 |
| | 15 | 5223 | n.d. | n.d. | n.d. | 5107 | 2.22 | 10800.0 | 145973 | 20000 |
| fnl4461 | 9 | 2108 | 32321 | 0.3 | 354.6 | 2108 | 0.00 | 8.7 | 283 | 505 |
| | 11 | 2336 | 556081 | 7.2 | n.d. | 2336 | 0.00 | 70.2 | 2907 | 1167 |
| | 13 | 2592 | 6198073 | 128.5 | n.d. | 2592 | 0.00 | 704.8 | 41447 | 3034 |
| | 15 | 2837 | 45726987 | 1401.5 | n.d. | 2837 | 0.00 | 1155.6 | 31038 | 3713 |
| | 17 | 3002 | n.d. | n.d. | n.d. | 2924 | 2.60 | 10800.0 | 186089 | 8994 |
| nrw1379 | 9 | 2708 | 1959041 | 11.4 | 603.7 | 2708 | 0.00 | 11.0 | 201 | 727 |
| | 11 | 2865 | 37121719 | 365.4 | n.d. | 2865 | 0.00 | 27.8 | 289 | 1192 |
| | 13 | 3294 | n.d. | n.d. | n.d. | 3294 | 0.00 | 3587.6 | 36598 | 20000 |
| | 15 | 3526 | n.d. | n.d. | n.d. | 3167 | 10.18 | 10800.0 | 70650 | 20000 |
| pr1002 | 9 | 13484 | 9483 | 0.1 | 155.1 | 13484 | 0.00 | 1.6 | 5 | 218 |
| | 11 | 14791 | 85261 | 1.2 | 2857.8 | 14791 | 0.00 | 3.4 | 11 | 212 |
| | 13 | 16478 | 1282958 | 26.0 | n.d. | 16478 | 0.00 | 11.9 | 57 | 554 |
| | 15 | 17209 | 5359043 | 167.0 | n.d. | 17209 | 0.00 | 27.3 | 117 | 843 |
| | 17 | 18683 | 102771998 | 3420.4 | n.d. | 18683 | 0.00 | 86.6 | 767 | 1795 |
| | 19 | 19930 | n.d. | n.d. | n.d. | 19930 | 0.00 | 240.7 | 1221 | 3219 |
| | 21 | 21348 | n.d. | n.d. | n.d. | 21348 | 0.00 | 511.7 | 5445 | 2955 |
| | 23 | 22653 | n.d. | n.d. | n.d. | 22653 | 0.00 | 890.8 | 8838 | 4238 |
| | 25 | 23790 | n.d. | n.d. | n.d. | 23790 | 0.00 | 8386.2 | 66092 | 10922 |
| ts225 | 9 | 23000 | 47732 | 0.4 | 7707.4 | 23000 | 0.00 | 4.6 | 147 | 340 |
| | 11 | 28000 | 318781 | 4.3 | n.d. | 28000 | 0.00 | 9.0 | 138 | 525 |
| | 13 | 38306 | 17849184 | 289.9 | n.d. | 38306 | 0.00 | 357.5 | 14905 | 3154 |
| | 15 | 44398 | n.d. | n.d. | n.d. | 41790 | 5.87 | 10800.0 | 272175 | 20000 |
| Averages | | | | | | 15721 | 0.01 | 2137.7 | 25461 | 5738 |