

The One-commodity Pickup-and-Delivery Traveling Salesman Problem: Inequalities and Algorithms

Hipólito Hernández-Pérez and Juan-José Salazar-González

*DEIOC, Facultad de Matemáticas, Universidad de La Laguna,
38271 La Laguna, Tenerife, Spain,
e-mail: {hhperez,jjsalaza}@ull.es*

Abstract

This article concerns the “One-commodity Pickup-and-Delivery Traveling Salesman Problem” (1-PDTSP), in which a single vehicle of fixed capacity must either pick up or deliver known amounts of a single commodity to a given list of customers. It is assumed that the product collected from the pickup customers can be supplied to the delivery customers, and that the initial load of the vehicle leaving the depot can be any quantity. The problem is to find a minimum-cost sequence of the customers in such a way that the vehicle’s capacity is never exceeded. This article points out a close connection between the 1-PDTSP and the classical “Capacitated Vehicle Routing Problem” (CVRP), and it presents new inequalities for the 1-PDTSP adapted from recent inequalities for the CVRP. These inequalities have been implemented in a branch-and-cut framework to solve to optimality the 1-PDTSP that outperforms a previous algorithm [12]. Larger instances (with up to 100 customers) are now solved to optimality. The classical “Traveling Salesman Problem with Pickups and Deliveries” (TSPPD) is a particular case of the 1-PDTSP, and this observation gives an additional motivation for this article. The here-proposed algorithm for the 1-PDTSP was able to solve to optimality TSPPD instances with up to 260 customers.

Key words: Traveling Salesman; Pickup-and-Delivery; Branch-and-Cut.

1 Introduction

This article presents an exact algorithm to solve the *One-commodity Pickup-and-Delivery Traveling Salesman Problem* (1-PDTSP). One specific city is considered to be the *depot* of a vehicle, while other cities are identified as

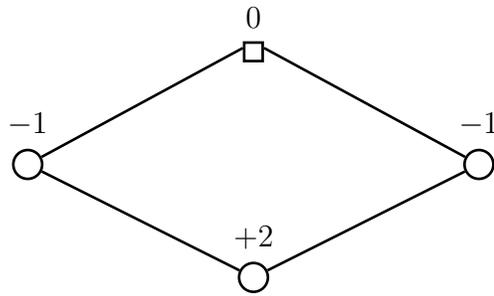


Fig. 1. A 1-PDTSP feasible solution

customers and are divided into two groups according to the type of service required (delivery or pickup). Each *delivery customer* requires a given amount of the product and each *pickup customer* provides a given amount of the product. The product collected from a pickup customer can be supplied to a delivery customer, on the assumption that there is no deterioration in the product. For example, the customers can be branches of a bank in an area providing or requiring a known amount of money (the product), and the depot is the main branch of the bank. Clearly, this is a very simple variant of a more realistic problem where several commodities (e.g., bills and coins) could be considered, but it is still an interesting problem in its own right. It is assumed that the vehicle has a fixed upper-limit *capacity* and must start and end the route at the depot. Also, we allow the initial load of the vehicle leaving from the depot to be any quantity. For example, Figure 1 shows a feasible solution of an instance with two customers requiring one unit, one customer providing two units, and a vehicle with a capacity of two units. The feasibility is based on the assumption that the vehicle leaves out from the depot (the box in the figure) with one extra unit, which will be returned to the depot at the end of the route. See [12] to find approaches for the 1-PDTSP to solve the variant where the vehicle must leave the depot with an empty load.

The 1-PDTSP is that of finding a minimum distance route for the vehicle to satisfy all the customer requirements without ever exceeding the vehicle capacity. The route must be a Hamiltonian tour through all the cities, i.e., each customer must be visited exactly once. The problem is a capacitated variant of the well-known *Traveling Salesman Problem* (TSP), in which a finite set of cities is given, and the travel distance between each pair of cities is assumed to be known.

A real-world application of the 1-PDTSP is shown by Anily and Bramel [2] in the context of inventory reposition. The 1-PDTSP was first introduced in Hernández and Salazar [12] together with an exact approach to solve in-

stances with up to 50 customers. Chalasani and Motwani [8] and Anily and Bramel [2] study the special case of the 1-PDTSP where the delivery and pickup quantities are all equal to one unit, which is known in the literature as the *Capacitated Traveling Salesman Problem*. An important related problem in the literature is the so-called *Traveling Salesman Problem with Pickups and Deliveries* (TSPPD), where two commodities are transported by a capacitated vehicle. See Hernández and Salazar [13] for references to these and other related problems, and for 1-PDTSP heuristic approaches.

This article exploits the relation between the 1-PDTSP and the *Capacitated Vehicle Routing Problem* (CVRP), where a homogeneous capacitated vehicle fleet located in a depot must collect a product from a set of pickup customers. CVRP combines the structure from the TSP and from the *Bin Packing Problem*, two well-known and quite different combinatorial problems. See Toth and Vigo [19] for a recent survey on this routing problem. A previous publication [12] proposed a simple branch-and-cut algorithm based on TSP inequalities and Benders' cuts. The innovative nature of this new article consists of adapting some recent known inequalities for the CVRP to make them valid for the 1-PDTSP and confirming their utility in practice. The multistar inequalities for the CVRP with unit demands were introduced by Araque, Hall and Magnanti [4]. Later, Achuthan, Caccetta and Hill [1], Fisher [9], and Gouveia [11] generalized the multistar inequalities to the CVRP with general demands. Blasum and Hochstättler [7] provide a separation procedure. Letchford, Eglese and Lysgaard [14] give a procedure to derive all known (and some new) multistar inequalities, and some procedures to embed them in a cutting-plane algorithm to solve the CVRP. These results are discussed and adapted to the 1-PDTSP, and computationally evaluated in a branch-and-cut algorithm. The benefit from this adaptation is that larger instances can now be solved to optimality.

Section 2 shows a 0-1 pure Integer Linear Programming (ILP) model for the symmetric 1-PDTSP. Section 3 enumerates new valid inequalities to strengthen the linear programming relaxation of the model. Section 4 describes the heuristic procedures to separate the new inequalities. These inequalities and their separation procedures are inspired by the works [7] and [14] for the CVRP. An extensive computational analysis is presented in Section 5, where 1-PDTSP instances with up to 100 location points and TSPPD instances with up to 261 location points are solved exactly. The article ends with some conclusions and acknowledgements.

2 Problem Formulation

This section summarizes a mathematical formulation of 1-PDTSP taken from Hernández and Salazar [12]. Let us start by introducing some notations. The depot will be denoted by 1 and each customer by i for $i = 2, \dots, n$. For each pair of customers $\{i, j\}$, the travel distance (or cost) c_{ij} of going from i to j is given. We assume $c_{ij} = c_{ji}$. Also given is a demand q_i associated with each customer i , where $q_i < 0$ if i is a delivery customer and $q_i > 0$ if i is a pickup customer. In the case of a customer with zero demand, it is assumed that it is (say) a pickup customer that must also be visited by the vehicle. Without loss of generality, the depot can be considered a customer by defining $q_1 := -\sum_{i=2}^n q_i$. The capacity of the vehicle is represented by Q and it is assumed to be a positive number. Let $V := \{1, 2, \dots, n\}$ be the vertex set and $E := \{[i, j] : i, j \in V\}$ the edge set between all vertices. For each subset $S \subset V$, let $\delta(S) := \{[i, j] \in E : i \in S, j \notin S\}$ and $E(S) := \{[i, j] : i, j \in S\}$. Finally, for $S, T \subset V$ we denote by $E(S : T) := \{[i, j] \in E : i \in S, j \in T\}$, and for $E' \subset E$ we denote by $x(E') := \sum_{e \in E'} x_e$.

The 0-1 ILP model for the 1-PDTSP is based on the following edge-decision variables:

$$x_e := \begin{cases} 1 & \text{if and only if } e \text{ is routed,} \\ 0 & \text{otherwise,} \end{cases}$$

and the formulation is:

$$\min \sum_{e \in E} c_e x_e \tag{1}$$

subject to

$$x(\delta(\{i\})) = 2 \quad \text{for all } i \in V \tag{2}$$

$$x(\delta(S)) \geq 2 \quad \text{for all } S \subset V \tag{3}$$

$$x(\delta(S)) \geq \frac{2}{Q} \left| \sum_{i \in S} q_i \right| \quad \text{for all } S \subset V \tag{4}$$

$$x_e \in \{0, 1\} \quad \text{for all } e \in E. \tag{5}$$

Equations (2) impose that each customer must be visited exactly once. Constraints (3) force the 2-connectivity between customers. Finally, constraints (4) guarantee that the vehicle capacity is never exceeded. We call these inequalities *capacity constraints* because they are equivalent to the known capacity constraints for the CVRP (see, e.g., Toth and Vigo [19]), although in (4) demands q_i are allowed to be negative numbers. The capacity constraints are

obtained through the projection of the flow variables from a one-commodity flow formulation through Benders' decomposition approach (see [12] for details).

In order to simplify the model, we can replace the inequalities (3) and (4) by the following family of inequalities:

$$x(\delta(S)) \geq 2r(S) \quad \text{for all } S \subset V, \quad (6)$$

where

$$r(S) := \max \left\{ 1, \left\lceil \left| \sum_{i \in S} q_i \right| / Q \right\rceil \right\}$$

is a lower bound on the number of times that the vehicle has to go inside/outside the customer set S . These inequalities are called *rounded capacity constraints*. As happens in other routing problems, the degree equations (2) imply that the above inequalities can be rewritten as:

$$x(E(S)) \leq |S| - r(S) \quad \text{for all } S \subset V.$$

3 Further valid inequalities

Model (1)–(5) was used in [12] to propose a simple branch-and-cut algorithm for solving the 1-PDTSP. This section presents new inequalities to strengthen the linear programming relaxation of this model by exploiting the relaxation of the 1-PDTSP with other combinatorial problems.

3.1 Constraints from the Stable Set Problem

The *Stable Set Problem* (SSP) is the problem of selecting a subset of vertices of an undirected graph $G_{\text{SSP}} = (V_{\text{SSP}}, E_{\text{SSP}})$ such that two selected vertices are not connected by an edge of G_{SSP} . This problem is modelled with a 0-1 variable y_i for each vertex i in V_{SSP} and a simple constraint $y_i + y_j \leq 1$ for each edge $[i, j]$ in E_{SSP} . A SSP can be obtained by relaxing the 1-PDTSP to consider only incompatibilities of edges in the graph G due to the load of the vehicle. These incompatibilities lead to inequalities based on the following consideration. If two customers i and j satisfy $|q_i + q_j| > Q$ then $x_{ij} = 0$; otherwise, the vehicle could route the edge $[i, j]$. Nevertheless, if $|q_i + q_j +$

$q_k| > Q$ then the vehicle could route at most one edge in $\{[i, j], [j, k], [i, k]\}$, and this observation generates the mentioned incompatible pairs of edges. A vertex in V_{SSP} corresponds to an edge in E , and an edge in E_{SSP} connects the vertices associated with two incompatible edges in E . Then any valid inequality for the SSP on $G_{\text{SSP}} = (V_{\text{SSP}}, E_{\text{SSP}})$ can be used to strengthen the linear relaxation of 1-PDTSP, and there are some known inequalities for the SSP in the literature (see, e.g., Mannino and Sassano [15]). For example, a large family of known (facet-defining) inequalities for the SSP are called *rank inequalities*, characterized by having all the variable coefficients equal to one. In other words, they have the form $\sum_{i \in T} y_i \leq p$ where T is the vertex set of a subgraph of G_{SSP} with a special structure (a clique, an odd hole, etc.). This large family of inequalities is extended to the 1-PDTSP in the following paragraph.

Incompatibilities between edges can be generalized into incompatibilities between subsets of customers in the original graph G . Indeed, let us consider a collection of subsets of customers in V such that all the customers of each subset can be visited consecutively by the vehicle, but the customers of the union of two non-disjoint subsets cannot be visited consecutively by the vehicle. Let us derive another SSP based on a new graph $G_{\text{SSP}} = (V_{\text{SSP}}, E_{\text{SSP}})$ which considers a collection of customer subsets with the described type of incompatibility. Each vertex in V_{SSP} corresponds to one of these subsets of V . There is an edge in E_{SSP} between two vertices if their associated subsets intersect, thus they cannot be visited consecutively by the vehicle. For simplicity, we assume that the intersection of two subsets is in, at most, one customer. Then, the inequalities valid for the SSP induce inequalities valid for the 1-PDTSP. For example, assume a rank inequality $\sum_{i \in T} y_i \leq p$ for G_{SSP} where $T \subset V_{\text{SSP}}$. Let W_1, \dots, W_m be the customer subsets in V associated with the vertices in T . As observed in Araque, Hall and Magnanti [4] and Naddef and Rinaldi [17], this means that, at most, p subsets of customers can have a *coboundary* that intersects a route on exactly two edges. More precisely, given the rank inequality $\sum_{i \in T} y_i \leq p$ for the SSP in G_{SSP} , at most p subsets W_i associated with vertices in T satisfy $x(\delta(W_i)) = 2$ and the other $m - p$ subsets W_j verify $x(\delta(W_j)) \geq 4$, thus yielding the following inequality for the 1-PDTSP on G :

$$\sum_{i=1}^m x(\delta(W_i)) \geq 4m - 2p.$$

A particular case of a rank inequality is induced by a complete subgraph of

G_{SSP} , where $p = 1$. It is called *clique inequality* in SSP. The above described procedure leads to a valid inequality in the 1-PDTSP called *clique cluster inequality*. To illustrate an inequality in this family, let us consider a customer $v \in V$ and a collection of customer subsets W_1, \dots, W_m such that:

$$\begin{aligned} W_i \cap W_j &= \{v\} && \text{for all } 1 \leq i < j \leq m, \\ r(W_i) &= 1 && \text{for all } i \in \{1, \dots, m\}, \\ r(W_i \cup W_j) &> 1 && \text{for all } 1 \leq i < j \leq m. \end{aligned}$$

Because at most one subset W_i can satisfy $x(\delta(W_i)) = 2$ then a clique cluster inequality is:

$$\sum_{i=1}^m x(\delta(W_i)) \geq 4m - 2.$$

By using the degree equations (2), an equivalent form of a clique cluster inequality is:

$$\sum_{i=1}^m x(E(W_i)) \leq \sum_{i=1}^m |W_i| - 2m + 1. \quad (7)$$

This family of inequalities was first proposed by Augerat [5] and Pochet [18] for the CVRP. They generalize the *star inequalities* introduced by Araque [3] for the special case of the CVRP arising when all demands are identical.

3.2 Constraints from the CVRP

The previous results show a very close connection between the CVRP and 1-PDTSP. In fact, both problems share the property that a subset of customers cannot be visited consecutively by a vehicle when the total demand exceeds the vehicle capacity. For the 1-PDTSP the order of the sequence is relevant, and this is based on the fact that, in contrast to the CVRP, in the 1-PDTSP $S' \subset S \subset V$ does not imply $r(S') \leq r(S)$. We take advantage of this relation in the rest of this section.

Another known family of inequalities for the CVRP is the *multistar inequalities*. Recent articles deal with the multistar inequalities for the CVRP with general demands (see, e.g., [7], [9], [11] and [14]). They can be adapted to the 1-PDTSP by considering the two signs of the demands. The customer demands in the 1-PDTSP differ from the CVRP in the fact that they can decrease the load of the vehicle. For that reason we have two types of inequalities. The

positive multistar inequality is:

$$x(\delta(N)) \geq \frac{2}{Q} \sum_{i \in N} \left(q_i + \sum_{j \in S} q_j x_{[i,j]} \right)$$

for all $N \subset V$ and $S = \{j \in V \setminus N : q_j > 0\}$. The *negative multistar inequality* is:

$$x(\delta(N)) \geq \frac{2}{Q} \sum_{i \in N} \left(-q_i - \sum_{j \in S} q_j x_{[i,j]} \right)$$

for all $N \subset V$ and $S = \{j \in V \setminus N : q_j < 0\}$. The set of customers outside N is reduced to S to strengthen the constraints and this is a fundamental difference with respect to the CVRP multistar inequalities. The two families of linear inequalities can be represented (for simplicity of notation) by the non-linear inequalities:

$$x(\delta(N)) \geq \frac{2}{Q} \left| \sum_{i \in N} \left(q_i + \sum_{j \in S} q_j x_{[i,j]} \right) \right|, \quad (8)$$

for all $N \subset V$ and $S = \{j \in V \setminus N : q_j > 0\}$ or $S = \{j \in V \setminus N : q_j < 0\}$.

These constraints are called *inhomogeneous multistar inequalities*, for analogy with the inequalities with the same name presented in [14] for the CVRP. The validity of these inequalities for the 1-PDTSP follows from the fact that the vehicle capacity should allow the loading of the demand of the customers in N , plus the demand of the customers outside N immediately visited after a customer in N . See Gouveia [11] for how the inequalities can be obtained from a one commodity flow formulation for the CVRP. The vertex set N is called the *nucleus* and the vertices in S *satellites*. Blasum and Hochstättler [7] prove that this family of inequalities can be separated in polynomial time for the CVRP. Unfortunately, the same procedure cannot be easily adapted to separate (8) in polynomial time for the 1-PDTSP when there is a customer i with $|q_i| > Q/2$. The fact that S is a subset of $V \setminus N$ added some difficulties to our research into trying to separate the inhomogeneous multistar inequalities for the 1-PDTSP as done by Gouveia [11] for the CVRP. Still, Section 4 presents a heuristic approach to find constraints (8) violated by a given solution x^* .

Blasum and Hochstättler [7] present a generalization of this family of inequalities for the CVRP that can also be adapted to the 1-PDTSP. It arises when the vertices in S are replaced by subsets of vertices. Let $\{S_1, \dots, S_m\}$ be a collection of subsets in $V \setminus N$ such that $\sum_{i \in N} q_i, \sum_{i \in S_1} q_i, \dots, \sum_{i \in S_m} q_i$ have

all the same sign. Then, a *generalized inhomogeneous multistar inequality* is:

$$x(\delta(N)) \geq \frac{2}{Q} \left| \sum_{i \in N} q_i + \sum_{j=1}^m \left(\sum_{i \in S_j} q_i \right) \left(x(E(N : S_j)) + 2 - x(\delta(S_j)) \right) \right|, \quad (9)$$

and it is valid for the 1-PDTSP. The validity is based on considering each S_j as a single vertex with demand $\sum_{i \in S_j} q_i$ when $x(\delta(S_j)) = 2$. More precisely, if the vehicle goes directly from N to S_j (i.e., $x(E(N : S_j)) \geq 1$) and visits all customers in S_j in a sequence (i.e., $x(\delta(S_j)) = 2$) then the load must allow the serving of $\sum_{i \in S_j} q_i$ additional units. See [7] for detailed proof of the validity of the generalized inhomogeneous multistar inequalities for the CVRP.

Observe that all these inequalities can be written in other forms using the degree equations (2). For example, the above inequality (9) can be written as:

$$x(E(N)) \leq |N| - \left| \frac{\sum_{i \in N} q_i}{Q} + \sum_{j=1}^m \frac{\sum_{i \in S_j} q_i}{Q} \left(x(E(N : S_j)) - |S_j| + 1 + x(E(S_j)) \right) \right|. \quad (10)$$

Araque, Hall and Magnanti [4] propose three classes of multistar inequalities for the CVRP with unit demands. Based on this work Letchford, Eglese and Lysgaard [14] propose another generalization of the multistar inequalities for the CVRP with general demands. Once again, by considering the fact that customer demands can be negative, this generalization is adapted to the 1-PDTSP as follows. Let us consider the subsets $N \subset V$, $C \subseteq N$ and $S \subseteq V \setminus N$. The subset N is called *nucleus*, C *connector* and S *set of satellites*. Then the inequality

$$\lambda x(E(N)) + x(E(C : S)) \leq \mu \quad (11)$$

is valid for appropriate values of λ and μ . These constraints are called *homogenous partial multistar inequalities* when $C \neq N$ and more simply called *homogenous multistar inequalities* when $C = N$.

For a fixed N , S , and C , the set of all valid homogeneous partial multistar inequalities (i.e., the set of pairs (λ, μ) for which (11) is a valid inequality) can be found by projecting the 1-PDTSP polyhedron onto a planar subspace having $x(E(N))$ and $x(E(C : S))$ as axes. Computing this projection in practice is a difficult problem. However, it is not so difficult to compute upper bounds of $x(E(C : S))$ and $x(E(N))$ for solutions of the 1-PDTSP, and to use these results to get valid inequalities for the 1-PDTSP. The procedure is derived from a detailed work in Letchford, Eglese and Lysgaard [14] for the

CVRP. We next compute some upper bounds for $x(E(C : S))$ and $x(E(N))$ which will be used later for generating a particular family of inequalities (11). However, this adaptation is not so immediate for the 1-PDTSP since the order of customers in a set creates difficulties when computing the bounds due to the different signs.

Let $N \subset V$, $C \subseteq N$ and $S \subseteq V \setminus N$. Then all feasible 1-PDTSP solutions satisfy:

$$x(E(C : S)) \leq \min\{2|C|, 2|S|, |C| + |S| - r(C \cup S)\}. \quad (12)$$

When $C = N$ and by considering a feasible 1-PDTSP solution x such that $x(E(C : S)) = \alpha$ for a parameter α , an upper bound for $x(E(N))$ is:

$$x(E(N)) \leq \begin{cases} |N| - r(N) & \text{if } \alpha = 0, \\ |N| - \min\{r(N \cup S') : S' \subset S, |S'| = \alpha\} & \text{if } 1 \leq \alpha \leq |S|, \\ |N \cup S| - r(N \cup S) - \alpha & \text{if } \alpha > |S|. \end{cases} \quad (13)$$

This bound is derived from the same reasoning of the inhomogeneous multistar inequality (8): a lower bound for $x(\delta(N))$ is twice the required capacity divided by Q , and this required capacity can consider some customers in S .

At first glance, the calculation of $\min\{r(N \cup S') : S' \subset S, |S'| = \alpha\}$ is a difficult task, but in practice it is a simple problem in some particular situations. An example of these situations arises when $\sum_{i \in N} q_i \geq -Q$ and $q_j > 0$ for all $j \in S$ or when $\sum_{i \in N} q_i \leq Q$ and $q_j < 0$ for all $j \in S$. In this case, the minimum is obtained when S' is equal to the α customers in S with the smallest absolute value demands.

Also, when $C \neq N$ and $|S| > |C|$ the following bound can be useful for $x(E(N))$. For a fixed value of $\alpha = |C|, \dots, 2|C|$ the inequality

$$x(E(N)) \leq |N| + |C| - \alpha - \min \left\{ r((N \setminus C) \cup C' \cup S') : \begin{array}{l} C' \subset C, S' \subset S, \\ |C'| = |S'| = 2|C| - \alpha \end{array} \right\} \quad (14)$$

is valid for all solutions x such that $\alpha = x(E(C : S))$. Equations (13) and (14) are different from the ones given in [14] due to the existence of negative and positive demands. However, the technical proof of the (13) and (14) for the 1-PDTSP can be adapted from the one given for the CVRP in [14].

As carried out in the case $C = N$, for the case where $C \neq N$ it is also possible

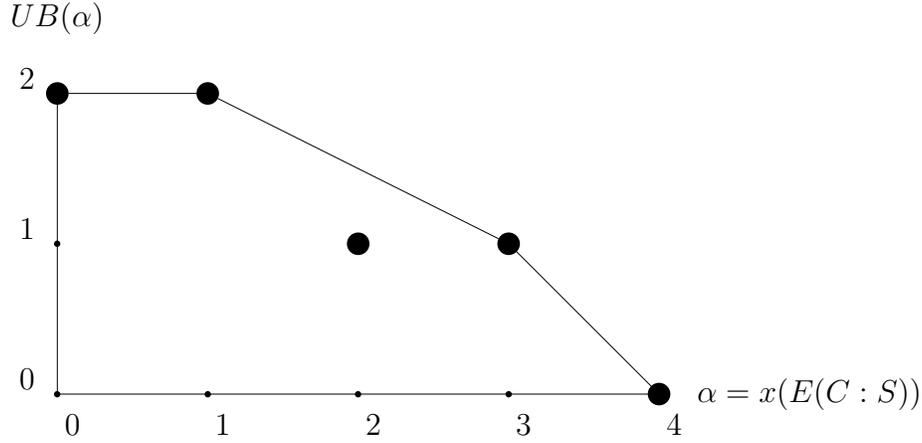


Fig. 2. Polygon

to find subsets N , C , S such that the right-hand side can be easily calculated. A procedure to do this is described in Section 4.

The above upper bounds provide a way of generating 1-PDTSP homogeneous partial multistar inequalities for a fixed N , C and S as done in [14] for the CVRP. Let UB_{CS} be the right-hand side in (12), which is an upper bound for $x(E(C : S))$. Using (13) and (14), for $\alpha = 0, \dots, UB_{CS}$ we compute an upper bound $UB(\alpha)$ for the value of $x(E(N))$ given that $x(E(C : S)) = \alpha$. Once $UB(\alpha)$ have been calculated, the inequalities can be easily found by computing the convex hull of a polygon in $(\alpha, UB(\alpha))$ -space. Finally, the inequalities obtained by the projection are transferred onto the 1-PDTSP polyhedron.

To illustrate this procedure let us consider an instance with capacity $Q = 10$ and at least six customers with demands $q_1 = +1$, $q_2 = -6$, $q_3 = +1$, $q_4 = -3$, $q_5 = -4$ and $q_6 = -4$, respectively. Let us consider $N = C = \{1, 2, 3\}$ and $S = \{4, 5, 6\}$. Then, from (12), we obtain $UB_{CS} = 4$. Using (13), the values $UB(\alpha)$ for $\alpha = 0, \dots, 4$ are calculated: $UB(0) = 2$, $UB(1) = 2$, $UB(2) = 1$, $UB(3) = 1$ and $UB(4) = 0$. Figure 2 plots these points. Apart from the trivial inequalities $x(E(N : S)) \geq 0$ and $x(E(N)) \geq 0$, we obtain another three inequalities: $x(E(N)) \leq 2$ is equivalent to the inequality (6) for N , $2x(E(N)) + x(E(N : S)) \leq 5$ is a homogeneous multistar inequality (11) and $x(E(N)) + x(E(N : S)) \leq 4$ is dominated by the inequality (6) for $N \cup S$.

The homogeneous multistar inequality (11) for $C = N$ and the homogeneous partial multistar inequality (11) for $C \neq N$ can be generalized when the

vertices in S are replaced by a collection of subsets $\{S_1, \dots, S_m\}$. As far we know, this extension of homogeneous multistar inequalities has not been done for the CVRP. Then, the equation (11) can be rewritten as:

$$\lambda x(E(N)) + \sum_{i=1}^m \left(x(E(C : S_i)) + x(E(S_i)) \right) \leq \mu + \sum_{i=1}^m (|S_i| - 1). \quad (15)$$

We call constraints (15) *generalized homogeneous multistar inequality* if $C = N$ and *generalized homogeneous partial multistar inequality* if $C \neq N$. This generalization allows the further strengthening of the linear programming relaxation of the model. As a consequence, a clique cluster constraint (7) can be considered a generalized homogeneous multistar inequality where the connector and the nucleus are a single vertex $C = N = \{v\}$, and the satellites are the subsets $S_i = W_i \setminus \{v\}$ for $i = 1, \dots, m$ (thus $\mu = 1$). Nevertheless, the clique cluster inequalities cannot be obtained by the above described procedure for the homogeneous multistar inequalities because $x(E(C : S)) \leq 1$ and $x(E(N)) = 0$, and therefore the projection procedure is useless.

4 Separation procedures

Given a solution x^* of a linear relaxation of the model (1)–(5) and considering the family of constraints \mathcal{F} valid for the integer solutions of the model (1)–(5), a *separation problem* for \mathcal{F} and x^* is the problem of proving that all constraints in \mathcal{F} are satisfied by x^* , or of finding a constraint in \mathcal{F} violated by x^* . The scope of this section is to develop procedures to solve the separation problems associated with the classes of constraints introduced in the previous section. Since these problems are difficult, we describe mainly heuristic procedures. To this aim, we introduce a capacitated graph $G^* := (V^*, E^*)$ built from the fractional solution x^* , where $V^* := V$ is the vertex set, $E^* = \{e \in E : x_e^* > 0\}$ is the edge set and x_e^* is the capacity of each $e \in E^*$.

4.1 Rounded capacity constraints

For the separation problem of the rounded capacity constraints (6) we use the techniques described in [12]. They are based on greedy procedures to construct subsets S which are candidates for defining violated constraints, and interchange approaches to shake up these subsets.

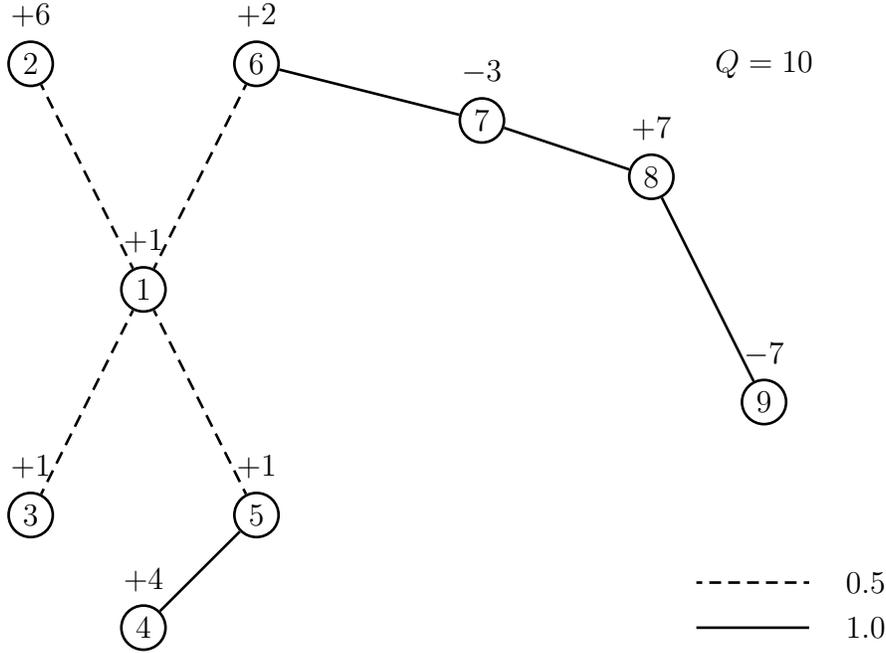


Fig. 3. A piece of a fractional solution: clique cluster inequality violated

4.2 Clique cluster inequalities

The separation algorithm for the clique cluster inequalities (7) looks for sets W_i that could generate a constraint (7) violated by x^* . To detect sets W_i , unlike in the CVRP case, the graph G^* should not be shrunk. Indeed, shrinking G^* could lead to the loss of good candidates because the demands of the customers are positive and negative, and various vertices of G^* would be shrunk to a vertex with a smaller absolute value demand than a non-shrunk vertex. In our implementation, for each vertex $v \in V$, a greedy heuristic searches all paths over the capacitated graph G^* such that they start with a vertex in $\{i \in V \setminus \{v\} : x_{[v,i]}^* > 0\}$ and the following vertices are joined by edges e with $x_e^* = 1$. We study two cases depending on whether $r(W_i \cup W_j) > 1$ for $i \neq j$ is due to $\sum_{k \in W_i \cup W_j} q_k > Q$ or due to $\sum_{k \in W_i \cup W_j} q_k < -Q$. The last vertices in a path are removed if they do not increment the sum of the demands in the first case or they do not decrement the sum of the demands in the second case. The vertices in each path and the vertex v form a possible set W_i . If two sets W_i and W_j satisfy $r(W_i \cup W_j) \leq 1$ then the one with smallest absolute total demand is removed from the list of candidate sets.

We now illustrate this procedure on the piece of a fractional solution shown

in Figure 3. For the vertex $v = 1$, the greedy heuristic obtains the paths $\{2\}$, $\{3\}$, $\{5, 4\}$ and $\{6, 7, 8, 9\}$. Searching for a clique cluster inequality with $\sum_{k \in W_i \cup W_j} q_k > Q$, the algorithm removes the vertex 9 in the last path (observe that $q_6 = +2$, $q_6 + q_7 = -1$, $q_6 + q_7 + q_8 = +6$ and $q_6 + q_7 + q_8 + q_9 = -1$ and the maximum is obtained when vertices 6, 7, and 8 are included). The possible sets are composed by the vertices in each path and the vertex 1, i.e., $W_1 = \{1, 2\}$, $W_2 = \{1, 6, 7, 8\}$, $W_3 = \{1, 2\}$ and $W_4 = \{1, 3\}$ with demands $+7$, $+6$, $+6$ and $+2$, respectively. These four sets do not verify $r(W_i \cup W_j) > 1$ for $i \neq j$. When W_4 is eliminated, subsets W_1 , W_2 and W_3 verify this condition. In fact, the fractional solution x^* violates the clique cluster inequality (7) defined by W_1 , W_2 and W_3 .

4.3 Generalized homogeneous multistar inequalities

The separation algorithm for the generalized homogeneous and generalized partial homogenous multistar inequalities (15) is based on greedy heuristics to find possible candidates for the nucleus N , connector C and satellites $\{S_1, \dots, S_m\}$. For each N , C and $\{S_1, \dots, S_m\}$, we compute $UB(\alpha)$ as described on the right-hand side of (13) and (14) for different values of α limited by the right-hand side of (12). The upper frontier of the convex hull of $(\alpha, UB(\alpha))$ induces valid inequalities (15), which are checked for violation of x^* . We next describe how to generate these subsets.

Given a fractional solution x^* , the candidates for nucleus sets are saved in a list. This list contains sets N with $2 \leq |N| \leq n/2$. In particular, it contains n subsets of cardinality two, $N = \{i, j\}$, such that $x^*_{[i,j]}$ is as large as possible. Each set of cardinality $k \geq 3$ is generated from a set N of cardinality $k - 1$ in the list, inserting a vertex j verifying that $j \in V \setminus N$, $x^*(E(N : \{j\})) > 0$ and $N \cup \{j\}$ is not already in the nucleus list. If there are various vertices j verifying these conditions, the one which maximizes $x^*(E(N : \{j\}))$ is chosen.

When the nucleus N has been selected from the nucleus list, to detect the satellite subsets S_i for the generalized homogeneous multistar inequalities (15) with $C = N$, we use the same greedy heuristic used to detect the sets W_i in the clique cluster inequalities (7). More precisely, the greedy heuristic searches for all paths over the capacitated graph G^* such that they start in a vertex in $\{i \in V \setminus N : x^*(N : \{i\}) > 0\}$ and the following vertices are joined by an edge e with capacity $x^*_e = 1$. Each path induces a possible satellite set. If $\sum_{i \in N} q_i >$

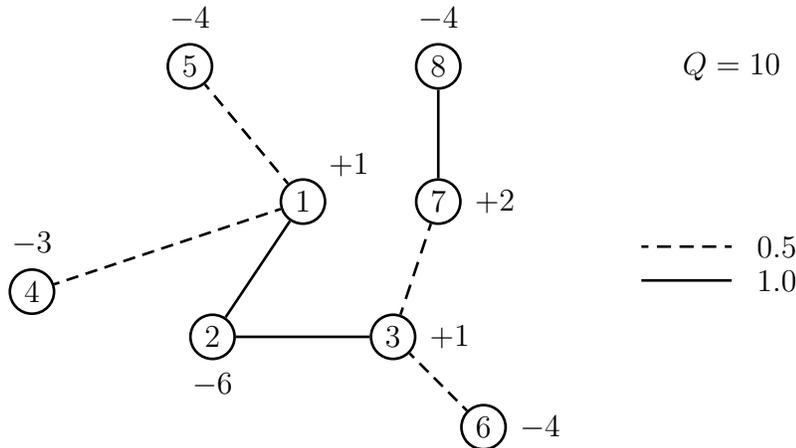


Fig. 4. A piece of a fractional solution

Q , the procedure searches for satellites with positive demands; if $\sum_{i \in N} q_i < -Q$, it searches for satellites with negative demands; otherwise, it executes two rounds, one searching for satellites with positive demands and another searching for satellites with negative demands. All the valid inequalities (15) with $C = N$ obtained by considering the projection from $\{N, S_1, \dots, S_m\}$ are checked for the violation of x^* . We also check the inequalities (15) with $C = N$ obtained by considering the projection from $\{N, S_1, \dots, S_m\} \setminus \{S_j\}$ where S_j is the satellite set with the smallest absolute value demand. This procedure is iteratively repeated removing a satellite set at a time until there are two satellite sets (i.e., $m = 2$).

To illustrate this procedure, let us consider the piece of a fractional solution shown in Figure 4. When $N = \{1, 2, 3\}$ is selected from the nucleus list, the greedy heuristic returns only the satellite set $S_1 = \{7\}$ if it is searching satellite sets with positive demands. Searching for satellite sets with negative demands, it obtains the satellites $S_1 = \{5\}$, $S_2 = \{6\}$, $S_3 = \{4\}$ and $S_4 = \{7, 8\}$ with associated demands -4 , -4 , -3 and -2 , respectively. The polygon associated with N , S_1 , S_2 , S_3 and S_4 in the representation of $(\alpha, UB(\alpha))$ is the square of vertices $(0, 0)$, $(0, 2)$, $(2, 2)$ and $(2, 0)$. This polygon does not induce a generalized homogeneous multistar inequality. Then the satellite of smallest absolute value demand S_4 is removed. The polygon associated with N , S_1 , S_2 and S_3 was shown in Figure 2 and yields a violated generalized homogeneous multistar inequality (15) with $C = N$. This inequality is (15) with $\lambda = 2$ and $\mu = 5$.

4.4 Generalized partial homogeneous multistar inequalities

The separation procedure for generalized homogeneous partial multistar inequalities (15) with $C \neq N$ starts only if no homogeneous multistar inequality was found. It takes the same list of nucleus sets. For each nucleus N and each cardinality from 1 to the maximum of 4 and $|N| - 1$, a connector C is built such that $x^*(C : V \setminus N)$ is as great as possible. Then, the right-hand sides of (13) and (14) define a planar polygon and its upper frontier is used to calculate valid inequalities. For a given N and C , the satellites S_i are computed using the same procedure described for the inequality (15) with $C = N$. To calculate the right hand side of (14) easily, we search for satellites with positive demands if $\sum_{i \in N \setminus C} q_i + \sum_{i \in C, q_i < 0} \geq -Q$. In this way, we guarantee that the minimum is obtained for C' equal to the $2|C| - \alpha$ vertices in C with the smallest demands, and for S' equal to the $2|C| - \alpha$ satellite sets with the smallest demands. If $\sum_{i \in N \setminus C} q_i + \sum_{i \in C, q_i > 0} \leq Q$, we search for satellites with negative demands. Finally, the induced inequalities are checked for violation of the fractional solution x^* .

4.5 Generalized inhomogeneous multistar inequalities

The separation algorithm for the generalized inhomogeneous multistar inequalities (10) uses the same list of nucleus sets described above and the same satellite constructor. As can be observed from (10), the inequality is stronger if we consider *all* candidate satellites. For example, Figure 4 also shows a piece of a fractional solution violating the generalized inhomogeneous multistar inequality defined over $N = \{1, 2, 3\}$, $S_1 = \{5\}$, $S_2 = \{6\}$, $S_3 = \{4\}$ and $S_4 = \{7, 8\}$. However, based on our computational experiments, we decided to call on this separation algorithm only when the other separation algorithms did not succeed in finding a violated constraint.

5 Computational Results

The separation procedures described in Section 4 have been implemented in the branch-and-cut framework of CPLEX 7.0, together with an LP-based heuristic and other classical ingredients present in this type of approach. See

[13] for details on the heuristic approach. The new implementation differs from [12] because we now consider the families of inequalities (7), (10) and (15). The whole algorithm ran on a personal computer with an AMD Athlon XP 2600+ (2.08 Ghz.) processor.

We tested the performance of the 1-PDTSP approaches on three classes of instances. Briefly, the first class is generated using the random generator proposed in Mosheiov [16] for the TSPPD. The second class contains the CVRP instances from the TSPLIB transformed into 1-PDTSP instances. The third class consists of the instances described by Gendreau, Laporte and Vigo in [10] for the TSPPD. Tables 1 to 3 concern the results in the first class, Table 4 the second class and Tables 5 to 7 the third class. This section describes the instances and analyzes the results for each instance class. The column headings of Table 2 will follow later. The column heading of Tables 1, 3, 4 and 5 have the following meanings:

- n*: the number of vertices;
- Q*: the capacity of the vehicle. Word *pd* means that it is a TSPPD instance with the tightest capacity;
- Cuts*: the numbers of violated inequalities found by the separation procedures;
 - Cap.* refers to rounded capacity constraints (6), *cliq.* refers to clique clusters inequalities (7), *hm* refers to generalized homogeneous multistar inequalities (15) with $C = N$, *hpm* refers to generalized homogeneous partial multistar inequalities (15) with $C \neq N$, and *im* refers to generalized inhomogeneous multistar inequalities (10);
- LB/Opt.*: the percentage of the lower bound at the root node over the optimal solution value when the different kinds of inequalities are inserted;
- UB/Opt.*: the percentage of the initial heuristic value over the optimal solution value;
- Opt.*: the optimal solution value;
- Opt./TSP*: the percentage of the optimal solution value over the TSP optimal solution value;
- B&C*: the number of explored branch-and-cut nodes;
- root*: the computational time at the end of root branch-and-cut node;
- Time*: the total computational time;
- t.l.*: the number of instances not solved within the time limit (2 hours).

Each row of all the tables, except Tables 1, 4 and 5, shows the average results of ten randomly-generated instances. A row in Tables 1 and 4 corresponds to one instance, while a row in Table 5 corresponds to the average results on 26

instances.

Let us describe the instances in our first class. To observe the main differences between the 1-PDTSP and the TSPPD, we have considered Problem 1 of Mosheiov [16]. Note that the definition of the 1-PDTSP does not require the vehicle to leave (or enter) the depot with a pre-specified load (e.g., an empty or full initial load). In fact, it is assumed that the depot can supply the vehicle with any extra initial load, which is an unknown value in the problem. Nevertheless, it is easy to fix the initial load of the vehicle, and in fact we have also considered a restricted version of the 1-PDTSP where the vehicle is required to leave the depot with a load equal to q_1 if $q_1 > 0$, or with a load equal to 0 if $q_1 \leq 0$. To solve this restricted problem with an algorithm for the unrestricted 1-PDTSP, the depot is split into two dummy customers with demands $+Q$ and $q_1 - Q$ if $q_1 > 0$, and with demands $-Q$ and $Q + q_1$ if $q_1 \leq 0$. The edge e between the pickup dummy vertex and the delivery dummy vertex must be routed by the vehicle (i.e. $x_e = 1$).

Table 1 shows the results of three kinds of instances: 1-PDTSP instances with capacity in $\{7, \dots, 16\}$, the restricted 1-PDTSP instances with capacity in $\{7, \dots, 20\}$, and TSPPD instances with capacity in $\{45, \dots, 50\}$. The features of the problems confirm that the optimal values of a 1-PDTSP instance without the initial load requirement and of a TSPPD instance coincide with the TSP value when the capacity Q is large enough. The optimal TSP value for this instance is 4431. However, the optimal route of the restricted 1-PDTSP with a sufficiently large capacity ($Q \geq 20$) coincides with the optimal solution of TSPPD with the smallest capacity ($Q = 45$). What is more, the additional initial load requirement on the 1-PDTSP solutions implies larger optimal values for the same capacities but not a more difficult problem to solve. On this data, only the separation procedure for the generalized homogeneous multi-star inequalities (15) with $C = N$ helped to improve the lower bound when Q was small. In all cases, the instances were solved to optimality.

The generator of random instances in the first class produces $n - 1$ random pairs in the square $[-500, 500] \times [-500, 500]$, each one corresponding to the location of a customer with an integer demand randomly generated in $[-10, 10]$. The depot was located at $(0,0)$ with demand $q_1 := -\sum_{i=2}^n q_i$. When $q_1 \notin [-10, 10]$, the customer demands were regenerated. The travel cost c_{ij} was computed as the Euclidean distance between the locations i and j . We generated a group of ten random instances for each value of n in $\{30, 40, \dots, 100\}$

and for each different value of Q . The solution of the 1-PDTSP instances with $Q = 100$ (i.e., when Q is large enough) coincided with an optimal TSP solution, and $Q = 10$ is the smallest capacity possible because for each n there was a customer in at least one instance of each group with demand $+10$ or -10 . In addition, we have also used this generator for creating the TSPPD instances mentioned in [16] in which Q is the smallest possible capacity (i.e., $Q = \max\{\sum_{q_i > 0, i \neq 1} q_i, -\sum_{q_i < 0, i \neq 1} q_i\}$). These instances are referred in the table as *pd* in the Q column.

Table 2 shows the results (branch-and-cut nodes and computational time) for small instances ($n \leq 70$) with five different strategies of separation:

- capacity*: when only rounded capacity constraints (6) are separated;
- cliques*: when rounded capacity constraints (6) and clique clusters inequalities (7) are separated;
- root MS*: when all families of multistar inequalities (15) and (10) are also separated (i.e., generalized homogeneous, generalized partial homogeneous and generalized inhomogeneous multistar), but only while exploring the root node;
- 150 MS*: when all families of multistar inequalities are separated in the 150 first branch-and-cut nodes;
- all MS*: when all families of multistar inequalities are separated at all nodes.

The first two columns are the size n of the problem and the capacity Q of the vehicle, and the following ten columns are the number of branch-and-cut nodes (*BC*) and computational time (*Time*) for each strategy. From this table, we confirm that the number of branch-and-cut nodes decreases when more cuts are inserted, which does not always lead to a reduction in the computational time. The insertion of clique cluster inequalities improves the performance of the algorithm. The insertion of all possible multistar inequalities in all branch-and-cut nodes does not lead to a better performance in general. Nevertheless, the insertion of some multistar inequalities in the first branch-and-cut nodes is better than the non-insertion of them when solving the hardest instances (i.e., with a larger number of customers and tighter capacities). This is the reason why we decided to include the multistar inequalities in the first 150 branch-and-cut nodes in the final exact algorithm. Other strategies were tested; for example, the insertion of generalized homogeneous multistar inequalities excluding generalized homogeneous partial multistar and generalized inhomogeneous multistar inequalities. These experiments showed similar computational

results. For the sake of completeness, the separations of all possible multistar inequalities are retained in the final exact algorithm.

Table 3 shows exhaustive results of the branch-and-cut algorithm on larger random instances in the first class. The average results are computed only for the instances solved before the time limit. When the size n is greater, the tighter capacitated instances are not considered in the table because almost all instances remain unsolved within the time limit. The amount of time consumed for solving the root node was very small compared with the total time (never more than 10 seconds). On average, the new inequalities increase the lower bound by about 1.5% when $Q = 10$, so it goes up to 3% under the optimal value at the root node. The heuristic procedure also provides a feasible solution which is 3% over the optimal value. The optimal value of these instances with $Q = 10$ is about 50% over the optimal TSP value, which partially explains the difficulty of these instances for our branch-and-cut algorithm. From these results, the `pd` instances (i.e., the TSPPD instances used in Mosheiov [16]) are observed to be much more easily solved by our algorithm than other 1-PDTSP instances with a small capacity.

We now describe the instances in our second class. This class is derived from the CVRP instances in the TSPLIB library. The demands of the even customers remain as they are in the CVRP instances while the demands of odd customers are changed to be negative numbers. The depot (customer 1) has demand $q_1 = -\sum_{i=2}^n q_i$. Table 4 shows the results of the exact algorithm for the second class of instances. The instances of size less than 50 are solved in less than one second and therefore are not shown in the table. The table shows the results solving the instances of size greater than 50 (i.e., three collections: `ei151`, `ei176` and `ei1101`) with different capacities. The first of each collection of instances corresponds to the tightest possible capacity (it is equal to the demand of customer 19 in `ei151` and equal to the depot demand in `ei176` and `ei1101`). The last of each collection corresponds to the instance with the minimum capacity which has an optimal solution coinciding with a TSP solution. The overall performance of the algorithm on these instances is similar to that observed in previous tables. The important observation from this table is that our algorithm succeeded in solving all the instances to optimality due to the new families of constraints introduced in this article. These results are not possible with the approach in [12].

Finally, we conducted experiments solving the same TSPPD instances used

by Gendreau, Laporte and Vigo [10] and by Baldacci, Hadjiconstantinou and Mingozzi [6], which are the instances in our third class. They are grouped into three collections (derived from the VRP, random Euclidean TSPPD and random symmetric TSPPD). The first collection contains 26 TSPPD instances generated for each value of an input parameter $\beta \in \{0, 0.05, 0.1, 0.2\}$, and the largest instance has 261 location points. The instances in the second and third collections are also grouped according to $\beta \in \{0, 0.05, 0.1, 0.2\}$, and the largest TSPPD instance has 200 location points. See [6,10] for details on how these instances are generated, and in particular for the meaning of β , an input parameter for the random generator to define the largest possible customer demand. Tables 5, 6 and 7 show the average results when solving the instances of each collection, respectively, with our 1-PDTSP approach. An important observation is that all the instances are solved to optimality by our approach while only some heuristic solutions are computed in [10]. It is interesting to note that these random instances are much easier to solve than the instances considered in Table 3 and 4 because the optimal TSPPD value of these instances is quite close to the optimal TSP value. To compare the results of our experiments with the ones reported in Baldacci, Hadjiconstantinou and Mingozzi [6], we executed our algorithm on a personal computer with a Pentium III (900 Mhz.) and observed that our computer is 2.9 times faster than their computer. Since their time limit becomes 1200 seconds on our computer, then our algorithm was able to solve all the TSPPD instances while [6] presents some unsolved instances. Moreover, even if [6] shows results of two selected instances for each group and we show average results of solving the 10 instances in [10] for each group, one can deduce that our algorithm performs better when solving these collections of TSPPD instances.

Conclusions

In this article we have presented some new valid inequalities for the 1-PDTSP. These inequalities exploit the close relationship between the 1-PDTSP and the CVRP. Indeed, inequalities recently developed for the CVRP are adapted in this article for the 1-PDTSP. Heuristic procedures to separate these inequalities are proposed and integrated within a branch-and-cut framework. The computational results show the usefulness of these families of inequalities when implemented in a branch-and-cut algorithm. A branch-and-cut algorithm without the new inequalities solved 1-PDTSP instances with up to 50

customers (see [12]), while with the new inequalities it can solve 1-PDTSP instances with up to 100 customers.

Acknowledgments

Work partially funded by “Ministerio de Educación y Ciencia” (TIC2003-05982-C05-02 and MTM2006-14961-C05-03), Spain. We thank Daniele Vigo for providing us with the instances used in [10].

References

- [1] N.R. Achuthan, L. Caccetta, S.P. Hill, “Capacitated vehicle routing problem: some new cutting planes”, *Asia-Pacific Journal of Operations Research* 15 (1998) 109–123.
- [2] S. Anily, J. Bramel, “Approximation Algorithms for the Capacitated Traveling Salesman Problem with Pickups and Deliveries”, *Naval Research Logistics* 46 (1999) 654–670.
- [3] J.R. Araque, “Contributions to the polyhedral approach to vehicle routing”, Working Paper 90–74, CORE, University of Louvain La Neuve, 1990.
- [4] J.R. Araque, L.A. Hall, T.L. Magnanti, “Capacitated trees, capacitated routing and associated polyhedra”, Working Paper, CORE, Catholic University of Louvain, 1990.
- [5] P. Augerat, “Approche Polyédrale du Problème de Tournées de Véhicules”, PhD thesis, Institut National Polytechnique de Grenoble, 1995.
- [6] R. Baldacci, E. Hadjiconstantinou, A. Mingozzi, “An Exact Algorithm for the Traveling Salesman Problem with Deliveries and Collections”, *Networks* 42 (2003) 26–41.
- [7] U. Blasum, W. Hochstättler, “Application of the Branch and Cut Method to the Vehicle Routing Problem”, Working Paper, University of Cologne, 2002.
- [8] P. Chalasani, R. Motwani, “Approximating Capacitated Routing and Delivery Problems”, *SIAM Journal on Computing* 28 (1999) 2133–2149.
- [9] M.L. Fisher, “Optimal solution of the vehicle routing problems using K-trees”, *Operations Research* 42 (1995) 621–642.
- [10] M. Gendreau, G. Laporte, D. Vigo, “Heuristics for the traveling salesman problem with pickup and delivery”, *Computers & Operations Research* 26 (1999) 699–714.

- [11] L. Gouveia, “A result on projection for the the Vehicle Routing Problem”, *European Journal of Operational Research* 83 (1995) 610–624.
- [12] H. Hernández-Pérez, J.J. Salazar-González, “A Branch-and-Cut Algorithm for a Traveling Salesman Problem with Pickup and Delivery”, *Discrete Applied Mathematics* 145 (2004) 126–139.
- [13] H. Hernández-Pérez, J.J. Salazar-González, “Heuristics for the One-Commodity Pickup-and-Delivery Traveling Salesman Problem”, *Transportation Science* 38 (2004) 245–255.
- [14] A.N. Letchford, R.W. Eglese, J. Lysgaard, “Multistars, partial multistars and the capacitated vehicle routing problem”, *Mathematical Programming* 94 (2002) 21–40.
- [15] C. Mannino, A. Sassano, “An exact algorithm for the maximum Stable Set problem”, *Computational Optimization and Applications* 3 (1994) 243–258.
- [16] G. Mosheiov, “The Travelling Salesman Problem with pick-up and delivery”, *European Journal of Operational Research* 79 (1994) 299–310.
- [17] D. Naddef, G. Rinaldi, “Branch-and-Cut Algorithms”, in P. Toth, D. Vigo (Editors), “The Vehicle Routing Problem”, *SIAM Monographs on Discrete Mathematics and Applications*, 2001.
- [18] Y. Pochet, “New valid inequalities for the vehicle routing problem”, Working Paper, Université Catholique de Louvain, 1998.
- [19] P. Toth, D. Vigo (Editors), “The Vehicle Routing Problem”, *SIAM Monographs on Discrete Mathematics and Applications*, 2001.

Q	Cuts					LB/Opt.	UB/Opt.	Opt.	B&C	Time
	Cap.	Cliq.	hm	hpm	im					
7	73	5	37	12	0	96.83	100.38	5734	41	0.3
8	36	1	20	1	0	99.50	100.00	5341	3	0.1
9	24	0	6	0	0	99.19	100.00	5038	3	0.1
10	23	0	0	0	0	99.44	100.00	4979	3	0.1
11	17	0	0	0	0	99.86	100.00	4814	1	0.1
12	16	0	3	0	0	99.11	100.00	4814	3	0.1
13	21	0	5	0	0	99.75	100.00	4627	3	0.1
14	8	0	0	0	0	100.00	100.00	4476	1	0.1
15	8	0	0	0	0	100.00	100.00	4476	1	0.1
16	2	0	0	0	0	99.50	100.00	4431	1	0.1
7	58	2	16	0	0	98.87	101.46	5771	6	0.1
8	29	1	14	0	0	100.00	100.00	5341	1	0.1
9	22	0	14	0	0	100.00	100.00	5038	1	0.1
10	21	0	3	0	0	99.48	100.00	5038	2	0.1
11	20	0	0	0	0	100.00	100.00	4821	1	0.1
12	22	0	0	0	0	99.09	100.00	4821	3	0.1
13	22	0	0	0	0	98.84	100.00	4758	7	0.1
14	22	0	2	1	0	99.59	100.00	4658	3	0.1
15	18	0	0	0	0	99.67	100.00	4648	3	0.1
16	11	0	0	0	0	99.52	100.00	4575	2	0.1
17	11	0	0	0	0	98.40	100.00	4575	4	0.1
18	11	0	0	0	0	98.40	100.00	4575	4	0.1
19	10	0	0	0	0	98.40	100.00	4575	4	0.1
20	7	0	0	0	0	100.00	100.00	4467	1	0.1
45	7	0	0	0	0	100.00	100.00	4467	1	0.1
46	6	0	0	0	0	100.00	100.00	4467	1	0.1
47	7	0	0	0	0	100.00	100.00	4467	1	0.1
48	7	0	0	0	0	100.00	100.00	4467	1	0.1
49	8	0	0	0	0	100.00	100.00	4467	1	0.1
50	4	0	0	0	0	99.50	100.00	4431	1	0.1

Table 1
Different variants based on Problem 1 in Mosheiov [16], where $n = 25$. These variants are 1-PDTSP, 1-PDTSP with initial load requirement, and TSPPD instances

n	Q	capacity		cliques		root MS		150 MS		all MS	
		B&C	Time	B&C	Time	B&C	Time	B&C	Time	B&C	Time
30	pd	4.2	0.03	4.2	0.03	3.5	0.03	3.6	0.03	3.6	0.03
30	10	74.0	0.36	81.3	0.44	34.3	0.33	22.8	0.43	22.8	0.43
30	15	31.7	0.22	22.3	0.19	15.0	0.24	10.1	0.31	10.1	0.30
30	20	60.0	0.26	47.0	0.20	37.7	0.21	26.2	0.30	25.3	0.31
30	25	4.5	0.04	4.5	0.04	3.6	0.04	2.9	0.04	2.9	0.04
30	30	3.9	0.03	3.9	0.03	3.1	0.03	2.7	0.03	2.7	0.03
30	35	2.5	0.02	2.5	0.02	2.3	0.02	2.1	0.03	2.1	0.02
30	40	1.2	0.01	1.2	0.01	1.0	0.01	1.0	0.01	1.0	0.01
30	45	2.3	0.01	2.3	0.01	1.2	0.01	1.2	0.02	1.2	0.01
30	100	1.0	0.01	1.0	0.01	1.0	0.01	1.0	0.01	1.0	0.01
40	pd	10.8	0.21	10.8	0.20	10.4	0.20	11.1	0.11	11.1	0.22
40	10	1019.5	8.07	628.1	5.89	710.3	6.94	666.3	11.18	608.5	24.11
40	15	124.9	0.79	133.8	0.83	61.9	0.51	56.0	0.75	55.5	0.80
40	20	24.1	0.21	24.9	0.21	23.6	0.22	23.9	0.31	23.9	0.31
40	25	5.3	0.07	5.3	0.07	5.7	0.08	3.8	0.09	3.8	0.09
40	30	6.3	0.05	7.0	0.05	5.8	0.05	5.4	0.07	5.4	0.07
40	35	6.7	0.04	6.7	0.04	6.7	0.04	6.6	0.06	6.6	0.06
40	40	6.1	0.03	6.1	0.03	6.1	0.04	6.1	0.06	6.1	0.06
40	45	6.1	0.03	6.1	0.03	6.1	0.04	6.1	0.06	6.1	0.05
40	100	6.4	0.03	6.4	0.03	6.4	0.04	6.4	0.06	6.4	0.05
50	pd	31.2	0.21	32.7	0.21	35.6	0.24	32.0	0.36	33.3	0.39
50	10	3355.8	113.13	2192.2	69.51	1466.1	54.44	697.0	38.72	592.0	54.70
50	15	801.5	25.20	646.0	22.79	457.6	16.50	160.5	13.78	158.0	16.97
50	20	366.2	5.31	285.1	4.81	186.9	3.56	124.7	4.80	117.0	5.32
50	25	131.5	1.60	130.4	1.68	73.2	1.11	63.3	1.72	63.2	1.79
50	30	20.4	0.31	21.1	0.32	18.8	0.33	14.0	0.41	14.0	0.41
50	35	22.4	0.27	22.4	0.27	24.0	0.30	23.5	0.46	23.5	0.46
50	40	21.6	0.20	21.6	0.20	18.3	0.20	16.2	0.32	16.2	0.31
50	45	26.4	0.19	26.4	0.19	26.6	0.21	27.2	0.34	27.2	0.35
50	100	10.5	0.07	10.5	0.07	10.5	0.08	10.5	0.12	10.5	0.11
60	pd	109.0	0.56	108.9	0.56	112.5	0.59	102.9	1.06	102.1	1.28
60	10	5801.1	296.20	3190.1	163.82	3148.4	187.07	2650.7	210.45	1901.2	334.04
60	15	1724.3	69.90	1328.5	55.24	1178.2	50.10	739.2	50.71	653.8	77.36
60	20	159.8	3.30	167.3	3.56	151.5	4.15	109.4	5.94	110.4	6.38
60	25	105.5	1.30	99.4	1.30	97.1	1.31	84.1	2.38	83.8	2.37
60	30	163.0	1.50	163.0	1.54	148.1	1.56	140.9	2.35	138.3	3.52
60	35	160.4	1.15	161.1	1.15	110.6	0.95	126.1	1.80	119.8	2.30
60	40	41.9	0.29	41.9	0.28	66.9	0.42	61.4	0.81	60.2	0.88
60	45	35.7	0.23	35.7	0.23	34.0	0.27	34.7	0.51	34.7	0.49
60	100	42.5	0.20	42.5	0.19	42.5	0.21	42.5	0.46	42.5	0.45
70	pd	942.0	6.58	868.5	5.86	684.8	4.25	745.6	6.07	687.3	11.13
70	10	11358.2	1865.57	8458.1	1438.71	7495.8	1389.45	6491.7	1202.32	3113.1	1770.46
70	15	14730.1	2282.84	15202.3	2438.76	10157.7	2068.57	6957.3	1819.66	3485.8	2076.97
70	20	4742.6	369.14	4698.8	367.59	2711.0	241.86	2090.7	247.50	1648.8	343.67
70	25	1040.1	39.29	1095.1	43.93	1277.7	53.07	954.9	51.48	776.9	69.23
70	30	623.4	12.88	609.2	12.75	607.9	11.84	517.1	14.75	480.0	20.97
70	35	280.7	3.91	281.0	3.92	210.7	3.41	194.5	5.70	194.8	6.92
70	40	184.3	2.25	183.5	2.29	211.3	2.56	177.9	4.28	176.7	5.45
70	45	135.3	1.15	135.3	1.15	142.8	1.51	121.7	2.40	121.5	2.61
70	100	72.7	0.39	72.7	0.39	72.7	0.41	72.7	0.90	72.7	1.01

Table 2
Average results of different separation strategies on the random 1-PDTSP instances

n	Q	Cuts					LB/Opt.	UB/Opt.	Opt./TSP	B&C	Time	t.l.
		cap.	cliq.	hm	hpm	im						
40	pd	32.1	0.0	1.6	0.2	0.0	98.87	100.00	102.02	11.1	0.11	0
40	10	283.5	48.5	217.6	26.5	4.7	96.97	101.10	139.91	666.3	11.18	0
40	15	122.3	2.0	44.8	2.6	0.0	98.84	100.61	117.22	56.0	0.75	0
40	20	68.8	1.0	13.8	0.5	0.0	99.12	100.14	107.02	23.9	0.31	0
40	25	33.5	0.0	10.6	0.0	0.0	99.52	100.01	103.07	3.8	0.09	0
40	30	26.3	0.1	5.0	0.0	0.0	99.54	100.08	101.16	5.4	0.07	0
40	40	17.2	0.0	0.9	0.0	0.0	99.34	100.07	100.00	6.1	0.06	0
40	100	17.3	0.0	0.0	0.0	0.0	99.31	100.10	100.00	6.4	0.06	0
50	pd	50.5	0.2	9.4	0.6	0.0	98.37	100.01	101.48	32.0	0.36	0
50	10	396.3	72.4	394.9	41.0	18.1	96.90	103.41	148.19	697.0	38.72	0
50	15	268.6	16.4	241.1	18.7	2.9	97.91	100.58	121.60	160.5	13.78	0
50	20	204.0	4.7	142.4	9.6	0.7	98.01	100.53	111.54	124.7	4.80	0
50	25	140.5	0.5	80.4	4.1	0.1	98.65	100.42	106.76	63.3	1.72	0
50	30	76.6	0.3	18.3	0.2	0.0	99.14	100.08	103.83	14.0	0.41	0
50	40	49.3	0.0	11.6	0.4	0.0	98.94	100.14	101.46	16.2	0.32	0
50	100	24.8	0.0	0.0	0.0	0.0	99.03	100.04	100.00	10.5	0.12	0
60	pd	63.4	0.0	10.5	0.6	0.0	98.51	100.42	101.33	102.9	1.06	0
60	10	705.6	147.1	590.0	65.2	22.1	96.60	103.23	144.86	2650.7	210.45	0
60	15	407.5	34.9	299.8	24.4	2.6	97.81	101.11	119.82	739.2	50.71	0
60	20	223.4	6.9	135.9	14.2	2.0	98.38	100.80	109.88	109.4	5.94	0
60	25	156.4	1.4	87.6	5.4	0.4	98.47	100.26	105.60	84.1	2.38	0
60	30	118.4	0.7	42.1	2.3	0.2	98.73	100.23	103.00	140.9	2.35	0
60	40	47.7	0.0	42.2	0.9	0.1	98.92	100.16	100.50	61.4	0.81	0
60	100	35.4	0.0	0.0	0.0	0.0	98.65	100.27	100.00	42.5	0.46	0
70	pd	116.4	0.4	28.3	1.3	0.0	98.32	100.28	101.45	745.6	6.07	0
70	10	1099.5	186.5	687.3	60.0	53.3	97.13	103.44	151.65	6491.7	1202.32	0
70	15	911.5	82.1	555.1	41.3	15.1	96.54	101.47	124.02	4040.5	474.31	2
70	20	736.5	32.6	308.6	28.5	5.7	97.27	100.82	113.95	2090.7	247.50	0
70	25	434.8	5.8	217.9	20.0	2.6	97.71	100.67	107.90	954.9	51.48	0
70	30	273.2	0.4	116.8	9.4	0.2	98.30	100.38	104.78	517.1	14.75	0
70	40	137.0	0.3	69.3	5.1	0.1	98.54	100.31	102.09	177.9	4.28	0
70	100	38.7	0.0	0.0	0.0	0.0	99.04	100.11	100.00	72.7	0.90	0
80	pd	90.8	0.1	49.1	0.4	0.0	98.81	100.14	101.06	72.3	2.28	0
80	10	1030.4	166.0	804.6	67.6	68.2	97.12	105.36	153.72	4615.0	704.26	5
80	15	1418.3	125.3	605.9	51.1	12.7	96.88	101.99	126.83	6151.7	989.76	3
80	20	1590.0	112.5	356.1	32.8	6.6	96.85	101.07	117.52	18357.1	2079.17	2
80	25	800.8	14.7	309.9	33.6	3.0	97.41	101.07	109.68	2646.4	193.74	1
80	30	465.5	4.8	139.2	16.0	0.5	98.16	100.80	106.34	1491.2	82.64	0
80	40	131.6	0.0	66.5	1.9	0.0	98.98	100.19	102.67	129.8	3.63	0
80	100	44.6	0.0	0.0	0.0	0.0	99.17	100.17	100.00	146.2	1.27	0
90	pd	146.8	0.3	28.2	0.3	0.0	98.67	100.16	101.24	839.6	9.30	0
90	20	1069.2	46.0	421.0	36.5	2.5	96.81	101.34	113.35	3690.7	472.63	4
90	25	933.0	7.9	194.6	14.7	0.4	97.75	101.08	107.65	9164.1	740.61	3
90	30	832.7	6.2	234.5	24.0	0.8	97.89	100.90	106.15	3472.8	412.30	0
90	40	248.3	0.2	131.0	1.9	0.0	98.96	100.48	102.05	369.5	13.97	0
90	100	62.5	0.0	0.3	0.0	0.0	99.28	100.25	100.00	360.4	2.84	0
100	pd	148.3	0.1	29.2	0.5	0.0	98.72	100.42	101.01	261.3	6.69	0
100	20	1684.3	73.0	435.3	27.7	1.0	97.22	102.07	110.79	18674.0	2733.24	7
100	25	1841.1	23.8	365.0	29.3	1.3	97.34	101.60	109.83	9404.0	1758.84	2
100	30	1404.0	7.7	248.4	21.5	0.9	97.74	101.30	106.84	11728.7	1401.44	0
100	40	527.1	0.4	86.5	7.2	0.0	98.71	100.49	103.14	2566.3	132.46	0
100	100	54.3	0.0	0.0	0.0	0.0	99.22	100.39	100.00	42.6	1.45	0

Table 3
Average results of the random 1-PDTSP instances

name	n	Q	Cuts					LB/Opt.	UB/Opt.	Opt.	B&C	root	Time
			cap.	cliq.	hm	hpm	im						
eil51	51	41	1041	170	382	19	4	95.82	101.79	504	4603	0.9	185.3
eil51	51	42	830	137	356	32	8	96.27	101.40	500	2560	0.9	88.1
eil51	51	43	363	34	453	19	2	97.14	102.44	491	357	1.0	18.6
eil51	51	44	334	34	327	34	3	97.24	101.22	490	392	1.0	16.2
eil51	51	45	401	33	453	35	3	96.92	100.41	486	420	1.1	22.1
eil51	51	50	318	19	331	17	0	97.26	102.55	470	246	0.6	10.7
eil51	51	60	254	3	256	17	0	97.41	100.00	452	263	0.4	6.2
eil51	51	70	296	2	152	7	0	97.21	100.00	445	391	0.4	5.5
eil51	51	80	100	1	49	2	0	98.49	101.15	434	52	0.2	1.1
eil51	51	90	67	0	47	0	0	98.76	100.46	432	33	0.2	0.7
eil51	51	100	42	0	86	0	0	98.95	100.23	430	8	0.2	0.3
eil51	51	125	17	0	4	0	0	99.06	100.00	427	4	0.1	0.2
eil51	51	150	20	0	63	0	0	98.95	100.00	427	11	0.1	0.2
eil51	51	155	12	0	0	0	0	99.18	100.23	426	7	0.1	0.2
eil76	76	134	104	0	102	1	0	99.28	100.73	547	23	0.5	1.5
eil76	76	135	105	0	66	1	0	99.28	100.73	547	29	0.3	1.7
eil76	76	136	103	0	35	0	0	99.28	100.37	547	22	0.4	1.2
eil76	76	137	87	0	58	1	0	99.28	100.37	547	19	0.4	1.2
eil76	76	138	91	0	69	1	0	99.27	100.37	547	30	0.3	1.3
eil76	76	139	30	0	0	0	0	99.82	101.10	544	2	0.0	0.4
eil76	76	140	29	0	0	0	0	99.82	100.00	544	2	0.0	0.4
eil76	76	150	3	0	1	0	0	100.00	100.37	539	1	0.0	0.2
eil76	76	160	18	0	8	0	0	99.85	101.11	539	1	0.0	0.4
eil76	76	166	17	0	4	0	0	100.00	100.74	538	1	0.0	0.4
eil101	101	82	3172	17	242	21	6	98.12	101.95	665	25099	3.1	3543.8
eil101	101	83	1659	13	331	27	4	98.19	102.26	664	6940	3.6	610.4
eil101	101	84	3098	36	259	18	0	98.11	101.81	664	38619	3.3	4329.5
eil101	101	85	3609	18	217	16	1	98.00	101.81	662	41383	2.7	6402.2
eil101	101	86	896	3	250	9	0	98.42	102.28	657	2825	2.1	130.6
eil101	101	87	1167	3	209	17	0	98.35	102.28	657	4784	2.3	271.2
eil101	101	88	1590	17	132	25	0	98.19	102.44	657	7101	2.2	457.2
eil101	101	89	1611	7	166	21	0	98.30	101.83	656	6029	2.8	479.9
eil101	101	90	1038	2	195	22	2	98.47	100.00	655	4391	2.4	194.0
eil101	101	95	1975	25	158	23	0	98.23	101.07	654	13967	1.8	928.1
eil101	101	100	878	3	191	11	2	98.78	100.62	647	979	2.4	80.4
eil101	101	125	345	0	147	5	0	99.08	100.16	637	489	2.2	35.3
eil101	101	150	222	0	163	1	0	99.05	100.16	635	312	1.9	22.6
eil101	101	175	109	0	102	3	0	99.29	100.16	633	116	1.3	9.6
eil101	101	185	21	0	30	0	0	99.76	100.00	629	3	0.0	0.6

Table 4
Results of the 1-PDTSP instances derived of the TSPLIB

β	LB/Opt.	UB/Opt	Opt/TSP	B&C	root	Time
0.00	99.67	100.20	100.00	1172.0	0.28	64.99
0.05	99.61	100.27	100.54	2888.5	0.29	187.37
0.10	99.54	100.27	100.66	816.1	0.30	48.10
0.20	99.49	100.23	100.85	1505.0	0.35	95.98

Table 5
Results of the TSPPD instances derived from VRP test problems described in [10]

β	n	LB/Opt.	UB/Opt.	Opt./TSP	B&C	root	Time
0.00	25	99.38	100.04	100.00	2.2	0.00	0.02
0.00	50	99.38	100.05	100.00	4.3	0.02	0.07
0.00	75	99.29	100.20	100.00	20.8	0.08	0.38
0.00	100	99.19	100.37	100.00	236.6	0.15	2.56
0.00	150	99.19	100.81	100.00	774.4	0.67	18.13
0.00	200	99.31	101.13	100.00	586.0	1.70	34.78
0.05	25	99.05	100.00	102.07	1.9	0.01	0.02
0.05	50	99.36	100.03	100.17	4.8	0.03	0.09
0.05	75	99.08	100.08	100.76	262.4	0.13	1.92
0.05	100	99.15	100.35	100.26	219.3	0.26	3.52
0.05	150	99.15	100.85	100.13	703.7	0.79	23.44
0.05	200	99.15	100.96	100.34	2650.9	2.01	113.84
0.10	25	99.12	100.04	101.18	5.7	0.01	0.03
0.10	50	99.49	100.05	100.44	3.7	0.05	0.11
0.10	75	98.82	100.21	101.20	910.3	0.15	7.11
0.10	100	98.96	100.44	100.48	209.0	0.25	4.86
0.10	150	99.08	100.92	100.33	5279.4	0.85	158.98
0.10	200	99.11	100.85	100.43	1255.0	2.38	77.85
0.20	25	98.62	100.04	102.59	6.3	0.01	0.04
0.20	50	99.39	100.02	100.79	5.9	0.04	0.13
0.20	75	98.50	100.27	101.69	1796.5	0.15	16.44
0.20	100	98.84	100.53	100.82	952.5	0.29	16.09
0.20	150	99.02	100.99	100.51	2812.7	0.89	101.16
0.20	200	99.02	101.14	100.59	5058.0	2.64	249.40
∞	25	99.07	100.00	101.32	8.1	0.01	0.04
∞	50	98.12	100.08	102.42	506.9	0.05	1.97
∞	75	98.85	100.07	100.91	80.3	0.13	1.34
∞	100	98.76	100.39	100.74	1646.2	0.29	18.33
∞	150	98.99	100.87	100.43	2108.7	1.08	57.32
∞	200	99.08	101.10	100.45	7967.5	2.63	513.01

Table 6
Results of the Euclidian TSPPD instances described in [10]

β	n	LB/Opt.	UB/Opt.	Opt/TSP	B&C	root	Time
0.00	25	99.37	100.00	100.00	1.5	0.00	0.01
0.00	50	99.87	102.03	100.00	1.9	0.02	0.06
0.00	75	99.64	104.36	100.00	2.9	0.08	0.17
0.00	100	99.88	106.39	100.00	1.4	0.16	0.31
0.00	150	99.80	114.27	100.00	2.3	0.48	0.98
0.00	200	99.67	122.54	100.00	4.9	1.11	2.51
0.05	25	98.96	100.24	101.41	2.1	0.00	0.02
0.05	50	99.72	101.48	100.20	1.4	0.02	0.06
0.05	75	99.40	103.34	100.53	7.1	0.14	0.40
0.05	100	99.82	106.39	100.64	7.7	0.23	0.81
0.05	150	99.83	113.88	100.53	3.4	0.91	1.78
0.05	200	99.77	123.08	100.07	3.5	1.82	3.51
0.10	25	98.48	100.00	102.29	3.4	0.01	0.03
0.10	50	99.45	101.62	100.97	2.7	0.04	0.09
0.10	75	99.10	104.16	101.27	16.1	0.15	0.78
0.10	100	99.75	107.24	100.86	4.8	0.31	0.82
0.10	150	99.77	115.05	100.60	9.4	0.89	2.99
0.10	200	99.70	122.33	100.23	4.9	2.19	4.42
0.20	25	98.92	100.19	102.53	2.2	0.01	0.02
0.20	50	99.43	101.83	101.08	5.0	0.04	0.13
0.20	75	99.06	104.54	101.68	18.7	0.17	0.86
0.20	100	99.66	106.17	100.98	26.2	0.23	2.23
0.20	150	99.66	114.42	100.79	28.4	1.15	7.20
0.20	200	99.70	121.89	100.46	14.5	2.09	8.60
∞	25	99.25	100.49	102.41	1.4	0.00	0.02
∞	50	99.41	101.69	101.28	7.2	0.04	0.16
∞	75	99.56	104.97	100.77	8.5	0.12	0.54
∞	100	99.69	106.46	100.63	16.1	0.31	1.52
∞	150	99.53	113.22	101.33	6.2	1.04	2.33
∞	200	99.40	122.60	100.52	17.8	2.56	10.38

Table 7
Results of the symmetric TSPPD instances described in [10]

Appendix: Pseudocodes of separation procedures

The appendix shows the pseudocodes of the separation procedures of clique cluster inequalities (7), generalized homogeneous multistar inequalities (15) with $C = N$, generalized partial homogeneous multistar inequalities (15) with $C \neq N$, and generalized inhomogeneous multistar inequalities (10). The pseudocodes of three auxiliary functions (FINDPATHS, FINDPOSITIVESELLITES and BUILDNUCLEUS) are also shown. The pseudo-code of function FINDNEGATIVESELLITES is not described because it is similar to the pseudo-code of function FINDPOSITIVESELLITES.

Function FINDPATHS (G^* , $N \subset V$, $C \subseteq N$): to construct paths of G^* starting from C .

```

1: function FINDPATHS( $G^*$ ,  $N \subset V$ ,  $C \subseteq N$ )
2:    $l \leftarrow 0$ 
3:   while  $i_1 \in V \setminus N$  and  $x^*(E(C : \{i_1\})) > 0$  do
4:      $l \leftarrow l + 1$ 
5:     Find the path  $P = (i_1, \dots, i_s)$  such that:
       a)  $i_j \in V \setminus N$  for  $j = 1, \dots, s$ ,
       b)  $x_e^* = 1$  for each  $e = [i_j, i_{j+1}]$ ,  $j = 1, \dots, s - 1$ , and
       c)  $P$  is the largest one.
6:      $P_l \leftarrow P$ 
7:   end while
8:   return  $\{P_1, \dots, P_l\}$ 
9: end function

```

Function FINDPOSITIVESELLITES($\{P_1, \dots, P_l\}$): to find satellites outside N for all clique cluster and generalized multistar inequalities.

```

1: function FINDPOSITIVESELLITES( $\{P_1, \dots, P_l\}$ )
2:    $m \leftarrow 0$ 
3:   while  $P = (i_1, \dots, i_s) \in \{P_1, \dots, P_l\}$  do
4:     Let be  $max = \max\{q_{i_1}, q_{i_1} + q_{i_2}, \dots, q_{i_1} + q_{i_2} + \dots + q_{i_s}\}$  and let be  $s'$ 
       the index where the maximum is obtained
5:     if  $max > 0$  then
6:        $m \leftarrow m + 1$ 
7:        $S_m \leftarrow \{i_1, \dots, i_{s'}\}$ 
8:     end if
9:   end while
10:  Resort the sets  $S_1, \dots, S_m$  in decreasing order to the sum of the demands
       of their vertices
11:  return  $\{S_1, \dots, S_m\}$ 
12: end function

```

Function SEPARATIONCLIQUECLUSTERS(G^*): to separate the clique cluster inequalities (7).

```

1: function SEPARATIONCLIQUECLUSTERS( $G^*$ )
2:    $m \leftarrow 0$ 
3:   while  $v \in V$  do
4:     Call FINDPATHS( $G^*$ ,  $\{v\}$ ,  $\{v\}$ ) returning the paths  $\{P_1, \dots, P_l\}$ 
5:     Call FINDPOSITIVESATELLITES( $\{P_1, \dots, P_l\}$ ) returning  $\{S_1, \dots, S_m\}$ 
6:     Let  $m' \leftarrow 1$ 
7:     while  $m' < m$  and  $q_v + \sum_{j \in S_{m'}} q_j + \sum_{j \in S_{m'+1}} q_j > Q$  do
8:        $m' \leftarrow m' + 1$ 
9:     end while
10:    Let  $W_1 \leftarrow S_1 \cup \{v\}$ ,  $\dots$ ,  $W_{m'} \leftarrow S_{m'} \cup \{v\}$ 
11:    Check if the clique cluster constraint defined by  $W_1, \dots, W_{m'}$  is violated
12:    Call FINDNEGATIVESATELLITES( $\{P_1, \dots, P_l\}$ ) returning  $\{S_1, \dots, S_m\}$ 
13:    Repeat the process for negative demands
14:  end while
15: end function

```

Function BUILDNUCLEUS(G^*): to construct the nucleus list.

```

1: function BUILDNUCLEUS( $G^*$ )
2:   Sort the edge set  $E^*$  in decreasing order of  $x_e^*$  for each  $e \in E^*$ 
3:   For the  $|V|$  first edges  $e = [i, j]$  insert the sets  $\{i, j\}$  in the nucleus list  $\mathcal{N}$ 
4:   for  $k \leftarrow 3$  to  $|V|/2$  do
5:     while  $N \in \mathcal{N}$  and  $|N| = k - 1$  do
6:       Search the vertex  $j$  (if any exist) verifying:
7:         a)  $j \in V \setminus N$ ,
8:         b)  $x^*(E(N : \{j\})) > 0$  is as large as possible, and
9:         c)  $N \cup \{j\}$  is not in  $\mathcal{N}$ 
10:      if  $j$  exists then
11:        insert  $N \cup \{j\}$  in  $\mathcal{N}$ 
12:      end if
13:    end while
14:  end for
15:  return  $\mathcal{N}$ 
16: end function

```

Function SEPARATIONHOMOGENEOUSMULTISTAR(G^*): to separate generalized homogeneous multistar inequalities (15) with $C = N$.

```

1: function SEPARATIONHOMOGENEOUSMULTISTAR( $G^*$ )
2:   Call BUILDNUCLEUS( $G^*$ ) returning the nucleus list  $\mathcal{N}$ 
3:   while  $N \in \mathcal{N}$  do
4:     Call FINDPATHS( $G^*$ ,  $N$ ,  $N$ ) returning the paths  $\{P_1, \dots, P_l\}$ 
5:     if  $\sum_{i \in N} q_i \geq -Q$  then
6:       Call FINDPOSITIVESATELLITES( $\{P_1, \dots, P_l\}$ ) returning  $\{S_1, \dots, S_m\}$ 
7:       Let  $m' \leftarrow m$ 
8:       for  $m' \leftarrow m$  to 2 do
9:         Build the projection of  $\{N, S_1, \dots, S_{m'}\}$ 
10:        Check if any inequality derived from the projection is violated
11:      end for
12:    end if
13:    if  $\sum_{i \in N} q_i \leq Q$  then
14:      Call FINDNEGATIVESATELLITES( $\{P_1, \dots, P_l\}$ ) returning  $\{S_1, \dots, S_m\}$ 
15:      Repeat the process for negative demands
16:    end if
17:  end while
18: end function

```

Function SEPARATIONPARTIALHOMOGENEOUSMULTISTAR(G^* , \mathcal{N}): to separate generalized partial homogeneous multistar inequalities (15) with $C \neq N$.

```

1: function SEPARATIONPARTIALHOMOGENEOUSMULTISTAR( $G^*$ ,  $\mathcal{N}$ )
2:   while  $N \in \mathcal{N}$  do
3:     for  $l \leftarrow \min\{4, |N| - 1\}$  to 1 do
4:       Let be  $C$  such that:  $C \subset N$ ,  $|C| = l$  and  $x^*(E(C : V \setminus N))$  is as
       largest as possible
5:       Call FINDPATHS( $G^*$ ,  $N$ ,  $C$ ) returning the paths  $\{P_1, \dots, P_l\}$ 
6:       if  $\sum_{i \in N \setminus C} q_i + \sum_{i \in C, q_i < 0} q_i \geq -Q$  then
7:         Call FINDPOSITIVESATELLITES( $\{P_1, \dots, P_l\}$ )
8:         Let  $m' \leftarrow m$ 
9:         for  $m' \leftarrow m$  to 2 do
10:          Build the projection of  $\{N, C, S_1, \dots, S_{m'}\}$ 
11:          Check if any inequality derived from the projection is violated
12:        end for
13:      end if
14:      if  $\sum_{i \in N \setminus C} q_i + \sum_{i \in C, q_i > 0} q_i \leq Q$  then
15:        Call FINDNEGATIVESATELLITES( $\{P_1, \dots, P_l\}$ )
16:        Repeat the process for negative demands
17:      end if
18:    end for
19:  end while
20: end function

```

Function SEPARATIONINHOMOGENEOUSMULTISTAR(G^*, \mathcal{N}): to separate generalized inhomogeneous multistar inequalities (10).

```

1: function SEPARATIONINHOMOGENEOUSMULTISTAR( $G^*, \mathcal{N}$ )
2:   while  $N \in \mathcal{N}$  do
3:     Call FINDPATHS( $G^*, N, N$ ) returning the paths  $\{P_1, \dots, P_l\}$ 
4:     if  $\sum_{i \in N} q_i \geq -Q$  then
5:       Call FINDPOSITIVESATELLITES( $\{P_1, \dots, P_l\}$ ) returning  $(S_1, \dots, S_m)$ 
6:       Check if any inhomogeneous multistar inequality defined by  $\{N, S_1, \dots, S_m\}$ 
       is violated
7:     end if
8:     if  $\sum_{i \in N} q_i \leq Q$  then
9:       Call FINDNEGATIVESATELLITES( $\{P_1, \dots, P_l\}$ ) returning  $\{S_1, \dots, S_m\}$ 
10:      Check if any inhomogeneous multistar inequality defined by  $\{N, S_1, \dots, S_m\}$ 
      is violated
11:    end if
12:  end while
13: end function

```