Branch and Price for the Vehicle Routing Problem with Discrete Split Deliveries and Time Windows

Matteo Salani * Ilaria Vacca *

December 24, 2009

Report TRANSP-OR 091224
Transport and Mobility Laboratory
Ecole Polytechnique Fédérale de Lausanne
transp-or.epfl.ch

*Transp-OR, Ecole Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland, { matteo.salani , ilaria.vacca } @epfl.ch

Abstract

The Discrete Split Delivery Vehicle Routing Problem with Time Windows (DSDVRPTW) consists of designing the optimal set of routes to serve, at least cost, a given set of customers while respecting constraints on vehicles' capacity and customer time windows. The delivery request of a customer is discrete since it consists of several items that cannot be split further. The problem belongs to the class of split delivery problems since each customer's demand can be split in orders, i.e. feasible combinations of items, and each customer can be visited by more than one vehicle. In this work, we model the DSDVRPTW assuming that all feasible orders are known in advance and that each vehicle can serve at most one order per customer. Remarkably, service time at customer's location depends on the serviced combination of items, which is a modeling feature rarely found in literature. We present a mixed integer program for the DSDVRPTW based on arc-flow formulation, we reformulate it via Dantzig-Wolfe and we apply column generation. We propose a branch-and-price algorithm, implemented using state-of-the-art techniques for the pricing and the master problem. Computational results on instances based on Solomon's data set are presented and discussed.

1 Introduction

The capacitated Vehicle Routing Problem (VRP) consists of designing the optimal routes for a set of vehicles with given capacity in order to serve a set of customers. Customer's demand must be delivered by exactly one vehicle and vehicles' capacity must cannot be violated (exceeded).

The Split Delivery Vehicle Routing Problem (SDVRP) is a relaxed version of the classical capacitated VRP in which the number of visits to customer locations is no longer constrained to be at most one. In the SDVRP each customer can be visited by more than one vehicle which serves a fraction of its demand. It has been shown that this relaxation could yield to substantial savings on the total traveled distance, up to 50% in some instances, as well as on the number of required vehicles (Archetti et al., 2006a; Archetti et al., 2008a). The problem and some properties have been introduced by Dror and Trudeau (1989) and Dror and Trudeau (1990), who solve the problem using heuristic schemes. Next, Dror et al. (1994) introduce a mathematical formulation based on integer programming, solved through a cutting plane approach. Lower bounds have been proposed by Belenguer et al. (2000); exact methods (Gueguen, 1999; Jin et al., 2007) as well as heuristic algorithms (Archetti et al., 2006b; Chen et al., 2007; Jin et al., 2008; Archetti et al., 2008b) have been proposed. Gendreau et al. (2006) and Desaulniers (2008) address the problem with time windows and present exact approaches based on column generation and branch-and-bound techniques. Lower bounds have been studied by Ceselli et al. (2009b) and a tabu search algorithm has been proposed by Ho and Haugland (2004).

In the Discrete Split Delivery Vehicle Routing Problem (DSDVRP) the demand of a customer consists of several items which cannot be split further. The problem belongs to the class of split delivery problems since each customer's demand can be fractionated and each customer can be visited by more than one vehicle. Nakao and Nagamochi (2007) present the problem and propose a dynamic programming based heuristic. The algorithm is compared to other existing heuristics for the VRP and computational results on real-world instances are provided. Ceselli et al. (2009a)

present an exact approach to a real-world VRP in which customers' orders can be split among several vehicles in a discrete fashion. The authors propose a three level order aggregation which end up, at the last level, in considering any possible combination of items. The VRP with splittable and discrete demand arises in some practical applications, such as the routing of helicopters for crew exchanges on off-shore locations (Sierksma and Tijssen, 1998) and the Field Technician Scheduling Problem (Xu and Chiu, 2001); however, authors do not specifically relate their problems to the DSDVRP.

In the reminder of the paper we study the Discrete Split Delivery Vehicle Routing Problem with Time Windows (DSDVRPTW). We assume that demand can be split in orders, i.e. feasible combinations of items, that each vehicle can serve at most one order per customer and that service time at customer's location depends on the delivered combination of items. Remarkably, this is a modeling feature rarely found in literature, where service times are usually assumed to be independent of the delivered quantities. We refer e.g. to Gendreau et al. (2006) and Desaulniers (2008), who make the simplifying assumption of constant service times: this is indeed the case in applications where the unloading time is negligible, but it is not an appropriate modeling assumption for applications where the unloading time is largely affected by the size of the delivery. In Section 2 we recall some known properties of split deliveries. Section 3 provides an arc-flow formulation for the DSDVRPTW. In Section 4 we reformulate the problem using Dantzig-Wolfe decomposition and we illustrate the column generation scheme. The branch-and-price implementation is presented in Section 5 and computational results are discussed in Section 6. Section 7 concludes the paper.

2 Properties

In this section we recall some known properties of the VRP with Split Deliveries, firstly introduced by Dror and Trudeau (1990), extended to the variant with time windows by Gendreau et al. (2006). In particular, we discuss the implications of the new modeling feature introduced in this

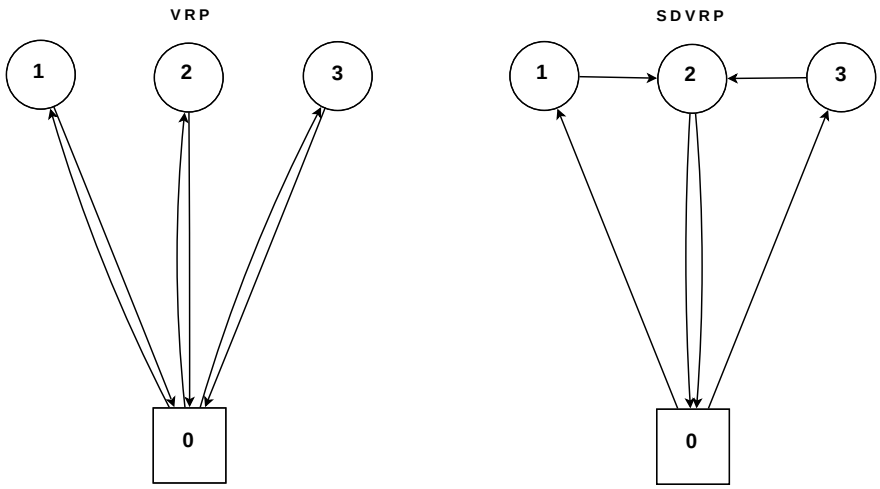

Figure 1: Dror and Trudeau's example.

paper, the quantity-dependant service time.

Property 1 *The SDVRP(TW) is a relaxation of the corresponding VRP(TW).*
Let z_s^* be the value of the optimal solution for the SDVRP(TW) and let z_f^* be the value of the optimal solution for the corresponding VRP(TW). Property 1 states that $z_s^* \leq z_f^*$. Clearly, $z_s^* \not> z_f^*$ for any problem instance since any VRP(TW) solution (and in particular, the optimal one) is a feasible solution for the corresponding SDVRP(TW). Furthrmore, there exists instances such that $z_s^* < z_f^*$, as for the following example.

Dror and Trudeau's example We consider three demand points with $d_1 = 3$, $d_2 = 4$ and $d_3 = 3$; the distances between the points including the depot (node 0) are $c_{0i} = 2M$ for $i = 1, 2, 3$; $c_{12} = c_{23} = \epsilon$ and $c_{13} = 2\epsilon$. All vehicles have a capacity of five units. The VRP solution has a total cost of $12M$ and requires 3 vehicles, whereas the SDVRP solution has a total cost of $8M + 2\epsilon$ and requires only 2 vehicles (cf. Fig. 1).

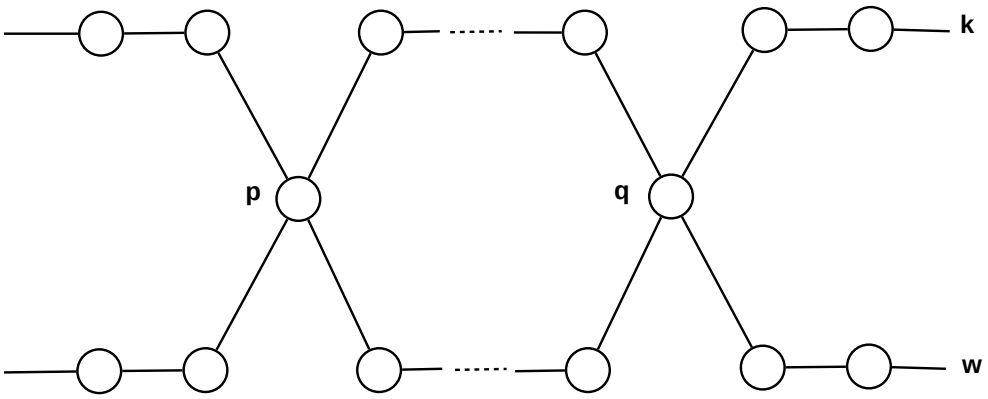

Figure 2: Dror and Trudeau's two-route two-split example.

Property 2 *The SDVRPTW is NP-Hard in the strong sense.*
The NP-Hardness has been proven by reducing the Traveling Salesman Problem to the SDVRPTW using a polynomial transformation.

Property 3 *When the cost matrix satisfies the triangular inequality, there exists an optimal solution of the SDVRP in which no two routes have more than one split demand in common.*
Consider an optimal SDVRP solution where two customers p and q are serviced by the same two routes k and w with split deliveries d_p^k, d_p^w, d_q^k and d_q^w (cf. Fig. 2). Without loss of generality, we assume that $d_p^k = \min\{d_p^k, d_p^w, d_q^k, d_q^w\}$. It is always possible to modify the quantities delivered by k and w to drop out customer p from route k, such that demands are still fulfilled, vehicles' capacity is not violated and the objective function does not increase its value. In particular, the new quantities are $d_p^{k\prime} = 0$, $d_p^{w\prime} = d_p^w + d_p^k$, $d_q^{k\prime} = d_q^k + d_p^k$ and $d_q^{w\prime} = d_q^w - d_p^k$.

Property 3 can be extended to the SDVRP with Time Windows only under the assumption of constant service times. In particular, since route w still visits customers p and q, service times are unchanged and time windows are not affected. With respect to route k, since customer p is not visited anymore, the vehicle may reach some subsequent customer earlier that allowed; in this case, the vehicle will just wait at customer's location

until it is allowed to start the delivery.

It can be easily shown that properties 1 and 2 also hold for the DSDVRPTW, whereas property 3 does not apply to the DSDVRPTW with quantity-dependant service times. In this case, the increased quantity delivered by route w to customer p implies an increased service time at location p. As a consequence, the arrival and the delivery to the next customers may not comply with the time windows constraints anymore.

3 Arc-flow formulation

In this section we present a mixed integer linear program for the DSDVRPTW based on arc-flow formulation.

Let $G(V, E)$ be a complete graph with $V = \{0\} \cup N$, where vertex $\{0\}$ represents the depot and $N = \{1, ..., n\}$ is the set of customers to be served. Each arc $(i, j) \in E$ has a cost c_{ij} and a travel time t_{ij}. The set of available vehicles with identical capacity Q is denoted by K. The set of items R is defined as $R = \bigcup_{i \in N} R_i$, where R_i represents the set of items to be delivered to customer $i \in N$. Furthermore, $R_i \cap R_j = \emptyset \ \forall i \neq j, \ \ i, j \in N$, meaning that any item $r \in R$ is univocally associated to a customer $i \in N$. Each item $r \in R$ has a size q^r and a service time t^r. Items are delivered in orders, i.e. combinations of items. The set of orders C is defined as $C = \bigcup_{i \in N} C_i$, where C_i represents the set of feasible orders for customer $i \in N$. Furthermore, $C_i \cap C_j = \emptyset \ \forall i \neq j, \ \ i, j \in N$, meaning that any order $c \in C$ is univocally associated to a customer $i \in N$. Each order $c \in C$ has a size $q_c = \sum_{r \in R} e_c^r q^r$ and a service time t_c such that $\max_{r \in R} e_c^r t^r \leq t_c \leq \sum_{r \in R} e_c^r t^r$, where e_c^r is a binary parameter equal 1 if item $r \in R$ is delivered in order $c \in C$ and 0 otherwise. Interval $[a_i, b_i]$ denotes the time window for customer $i \in N$.

We define the following decision variables:

- x_{ij}^k binary, equal to 1 if arc $(i, j) \in E$ is used by vehicle $k \in K$;
- y_c^k binary, equal to 1 if vehicle $k \in K$ delivers order $c \in C$;
- T_i^k ≥ 0, represents the arrival time of vehicle $k \in K$ at customer $i \in N$.

The discrete split delivery vehicle routing problem with time windows can be formulated as follows:

$$z^*_{IP} = \min \sum_{k \in K} \sum_{(i,j) \in E} c_{ij} x^k_{ij} \tag{1}$$

$$\sum_{j \in V} x^k_{0j} = 1 \qquad \forall k \in K, \tag{2}$$

$$\sum_{j \in V} x^k_{ij} - \sum_{j \in V} x^k_{ji} = 0 \qquad \forall k \in K, \forall i \in V, \tag{3}$$

$$\sum_{j \in V} x^k_{ij} = \sum_{c \in C_i} y^k_c \qquad \forall k \in K, \forall i \in N, \tag{4}$$

$$\sum_{k \in K} \sum_{c \in C} e^r_c y^k_c = 1 \qquad \forall r \in R, \tag{5}$$

$$\sum_{c \in C_i} y^k_c \leq 1 \qquad \forall k \in K, \forall i \in N, \tag{6}$$

$$T^k_i + \sum_{c \in C_i} t_c y^k_c + t_{ij} - T^k_j \leq (1 - x^k_{ij}) M \qquad \forall k \in K, \forall i \in N, \forall j \in V, \tag{7}$$

$$T^k_i - t_{0i} \geq (1 - x^k_{0i}) M \qquad \forall k \in K, \forall i \in N, \tag{8}$$

$$T^k_i \geq a_i \sum_{j \in V} x^k_{ij} \qquad \forall k \in K, \forall i \in N, \tag{9}$$

$$T^k_i + \sum_{c \in C_i} t_c y^k_c \leq b_i \sum_{j \in V} x^k_{ij} \qquad \forall k \in K, \forall i \in N, \tag{10}$$

$$\sum_{c \in C} q_c y^k_c \leq Q \qquad \forall k \in K, \tag{11}$$

$$x^k_{ij} \in \{0, 1\} \qquad \forall k \in K, \forall (i,j) \in E, \tag{12}$$

$$y^k_c \in \{0, 1\} \qquad \forall k \in K, \forall c \in C, \tag{13}$$

$$T^k_i \geq 0 \qquad \forall k \in K, \forall i \in N. \tag{14}$$

where M is a sufficiently large constant. The objective function (1) minimizes the total traveling costs. Flow conservation is ensured by constraints (2)–(4), which also link x and y variables. Demand satisfaction is ensured by constraints (5): all items must be delivered (but not all combinations). Constraints (6) ensure that every vehicle delivers at most one order per customer. Precedence, time windows and capacity constraints are ensured by constraints (7)–(8), (9)–(10) and (11). Finally, the domain of variables is defined by (12), (13) and (14).

The service time at customer location depends on the selected order. This feature is modeled by the term $\sum_{c \in C_i} t_c y_c^k$ in constraints (7): it increases the complexity of the model, with respect to the same type of precedence constraints in classical VRP formulations with time windows.

4 Column generation

In this section we reformulate the DSDVRPTW model (1)–(14) via Dantzig-Wolfe decomposition (Dantzig and Wolfe, 1960) and provide the formulations of the master problem and pricing subproblem. The master problem is solved by means of column generation.

4.1 Master problem

Let (2)-(4) and (6)-(14) be the constraints that define the subproblem and let $D^k = conv\{(x^k, y^k, T^k) \,|\, (x^k, y^k, T^k)$ satisfies $(2)-(4); (6)-(14)$ for $k\}$ be the feasible bounded domain of the subproblem associated to vehicle $k \in K$. Let P^k be the set of extreme points of D^k. Each extreme point $d_p = (x_p^k, y_p^k, T_p^k)$, $p \in P^k$ represents a feasible route for vehicle k with respect to vehicle's capacity and customers' time windows, delivering a unique order to every customer visited by the tour.

Since vehicles $k \in K$ present identical restrictions (in this case, the same capacity), all subproblems are identical and can therefore be aggregated into a single subproblem. We denote as $D = conv\{(x, y, T) \,|\, (x, y, T)$ satisfies $(2)-(4); (6)-(14)\}$ the feasible domain of the subproblem and P the set of extreme points of D. Each extreme point $d_p = (x_p, y_p, T_p)$, $p \in P$ represents now a feasible route that can be covered by any vehicle among the $|K|$ available.

The definition of the master problem requires the following additional notation: we denote c_p the cost of path $p \in P$, defined as $c_p = \sum_{(i,j) \in p} c_{ij}$, while α_p^r denotes a binary parameter equal to 1 if path $p \in P$ delivers item $r \in R$. After some standard adjustments and aggregation, the master

problem can be formulated as follows:

$$\min \sum_{p \in P} c_p \lambda_p \tag{15}$$

$$\sum_{p \in P} \alpha_p^r \lambda_p = 1 \quad \forall r \in R \quad (\pi_r) \tag{16}$$

$$\sum_{p \in P} \lambda_p \leq |K| \qquad (\pi_0) \tag{17}$$

$$\lambda_p \geq 0 \quad \forall p \in P. \tag{18}$$

where λ_p are the decision variables associated to paths $p \in P$. The dual variables associated to constraints (16) are denoted as π_r while π_0 is the dual variable associated to constraint (17).

The objective function (15) minimizes the total traveling costs. Constraints (16) ensure that all items are delivered to customers, while constraint (17) ensures that the number of chosen routes does not exceed the number of available vehicles.

We remark that constraints (16) need to be modeled as partitioning constraints in the DSDVRPTW, unlike common reformulations for routing problems that generally make use of covering constraints. This is due to the fact that, for every customer $i \in N$, the set of orders C_i does not necessarily contain all subsets of items $r \in R_i$, but only the subsets that are considered feasible with respect to the problem definition (incompatibilities between specific items, restrictions on the order size, etc.). As a consequence, a partitioning solution equivalent to the optimal covering solution may not exist.

4.2 Pricing subproblem

We denote $\tilde{c}_p := c_p - \sum_{r \in R} \pi_r \alpha_p^r - \pi_0$ the reduced cost of a route $p \in P$. In a column generation scheme, given a dual solution of the (restricted) master problem, the pricing subproblem identifies the route p^* with the minimum reduced cost:

$$p^* = \arg\min_{p \in P} \{\tilde{c}_p\} = \arg\min_{p \in P} \{c_p - \sum_{r \in R} \pi_r \alpha_p^r - \pi_0\} \tag{19}$$

The subproblem formulation relies on variables x, y and T defined in Section 3 (without index k, since we have aggregated the subproblems) and can be written as follows:

$$\min \sum_{(i,j)\in E} c_{ij}x_{ij} - \sum_{r\in R} \pi_r \Big(\sum_{c\in C} y_c e_c^r\Big) - \pi_0 \tag{20}$$

$$\sum_{j\in V} x_{0j} = 1 \tag{21}$$

$$\sum_{j\in V} x_{ij} - \sum_{j\in V} x_{ji} = 0 \qquad \forall i \in V, \tag{22}$$

$$\sum_{j\in V} x_{ij} = \sum_{c\in C_i} y_c \qquad \forall i \in N, \tag{23}$$

$$\sum_{c\in C_i} y_c \leq 1 \qquad \forall i \in N, \tag{24}$$

$$T_i + \sum_{c\in C_i} t_c y_c + t_{ij} - T_j \leq (1 - x_{ij})M \qquad \forall i \in N, \forall j \in V, \tag{25}$$

$$T_i - t_{0i} \geq (1 - x_{0i})M \qquad \forall i \in N, \tag{26}$$

$$T_i \geq a_i \sum_{j\in V} x_{ij} \qquad \forall i \in N, \tag{27}$$

$$T_i + \sum_{c\in C_i} t_c y_c \leq b_i \sum_{j\in V} x_{ij} \qquad \forall i \in N, \tag{28}$$

$$\sum_{c\in C} q_c y_c \leq Q \tag{29}$$

$$x_{ij} \in \{0, 1\} \qquad \forall (i, j) \in E, \tag{30}$$

$$y_c \in \{0, 1\} \qquad \forall c \in C, \tag{31}$$

$$T_i \geq 0 \qquad \forall i \in N. \tag{32}$$

Analyzing the objective function, we can observe that two major decisions are made in the subproblem:

a) the sequence of customers $i \in N$ visited in the route (cost component c_{ij});

b) for each customer in the route, the order $c \in C$ to be delivered, and therefore the subset of items $r \in R$ delivered by the route (cost component e_c^r).

The pricing problem (20)–(32) is an Elementary Shortest Path Problem with Resource Constraints (ESPPRC) defined on a network which has one node for every order $c \in C$ and whose arcs have transit time equals to $(t_{ij} + t_c)$. In particular, the choice on the orders to be delivered by the route has impact on the complexity to the subproblem.

5 Branch-and-price implementation

For solving the DSDVRPTW we have implemented a branch-and-price algorithm (Barnhart et al., 1998; Lübbecke and Desrosiers, 2005) with state-of-the-art solution techniques for the pricing and the master problem.

The pricing problem is solved using bounded bi-directional dynamic programming (Righini and Salani, 2006) with decremental state space relaxation (Righini and Salani, 2008). The algorithm is initialized by a preprocessing phase, used to identify and remove trivially dominated combinations, and by a simple greedy algorithm used to find a feasible solution to the problem. Such solution allows to compute an upper bound on the cost of the solution and on the number of vehicles. The search tree is explored using a best-first strategy.

5.1 Branching scheme

In the search tree, branching is required when the master problem is solved at optimality and the corresponding solution of the arc-flow formulation is not integer. We have implemented a branching scheme consisting of four hierarchical levels:

1. if the total number of vehicles is fractional ($\sum_{p \in P} \lambda_p = \tilde{K}$), branching is performed on constraint (17) by enforcing $\sum_{p \in P} \lambda_p \leq \lfloor \tilde{K} \rfloor$ on the first child node and $\sum_{p \in P} \lambda_p \geq \lceil \tilde{K} \rceil$ on the second child node.

2. if the number of vehicles visiting a customer $i \in N$ is fractional ($\sum_{p \in P} \alpha_p^i \lambda_p = \tilde{K}_i$), branching is performed by enforcing $\sum_{p \in P} \alpha_p^i \lambda_p \leq \lfloor \tilde{K}_i \rfloor$ on the first child node and $\sum_{p \in P} \alpha_p^i \lambda_p \geq \lceil \tilde{K}_i \rceil$ on the second child node. This branching requires additional constraints in the master

problem and associated dual values to be collected in the pricing; however, the pricing structure is not affected.

3. if there is an arc $(i,j) \in E$ visited a fractional number of times ($\sum_{k\in K} x_{ij}^k = \tilde{x}_{ij}$), branching is performed by enforcing $x_{ij} \leq \lfloor \tilde{x}_{ij} \rfloor$ on the first child node and $x_{ij} \geq \lceil \tilde{x}_{ij} \rceil$ on the second child node. This branching requires additional constraints in the master problem and associated dual values to be collected in the pricing; however, the pricing structure is not affected.

4. if none of the above conditions holds, then there exist two consecutive arcs $(i,j) \in E$ and $(j,l) \in E$ visited consecutively a fractional number of times: $\sum_{p\in P_{ijl}} \lambda_p = \tilde{z}_{ijl}$, where P_{ijl} denotes the set of paths containing arc (j,l) immediately after arc (i,j). In this case, branching is performed by enforcing $z_{ijl} \leq \lfloor \tilde{z}_{ijl} \rfloor$ on the first child node and $z_{ijl} \geq \lceil \tilde{z}_{ijl} \rceil$ on the second child node. This branching requires modifying the pricing structure as well as additional constraints in the master problem. However, it is rarely needed (<1% of instances in our tests).

5.2 2-Path Cuts

At the root node we try to identify valid 2-path inequalities that are violated by the current linear relaxation solution.

The basic idea of k-path inequalities (Kohl et al., 1999) is to identify a subset of customers that is visited by less than k vehicles in the current fractional solution, although it requires, in the optimal solution, at least k vehicles to be serviced. For any subset of customers $S \subseteq N$, $|S| \geq 1$ we define the flow into S, denoted $x(S)$, as $x(S) = \sum_{i\in\bar{S}} \sum_{j\in S} x_{ij}$ where $\bar{S} = N \setminus S$. Given the smallest number of vehicles needed to service all the customers in S, denoted by $k(S)$, a valid k-path inequality is defined by $x(S) \geq k(S)$.

Since calculating $k(S)$ is very time consuming, we have limited the search to the 2-path inequalities. This reduces to identify some set S such that $x(S) < 2$ and $k(S) > 1$. To determine whether $k(S) > 1$, we solve

a Traveling Salesman Problem with Time Windows (TSPTW) for S: if a TSPTW solution cannot be found, then $k(S) > 1$. Since the number of sets S grows exponentially, in our search we limited the size of S to twice the average number of customer per vehicle. All 2-path inequalities that are violated by more than a predetermined threshold value (0.2 for our tests) are added to the master problem, defining a new linear relaxation to solve.

6 Computational results

Algorithms are coded in ANSI C and compiled with gcc 4.1.2. Computational experience is run under a linux operating system on a 2Ghz Intel processor equipped with 2GB of RAM. All restricted master problems are solved using CPLEX version 10.2.

6.1 Instances

To the best of our knowledge there is no standard dataset used in the literature for the DSDVRPTW. The most related contribution is that of Nakao and Nagamochi (2007) for which the instances are not available.

We generated our test bed from the well-known Solomon's data set (Solomon, 1983). For all instances of classes R1, C1 and RC1 we considered the first $n = 25, 50$ customers and we discretized the demand of each customer in 12 items ($|R_i| = 12 \ \forall i \in N$).

For each customer, we generated 7 orders: 1 full order (containing 12 items); 2 complementary orders 50%-50% (containing 6 items each, partitioned); 2 complementary orders 75%-25% (containing 9 and 3 items respectively, partitioned); 2 complementary 90%-10% orders (containing 11 and 1 items respectively, partitioned).

We considered 3 possible scenarios:

A: full order + 50-50% orders ($|C_i| = 3$);
B: full order + 50-50% orders + 75-25% orders ($|C_i| = 5$);
C: full order + 50-50% orders + 75-25% orders + 90-10% orders ($|C_i| = 7$).

The full order has been always included in order to allow the comparison of the DSDVRPTW with the classical VRP with Time Windows

(VRPTW). The unsplittable case, which is trivially composed of the full order only ($|C_i| = 1$), is denoted as scenario O.

In order to enhance splitting, we considered more restrictive capacities than Solomon's, as already suggested by Gendreau et al. (2006). Instances have been tested with $Q = 30, 50$ and 100.

From the 29 original Solomon's instances (12 for class R1, 9 for class C1 and 8 for class RC1), we derived 174 instances: 29×2 (customers) $\times$ 3 (capacities). Each instance has been tested under the 4 scenarios A, B, C and O.

6.2 Results

Table 1 presents a summary of the instances solved by the branch-and-price within 1 hour of computational time. Instances are grouped by the number of customers (n) and the capacity (Q). The number of instances of each class is also provided (nb_inst). For each group, the table provides the number of instances solved at optimality (nb_solved) and the average computational time in seconds (t) for each DSDVRPTW scenario.

We were able to solve 88, 67 and 47 out of 174 instances for scenarios A, B and C, respectively. The difficulty of solving the instances increases with the size of $|C|$: 75, 125 and 175 orders with 25 customers and 150, 250, and 350 orders with 50 customers for scenarios A, B and C, respectively. This difficulty also increases with the number of customers: we were able to solve 76% (A), 60% (B) and 48% (C) of instances with $n = 25$, whereas only 25% (A), 17% (B) and 6% (C) of instances with $n = 50$ were solved at optimality. The average computational time is also affected by the size of $|C|$ and the number of customers.

For $n = 25$ customers, instances of class R1 are the easiest to solve. There are 36 (A), 36 (B) and 32 (C) solved instances out of 36 for class R1; 20 (A), 8 (B) and 2 (C) solved instances out of 27 for class C1; 10 (A), 8 (B) and 8 (C) solved instances out of 24 for class RC1. On average, 96% of instances were solved in class R1, 37% in class C1 and 36% in class RC1.

For $n = 50$ customers, class RC1 seems easier to solve than class R1 (on average, 42% versus 11% of solved instances), while no instances in class

				A		B		C	
n	class	nb_inst	Q	nb_solved	t	nb_solved	t	nb_solved	t
25	R1	12	30	12	7	12	75	8	466
			50	12	6	12	60	12	430
			100	12	9	12	41	12	113
25	C1	9	30	4	1108	0	x	0	x
			50	9	37	4	2137	0	x
			100	7	706	4	705	2	1876
25	RC1	8	30	2	1988	0	x	0	x
			50	0	x	0	x	0	x
			100	8	3	8	11	8	35
50	R1	12	30	1	1010	0	x	0	x
			50	3	1572	1	385	0	x
			100	3	1035	2	167	2	535
50	RC1	8	30	0	x	0	x	0	x
			50	7	54	6	902	0	x
			100	8	529	6	809	3	2832

Table 1: Summary of the branch-and-price results.

C1 were solved.

Optimal solutions are detailed in tables 2, 3, 4, 5 and 6. For each instance, we provide the value of the optimal integer solution (z_{IP}), the number of vehicles (veh) and the computational time in seconds (t). The three DSDVRPTW scenarios A, B, C and compared to the unsplittable VRPTW scenario O: figures highlighted in bold denote savings due to split deliveries. Instances that are not feasible for the unsplittable case because of insufficient capacity are denoted by "Q $<$ d". Instances not solved at optimality within 1 hour of computational time are denoted by "x".

We can observe that split deliveries are more frequent for instances with small Q values, although they also occur for certain instances with Q $= 100$.

In a few cases, split deliveries not only decrease the total traveling costs but also allow to save one vehicle.

7 Conclusions

Analyzing the results, we can conclude that obtaining optimal solutions is difficult, even with a small number of orders per customer. Furthermore, only a limited number of instances with 50 customers could be solved.

We guess that the bottleneck is in the pricing problem. Indeed, the underlying ESPPRC network is huge, since, in the worst case scenario, for every customer $i \in N$ we have that set C_i corresponds to the set of all subsets of R_i and therefore its size grows exponentially with the number of items. Computational results show that solving the ESPPRC on such a network may be impractical. Therefore, more efficient solution techniques need to be investigated.

References

Archetti, C., Savelsbergh, M. and Speranza, M. (2006a). Worst-case analysis for split delivery vehicle routing problems, *Transportation Science* **40**: 226–234.

Archetti, C., Savelsbergh, M. and Speranza, M. (2008a). To split or not to split: That is the question, *Transportation Research Part E* **44**: 114–123.

Archetti, C., Speranza, M. and Hertz, A. (2006b). A tabu search algorithm for the split delivery vehicle routing problem, *Transportation Science* **40**: 64 – 73.

Archetti, C., Speranza, M. and Savelsbergh, M. (2008b). An optimization-based heuristic for the split delivery vehicle routing problem, *Transportation Science* **42**(1): 22–31.

Barnhart, C., Johnson, E., Nemhauser, G., Savelsbergh, M. and Vance, P. (1998). Branch-and-price: Column generation for solving huge integer programs, *Operations Research* **46**(3): 316–329.

Belenguer, J., Martinez, M. and Mota, E. (2000). A lower bound for the split delivery vehicle routing problem, *Operations Research* **48**: 801–810.

Ceselli, A., Righini, G. and Salani, M. (2009a). A column generation algorithm for a vehicle routing problem with economies of scale and additional constraints, *Transportation Science* **43**(1): 56–69.

Ceselli, A., Righini, G. and Salani, M. (2009b). Column generation for the split delivery vehicle routing problem, *Technical report*, Note del Polo - Ricerca 118, Dipartimento di Tecnologie dell'Informazione, Università degli Studi di Milano.

Chen, S., Golden, B. and Wasil, E. (2007). The split delivery vehicle routing problem: Applications, algorithms, test problems and computational results, *Networks* **49**(4): 318–329.

Dantzig, G. and Wolfe, P. (1960). Decomposition principle for linear programs, *Operations Research* **8**: 101–111.

Desaulniers, G. (2008). Branch-and-price-and-cut for the split delivery vehicle routing with time windows, *Technical Report G-2008-32*, Les Chaiers du GERAD.

Dror, M., Laporte, G. and Trudeau, P. (1994). Vehicle routing with split deliveries, *Discrete and Applied Mathematics* **50**: 239–254.

Dror, M. and Trudeau, P. (1989). Savings by split delivery routing, *Transportation Science* **23**: 141–145.

Dror, M. and Trudeau, P. (1990). Split delivery routing, *Naval Research Logistics* **37**: 383–402.

Gendreau, M., Dejax, P., Feillet, D. and Gueguen, C. (2006). Vehicle routing with time windows and split deliveries, *Technical report*, Technical report 2006-851, Laboratoire d'Informatique d'Avignon.

Gueguen, C. (1999). *Méthodes de résolution exacte pour les problèmes de tournées de véhicules*, PhD thesis, Ecole centrale Paris.

Ho, S. and Haugland, D. (2004). A tabu search heuristic for the vehicle routing problem with time windows and split deliveries, *Computers and Operations Research* **31**: 1947–1964.

Jin, M., Liu, K. and Bowden, R. (2007). A two stage algorithm with valid inequalitites for the split delivery vehicle routing problem, *International Journal of Production Economics* **105**: 228–242.

Jin, M., Liu, K. and Eksiouglu, B. (2008). A column generation approach for the split delivery vehicle routing problem, *Operations Research Letters* **36**: 265–270.

Kohl, N., Desrosiers, J., Madsen, O., Solomon, M. and Soumis, F. (1999). 2-path cuts for the vehicle routing problem with time windows, *Transportation Science* **33**(1): 101–116.

Lübbecke, M. and Desrosiers, J. (2005). Selected topics in column generation, *Operations Research* **53**(6): 1007–1023.

Nakao, Y. and Nagamochi, H. (2007). A dp-based heuristic algorithm for the discrete split delivery vehicle routing problem, *Journal of Advanced Mechanical Design, Systems, and Manufacturing* **1**(2): 217–226.

Righini, G. and Salani, M. (2006). Symmetry helps: Bounded bi-directional dynamic programming for the elementary shortest path problem with resource constraints, *Discrete Optimization* **3**(3): 255–273.

Righini, G. and Salani, M. (2008). New dynamic programming algorithms for the resource constrained shortest path problem., *Networks* **51**(3): 155–170.

Sierksma, G. and Tijssen, G. (1998). Routing helicopters for crew exchanges on off-shore locations, *Annals of Operations Research* **76**: 261–286.

Solomon, M. (1983). *Vehicle Routing and Scheduling with Time Windows Constraints: Models and Algorithms*, PhD thesis, University of Pennsylvania.

Xu, J. and Chiu, S. Y. (2001). Effective heuristic procedures for a field technician scheduling problem, *Journal of Heuristics* **7**(5): 495–509.

		O			A			B			C		
Q	id	z_{IP}	veh	t	z_{IP}	veh	t	z_{IP}	veh	t	z_{IP}	veh	t
30	r101	795.6	13	0	**795.1**	13	1	**782.5**	13	3	**782.5**	13	11
	r102	789.1	13	0	**772.3**	13	4	**765.9**	**12**	161	**761.2**	**12**	291
	r103	759.6	12	0	759.6	12	19	**751.7**	12	176	**745.3**	12	70
	r104	759.6	12	0	759.6	12	33	**747.0**	12	32	**745.3**	12	140
	r105	775.7	12	0	**775.3**	12	3	**773.2**	12	47	**773.2**	12	558
	r106	772.6	13	0	**763.7**	**12**	4	**756.6**	**12**	50	**753.4**	**12**	115
	r107	748.5	12	0	748.5	12	3	**744.1**	12	57	x		
	r108	748.5	12	0	748.5	12	4	**744.1**	12	100	x		
	r109	754.6	12	0	754.6	12	1	**750.2**	12	20	**750.2**	12	1041
	r110	748.5	12	0	748.5	12	4	**744.1**	12	37	**744.1**	12	1498
	r111	754.6	12	0	754.6	12	2	**750.2**	12	102	x		
	r112	748.5	12	0	748.5	12	5	**744.1**	12	118	x		
50	r101	635.0	9	0	**631.5**	**8**	0	**631.5**	**8**	1	**631.5**	**8**	1
	r102	580.7	8	0	580.7	8	7	580.7	8	35	580.7	8	221
	r103	534.3	7	0	534.3	7	3	534.3	7	65	534.3	7	333
	r104	527.3	7	0	527.3	7	7	527.3	7	76	527.3	7	437
	r105	596.1	8	0	**588.9**	8	1	**585.4**	8	4	**585.4**	8	13
	r106	543.3	7	0	**542.5**	7	4	**542.3**	7	52	**542.3**	7	233
	r107	527.7	7	0	527.7	7	14	527.7	7	187	527.7	7	1309
	r108	521.6	7	0	521.6	7	16	521.6	7	185	521.6	7	2175
	r109	524.6	7	0	524.6	7	1	524.6	7	5	524.6	7	11
	r110	536.7	7	0	**529.1**	7	3	**526.0**	7	17	**526.0**	7	119
	r111	521.6	7	0	521.6	7	7	521.6	7	45	521.6	7	178
	r112	515.8	7	0	515.8	7	8	515.8	7	46	515.8	7	135
100	r101	617.1	8	0	617.1	8	0	617.1	8	1	617.1	8	1
	r102	547.1	7	0	547.1	7	1	547.1	7	7	547.1	7	15
	r103	454.6	5	0	454.6	5	2	454.6	5	8	454.6	5	14
	r104	416.9	4	0	416.9	4	5	416.9	4	14	416.9	4	58
	r105	530.5	6	0	530.5	6	1	530.5	6	3	530.5	6	5
	r106	465.4	5	0	465.4	5	7	465.4	5	55	465.4	5	201
	r107	428.4	4	0	428.4	4	7	428.4	4	32	428.4	4	87
	r108	403.2	4	0	403.2	4	10	403.2	4	28	403.2	4	111
	r109	441.3	5	0	441.3	5	2	441.3	5	7	441.3	5	12
	r110	444.1	5	0	444.1	5	13	444.1	5	96	444.1	5	229
	r111	428.8	4	0	428.8	4	6	428.8	4	30	428.8	4	101
	r112	401.7	4	1	**401.3**	4	59	**401.3**	4	209	**401.3**	4	519

Table 2: Optimal solutions for class R1, $n = 25$ customers.

		O			**A**			**B**			**C**		
Q	id	z_{IP}	veh	t	z_{IP}	veh	t	z_{IP}	veh	t	z_{IP}	veh	t
30	c101	$Q < d$			**825.3**	16	532	x			x		
	c105	$Q < d$			**825.7**	16	985	x			x		
	c106	$Q < d$			**826.4**	16	630	x			x		
	c107	$Q < d$			**825.7**	16	2285	x			x		
50	c101	516.9	10	0	**516.8**	10	5	**516.8**	10	1242	x		
	c102	516.6	10	0	**516.5**	10	29	x			x		
	c103	516.6	10	0	**516.5**	10	56	x			x		
	c104	516.6	10	0	**516.4**	10	142	x			x		
	c105	516.9	10	0	**516.8**	10	9	**516.8**	10	2030	x		
	c106	516.9	10	0	**516.8**	10	7	**516.8**	10	1721	x		
	c107	516.9	10	0	**516.8**	10	17	**516.8**	10	3555	x		
	c108	516.8	10	0	**516.7**	10	29	x			x		
	c109	516.8	10	0	**515.9**	10	42	x			x		
100	c101	291.9	5	0	291.9	5	17	291.9	5	175	291.9	5	1858
	c102	291.9	5	10	291.9	5	1010	x			x		
	c105	291.9	5	1	291.9	5	47	291.9	5	687	x		
	c106	291.9	5	1	291.9	5	24	291.9	5	231	291.9	5	1894
	c107	291.9	5	1	291.9	5	86	291.9	5	1726	x		
	c108	291.9	5	2	291.9	5	530	x			x		
	c109	289.5	5	15	289.5	5	3226	x			x		

Table 3: Optimal solutions for class C1, $n = 25$ customers.

		O			A			B			C		
Q	id	z_{IP}	veh	t	z_{IP}	veh	t	z_{IP}	veh	t	z_{IP}	veh	t
30	rc101	Q < d			**1438.0**	18	453	x			x		
	rc106	Q < d			**1438.0**	18	3523	x			x		
100	rc101	534.3	6	0	534.3	6	1	534.3	6	6	534.3	6	19
	rc102	523.7	6	0	523.7	6	2	523.7	6	11	523.7	6	34
	rc103	514.7	6	0	**513.7**	6	3	**513.7**	6	11	**513.7**	6	54
	rc104	506.7	6	0	506.7	6	3	506.7	6	18	506.7	6	34
	rc105	527.5	6	0	527.5	6	3	527.5	6	6	527.5	6	32
	rc106	515.6	6	0	515.6	6	1	515.6	6	4	515.6	6	12
	rc107	505.7	6	0	505.7	6	3	505.7	6	13	505.7	6	39
	rc108	505.7	6	0	505.7	6	4	505.7	6	16	505.7	6	56

Table 4: Optimal solutions for class RC1, $n = 25$ customers.

		O			A			B			C		
Q	id	z_{IP}	veh	t	z_{IP}	veh	t	z_{IP}	veh	t	z_{IP}	veh	t
30	r101	Q < d			**1664.6**	26	1010	x			x		
50	r101	1222.0	16	1	**1211.1**	16	127	**1198.7**	**15**	385	x		
	r102	1134.9	16	2	**1125.1**	16	3404	x			x		
	r105	1166.3	16	17	**1148.5**	16	1185	x			x		
100	r101	1044.0	12	0	**1044.0**	12	9	**1040.6**	12	22	**1040.6**	12	54
	r102	913.2	11	1	913.2	11	58	**911.9**	11	311	**911.9**	11	1016
	r105	918.2	9	7	918.2	9	3038	x			x		

Table 5: Optimal solutions for class R1, $n = 50$ customers.

		O			A			B			C		
Q	id	z_{IP}	veh	t	z_{IP}	veh	t	z_{IP}	veh	t	z_{IP}	veh	t
50	rc101	1713.2	20	0	**1708.9**	20	13	**1708.3**	20	594	x		
	rc102	1704.3	20	0	**1700.5**	20	62	**1700.5**	20	1938	x		
	rc103	1703.4	20	1	**1696.8**	20	37	**1696.8**	20	427	x		
	rc104	1702.2	20	1	**1696.7**	20	54	**1696.7**	20	677	x		
	rc105	1703.9	20	0	**1700.1**	20	73	**1700.1**	20	1132	x		
	rc107	1704.1	20	1	**1698.6**	20	58	x			x		
	rc108	1702.2	20	2	**1696.7**	20	83	**1696.7**	20	645	x		
100	rc101	994.6	10	2	**993.8**	10	257	**984.4**	10	524	x		
	rc102	961.0	10	1	**960.2**	10	2657	x			x		
	rc103	936.2	10	4	936.2	10	837	x			x		
	rc104	915.9	10	4	915.9	10	198	915.9	10	2140	x		
	rc105	957.4	10	2	957.4	10	82	957.4	10	536	957.4	10	2940
	rc106	937.0	10	1	937.0	10	58	937.0	10	742	x		
	rc107	915.1	10	1	915.1	10	33	915.1	10	515	915.1	10	2064
	rc108	911.9	10	3	911.9	10	110	911.9	10	398	911.9	10	3491

Table 6: Optimal solutions for class RC1, $n = 50$ customers.