# The split-demand one-commodity pickup-and-delivery travelling salesman problem

Juan-José Salazar-González, Beatriz Santos-Hernández

*Department of Mathematics, Statistics and Operations Research*
*University of La Laguna, Tenerife, Spain*

**Abstract**

This paper introduces a new vehicle routing problem transferring one commodity between customers with a capacitated vehicle that can visit a customer more than once, although a maximum number of visits must be respected. It generalizes the capacitated vehicle routing problem with split demands and some other variants recently addressed in the literature. We model the problem with a single commodity flow formulation and design a branch-and-cut approach to solve it. We make use of Benders Decomposition to project out the flow variables from the formulation. Inequalities to strengthen the linear programming relaxation are also presented and separated within the approach. Extensive computational results illustrate the performance of the approach on benchmark instances from the literature.

*Keywords:* Vehicle Routing Problem, Branch and Cut, Split Demand.

## 1. Introduction

In the Capacitated Vehicle Routing Problem (VRP) there are products originally in a depot that must be delivered to a set of customers by an homogeneous fleet of vehicles. Since all products departure from the same source location, they can all be considered as *one commodity*. A route consists of paths, starting from and ending at the depot. Each path is called here *trip*. The trips in a

*Email addresses:* jjsalaza@ull.es (Juan-José Salazar-González), besanher@ull.es (Beatriz Santos-Hernández)

route can be performed in parallel by a set of identical vehicles (each vehicle performs a different trip) or in sequence by a single vehicle. Each customer is a delivery location that must be visited once, while the depot is a pickup location. The load of the vehicle through a trip should never exceed its capacity. The aim of the VRP is to compute a route to satisfy the demand of each customer while minimizing the travel cost. See e.g. Toth and Vigo [30] for a textbook on VRP and variants. In the Split Delivery Vehicle Routing Problem (SDVRP) the demand of a customer is allowed to be served in more than one visit, thus allowing routes with smaller travel cost. See e.g. Archetti and Speranza [3] for a survey on SDVRP and related problems. In Pickup-and-Delivery problems the customers are also allowed to be pickup locations, and therefore it is important to distinguish between one-commodity and multi-commodity variants. See, e.g., Berbeglia et al. [5] and Parragh et al. [28] for surveys on static problems in this area. Our paper merges the three topics in the following problem.

A finite set of locations is given and the travel cost from one location to another location is assumed to be known. One specific location is considered to be a depot and the other locations are identified as customers. Each customer requires or provides a given demand of a single commodity (the product). A product unit collected from a pickup location can be supplied to any delivery location. It is assumed that there is a vehicle with a given capacity, originally at the depot, to serve the demands of all customers through a route. A route is a set of trips that cover each customer *at least* once. While following the route, the vehicle can either deliver or collect product in each location. All the visits to a customer must end up with exactly its required demand. The number of times that the vehicle visits a customer is limited by a given parameter $m$. The *Split-Demand One-Commodity Pickup-and-Delivery Travelling Salesman Problem* (SD1PDTSP) is the problem of finding a min-cost route for the capacitated vehicle such that it satisfies the demand of all customers.

In this paper, the initial load of the vehicle when leaving the depot is a decision that must be computed within the optimization problem. As shown later, the model and algorithm proposed in this paper for the SD1PDTSP can

easily be adapted to the variant of the problem where the vehicle is required to leave the depot with full (or empty) load. Also in this paper, the depot is treated as a customer, providing (or absorbing) the sum of the customers' demands so the balance of the commodity in the system is zero. As any customer, the depot is allowed to be visited several times by the vehicle through the route. The number of visits to the depot is the number of trips in the route, and is limited to at most a given parameter $k$. For simplicity in notation, we assume $k = m$, although in some applications the value $k$ could be desired to be different than $m$.

Although it could make sense to require that product collected in a trip must be delivered in the same trip, we do not impose this requirement in the SD1PDTSP. Therefore all trips in a SD1PDTSP route are considered to be performed by a single vehicle sequentially. The problem name contains the words "Travelling Salesman Problem" (TSP) to emphasize this assumption. When the depot is the only pickup location, as on VRP instances, or when the vehicle is forced to leave the depot with full load, then the trips may be executed in parallel when a fleet of $k$ identical vehicles are available. In general, however, product collected in a customer may go through the depot before being delivered in another customer, and therefore the trips cannot be performed in parallel by different vehicles. We discuss later how to adapt the mathematical model to ensure that product collected in a trip are also delivered in the same trip.

In the SD1PDTSP each location is assumed to have a known inventory of the product before starting the vehicle service. Also, each location is associated with a desired inventory that must have after the last vehicle service. The difference between the inventories is the demand of the location. Inventories in a location can be reduced or augmented during intermediate visits of the vehicle with the only constraint of considering a given capacity associated with the location. In other words, we allow preemption in the SD1PDTSP, i.e. product units collected in a location can be unloaded (fully or partially) at any intermediate location to be picked up later and delivered in another location. We will observe how to

adapt the model and algorithm described in this paper for the non-preemptive variant. We do not consider inventory holding cost in the SD1PDTSP. See e.g. Coelho et al. [8] for a recent survey on the Inventory-Routing Problem.

A practical application of SD1PDTSP arises in the context of a self-service bike-sharing system, where every night a capacitated vehicle visits all the bike stations in the district of a city to move bikes and restore the initial configuration of the system (see, e.g., Chemla et al. [6], Raviv et al. [29] and Dell'Amico et al. [9]).

This paper is organized as follows. Section 2 shows the relation between SD1PDTSP and other problems addressed in the literature. Section 3 presents a mixed integer linear programming formulation for SD1PDTSP and discusses several minor modifications to also model other problem variants. Section 4 describes a branch-and-cut algorithm based on projecting the continuous variables and on some new valid inequalities. Section 5 analyzes computational results obtained by applying the algorithm to solve SD1PDTSP, SDVRP and VRP instances.

## 2. Related problems

When the number of visits to a location is unbounded, the uncapacitated variant of the SD1PDTSP becomes the *Graphical TSP* (see e.g. Naddef and Rinaldi [25]), where the aim is to find a min-cost route visiting each location at least once. When the number of visits to a location is bounded, the uncapacitated variant of the SD1PDTSP is a particular case of the *Generalized TSP* (see e.g. Fischetti et al. [12]), where each location is represented by a cluster of nodes (visits to a customer) in a graph. When the maximum number of visits to a location is one, the uncapacitated variant is the TSP.

The VRP can be considered as a SD1PDVRP where the customers are delivery locations, the depot is the only pickup location, and split is allowed only at the pickup location. In standard VRP instances, each customer demand is not larger than the vehicle capacity and the total delivery demand is larger than the vehicle capacity. Then, the depot must be visited several times (or equivalently,

several trips are necessary). Some VRP articles assume that the number of trips must be the optimal objective value $k$ of a Bin Packing Problem defined by the customer demands and the vehicle capacity. In other VRP articles, $k$ is a given upper bound of this optimal objective value. In other VRP articles, the number of trips is unbounded (e.g. $k$ is the number of customers).

The SD1PDTSP is an immediate generalization of the so-called *One-Commodity Pickup-and-Delivery Travelling Salesman Problem* (1PDTSP) considered in, e.g., Hernández-Pérez and Salazar-González [16] and where each location must be visited exactly once. It is worth mentioning that, when assuming an unlimited number of trips, finding a feasible VRP solution is trivial (e.g. trips consist of single customers). However, for the 1PDTSP, even checking whether a feasible solution exists is an strongly $\mathcal{NP}$-complete problem (see Hernández-Pérez and Salazar-González [16]). Therefore, allowing split-demand on the 1PDTSP (and so, a location is allowed to be visited several times) has not only the advantage of reducing the travel cost of the route, but it may also help to find routes respect to the non-split variant.

The SDVRP is the particular case of the SD1PDTSP where the depot is the only delivery location, the customers represent pickup locations, and the parameter limiting the number of visits to a location is unlimited. An interesting difference of the SDVRP respect to the VRP is that there exists always a SDVRP route with a number of trips equal to the total demand divided by the vehicle capacity, rounded up to the nearest integer. When the number of trips is unlimited and the travel cost satisfies the triangular inequality, Dror and Trudeau [11] shows that there exists an optimal SDVRP solution where two trips share at most one customer, and Archetti et al. [2] shows that there exists an optimal SDVRP solution where the sum of the number of splits over all customers is less than the number of trips, and Archetti and Speranza [3] shows that the optimal value of the VRP may be twice the optimal value of the SDVRP. There are many articles in the SDVRP literature, specially on heuristic techniques. Regarding exact methods, Dror et al. [10] gives a branch-and-cut algorithm, Belenguer et al. [4] and Moreno et al. [23] describe lower bound procedures, Lee

et al. [22] develops a dynamic programming approach, Jin et al. [19] proposes an iterative procedure where a sequence of TSPs are solved, and Archetti et al. [1] describes a column generation technique. Although our SD1PDTSP algorithm is not intended to compete with specific exact SDVRP methods on benchmark SDVRP instances, Section 5 shows that our SD1PDTSP implementation found optimal solutions to benchmark instances that the recent technique in Archetti et al. [1] was not able to solve.

In practice, visiting a customer is costly to both the transportation company and the customer. As observed in Gulczynski et al. [13], it takes time, involves paperwork and data processing, and often distracts the customer from primary activities. It is especially undesirable for the customer to be interrupted and distracted too many times. As a result both parties may impose a maximum number $m$ of visits to a location. In addition, other conditions can be required. For example, as analyzed in Gulczynski et al. [13], only SDVRP solutions with a minimum delivery amount in each visit could be accepted.

There are few publications on optimization problems including vehicle routing, split demands and pickup-and-deliveries. Nowack et al. [26] considers a related problem where several commodities must be transported. Each commodity goes from one origin to one destination, a location may simultaneously serve as both an origin and a destination, and split demands are allowed. The authors showed that the maximum travel-cost reduction of the split-allowed problem versus the split-forbidden problem is obtained when all the customer demands are just above one half of vehicle capacity. They also made the conjecture that this reduction is at most 50%, and described a heuristic approach to solve the split-allowed problem starting from a feasible split-forbidden solution. The split-forbidden problem is the problem addressed in Hernández-Pérez and Salazar-González [17]. Kerivin et al. [21] and Kerivin et al. [20] extend the problem by allowing preemption, i.e. a commodity collected in a location can be unloaded (fully or partially) at any intermediate location to be picked up later by the same or another vehicle. This unloading/picking-up process can be repeated several times for a demand until its destination is reached. There

is no restriction on the number of times that a location may be visited, and the vehicle capacity is not on each vehicle routing an arc but on all vehicles traversing the same arc. Nowak et al. [27] extends the problem with the additional constraint that, on each trip, all pickup customers must be visited before any delivery customer, and describes a dynamic programming algorithm that exploits the precedence constraints between origins and destinations.

As far as we know no research has been conducted previously on the SD1PDTSP.

## 3. Problem Formulation

This section presents a mathematical model for the SD1PDTSP. We first set up the notation. Let $n$ be the number of locations (i.e. depot and customers), and $m$ the maximum number of visits to each location. Let $I = \{1, \ldots, n\}$ be the set of locations, and $V_i$ a set of $m$ nodes representing potential visits to location $i$. $V_i$ is an ordered set, so (e.g.) $i_1$ represents the first visit to location $i$. Given a node $v \in V$, the location associated with this node is denoted by $i(v)$. The set $V = \cup_{i \in I} V_i$ is the node-set of a graph $G = (V, A)$, where $A$ contains the arcs connecting nodes associated with different locations. For a given subset $S$ of nodes, we write $\delta^+(S) = \{(v, w) \in A : v \in S, w \notin S\}$ and $\delta^-(S) = \{(v, w) \in A : v \notin S, w \in S\}$.

The cost $c_a$ of each arc $a$ is given. For each location $i \in I$, let $p_i$ be the units of product in $i$ before starting the service, and $p'_i$ the desired units of product in $i$ after the end of the service. Let $d_i = p'_i - p_i$ be the demand of location $i$. When $d_i > 0$ the location $i$ is a delivery customer, which means that it needs product from the system. When $d_i < 0$ the location $i$ is a pickup customer, which means that it provides product to the system. We assume that $\sum_{i \in I} d_i = 0$, so the number of product units in the system remains equal before and after performing the vehicle service. We also assume that all locations must be served by the vehicle, including those customers with zero demand. In addition, we are given with a capacity $q_i$ associated to location $i$, meaning that this location can store between 0 and $q_i$ units of the commodity. The capacity of the vehicle is $Q$, and is also assumed to be known.

In the bike-sharing application, $I$ represents the stations, $p_i$ and $p'_i$ are the initial and final (respectively) numbers of bikes at station $i$, $q_i$ is the number of slots for bikes at station $i$, and $Q$ is the maximum number of bikes that the vehicle can transport at each time.

To present a mathematical formulation, let us consider the following variables to represent a SD1PDTSP solution. For each arc $a \in A$, a binary variable $x_a$ assumes value 1 if and only if $a$ is part of the route, and a continuous variable $f_a$ is the number of units of product in the vehicle when going through arc $a$. For each node $v \in V$, a binary variable $y_v$ assumes value 1 if and only if the visit $v$ is performed by the vehicle, and a continuous variable $g_v$ is the number of units collected (if positive) or delivered (if negative) when performing the visit $v$. Note that, given a solution, the final number of units of product at customer $i$ is given by $p_i + \sum_{v \in V_i} g_v$, which must be $p'_i$.

Then a mathematical formulation of SD1PDTSP is given by:

$$\min \sum_{a \in A} c_a x_a \tag{1}$$

subject to:

$$y_{i_1} = 1 \qquad \forall i \in I \tag{2}$$

$$y_{i_k} \geq y_{i_{k+1}} \qquad \forall i \in I, \forall k = 1, \ldots, m-1 \tag{3}$$

$$\sum_{a \in \delta^+(v)} x_a = \sum_{a \in \delta^-(v)} x_a = y_v \qquad \forall v \in V \tag{4}$$

$$\sum_{a \in \delta^+(S)} x_a \geq y_v + y_w - 1 \qquad \forall S \subseteq V, \forall v \in S, \forall w \in V \setminus S \tag{5}$$

$$\sum_{a \in \delta^+(v)} f_a - \sum_{a \in \delta^-(v)} f_a = g_v \qquad \forall v \in V \tag{6}$$

$$0 \leq f_a \leq Q x_a \qquad \forall a \in A \tag{7}$$

$$\sum_{l=1}^{m} g_{i_l} = d_i \qquad \forall i \in I \tag{8}$$

$$0 \leq p_i + \sum_{1 \leq k \leq l} g_{i_k} \leq q_i \qquad \forall i \in I, \forall l = 1, \ldots, m-1 \tag{9}$$

$$-q_i y_{i_l} \leq g_{i_l} \leq q_i y_{i_l} \qquad \forall i \in I, \forall l = 2, \ldots, m \tag{10}$$

$$y_v, x_a \in \{0, 1\} \qquad \forall v \in V, \forall a \in A. \tag{11}$$

Equations (2) force to visit once each location. Inequalities (3) impose an order to use the nodes of a location. These constraints avoid symmetrical solutions representing the same route. Equations (4) are the degree equations, ensuring that the vehicle enters and leaves each node $v$ with $y_v = 1$. Inequalities (5) are the subtour elimination constraints. Constraints (6)–(8) ensure that the load of the vehicle is able to satisfy the demand of each location. Constraints (9) guarantee that the storage of product in a location is always between 0 and its capacity. Note that case $l = m$ is useless because (8) is in the model and $0 \leq p'_i, p_i \leq q_i$ is assumed. Inequalities (10) impose that product can be delivered to or collected from a location in each visit. Note that case $l = 1$ is useless because (2) and (9).

Trivially, for a given $S$, inequality (5) is dominated by

$$\sum_{a \in \delta^+(S)} x_a \geq 1 \tag{12}$$

when there are locations $i$ and $j$ such that $i_1 \in S$ and $j_1 \in V \setminus S$. Otherwise, it is dominated by

$$\sum_{a \in \delta^+(S)} x_a \geq y_v \tag{13}$$

for all $v \in V \setminus S$ when $i_1 \in S$ for each $i \in I$, or for all $v \in S$ when $i_1 \in V \setminus S$ for each $i \in I$.

As pointed before, model (1)–(11) is also valid for special cases of the SD1PDTSP like the VRP and the SDVRP. In this problems the depot is the only pickup location that can be visited at most $k$ times while the customers are delivery locations. On the VRP the number of visits to a delivery location is limited to 1, while on the SDVRP it is limited to $m$. The next section propose a technique to solve the SD1PDTSP to optimality based on the formulation (1)–(11) and therefore it is an approach to solve VRP and SDVRP.

We conclude this section by observing that some variants of the SD1PDTSP can be easily modelled by slightly modifying the formulation (1)–(11):

- A first variant concerns relaxing the constraints that customers with zero-demand must be visited. This is easily obtained by simply removing equations (2) from the formulation. In this case inequalities (12) and (13) are valid only under some conditions. For example, inequalities (12) is valid when there are customers $i$ and $j$ with non-zero demand such that $V_i \subseteq S$ and $V_j \subseteq V \setminus S$.

- On some applications it is desired that any product collected from a customer must be delivered by the vehicle before returning to the depot. This constraint is important when the whole route will not be performed by a single vehicle, but by a fleet of vehicles. If $j$ represents the depot with zero demand, the constraint can be modelled by simply adding $f_a = 0$ for all $a \in \delta^+(V_j) \cup \delta^-(V_j)$.

- More general, one can be interested in fixing the load of the vehicle when leaving (or entering) the depot, For example, if $j$ represents the depot, adding $f_a = Qx_a$ for all $a \in \delta^+(V_j)$ forces the vehicle to leave the depot with full load. These equations are useful to solve VRP and SDVRP instances with the model and algorithm described in this paper.

- By replacing (10) with

$$0 \leq g_v \leq q_{i(v)} y_v \qquad \forall v \in V : d_{i(v)} > 0$$
$$-q_{i(v)} y_v \leq g_v \leq 0 \qquad \forall v \in V : d_{i(v)} < 0$$

  one can forces that a vehicle never stores a unit of product in an intermediate location between it was collected and it will be delivered. In other words, these inequalities avoid preemption during the route which may be considered as disturbing the customer in some applications.

- Archetti et al. [1] assumes that, if a customer is visited, it has also to be served. This implies that the customers visited in any trip receive or deliver at least one unit, and it can be modelled with

$$1 \leq g_v \leq q_{i(v)} y_v \qquad \forall v \in V : d_{i(v)} > 0$$

$$-q_{i(v)}y_v \leq g_v \leq -1 \qquad \forall v \in V : d_{i(v)} < 0.$$

- Gulczynski et al. [13] imposes a minimum amount of product served to a customer in each visit. In this way, a customer is interrupted with a visit only when the delivery or pickup is substantial in amount or value each time. To this end, (10) should be replaced by:

$$rd_{i(v)}y_v \leq g_v \leq q_{i(v)}y_v \qquad \forall v \in V : d_{i(v)} > 0$$
$$-q_{i(v)}y_v \leq g_v \leq -rd_{i(v)}y_v \qquad \forall v \in V : d_{i(v)} < 0,$$

  where $r$ is the percentage of the customer demand defining the minimum amount (e.g. 35%). Clearly this parameter has a high impact on the maximum number of visits (e.g. $r = 0.35$ implies $m = 2$). To our knowledge, this is the first mathematical formulation known for the SDVRP with minimum delivery amounts, and the solution approach described in the next section is the first exact method to solve it.

## 4. Solution Approach

The formulation given in the previous section suggests a branch-and-cut algorithm to solve the SD1PDTSP. We now describe important elements for an effective implementation.

### *4.1. Lower Bound*

Computing a good lower bound on the optimal objective value is fundamental for a successful branch-and-bound approach. We propose to achieve it by solving the linear programming relaxation of the integer model described in the previous section. The large set of variables is the major drawback, but it can be afforded by using the min-cut max-flow equivalence to eliminate the $f$ variables. Indeed, the linear system defined by (6) and (7) can be replaced by

$$\sum_{a \in \delta^+(S)} x_a \geq \frac{1}{Q} \sum_{v \in S} g_v \qquad \forall S \subset V. \tag{14}$$

These inequalities are similar to the so-called *fractional capacity cuts*, known in the vehicle routing literature. The major difference is that the right-hand side

is not a constant for us, and therefore rounding up would create a non-linear inequality. Still this is not always the case, and we now present a particular situation where rounding up keeps the linearity of the inequality, while at the same time it strengthen the linear programming relaxation of the formulation.

Consider $S$ defined by the union of a collection of $V_i$. Let $C$ be a subset of customers, and assume that $S = \cup_{i \in C} V_i$. Then the previous inequality for $S$ is dominated by the following one:

$$\sum_{a \in \delta^+(S)} x_a \geq \left\lceil \frac{1}{Q} \left| \sum_{i \in C} d_i \right| \right\rceil . \tag{15}$$

These inequalities can be heuristically separated by first finding the most violated inequality without the rounded up operation. To this end, one must shrink each set $V_i$ of the graph $G$ in a single node, and then solve a max-flow problem on a properly defined network with an artificial node. We skip the details of this procedure because they are well-known in the VRP literature.

Another interesting family of inequalities are

$$g_u x_{(u,v)} \leq f_{(u,v)} \leq (Q - g_v) x_{(u,v)} \qquad \forall (u,v) \in A.$$

They can be linearized by using that

$$g_u x_{(u,v)} \geq g_u - q_{i(u)}(1 - x_{(u,v)}) \qquad \forall (u,v) \in A.$$

The system defined by (6) and these linear bounds on the continuous variables can be replaced by

$$Qx(\delta^+(S)) \geq \sum_{u \in S} \left( g_u + \sum_{v \notin S} (2g_v - q_{i(u)} - q_{i(v)} + q_{i(u)} x_{(u,v)} + q_{i(v)} x_{(v,u)}) \right) \qquad \forall S \subset V. \tag{16}$$

Inequalities (14) and (16) ensure the existence of a load for the vehicle on a route such that $\max\{0, g_u\} x_{(u,v)} \leq f_{(u,v)} \leq \min\{Q - g_v, Q\} x_{(u,v)}$.

Further strengthenings arise by using valid inequalities for the Generalized TSP polytope. In particular, we use the generalized 2-matching inequalities:

$$\sum_{a \in \delta^+(H) \setminus T} x_a \geq \sum_{a \in T} x_a - \sum_{v \in H} y_v + 1 \tag{17}$$

for all $H \subset V$ and $T \subset \delta^+(H)$ with $|T|$ odd. Although these inequalities can be separated in polynomial time, we use the heuristic procedure described in Fischetti et al. [12] to incorporate them in our branch-and-cut approach.

Using Benders' Decomposition one can also eliminate the $g$ variables from the model, and iteratively works with a master model based only on the binary variables $(x, y)$ and with a subproblem defined by the continuous variables $(f, g)$. The master model is defined by (1)–(5), the Benders' cuts that we define later in this section, and the integrality constraints (11). At each iteration, the master problem is solved. If it is infeasible, then the original SD1PDTSP is infeasible ($m$ or $Q$ are too small). Otherwise, let $(x^*, y^*)$ the optimal solution of the master problem. Then, the subproblem checks the feasibility of the linear system (6)–(10) on the variables $(f, g)$. If this system is feasible then the master includes all the necessary constraints and $(x^*, y^*)$ defines an optimal SD1PDTSP route. Otherwise, the infeasible system can be considered as a primal problem whose dual problem is unbounded, the ray proving this problem status defines a new Benders' cut that must be included in the master problem, and another iteration is required.

Although, the classical Benders' Decomposition would suggest to keep (11) in the master problem solved at each iteration, a more efficient approach is to replace such requirement by its linear-programming relaxation and apply a branching scheme when the iterative procedure stops and $(x^*, y^*)$ is not integer. Indeed, the subproblem is a linear program even when $(x^*, y^*)$ contains fraction values. The overall approach is then a branch-and-bound procedure where the iterative procedure is applied on each node.

We now show the details of the Benders' cut that can be generated on each iteration. Let $\alpha_v$ be the dual variable of equation (6), $\beta_a$ of equation (7), $\gamma_i$ of the right-hand side inequality in (8), $\delta^1_{il}$ of the left-hand side inequality in (9), $\delta^2_{il}$ of the right-hand side inequality in (9), $\lambda^1_{il}$ of the left-hand side inequality in (10), and $\lambda^2_{il}$ of the right-hand side inequality in (10). When $(x^*, y^*)$ is the

solution from the masster problem, the dual problem maximizes

$$\sum_{i\in I}\gamma_i d_i + \sum_{a\in A}\beta_a Q x_a^* + \sum_{i\in I}\sum_{l=1}^{m-1}\left(\delta_{i_l}^2(q_i-p_i)-\delta_{i_l}^1 p_i\right) + \sum_{i\in I}\sum_{l=2}^{m}(\lambda_{i_l}^2-\lambda_{i_l}^1)q_i y_{i_l}^*$$

subject to:

$$\begin{aligned}
\alpha_u - \alpha_v + \beta_{(u,v)} &\le 0 \qquad && \forall (u,v)\in A\\
-\alpha_{i_1} + \gamma_i + \sum_{k=1}^{m-1}(\delta_{i_k}^1+\delta_{i_k}^2) &= 0 && \forall i\in I\\
-\alpha_{i_l} + \gamma_i + \sum_{k=l}^{m-1}(\delta_{i_k}^1+\delta_{i_k}^2) + \lambda_{i_l}^1 + \lambda_{i_l}^2 &= 0 && \forall i\in I, \forall l = 2,\ldots,m
\end{aligned}$$

with $\alpha_v$ unsigned, $\beta_a \le 0$, $\gamma_i$ unsigned, $\delta_v^1 \ge 0$, $\delta_v^2 \le 0$, $\lambda_v^1 \ge 0$ and $\lambda_v^2 \le 0$.

Note that $\lambda_{i_l}^1 + \lambda_{i_l}^2 = \alpha_{i_l} - \gamma_i - \sum_{k=l}^{m-1}(\delta_{i_k}^1+\delta_{i_k}^2)$ is unsigned. Since $\lambda_{i_l}^1 \ge 0$ and $\lambda_{i_l}^2 \le 0$ then $\lambda_{i_l}^1 - \lambda_{i_l}^2 = |\alpha_{i_l} - \gamma_i - \sum_{k=l}^{m-1}(\delta_{i_k}^1+\delta_{i_k}^2)|$. Observe also that $\gamma_i = \alpha_{i_1} - \sum_{k=1}^{m-1}(\delta_{i_k}^1+\delta_{i_k}^2)$ for all $i\in I$. Since the total demand of product in the system is zero, equations (6) can all be replaced by smaller-or-equal inequalities, and equations (7) can all be replaced by greater-or-equal inequalities if $d_i > 0$ and by smaller-or-equal inequalities otherwise. Then $\beta_a$ can be required to be non-negative for all $a$, and $\gamma_i$ can be required to be non-negative if $d_i > 0$ and non-positive otherwise. Introducing also a new variable $\beta'_a$ to replace $-\beta_a$, the above dual problem can be reformulated as maximizing

$$\sum_{i\in I}\left(\alpha_{i_1} - \sum_{k=1}^{m-1}(\delta_{i_k}^1+\delta_{i_k}^2)\right)d_i - \sum_{a\in A}\beta'_a Q x_a^* +$$

$$\sum_{i\in I}\sum_{l=1}^{m-1}\left(\delta_{i_l}^2(q_i-p_i)-\delta_{i_l}^1 p_i\right) - \sum_{i\in I}\sum_{l=2}^{m}\left|\alpha_{i_l} - \alpha_{i_1} + \sum_{k=1}^{l-1}(\delta_{i_k}^1+\delta_{i_k}^2)\right|q_i y_{i_l}^*$$

subject to:

$$\alpha_u - \alpha_v \le \beta'_{(u,v)} \qquad \forall (u,v)\in A$$

with $\alpha_v \ge 0$, $\beta'_a \ge 0$, $\delta_v^1 \ge 0$ and $\delta_v^2 \le 0$.

The extreme rays of the polyhedral cone on the $(\alpha,\beta')$ space are known. Hernández-Pérez and Salazar-González [14] shows that each extreme ray is associated with a set $S\subset V$ and defined by $\alpha_v = 1$ iff $v\in S$ and by $\beta'_a = 1$ iff

$a \in \delta^+(S)$. This allows us to avoid the unboundness of the dual problems by strengthening the master problem with the following set of constrains for each subset $S \subset V$:

$$Q \sum_{a \in \delta^+(S)} x_a + \sum_{i \in I} \sum_{l=2}^{m} \left| \alpha_{i_l} - \alpha_{i_1} + \sum_{k=1}^{l-1} (\delta^1_{i_k} + \delta^2_{i_k}) \right| q_i y_{i_l}$$

$$\geq \sum_{i \in I} \left( \alpha_{i_1} d_i - \sum_{k=1}^{m-1} \left( (\delta^1_{i_k} + \delta^2_{i_k}) d_i + \delta^1_{i_k} p_i - \delta^2_{i_k} (q_i - p_i) \right) \right) \tag{18}$$

where $\delta^1_v \geq 0$, $\delta^2_v \leq 0$ and $\alpha_v = 1$ iff $v \in S$. These are the *Benders' cuts*. Although this family contains an infinite number of inequalities, only an exponential number are necessary. They could be described by introducing another set $S' \subset V$ to represent the nodes $v$ with $\delta^1_v + \delta^2_v \neq 0$, which motivate an alternative procedure to find a violated Benders' cut by a given $(x^*, y^*)$, i.e. solving the separation problem. To be more precise, one way of solving this separation problem is using a linear-programming solver to find a ray $(\alpha, \beta, \gamma, \delta)$ that proves the unboundedness of the dual subproblem when 0 is not the optimal solution. Not surprisingly, the separation problem can be solved in polynomial time. However, we now describe another alternative approach based on considering the subproblem (6)–(10) for $(x^*, y^*)$ as a max-flow problem in a graph.

Let us consider a graph $G^*$ where the node set contains two dummy nodes ($s$ and $t$), and two nodes ($i_l$ and $i^l$) for each $i_l \in V$. See Figure 1, where we illustrate the graph associated to a location $i \in I$ on a SD1PDTSP instance with $m = 4$. The node $i_l$, represented by a circle in the figure, controls the product loaded or unloaded from the vehicle to the customer in the $l$-th visit. The node $i^l$, represented by a box in the figure, controls the inventory of the customer before and after the $l$-th visit. The arcs of $G^*$ join the nodes $i_l$ exactly as in $A$, each one $a$ with capacity $Qx^*_a$. $G^*$ also has arcs connecting $i_l$ and $i^l$ for all $l = 1, \ldots, m$ to model whether the service during visit $i_l$ is a delivery (flow through arc $(i_l, i^l)$) or a pickup (flow through arc $(i^l, i_l)$). These arcs represent the product transferred between the vehicle and the inventory of a customer, and they both have capacity equal to $q_i y^*_{i_l}$. In addition, the arc set

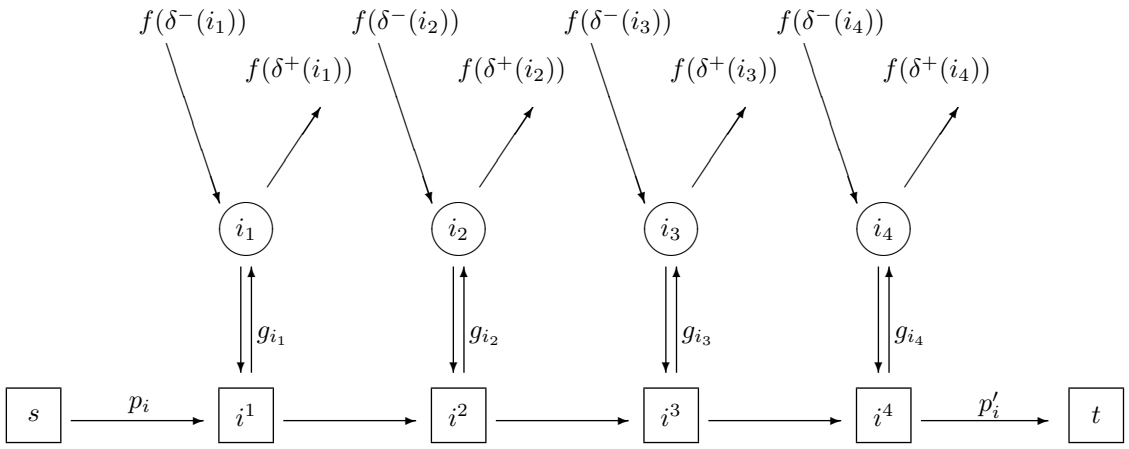


Figure 1: Graph representation to check feasibility of a route at location $i$ with $m = 4$.

of $G^*$ contains the $(s, i^1)$, $(i^m, t)$ and $(i^l, i^{l+1})$ for each $l = 1, \ldots, m-1$ and each customer $i \in I$. The flow through $(s, i^1)$ is fixed to $p_i$, representing the inventory at $i$ before the first service, and the flow through $(i^m, t)$ is fixed to $p'_i$, representing the inventory at $i$ after the last service. The capacity of arc $(i^l, i^{l+1})$ is $q_i$.

Using Hoffman [18], the linear system (6)–(10) for $(x^*, y^*)$ is feasible if and only if each cut in $G^*$ separating $t$ from $s$ has capacity at least $\sum_{i \in I} p_i$. Let us consider a cut separating $t$ from $s$. Let $S$ and $S'$ be the nodes of type $i_l$ and $i^l$, respectively, in the shore of $s$ in the cut. The capacity of this cut is given by $Q \sum_{a \in \delta^+(S)} x^*_a$, plus $q_i y^*_{i_l}$ for each $i \in I$ and $l = 1, \ldots, m$ such that $i_l \in S, i^l \notin S'$ or $i_l \notin S, i^l \in S'$, plus $q_i$ for each $i \in I$ and $l = 1, \ldots, m-1$ such that $i^l \in S', l^{l+1} \notin S'$, plus $p_i$ if $i^1 \notin S'$, plus $p'_i$ if $i^m \in S'$. Then, computing the maximum flow from $s$ to $t$ in $G^*$ allows us to detect whether the linear system if feasible or not, and to find an inequality violated by $(x^*, y^*)$ to include in the master problem when the system is infeasible.

Obsserve that it is possible to adapt the Benders' Decomposition and $G^*$ to also work with the variants annotated at the end of Section 3. For example, the requirement on the minimum amount of product served to a customer in each visit can be addressed in $G^*$ with one arc between $i_l$ and $i^l$, which is $(i_l, i^l)$ if

$d_i > 0$ and $(i^l, i_l)$ otherwise, with lower capacity $r|d_i|$.

Our lower bound technique considers dynamically the inequalities (5), (12), (13), (15), (17) and (18), i.e. we have separation procedures to add them to the master problem when needed. Inequalities (14) and (16) are not included in our master problem because they involve $g$ variables.

*4.2. Upper Bound*

The effectiveness of any branch-and-bound algorithm depends not only on the quality of the lower bound, but also on the procedure to find good upper bounds during the enumeration scheme. This latter requirement implies generating heuristic solutions for the SD1PDTSP, which is an $\mathcal{NP}$-hard problem. This is a major difference between SD1PDTSP and other related problems like SDVRP. For these other problems, heuristic approaches typically adapt solutions of the split-forbidden variant by using swaps and insertions to reduce the travel cost. On the SD1PDTSP, instead, there may not exist any solution to the demand-forbidden problem. Therefore, finding a heuristic solution for SD1PDTSP is a challenging optimization problem in itself.

We make use of the *infeasibility function* in Hernández-Pérez and Salazar-González [15] for a peace of route $P$ of the 1PDTSP. It is based on the following idea. Let $P$ be a location sequence $v_1, \cdots, v_t$, and let $l_i(P) = l_{i-1}(P) + g_{v_i}$ be the load of the vehicle leaving $v_i$ if it goes inside $v_1$ with load $l_0(P)$. Value $\max_{i=0}^{t}\{l_i(P)\} - \min_{i=0}^{t}\{l_i(P)\}$ represents the maximum variability of the load of the vehicle when going through $P$. This value must be in $[0, Q]$ in a feasible route $P$. For that reason, defining

$$\text{infeas}(P) := \max_{i=0}^{t}\{l_i(P)\} - \min_{i=0}^{t}\{l_i(P)\} - Q,$$

a path $P$ is feasible when $\text{infeas}(P) \leq 0$. Hernández-Pérez and Salazar-González [15] describe several operators to modified $P$ when it is not feasible, and potentially produce a feasible path $P'$ for 1PDTSP. On this problem, $g_v$ is the demand of the customer, which is a-priori known. On the SD1PDTSP the demand $g_v$ served when visiting $v$ is unknown, and therefore it is not possible to compute

the maximum variability for a given peace of route $P$. This may be considered as an advantage because there may be choices for which a path $P$ is feasible. On the other hand, this choices may not be enough to satisfy the demand of the customers. To overcome this disadvantage other paths may be necessary. This reasoning motivates us the following approach.

Our heuristic approach consists of at most $m$ iterations. Each iteration solves a TSP on a graph $G'$ defined by a subset of $I$. Although a heuristic TSP procedure would be acceptable, we used an optimal TSP procedure because the size of $I$ in our benchmark instances allows it. For $l = 1, \ldots, m$, the TSP solution at iteration $l$ defines a tour $P^l$ on $G$ visiting nodes $i_l$. In addition to compute a TSP solution, at iteration $l$ we also compute values $g_{i_l}$. When $l = 1$, they are chosen in $[-p_i, q_i - p_i]$ to keep infeas$(P^l) \leq 0$ while minimizing $\sum_{i \in I} |g_{i_1} - d_i|$. At iteration $l$, with $l = 2, \ldots, m$, we minimize $\sum_{i \in I} |\sum_{k=1}^{l} g_{i_k} - d_i|$ by selecting $g_{i_l}$ between $-p_i + \sum_{k=1}^{l-1} g_{i_k}$ and $q_i - p_i + \sum_{k=1}^{l-1} g_{i_k}$ such that infeas$(P^l) \leq 0$. This optimization problem is solved with a linear programming solver. Iteration 1 solves a TSP on a graph with $|I|$ nodes, while Iteration $l$ with $l \geq 2$ solves a TSP considering a node for each customer $i$ such that $\sum_{k=1}^{l-1} g_{i_k} \neq d_i$.

The iterative approach stops after iteration $l$ if $\sum_{k=1}^{l} g_{i_k} = d_i$ for all $i \in I$. In such case, a feasible SD1PDTSP route has been obtained by simply merging the tours $P^1, \ldots, P^l$ in a single tour. Tours $P^l$ and $P^{l+1}$ are merged a common node $i$ with the largest value $|g_{i_l} + g_{l+1}|$, and then the values $g_v$ in the new tour are recomputed with the above-described aim. If there is a location with $\sum_{k=1}^{m} g_{i_k} \neq d_i$ after iteration $m$ then the heuristic failed to produce a feasible SD1PDTSP solution.

The previous heuristic approach is the first step of our branch-and-bound algorithm. It is also executed when the linear-programming relaxation gives a solution $(x^*, y^*)$ and no violated cuts is found, i.e. at each node of search. In this case, the solution $(x^*, y^*)$ is used to affect the TSP's through the iterative approach. This is done by changing the cost $c_a$ of an arc $a$ with $(1 - x_a^*)c_a$.

*4.3. Branching*

Once no violated inequalities has been generated from a fractional solution $(x^*, y^*)$ and the heuristic procedure has been executed on this solution, the branching step creates two problems. To this end, we select a binary variable $y_v$ with fractional value $y_v^*$. When all values $y_v^*$ are integer, then we select a binary variable $x_a$ with fractional value $x_a^*$.

## 5. Computational Experiments

The branch-and-cut approach described in the previous section has been implemented in C++, using Cplex 12.5 as a framework. The code was executed on a computer with a Core 2 Duo CPU E8600 3.3 Ghz running Microsoft Windows 7. To evaluate the performance of our implementation we have considered five classes of instances.

Class I is based on the benchmark instance introduced in Mosheiov [24]. It has 25 locations, Euclidean distances, and demands $d_i$ between $-7$ and $+7$. We have defined $p_i = 7 - d_i$, $p'_i = 7$ and $q_i = 14$ for all locations. Different SD1PDTSP instances have been created by varying $Q$ and $m$. When $m = 1$ and $Q \geq 16$ then the optimal SD1PDVRP route is the optimal TSP tour. On these instances $\max\{|d_i| : i \in I\} = 7$, thus $m > 1$ when $Q < 7$.

Class II is based on the randomly-generated instances described in Hernández-Pérez and Salazar-González [15] for the 1PDTSP. They use Euclidean distances and demands $d_i$ between $-10$ and $+10$. We have defined $p_i = 10 - d_i$, $p'_i = 10$ and $q_i = 20$ for each location $i$, and created ten SD1PDTSP instances with $n = 20$ following their description. Then, for $m \in \{1, 2, 3\}$, we have considered $Q \in \{5, 6, 7\}$ to have instances where split may not be necessary, and $Q \in \{10, 12, 15\}$ to have instances where split is necessary.

Class III is based on benchmark instances from the VRP library [30]. These instances are Eil23, Eil30 and Eil50. They includes Euclidean distances and given customer demands $d'_i$, vehicle capacity $Q'$ and fleet size $k'$. For Eil23, $Q' = 4500$ and $k' = 3$. For Eil30, $Q' = 4500$ and $k' = 3$. For Eil50, $Q' = 160$

and $k' = 5$. For defining the SD1PDTSP instances we have used the given distances. The demands for the customers are the given demands, but with positive sign on even customers and negative sign on odd customers. More precisely, if $d'_i$ is the given VRP demand then we have used $p_i = 0$, $p'_i = d'_i$ and $q_i = 2p'_i$ when $i$ is even, and we have used $p_i = d'_i$, $p'_i = 0$ and $q_i = 2p_i$ when $i$ is odd. The depot is considered as a customer such that the total demand is zero. Each customer (including the depot) is allowed to be visited at most $m$ visits. We have enumerated a few values of $Q$ around $Q'$, and $m \in \{1, 2, 3\}$.

Class IV is also based on Eil23, Eil30 and Eil50, but now defined to solve the corresponding VRP and SDVRP instances. To this end, we have used the given distances and demands. We have used $p_i = 0$ and $q_i = p'_i = d'_i$ for each customer $i$. If $j$ represents the depot, we used $p_j = -\sum_i d'_i$, $p'_j = 0$ and $q_j = -2p_j$. We allowed the depot to have at most $k'$ visits, and each customer was allowed to be visited at most $m$ visits. No preemption of the product in a customer is allowed. Again, we have enumerated a few values of $Q$ around $Q'$, and $m \in \{1, 2, 3\}$. Note that in the SDVRP literature, the fleet size is typically ignored, thus allowing an unlimited number of return trips to the depot. Instead, we used the fleet size $k'$ to impose a maximum number of visits to the depot.

Class V are five SDVRP instances with 50 customers from Belenguer et al. [4] and nine SDVRP instances with up to 48 customers from Chen et al. [7]. Table 8 shows the number of customers $n - 1$, the fleet size $k$ (i.e. maximum number of visits to the depot) and the vehicle capacity $Q$ given in these instances. The first 5 instances are from Belenguer et al. [4] and, according to their description, the customers were placed randomly around a central depot, and each customer demand was generated randomly based on a high and low threshold, with a vehicle capacity of 160. The last 9 instances are from Chen et al. [7] and, according to their description, the customer were placed on rings surrounding a central depot and each customer demand was either 60 or 90, with a vehicle capacity of 100. The parameters $p_i$, $p'_i$ and $q_i$ have been defined as in Class IV. We have considered the distances to be rounded integer numbers. The algorithm performances are similar when these numbers are not rounded.

Tables 1 and 2 refer to Class I, i.e. SD1PDTSP instances, all identical except with different vehicle capacities. Table 1 shows results when $Q \geq 7$, i.e. when the vehicle capacity is enough to serve each customer individually. The table shows the optimal value $LB$ of the linear programming relaxation before the first branch, the travel cost $UB$ of the optimal SD1PDTSP solution, the computational time *rtime* when $LB$ was computed, and the total computational time *ttime*. Times are in seconds of our computer. The table shows that all instances were solved to optimality in a few seconds, although the computational time increases with $m$ due to the larger size of the mathematical formulation. Another observation is that there is benefit of splitting when $Q < 10$ but not when $m > 2$. Table 2 shows results when $Q < 7$, i.e. when some customers must be visited more than once. In this situation we have tried several values of $m$ such that $m \cdot Q \geq 7$, as reported in the table. It is worth to observe that the best route was found with $m := \max\lceil\{|d_i| : i \in I\}/Q\rceil$, as it happens when $Q \geq 10$.

Tables 3 and 4 refer to instances in Class II. As it happens with Class I, split helps to reduce the travel cost when the vehicle capacity is slightly over the maximum customer demand. On these instances the gain is obtained by allowing two visits, while when $m > 2$ the mathematical formulation is simply larger but with no better solution than $m = 2$. Indeed, we achieved our time limit (2 hours) on some instances when $m = 3$. Table 5 allows us to measure the benefit of using our cut-generation approach versus using a model with the flow variables explicitly. We have used the instances in Table 3 with $m = 2$. Since the cut-generation approach is a Benders' decomposition algorithm strengthened with additional inequalities, the lower bounds computed during the branch-and-bound process may be larger. The table shows the lower bound and the computational time before the first branching when $m = 2$. It also shows the numbers of inequalities generated before the first branching during the cutting-generation approach: *bend* is the number of (18), *sec* is the number of (12)–(13), *round* is the number of (15), and *2m* is the number of (17). Comparing the lower bounds and computational times, there is a clear benefit of using the

cut-generation approach versus using the flow variables explicitly.

Table 6 refers to instances in Class III, showing similar performances than SD1PDTSP instances in Classes I and II. Increasing $m$ implies weakening the lower bound at the root node, and increasing the computational time due to the size of the model. On our instances, the splitting variant allow to find shorter routes than the non-splitting variant only when $m = 2$ and on two instances (`Eil30` with $Q = 3100, 3200$). On these particular instances, one customer is visited twice: in the first visit, the vehicle delivers more than the customer demand, and in the second visit the vehicle pickups the extra amount that was delivered before. In other words, there is a cost reduction in the route due to the preemption that our formulation allows. It is worth mentioning that our results for $m = 1$ are similar to the ones in Hernández-Pérez and Salazar-González [14] and therefore one could expect to solve to proven optimality instances with a larger number of customers. However, the scope of our study is oriented to $m > 1$ and, under this condition, solving larger instances requieres higher computational times.

Table 7 shows the performance of our implementation to solve VRP instances ($m = 1$) and SDVRP instances ($m > 1$). Once again, using data from the VRP library, the splitting variant found routes with smaller travel costs than the non-splitting variant on a few instances, but no advantage was found on allowing a customer to be visited more than twice. Column $\#s$ shows the number of customers needing more than one visit in the optimal route. It is interesting to note that this number is small. Column $\%a$ shows the minimum percentage of demand served to a customer in a visit.

Table 8 shows the performance of our implementation to solve SDVRP instances from the literature. These instances have been introduced by other authors to evaluate lower bounds and heuristic approaches. Column UB' shows the value of the best known solution, taken from [1], except S51D4 which is from [7]. We performed experiments with $m = 2$ and 3, but the table reports the results with $m = 2$ because (as it occurred with other instances) no better solution was found with $m = 3$. It is interesting to observe that our approach,

| | $m=1$ | | | | $m=2$ | | | | $m=3$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $Q$ | LB | UB | rtime | ttime | LB | UB | rtime | ttime | LB | UB | rtime | ttime |
| 7 | 5433.5 | **5729** | 4.1 | 61.4 | 5272.9 | **5326** | 7.5 | 8.4 | 5093.0 | 5326 | 19.4 | 278.5 |
| 8 | 5253.5 | **5340** | 2.6 | 11.7 | 5055.5 | **5262** | 6.2 | 19.9 | 4668.2 | 5262 | 15.7 | 383.9 |
| 9 | 5040.0 | **5040** | 0.7 | 0.8 | 4924.8 | **5033** | 5.1 | 19.2 | 4610.4 | 5033 | 14.3 | 83.7 |
| 10 | 4949.5 | 4981 | 0.6 | 0.7 | 4764.9 | 4981 | 2.7 | 3.8 | 4506.3 | 4981 | 10.1 | 26.1 |
| 11 | 4778.9 | 4817 | 0.7 | 0.8 | 4684.5 | 4817 | 4.3 | 6.5 | 4469.1 | 4817 | 9.9 | 25.8 |
| 12 | 4757.2 | 4817 | 0.7 | 1.6 | 4479.8 | 4817 | 5.1 | 38.1 | 4435.4 | 4817 | 13.9 | 81.9 |
| 13 | 4584.0 | 4631 | 0.5 | 0.6 | 4436.2 | 4631 | 1.0 | 5.4 | 4425.5 | 4631 | 9.2 | 22.4 |
| 14 | 4479.0 | 4479 | 0.1 | 0.1 | 4423.1 | 4479 | 1.4 | 1.6 | 4433.6 | 4479 | 1.1 | 3.8 |
| 15 | 4479.0 | 4479 | 0.0 | 0.0 | 4479.0 | 4479 | 1.2 | 1.7 | 4434.3 | 4479 | 2.3 | 3.6 |
| 16 | 4434.0 | 4434 | 0.0 | 0.0 | 4434.0 | 4434 | 0.1 | 1.1 | 4434.0 | 4434 | 3.2 | 7.2 |

Table 1: Class I. SD1PDTSP instances based on data inMosheiov [24]

| $Q$ | $m$ | LB | UB | rtime | ttime |
|---|---|---|---|---|---|
| 1 | 7 | 21832.0 | 21832 | 85.7 | 209.0 |
| 2 | 4 | 11390.5 | 12070 | 721.5 | 3782.4 |
| 3 | 3 | 8527.8 | 9199 | 14.1 | 2503.1 |
| 4 | 2 | 7243.1 | 7791 | 8.4 | 1699.9 |
| 4 | 3 | 7018.7 | 7791 | 28.2 | 415.7 |
| 5 | 2 | 6565.0 | 6760 | 4.1 | 25.5 |
| 5 | 3 | 6262.8 | 6760 | 23.8 | 320.5 |
| 6 | 2 | 5434.9 | 5773 | 6.0 | 11.8 |
| 6 | 3 | 5362.2 | 5773 | 20.7 | 51.6 |

Table 2: Class I. SD1PDTSP instances based on data in Mosheiov [24]

even if it is intended to solve a more general and complex problem, it succeeded in finding better heuristic solutions than previously known for several SDVRP instances.

Table 9 shows the results when a minimum amount of product must be pickup from or delivery to the customer in each visit. This amount is measure as a percentage $r\%$ shown in the table. As expected, the problem becomes more complicated to solve and the objective value of the routes increase.

## 6. Conclusions

This paper has introduced a new routing problem in which a capacitated vehicle is used to transport a single commodity from pickup locations to delivery locations, and multiple visits to the same location are allowed. Several

| | | $m = 1$ | | | | $m = 2$ | | | | $m = 3$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | $Q$ | LB | UB | rtime | ttime | LB | UB | rtime | ttime | LB | UB | rtime | ttime |
| n30A | 10 | 5724.8 | **6727** | 12.6 | 1253.9 | 5654.1 | **6256** | 12.1 | 2266.4 | 5597.4 | 6256 | 22.8 | 4716.4 |
| n30A | 12 | 5583.8 | 5782 | 7.0 | 131.6 | 5111.6 | 5782 | 10.7 | 281.7 | 5008.6 | 5782 | 18.6 | 1937.2 |
| n30A | 15 | 5105.2 | **5595** | 5.8 | 91.2 | 5024.8 | **5465** | 7.5 | 137.6 | 5007.8 | 5465 | 15.1 | 399.4 |
| n30B | 10 | 6193.2 | 6603 | 7.3 | 165.8 | 5478.6 | 6603 | 6.8 | 591.1 | 5304.5 | 6603 | 14.8 | 1527.0 |
| n30B | 12 | 5522.0 | **6229** | 6.1 | 96.3 | 5003.3 | **6152** | 8.3 | 416.2 | 4869.7 | 6152 | 8.3 | 619.7 |
| n30B | 15 | 5094.0 | 5631 | 3.8 | 53.5 | 4632.0 | 5631 | 4.2 | 119.1 | 4454.7 | 5631 | 6.9 | 407.3 |
| n30C | 10 | 5215.5 | **6486** | 14.9 | 1197.9 | 5149.3 | **6348** | 13.4 | 2278.5 | 5005.1 | 6348 | 30.8 | 4289.7 |
| n30C | 12 | 5334.3 | **5456** | 5.3 | 69.1 | 5245.3 | **5367** | 5.3 | 92.9 | 5069.6 | 5367 | 12.7 | 293.4 |
| n30C | 15 | 4876.5 | 5181 | 1.9 | 14.2 | 4788.8 | 5181 | 2.7 | 42.8 | 4715.2 | 5181 | 7.7 | 176.8 |
| n30D | 10 | 5450.0 | **6577** | 22.9 | 2698.4 | 5279.2 | **6380** | 20.4 | 3811.8 | 5125.6 | 6380 | 26.9 | 4900.9 |
| n30D | 12 | 5368.5 | **6256** | 8.0 | 201.9 | 5095.9 | **6025** | 23.7 | 1907.1 | 5046.1 | 6025 | 23.5 | 3501.8 |
| n30D | 15 | 5285.5 | **5577** | 4.2 | 33.6 | 5170.6 | **5568** | 8.2 | 95.2 | 5029.4 | 5568 | 9.4 | 518.1 |
| n30E | 10 | 5691.2 | **6070** | 6.8 | 588.1 | 5402.5 | **6052** | 8.1 | 942.6 | 5253.1 | 6052 | 15.5 | 1433.8 |
| n30E | 12 | 5293.0 | **5876** | 1.5 | 18.3 | 5213.0 | **5762** | 3.9 | 71.7 | 5107.8 | 5762 | 11.7 | 234.9 |
| n30E | 15 | 5326.0 | 5416 | 1.7 | 7.8 | 4905.3 | 5416 | 3.1 | 42.8 | 4745.6 | 5416 | 5.1 | 97.6 |
| n30F | 10 | 5392.5 | **5737** | 11.5 | 600.4 | 5225.0 | **5727** | 14.7 | 1008.4 | 5006.9 | 5727 | 22.5 | 2284.3 |
| n30F | 12 | 5189.6 | **5260** | 0.7 | 3.2 | 5176.5 | **5259** | 4.3 | 62.1 | 4901.7 | 5259 | 10.9 | 188.2 |
| n30F | 15 | 4692.0 | 4893 | 0.1 | 1.5 | 4591.0 | 4893 | 2.7 | 30.8 | 4349.5 | 4893 | 7.1 | 145.7 |
| n30G | 10 | 8705.8 | **9305** | 16.0 | 2135.1 | 8641.7 | **9005** | 26.9 | 3994.7 | 8526.1 | 9005 | 28.1 | 7200.0 |
| n30G | 12 | 7438.0 | **8497** | 10.5 | 692.2 | 6066.8 | **8264** | 24.1 | 3049.5 | 5973.5 | 8264 | 21.9 | 6433.7 |
| n30G | 15 | 6530.0 | **7424** | 6.3 | 211.4 | 6453.3 | **7226** | 6.4 | 691.7 | 6324.5 | 7226 | 27.2 | 4659.7 |
| n30H | 10 | 5191.5 | **6433** | 10.7 | 457.9 | 5020.4 | **6164** | 30.5 | 4598.6 | 4913.7 | 6164 | 26.3 | 6973.4 |
| n30H | 12 | 5481.9 | **6025** | 5.3 | 219.5 | 5367.5 | **5947** | 20.3 | 1533.5 | 5239.3 | 5947 | 24.7 | 4217.6 |
| n30H | 15 | 5244.0 | **5613** | 7.6 | 82.7 | 5143.5 | **5447** | 7.0 | 241.9 | 4997.5 | 5447 | 21.7 | 1815.3 |
| n30I | 10 | 5156.2 | **5864** | 8.1 | 310.4 | 4969.5 | **5596** | 26.6 | 4839.8 | 4705.4 | 5596 | 25.1 | 7200.0 |
| n30I | 12 | 4646.0 | **5154** | 5.7 | 59.7 | 4572.6 | **4991** | 6.8 | 140.6 | 4228.6 | 4991 | 19.8 | 1683.1 |
| n30I | 15 | 4671.5 | 4762 | 0.9 | 1.9 | 4547.9 | 4762 | 4.0 | 11.4 | 4399.1 | 4762 | 3.5 | 415.1 |
| n30J | 10 | 5865.7 | **6192** | 7.3 | 224.8 | 5601.4 | **6090** | 19.6 | 1712.5 | 5389.5 | 6090 | 22.9 | 3556.4 |
| n30J | 12 | 5322.8 | **5874** | 4.6 | 99.7 | 5241.8 | **5647** | 8.1 | 179.3 | 5165.8 | 5647 | 10.2 | 654.7 |
| n30J | 15 | 4894.0 | 5349 | 3.7 | 19.2 | 4476.3 | 5349 | 3.7 | 141.7 | 4301.6 | 5349 | 6.0 | 362.7 |

Table 3: Class II. SD1PDTSP instances based on Hernández-Pérez and Salazar-González [15]

| | | $m = 2$ | | | | $m = 3$ | | | |
|---|---|---|---|---|---|---|---|---|---|
| Name | $Q$ | LB | UB | rtime | ttime | LB | UB | rtime | ttime |
| n30A | 5 | 10096.5 | 10157 | 21.9 | 3376.1 | 9984.2 | 10157 | 27.9 | 6472.6 |
| n30A | 6 | 8841.9 | 8957 | 18.2 | 3203.8 | 8764.1 | 8957 | 24.4 | 5997.5 |
| n30A | 7 | 7941.5 | 8048 | 14.7 | 2152.7 | 7793.8 | 8048 | 18.9 | 4734.1 |
| n30B | 5 | 10114.0 | 10515 | 25.5 | 3614.9 | 10060.7 | 10515 | 26.1 | 6916.3 |
| n30B | 6 | 9155.3 | 9321 | 19.5 | 3004.7 | 9003.5 | 9321 | 22.3 | 6209.7 |
| n30B | 7 | 8200.7 | 8358 | 17.3 | 2971.3 | 8150.6 | 8358 | 20.8 | 5004.6 |
| n30C | 5 | 9058.8 | 9337 | 18.2 | 3519.5 | 8909.9 | 9337 | 27.8 | 6705.2 |
| n30C | 6 | 8125.9 | 8231 | 8.3 | 2386.4 | 8033.7 | 8231 | 25.3 | 4596.1 |
| n30C | 7 | 7287.8 | 7487 | 12.9 | 1246.9 | 7164.8 | 7487 | 24.6 | 4120.7 |
| n30D | 5 | 8749.5 | 8981 | 18.8 | 3839.6 | 8669.1 | 8981 | 28.0 | 7200.0 |
| n30D | 6 | 7904.6 | 8022 | 16.1 | 2477.9 | 7882.7 | 8022 | 27.3 | 6191.4 |
| n30D | 7 | 7213.4 | 7332 | 11.8 | 329.1 | 7148.3 | 7332 | 22.1 | 2530.6 |
| n30E | 5 | 9066.2 | 9273 | 22.4 | 4335.7 | 8994.5 | 9733 | 28.6 | 7200.0 |
| n30E | 6 | 7794.6 | 7936 | 19.1 | 3792.5 | 7646.5 | 7936 | 26.9 | 7200.0 |
| n30E | 7 | 7800.3 | 7936 | 18.7 | 3309.6 | 7754.9 | 7936 | 24.5 | 6008.4 |
| n30F | 5 | 8947.1 | 9121 | 21.5 | 4681.8 | 8788.6 | 9567 | 26.1 | 7200.0 |
| n30F | 6 | 8035.7 | 8127 | 10.6 | 2341.8 | 7998.2 | 8127 | 23.3 | 6413.5 |
| n30F | 7 | 7262.4 | 7449 | 7.7 | 490.3 | 7191.8 | 7449 | 12.7 | 1344.8 |
| n30G | 5 | 14893.2 | 15074 | 20.4 | 4375.3 | 14601.5 | 15601 | 29.9 | 7200.0 |
| n30G | 6 | 12789.2 | 12978 | 16.5 | 3661.9 | 12661.2 | 12978 | 29.0 | 6628.9 |
| n30G | 7 | 11149.7 | 11560 | 12.7 | 946.6 | 11063.7 | 11560 | 19.1 | 2671.2 |
| n30H | 5 | 9261.2 | 9463 | 26.1 | 6505.7 | 9210.6 | 9506 | 22.4 | 7200.0 |
| n30H | 6 | 8292.4 | 8399 | 20.9 | 5016.4 | 8201.1 | 8462 | 20.6 | 7200.0 |
| n30H | 7 | 7776.3 | 7813 | 19.9 | 3958.3 | 7619.5 | 7813 | 20.1 | 6193.7 |
| n30I | 5 | 7800.6 | 7983 | 23.7 | 6867.1 | 7768.1 | 8012 | 24.3 | 7200.0 |
| n30I | 6 | 7107.3 | 7249 | 20.6 | 3816.7 | 7081.4 | 7249 | 23.3 | 6843.6 |
| n30I | 7 | 6398.5 | 6512 | 8.1 | 1268.6 | 6322.7 | 6512 | 17.5 | 2997.9 |
| n30J | 5 | 8564.2 | 8793 | 11.3 | 2497.2 | 8474.8 | 8793 | 25.5 | 6846.1 |
| n30J | 6 | 7896.4 | 8017 | 6.9 | 957.5 | 7743.9 | 8017 | 25.0 | 5944.8 |
| n30J | 7 | 7268.4 | 7406 | 4.5 | 688.6 | 7167.8 | 7406 | 24.1 | 5041.9 |

Table 4: Class II. SD1PDTSP instances based on Hernández-Pérez and Salazar-González [15]

| | | without Benders | | | | | with Benders | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | $Q$ | LB | rtime | ttime | sub | Nnode | bend | sub | round | 2m | Nnode |
| n30A | 10 | 5422.6 | 36.1 | 6794.1 | 175 | 4799 | 101 | 95 | 75 | 9 | 2179 |
| n30A | 12 | 5082.0 | 14.9 | 595.3 | 125 | 3931 | 84 | 72 | 48 | 5 | 1871 |
| n30A | 15 | 4989.9 | 14.9 | 351.3 | 95 | 2865 | 51 | 31 | 31 | 2 | 1101 |
| n30B | 10 | 5397.3 | 15.3 | 1764.4 | 234 | 1543 | 110 | 85 | 46 | 21 | 757 |
| n30B | 12 | 4748.0 | 15.3 | 1201.7 | 193 | 1147 | 92 | 49 | 35 | 13 | 406 |
| n30B | 15 | 4534.6 | 10.5 | 396.1 | 104 | 1002 | 65 | 51 | 46 | 5 | 299 |
| n30C | 10 | 4953.4 | 35.7 | 6009.7 | 181 | 7868 | 162 | 105 | 84 | 14 | 3246 |
| n30C | 12 | 5018.1 | 12.4 | 304.9 | 148 | 3429 | 98 | 75 | 55 | 10 | 979 |
| n30C | 15 | 4676.7 | 12.1 | 209.5 | 94 | 979 | 84 | 78 | 31 | 5 | 297 |
| n30D | 10 | 5134.8 | 31.1 | 6998.3 | 219 | 7999 | 116 | 63 | 64 | 16 | 4543 |
| n30D | 12 | 4828.6 | 25.9 | 4741.5 | 131 | 5225 | 81 | 59 | 48 | 6 | 2334 |
| n30D | 15 | 5001.9 | 16.2 | 669.3 | 81 | 2954 | 54 | 58 | 35 | 2 | 1272 |
| n30E | 10 | 5390.1 | 25.4 | 3050.2 | 141 | 1806 | 254 | 96 | 69 | 7 | 595 |
| n30E | 12 | 5121.5 | 12.3 | 322.7 | 111 | 1035 | 138 | 82 | 45 | 2 | 410 |
| n30E | 15 | 4857.5 | 11.7 | 211.8 | 99 | 752 | 90 | 79 | 31 | 5 | 100 |
| n30F | 10 | 5035.3 | 26.6 | 3551.9 | 193 | 6571 | 85 | 55 | 43 | 10 | 3065 |
| n30F | 12 | 4640.5 | 9.3 | 197.1 | 182 | 4256 | 96 | 62 | 41 | 7 | 725 |
| n30F | 15 | 4446.5 | 9.1 | 115.1 | 121 | 165 | 52 | 48 | 25 | 8 | 14 |
| n30G | 10 | 8409.7 | 34.2 | 6977.5 | 135 | 8656 | 171 | 89 | 69 | 20 | 4659 |
| n30G | 12 | 5918.0 | 32.7 | 6032.4 | 105 | 6899 | 93 | 57 | 60 | 19 | 2345 |
| n30G | 15 | 6300.9 | 22.0 | 1673.9 | 96 | 5119 | 99 | 66 | 29 | 3 | 1987 |
| n30H | 10 | 4901.5 | 36.9 | 7200.0 | 174 | 3975 | 156 | 112 | 65 | 13 | 1663 |
| n30H | 12 | 5148.5 | 23.8 | 4147.9 | 112 | 2178 | 194 | 84 | 61 | 4 | 986 |
| n30H | 15 | 5001.4 | 16.1 | 809.4 | 89 | 1543 | 102 | 51 | 36 | 4 | 764 |
| n30I | 10 | 4811.2 | 37.4 | 7200.0 | 188 | 4710 | 69 | 69 | 57 | 8 | 2003 |
| n30I | 12 | 4435.3 | 10.3 | 359.9 | 123 | 1387 | 78 | 48 | 42 | 7 | 712 |
| n30I | 15 | 4336.8 | 8.9 | 101.7 | 91 | 867 | 63 | 32 | 18 | 9 | 122 |
| n30J | 10 | 5321.9 | 28.4 | 5908.6 | 146 | 10589 | 151 | 91 | 47 | 6 | 5416 |
| n30J | 12 | 5030.5 | 12.6 | 595.2 | 135 | 9939 | 73 | 80 | 35 | 5 | 4977 |
| n30J | 15 | 4268.7 | 7.4 | 395.6 | 94 | 9108 | 110 | 43 | 38 | 5 | 5101 |

Table 5: Class II. SD1PDTSP instances based on Hernández-Pérez and Salazar-González [15] with $m = 2$

| | | $m = 1$ | | | | $m = 2$ | | | | $m = 3$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | $Q$ | LB | UB | rtime | ttime | LB | UB | rtime | ttime | LB | UB | rtime | ttime |
| eil22 | 2500 | | | | | 330.3 | 371 | 1.2 | 2.6 | 329.0 | 371 | 3.4 | 15.6 |
| eil22 | 3000 | | | | | 320.0 | 359 | 1.1 | 2.3 | 319.5 | 359 | 3.1 | 14.7 |
| eil22 | 3300 | 307.0 | 327 | 0.0 | 0.1 | 302.5 | 327 | 1.0 | 2.1 | 301.6 | 327 | 3.0 | 13.2 |
| eil22 | 4000 | 297.0 | 311 | 0.0 | 0.1 | 293.0 | 311 | 0.3 | 1.7 | 288.0 | 311 | 2.6 | 9.9 |
| eil22 | 5000 | 286.0 | 294 | 0.0 | 0.0 | 285.5 | 294 | 0.1 | 1.1 | 282.0 | 294 | 2.1 | 7.1 |
| eil22 | 6000 | 275.1 | 278 | 0.0 | 0.0 | 274.0 | 278 | 0.1 | 0.9 | 274.0 | 278 | 0.8 | 2.9 |
| eil23 | 3500 | | | | | 450.9 | 527 | 1.1 | 3.5 | 435.1 | 527 | 4.1 | 16.6 |
| eil23 | 4100 | | | | | 461.5 | 527 | 0.7 | 2.9 | 458.0 | 527 | 4.2 | 14.0 |
| eil23 | 4200 | 526.0 | 527 | 0.1 | 0.1 | 517.3 | 527 | 0.9 | 3.2 | 499.2 | 527 | 4.9 | 11.3 |
| eil23 | 4300 | 505.2 | 525 | 0.1 | 0.1 | 497.5 | 525 | 0.2 | 2.8 | 462.6 | 525 | 4.7 | 8.7 |
| eil23 | 4400 | 505.2 | 525 | 0.1 | 0.2 | 497.5 | 525 | 0.9 | 2.9 | 462.6 | 525 | 2.4 | 8.6 |
| eil23 | 4500 | 502.0 | 506 | 0.1 | 0.2 | 491.2 | 506 | 1.0 | 2.2 | 473.9 | 506 | 1.2 | 7.1 |
| eil23 | 4700 | 493.0 | 496 | 0.0 | 0.1 | 446.0 | 496 | 0.3 | 2.0 | 422.8 | 496 | 1.8 | 6.9 |
| eil23 | 5000 | 496.0 | 496 | 0.0 | 0.1 | 446.0 | 496 | 0.1 | 1.9 | 422.8 | 496 | 1.0 | 3.4 |
| eil30 | 2900 | | | | | 401.8 | 417 | 2.1 | 43.9 | 389.1 | 417 | 2.0 | 72.5 |
| eil30 | 3000 | | | | | 401.8 | 417 | 1.5 | 32.4 | 389.1 | 417 | 1.6 | 62.9 |
| eil30 | 3100 | 402.0 | **411** | 0.2 | 0.7 | 400.1 | **406** | 0.9 | 11.5 | 381.6 | 406 | 1.4 | 50.4 |
| eil30 | 3200 | 402.0 | **411** | 0.2 | 0.9 | 400.1 | **406** | 1.1 | 11.0 | 381.6 | 406 | 1.9 | 53.1 |
| eil30 | 3300 | 400.6 | 403 | 0.2 | 0.5 | 399.4 | 403 | 0.7 | 12.8 | 380.5 | 403 | 1.8 | 48.3 |
| eil30 | 3500 | 398.0 | 399 | 0.2 | 0.3 | 398.0 | 399 | 0.6 | 13.1 | 378.1 | 399 | 2.1 | 47.0 |
| eil30 | 3800 | 390.6 | 391 | 0.1 | 0.4 | 390.6 | 391 | 0.4 | 7.2 | 375.6 | 391 | 1.1 | 39.7 |
| eil30 | 4100 | 381.0 | 381 | 0.0 | 0.1 | 381.0 | 381 | 0.3 | 4.7 | 372.2 | 381 | 0.8 | 38.9 |
| eil33 | 4900 | | | | | 522.0 | 548 | 4.1 | 1106.4 | 520.0 | 548 | 7.6 | 3944.2 |
| eil33 | 5100 | | | | | 515.5 | 541 | 3.6 | 864.2 | 514.5 | 541 | 5.1 | 2032.1 |
| eil33 | 5600 | 453.6 | 474 | 0.2 | 0.3 | 451.8 | 474 | 3.3 | 246.8 | 436.0 | 474 | 5.3 | 1511.8 |
| eil33 | 6000 | 453.6 | 463 | 0.1 | 0.2 | 451.4 | 463 | 2.9 | 63.1 | 450.8 | 463 | 4.9 | 1151.6 |
| eil33 | 7000 | 445.5 | 449 | 0.1 | 0.3 | 445.0 | 449 | 1.8 | 42.7 | 444.6 | 449 | 5.1 | 967.4 |
| eil33 | 8000 | 445.1 | 448 | 0.0 | 0.1 | 444.9 | 448 | 1.7 | 49.3 | 444.3 | 448 | 4.3 | 697.9 |
| eil51 | 80 | 424.0 | 434 | 0.2 | 0.5 | 422.5 | 434 | 3.1 | 2309.4 | 418.5 | 434 | 8.0 | 4367.1 |
| eil51 | 100 | 421.5 | 430 | 0.1 | 0.3 | 416.7 | 430 | 2.4 | 2258.2 | 412.0 | 430 | 6.3 | 3522.9 |
| eil51 | 160 | 416.5 | 426 | 0.1 | 0.1 | 414.5 | 426 | 0.9 | 110.3 | 410.1 | 426 | 4.7 | 668.9 |

Table 6: Class III. SD1PDTSP instances based on data in the VRP library

| | | $m=1$ | | | | $m=2$ | | | | | | $m=3$ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Name | Q | LB | UB | rtime | ttime | LB | UB | rtime | ttime | #s | %a | LB | UB | rtime | ttime |
| eil22 | 5000 | 410.6 | 424 | 0.5 | 2.3 | 396.1 | 424 | 1.1 | 26.7 | 0 | 100 | 355.3 | 424 | 2.9 | 59.4 |
| eil22 | 6000 | 330.0 | 375 | 0.4 | 1.9 | 327.1 | 375 | 1.5 | 16.1 | 0 | 100 | 318.0 | 375 | 3.1 | 54.1 |
| eil22 | 7000 | 331.2 | 370 | 0.1 | 1.0 | 329.4 | 370 | 0.8 | 15.6 | 0 | 100 | 296.7 | 370 | 2.3 | 51.5 |
| eil23 | 4000 | | | | | 570.0 | 571 | 0.4 | 3.9 | 2 | 19 | 548.2 | 571 | 1.7 | 7.9 |
| eil23 | 4500 | 568.0 | 569 | 0.1 | 0.4 | 564.0 | 569 | 0.6 | 2.7 | 0 | 100 | 555.1 | 569 | 2.6 | 7.2 |
| eil23 | 5000 | 514.5 | 544 | 0.0 | 0.3 | 491.5 | 544 | 0.9 | 2.0 | 0 | 100 | 461.8 | 544 | 1.3 | 6.8 |
| eil30 | 3100 | 546.5 | 653 | 2.0 | 68.4 | 523.1 | 653 | 3.5 | 57.2 | 0 | 100 | 512.7 | 653 | 2.1 | 166.5 |
| eil30 | 4500 | 492.0 | **534** | 1.9 | 31.9 | 486.1 | **510** | 0.7 | 35.4 | 1 | 26 | 395.0 | 510 | 1.6 | 70.6 |
| eil30 | 9000 | 534.0 | **534** | 0.1 | 14.3 | 461.0 | **510** | 0.7 | 4.9 | 1 | 26 | 372.0 | 510 | 1.5 | 26.7 |
| eil33 | 7500 | 806.0 | **846** | 1.2 | 21.4 | 775.2 | **841** | 2.8 | 200.7 | 5 | 23 | 761.2 | 841 | 4.6 | 782.1 |
| eil33 | 8000 | 822.5 | 835 | 1.0 | 3.9 | 810.8 | 835 | 3.0 | 123.6 | 0 | 100 | 800.7 | 835 | 3.6 | 599.4 |
| eil33 | 9000 | 784.3 | 805 | 0.9 | 2.5 | 768.6 | 805 | 1.4 | 60.8 | 0 | 100 | 752.1 | 805 | 2.2 | 352.8 |
| eil51 | 160 | 509.3 | 521 | 1.0 | 7.8 | 508.4 | 521 | 2.1 | 891.3 | 0 | 100 | 489.1 | 521 | 7.6 | 2097.6 |

Table 7: Class IV. VRP and SDVRP instances from the VRP library.

| Name | $n-1$ | $k$ | $Q$ | LB | UB | rtime | ttime | #s | %a | UB' |
|---|---|---|---|---|---|---|---|---|---|---|
| S51D1 | 50 | 3 | 160 | 454.4 | 458 | 0.1 | 304.2 | 0 | 100 | 458 |
| S51D2 | 50 | 9 | 160 | 653.5 | 715 | 2.0 | 7200.0 | 2 | 19 | 726 |
| S51D3 | 50 | 15 | 160 | 921.1 | 950 | 1.7 | 7200.0 | 2 | 27 | 972 |
| S51D4 | 50 | 27 | 160 | 1517.3 | 1584 | 1.6 | 7200.0 | 4 | 32 | 1677 |
| S51D5 | 50 | 23 | 160 | 1279.8 | 1341 | 2.1 | 7200.0 | 3 | 27 | 1440 |
| S51D6 | 50 | 41 | 160 | 2110.5 | 2166 | 0.9 | 7200.0 | 2 | 18 | 2327 |
| S1 | 8 | 6 | 100 | 21414.0 | 22828 | 0.0 | 0.8 | 4 | 21 | 22828 |
| S2 | 16 | 12 | 100 | 70808.0 | 70828 | 0.9 | 45.9 | 11 | 16 | 70828 |
| S3 | 16 | 12 | 100 | 41076.6 | 43040 | 1.2 | 50.0 | 8 | 23 | 43040 |
| S4 | 24 | 18 | 100 | 56140.0 | 63062 | 2.3 | 671.5 | 12 | 25 | 63062 |
| S5 | 32 | 24 | 100 | 133773.0 | 138258 | 4.4 | 7200.0 | 10 | 27 | 138994 |
| S6 | 32 | 24 | 100 | 78220.2 | 83086 | 4.3 | 4004.8 | 16 | 32 | 83086 |
| S7 | 40 | 30 | 100 | 363794.1 | 363899 | 7.0 | 7200.0 | 11 | 33 | 364000 |
| S8 | 48 | 36 | 100 | 496695.0 | 506828 | 9.0 | 4992.2 | 4 | 24 | 506828 |
| S9 | 48 | 36 | 100 | 203094.4 | 203221 | 9.1 | 7200.0 | 8 | 13 | 204288 |

Table 8: Class V. SDVRP instances from Belenguer et al. [4] and Chen et al. [7], with $m = 2$.

| Name | $r$ | LB | UB | rtime | ttime | #s | %a |
|---|---|---|---|---|---|---|---|
| S51D2 | 20% | 642.7 | 716 | 1.2 | 7200.0 | 1 | 20 |
| S51D3 | 30% | 891.0 | 962 | 1.1 | 7200.0 | 1 | 32 |
| S51D4 | 40% | 1438.4 | 1610 | 0.7 | 7200.0 | 2 | 40 |
| S51D5 | 30% | 1255.9 | 1377 | 1.9 | 7200.0 | 1 | 31 |
| S51D6 | 20% | 1901.5 | 2296 | 1.6 | 7200.0 | 1 | 23 |
| S1 | 30% | 21406.1 | 22901 | 0.9 | 3.1 | 1 | 32 |
| S2 | 20% | 70795.2 | 70890 | 2.5 | 72.7 | 8 | 25 |
| S3 | 30% | 41174.2 | 43084 | 2.1 | 1195.8 | 4 | 30 |
| S4 | 30% | 62007.1 | 63122 | 4.4 | 1701.4 | 9 | 31 |
| S5 | 30% | 135090.1 | 138312 | 10.6 | 7200.0 | 8 | 33 |
| S6 | 40% | 78671.5 | 83107 | 14.6 | 5939.3 | 10 | 40 |
| S7 | 40% | 363411.6 | 363975 | 20.7 | 7200.0 | 6 | 42 |
| S8 | 30% | 502618.8 | 504007 | 21.4 | 7200.0 | 2 | 30 |
| S9 | 20% | 202999.8 | 203452 | 34.2 | 7200.0 | 5 | 27 |

Table 9: Class V: SDVRP instances from Belenguer et al. [4] and Chen et al. [7], with $m = 2$ and minimum delivery amount.

assumptions have been presented to properly define one problem, but we have also showed how other variants of this problem can be addressed with the results in this paper. Some examples of these assumptions are: the vehicle must visit each customer at least once; the initial load of the vehicle must be computed; preemption is allowed.

We have presented a single-commodity flow formulation for the problem, and described a branch-and-cut approach to solve it. A primal heuristic approach has been described, and also a Benders' decomposition on the flow variables. The Benders' cuts has been analyzed and a procedure based on solving a max-flow problem has been described. We have also described other inequalities to strengthen the linear programming relaxation. The branch-and-cut approach has been implemented and evaluated on five clases of benchmark instances. As it occurs in other split-demand routing problems in the literature, our experiments confirms that splitting the demand may reduce the travel cost. The approach in this paper can be applied to solve both the classical Capacitated Vehicle Routing Problem and also its split-demand variant. While our implementation is not competitive with the last developments on the first problem, we have shown that it gives good results on the second.

This paper also opens interesting questions. One concerns a worst-case analysis, trying to find instances where the travel cost of a SD1PDTSP route is smaller than half of the travel cost of an optimal 1PDTSP route, or proves that no one exists under some conditions (e.g. costs satisfying the triangular inequality). Another question could be to extend the results to the multi-commodity case in the line of the splittable pickup and delivery problem with reloads in Kerivin et al. [21].

**Acknowledgements**

This work has been partially supported by the Spanish Government through the research project MTM2012-36163-C06-01.

**References**

[1] Archetti, C., Bianchessi, N., Speranza, M. G., 2011. A column generation approach for the split delivery vehicle routing problem. Networks 58, 241–254.

[2] Archetti, C., Savelsbergh, M., Speranza, M. G., 2006. Worst-case analysis for split delivery vehicle routing problems. Transportation Science 40, 226–234.

[3] Archetti, C., Speranza, M. G., 2008. The split delivery vehicle routing problem: A survey. In: Golden, B., Raghavan, S., Wasil, E. (Eds.), The Vehicle Routing Problem: Latest Advances and New Challenges. Vol. 43 of Operations Research/Computer Science Interfaces. Springer US, pp. 103–122.

[4] Belenguer, J. M., Martinez, M. C., Mota, E., 2000. A lower bound for the split delivery vehicle routing problem. Operations Research 48, 801–810.

[5] Berbeglia, G., Cordeau, J. F., Gribkovskaia, I., Laporte, L., 2007. Static pickup and delivery problems: a classification scheme and survey. TOP 15 (1), 1–31.

[6] Chemla, D., Meunier, F., Wolfler-Calvo, R., 2013. Bike hiring system: Solving the static rebalancing problem. Discrete Optimization 10 (2), 120–146.

[7] Chen, S., Golden, B., Wasil, E., 2007. The split delivery vehicle routing problem: Applications, algorithms, test problems, and computational results. Networks 49, 318–329.

[8] Coelho, L. C., Cordeau, J. F., Laporte, G., 2014. Thirty years of inventory routing. Transportation Science 48, 1–19.

[9] Dell'Amico, M., Hadjicostantinou, E., Iori, M., Novellani, S., 2014. The bike sharing rebalancing problem: Mathematical formulations and benchmark instances. Omega 45, 7–19.

[10] Dror, M., Laporte, G., Trudeau, P., May 1994. Vehicle routing with split deliveries. Discrete Applied Mathematics 50, 239–254.

[11] Dror, M., Trudeau, P., 1990. Split delibery routing. Naval Research Logistics 37, 383–402.

[12] Fischetti, M., Salazar-González, J. J., Toth, P., 1997. A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. Operations Research 45 (3), 378–394.

[13] Gulczynski, D., Golden, B., Wasil, E., 2010. The split delivery vehicle routing problem with minimum delivery amounts. Transportation Research Part E: Logistics and Transportation Review 46, 612–626.

[14] Hernández-Pérez, H., Salazar-González, J. J., 2002. The one-commodity pickup-and-delivery travelling salesman problem. Lecture Notes in Computer Science 2570, 89–104.

[15] Hernández-Pérez, H., Salazar-González, J. J., 2004. Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. Transportation Science 38, 245–255.

[16] Hernández-Pérez, H., Salazar-González, J. J., 2007. The one-commodity pickup-and-delivery traveling salesman problem: Inequalities and algorithms. Networks 50, 258–272.

[17] Hernández-Pérez, H., Salazar-González, J. J., 2009. The multi-commodity one-to-one pickup-and-delivery traveling salesman problem. European Journal of Operational Research 196 (3), 987–995.

[18] Hoffman, A. J., 1960. Some recent applications of the theory of linear inequalities to extremal combinatorial analysis. In Bellman, R. , Hall, Jr. , M. (Eds. ): Proc. Symp. in Applied Mathematics. Amer. Math. Soc., 113–127.

[19] Jin, M., Liu, K., Bowden, R. O., 2007. A two-stage algorithm with valid inequalities for the split delivery vehicle routing problem. International Journal of Production Economics 105, 228–242.

[20] Kerivin, H. L. M., Lacroix, M., Mahjoub, A. R., 2012. Models for the single-vehicle preemptive pickup and delivery problem. Journal of Combinatorial Optimization 23, 196–223.

[21] Kerivin, H. L. M., Lacroix, M., Mahjoub, A. R., Quilliot, A., 2008. The splittable pickup and delivery problem with reloads. European Journal of Industrial Engineering 2 (2), 112–133.

[22] Lee, C. G., Epelman, M. A., White, C. C., Bozer, Y. A., 2006. A shortest path approach to the multiple-vehicle routing problem with split pick-ups. Transportation Research Part B: Methodological 40, 265–284.

[23] Moreno, L., de Aragão, M. P., Uchoa, E., 2010. Improved lower bounds for the split delivery vehicle routing problem. Operations Research Letters 38, 302–306.

[24] Mosheiov, G., 1994. The travelling salesman problem with pick-up and delivery. European Journal of Operational Research 79 (2), 299–310.

[25] Naddef, D., Rinaldi, G., 1992. The graphical relaxation: A new framework for the symmetric travelling salesman polytope. Mathematical Programming 58, 53–88.

[26] Nowack, M., Ergun, O., White, C. C., 2008. Pickup and delivery with split loads. Transportation Science 42, 32–43.

[27] Nowak, M., Hewitt, M., White, C. C., 2012. Precedence constrained pickup and delivery with split loads. International Journal of Logistics Research and Applications 15, 1–14.

[28] Parragh, S., Doerner, K., Hartl, R., 2008. A survey on pickup and delivery problems. Journal fr Betriebswirtschaft 58 (1), 21–51.

[29] Raviv, T., Tzur, M., Forma, I. A., 2013. Static repositioning in a bike-sharing system: models and solution approaches. EURO Journal on Trasnportation and Logistics 2, 187–229.

[30] Toth, P., Vigo, D., 2001. The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia.