An exact algorithm for a rich vehicle routing problem with private fleet and common carrier
Dabia, Said; Lai, David; Vigo, Daniele

published in
Transportation Science
2019

DOI (link to publisher)
10.1287/trsc.2018.0852

document version
Publisher's PDF, also known as Version of record
document license
Article 25fa Dutch Copyright Act

Link to publication in VU Research Portal

citation for published version (APA)

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

E-mail address:
vuresearchportal.ub@vu.nl

Download date: 16. Jun. 2021
An Exact Algorithm for a Rich Vehicle Routing Problem with Private Fleet and Common Carrier

Said Dabia, David Lai, Daniele Vigo

1. Introduction

We consider the vehicle routing problem (VRP) with private fleet and common carrier (VRPPC), which is a generalization of the classical vehicle routing problem (see, e.g., Toth and Vigo 2014) in which the dispatcher may either serve the customers by using the vehicles of the owned fleet (called the private fleet) or assign them to a common carrier, for example, a third-party logistics provider. The latter case occurs when either the total customer demand exceeds the capacity of the private fleet or if it is more economically convenient to do so, for example, because the customer is isolated and far from the private fleet depot.

This type of problem has many practical applications, particularly in the design of last-mile distribution services with which outsourcing of unprofitable services are frequently considered options. Despite its practical relevance, the VRPPC has received relatively scarce attention in the literature. During the last decade only, some heuristic approaches, examined in more detail later, were proposed to solve the VRPPC and some of its variants considering, for example, a heterogeneous fleet or multiple depots. Moreover, in all the proposed approaches except the recent paper by Gahm, Brabänder, and Tuma (2017), the modeling of the outsourcing costs paid to the common carrier is rather simplistic, assuming that they are separable and only dependent on the demand of each individual customer. To the best of our knowledge, no exact algorithm has been proposed so far for the VRPPC or its variants.

In this paper, we consider a general and practical variant of the VRPPC in which both time windows and a heterogeneous fleet are present. In addition, although in the current literature the cost charged by the external common carrier is generally modeled as a fixed fee—sometimes proportional to the customer demand—in this paper, we consider cost structures inspired from practice and with which the cost for outsourcing a unit of demand depends on the total quantity assigned to the common carrier. Such cost structures account for potential quantity discounts that the shipper may achieve by outsourcing larger quantities and are clearly relevant for both the strategic or midterm design of mixed delivery systems and operational settings in which such quantity discounts may be present. We call, for short, our problem the rich VRPPC (RVRPPC) because it contains as special cases the known variants of the VRPPC with a single depot.

We present here an exact approach for the RVRPPC, which, to the best of our knowledge, is the first exact method proposed for a problem of the VRPPC family so far.
The algorithm is based on a branch-and-cut-and-price (BCP) algorithm, which incorporates several new features. More precisely, we considered two alternative set partitioning formulations for which we developed specialized pricing procedures and which are strengthened by using problem-adjusted subset-row inequalities, which turned out particularly effective. Finally, a new dominance procedure is introduced exploiting the problem structure. Such procedure turned out to be very effective and can be generalized to other problems that suffer from the inclusion of nonrobust cuts.

The paper is organized as follows. In Section 2, we review the literature related to the VRPPC and the proposed branch-and-cut-and-price algorithms. In Section 3, the problem description and formulations are presented. Sections 4, 5, and 6 describe the proposed branch-and-cut-and-price algorithms and implementation details. The computational results can be found in Section 7.

2. Literature Review

An abundant number of publications is devoted to the vehicle routing problem (see, e.g., Laporte 1992, 2007; and Toth and Vigo 2014 for some reviews). For specific reviews on the vehicle routing problem with time windows (VRPTW), the reader is referred to Bräysy and Gendreau (2005a, b), Kallevaatjage (2008), Gendreau and Tarantilis (2010), and Desaulniers, Madsen, and Ropke (2014). For the heterogeneous VRP, a recent survey is included in Irnich, Schneider, and Vigo (2014).

In this section, we review the main literature on the VRPPC and its variants. Later, we examine the current state of the art for branch-and-cut-and-price algorithms applied to the VRP.

2.1. Literature on the VRPPC

As previously mentioned, the existing literature on the VRPPC is entirely devoted to heuristic approaches. Chu (2005) considered a single-depot routing problem with outsourcing options, which can be considered the first paper on the VRPPC. The problem considers a limited private fleet of vehicles with given capacity and fixed costs associated with their use. A set of customers with known demand can be served by the private fleet, which incurs travel costs as in the standard VRP. In addition, customers may be outsourced to a common carrier (therein called the less-than-truckload carrier) for which only a fixed cost per customer has to be paid. The objective is to minimize the total costs of the private fleet service involving both fixed costs for vehicles and variable travel costs and the fixed costs for orders performed by the common carrier. Chu (2005) introduced an integer linear programming (ILP) model for the VRPPC and proposed a heuristic solution method, consisting of a modified savings algorithm (see Clarke and Wright 1964) and a simple improvement phase. The computational testing was performed on five instances with up to 30 customers and showed that the heuristics produced solutions within less than 8% from the optimal solution value determined by using the ILP model. Bolduc et al. (2008) showed that the VRPPC can be modeled as a heterogeneous fleet VRP and presented a simple metaheuristic that constructs an initial solution through a repeated execution of a randomized savings algorithm and then iterates between a local-search improvement and a random perturbation step. The whole construction—improvement—perturbation cycle is repeated a prefixed number of times, and the best solution found is returned. The algorithm by Bolduc et al. (2008) substantially improved the results of Chu (2005). In addition, they provided results of their algorithm on two new benchmark sets: one with a homogeneous and one with a heterogeneous vehicle fleet with up to 480 customers. A tabu search heuristic for the VRPPC, which outperformed the approach of Bolduc et al. (2008) for the case of homogeneous vehicles, was presented in Côté and Potvin (2009). Furthermore, Potvin and Naud (2011) proposed a tabu search heuristic with ejection chains that further improved results in both homogeneous and heterogeneous instances at the cost of significantly larger total computing time. The current best results on the VRPPC are reported by Vidal et al. (2016), who extended to several VRP variants with profits, including the VRPPC, the unified hybrid genetic search framework of Vidal et al. (2012) and simple large neighborhood search algorithms. Gahm et al. (2017) introduced a generalization of the VRPPC in which a heterogeneous private fleet and different cost options for the common carrier are considered. For the private fleet, contrary to what is commonly done in the literature, only variable costs are considered, and the common carrier’s services may be acquired under several options: the first one considers distance-dependent (or time-dependent) variable costs associated with a minimum distance (or time) guaranteed fee. In the second case, no variable costs are paid, and only a flat fee per vehicle is due. In the last option, the cost of serving a customer with the common carrier depends on its distance from the depot and the number of items to be delivered. In addition, all-unit volume discounts are also considered for the common-carrier options. The resulting problem is called the vehicle routing problem with private fleet, multiple common carriers offering volume discounts, and rental options, for which the authors propose a heuristic algorithm based on variable neighborhood search (VNS). The algorithm is compared with simple VNS approaches on several classes of test instances adapted from the VRPPC and VRP instances.

A multiple-depot version of the VRPPC, denoted as MDVRPPC, was introduced in Stenger et al. (2013) in which they defined a VNS algorithm, enhanced by an adaptive mechanism, to select the shaking operator. The resulting algorithm was tested on a benchmark set of
instances derived from multidepot VRP (MDVRP) instances, showing the potential benefits associated with outsourcing. The algorithm is able to obtain state-of-the-art results on both the single-depot VRPPC and the MDVRP. The adaptive VNS was later extended in Stenger, Schneider, and Goeke (2013), who adopted different neighborhood structures and a randomization mechanism to increase diversification. The algorithm has been applied to several variants of the prize-collecting VRP with nonlinear objective function and also to simple variants, such as the VRPPC and MDVRPPC, for which they improved the best-known results. The problem under study is also related to the wide family of VRP with profits: we refer the interested reader to the survey of Archetti, Speranza, and Vigo (2014) for an overview of the literature in this field.

2.2. Branch-and-Cut-and-Price Approaches for the VRP
Column generation is successfully applied in the solution of various combinatorial optimization problems. Lübbecke and Desrosiers (2005) provide a good overview of column-generation algorithms. Desrochers, Desrosiers, and Solomon (1992) are the first to apply column generation in the context of the VRPTW, resulting in a branch-and-price (BP) algorithm. The BP algorithm is later improved by Kohl et al. (1999), who introduced subtour elimination constraints and two-path inequalities into the column-generation approach, and Cook and Rich (1999) applied the more general $k$-path inequalities. Adding valid inequalities to the BP algorithm results in a BCP algorithm. In the 1990s, the pricing subproblem of column generation was the shortest-path problem with resource constraints and two-cycle elimination. Irnich and Villeneuve (2006) introduced an algorithm for $k$-cycle elimination, which led to tighter bounds. Later, Feillet, Dejax, and Gendreau (2004) and Chabrier (2006) proposed algorithms for the elementary shortest-path problem with resource constraints (ESPPRC), which further improved lower bounds. Righini and Salani (2006, 2008) proposed various techniques to speed up the ESPPRC algorithm, such as bidirectional search and decremental state space relaxation. In opposition to all previous work in which valid inequalities in a BCP algorithm had been expressed in the variables of the previous work in which valid inequalities in a BCP algorithm, such as bidirectional search and problem variables, named subset–row (SR) inequalities. Including SR inequalities in a BCP algorithm complicates the solution of the pricing subproblem. However, if they are efficiently treated, SR inequalities may significantly improve the lower bounds. To accelerate the pricing subproblem solution, Desaulniers, Lessard, and Hadjar (2008) proposed a tabu search heuristic for the ESPPRC. Furthermore, decremental state space relaxation and both two-path and subset–row inequalities were used.

Baldacci, Mingozzi, and Roberti (2011) introduced a new route relaxation, called $ng$–route, used to solve the pricing subproblem; $ng$–routes proved to be very effective in solving difficult instances of the VRPTW with wide time windows. Baldacci, Mingozzi, and Roberti (2011) solved all but one of the 56 famous Solomon instances.

3. Problem Description
3.1. The Outsourcing Cost Structure
The cost charged by the common carrier is inspired from cost structures used in practice. We consider a cost structure with quantity bands. Rather than charging a cost based on individual customer orders, the cost depends on the total outsourced quantity. We assume the outsourcing cost to be a piecewise linear function of the total outsourced demand quantity $q$. Let $\mathcal{F} = \{1, 2, \ldots, |\mathcal{F}|\}$ be the set of all nonoverlapping linear segments of the piecewise linear function. For all $s \in \mathcal{F}$, let $[L_s, U_s]$ denote the quantity interval of segment $s$ and $M_s$ and $C_s$ denote the corresponding cost rate and intercept, respectively. The outsourcing cost for total demand quantity $q$ is given by the following piecewise linear function:

$$\mathcal{F}(q) = \begin{cases} M_1 q + C_1, & L_1 \leq q \leq U_1, \\ M_2 q + C_2, & L_2 \leq q \leq U_2, \\ \ldots \\ M_{|\mathcal{F}|} q + C_{|\mathcal{F}|}, & L_{|\mathcal{F}|} \leq q \leq U_{|\mathcal{F}|}, \end{cases}$$

where the intervals $[L_s, U_s]$ are defined such that $U_s \leq L_{s+1}$. The function $\mathcal{F}(q)$ is also called the tariff sheet.

Figure 1 depicts an example of a tariff sheet with four segments. The third segment, for instance, is defined in the interval $[L_3, U_3]$ and has equation $M_3 q + C_3$. This structure implies that the more that is outsourced to the common carrier, the more is paid. Furthermore, the cost per outsourced unit decreases with the total quantity outsourced. We should note that the framework developed in this paper allows other cost structures (e.g., with discontinuous segments in which $U_s < L_{s+1}$).

3.2. Mixed-Integer Linear Formulation
The RVRPPC is defined on a directed complete graph $G(N, \mathcal{A})$, where $N = \{0, 1, \ldots, n\}$ is the set of nodes and $\mathcal{A} = \{(i, j) \in N \times N : i \neq j\}$ is the set of arcs. Node 0 represents the depot at which all the vehicle routes start and end. The nodes in $N \setminus \{0\}$ represent the customers and are denoted as $N_0$. Each node $i \in N$ is associated with a positive demand $d_i$, a time window $[l_i, h_i]$, and a service time $s_i$. Each arc $(i, j) \in \mathcal{A}$ is associated with a nonnegative travel time $t_{ij}$. A heterogeneous fleet of vehicles is available. Let $K$ be the set of vehicle types. Each vehicle of type $k \in K$ has a finite capacity $Q_k$, a fixed cost for using the vehicle $f_k$, and the fleet size $m_k$ (number of vehicles of type $k$ available). Furthermore, for all arcs $(i, j) \in \mathcal{A}$ and vehicle types $k \in K$, let $c_{ij}^k$ be the...
travel cost from node \(i\) to node \(j\). A common carrier is available for which demand can be subcontracted. For a total demand quantity \(q\) subcontracted to the common carrier, the outsourcing cost is given by the piecewise linear function \(F(q)\). If \(F\) is the set of segments consisting of the function \(F(q)\), we define \(z_s\) for all \(s \in F\) to be a binary variable that takes value one if and only if segment \(s\) is selected and \(q_s\) as a continuous variable indicating the quantity within the range \([L_s, U_s]\) of segment \(s\) to be subcontracted. Let \(x_{ij}^k\) be a binary variable that takes value one if and only if a type-\(k\) vehicle traverses arc \((i, j)\), \(y_{ij}\) be a continuous variable representing a vehicle’s load on arc \((i, j)\), \(a_i\) be a continuous variable representing a vehicle’s arrival time at customer \(i\), and \(b_j\) be a binary variable that takes value one if and only if customer \(i\) is assigned to the common carrier. The objective is to determine the vehicle routes, minimizing the total fixed costs, travel costs, and outsourcing costs, subject to the following requirements:

- All routes start and end at the same depot 0.
- Each route is performed by exactly one private vehicle.
- The total demand in a route does not exceed the vehicle’s capacity.
- A customer is either visited by a private vehicle or subcontracted to the common carrier.
- The customers that are not outsourced must be visited exactly once by one vehicle.
- A vehicle visits a customer node within its corresponding time window.

The RVRPPC can be formulated as the following mixed-integer programming model:

\[
\begin{align*}
\min \sum_{k \in K} \sum_{(i,j) \in A} f_{ij} x_{ij}^k + \sum_{k \in K} \sum_{(i,j) \in A} \sum_{s \in F} c_{ij} x_{ij}^k + \sum_{s \in F} (M_s q_s + C_s z_s) \quad (1) \\
\text{s.t.} \quad \sum_{k \in K} \sum_{(i,j) \in A} x_{ij}^k + \sum_{k \in K} \sum_{(j,i) \in A} x_{ji}^k = 1, \quad \forall j \in N_0, \quad (2) \\
\sum_{(i,j) \in A} x_{ij}^k - \sum_{(j,i) \in A} x_{ji}^k = 0, \quad \forall k \in K, \quad \forall i \in N, \quad (3) \\
\sum_{(i,j) \in A} y_{ij} - \sum_{(j,i) \in A} y_{ji} = d_i (1 - \delta_i), \quad \forall i \in N_0, \quad (5) \\
y_{ij} \leq \sum_{k \in K} (Q_k - d_i) x_{ij}^k, \quad \forall k \in K, (i, j) \in A, \quad (6) \\
a_j - a_i \geq s_i + t_j - M \left(1 - \sum_{k \in K} x_{ij}^k\right), \quad \forall (i, j) \in A: i \neq 0, \quad (7) \\
a_j \geq t_0 \sum_{k \in K} x_{ij}^k, \quad \forall j \in N: j \neq 0, \quad (8) \\
\sum_{s \in F} d_i b_i \leq \sum_{s \in F} q_s, \quad \forall i \in N, \quad (9) \\
L_s z_s \leq q_s \leq U_s z_s, \quad \forall s \in F, \quad (10) \\
\sum_{s \in F} z_s = 1, \quad (11) \\
e_i \leq a_i \leq l_i, \quad \forall i \in N, \quad (12) \\
y_{ij} q_s \geq 0, \quad \forall s \in F, (i, j) \in A, \quad (13) \\
x_{ij}^k z_s \in \mathbb{B}, \quad \forall k \in K, (i, j) \in A, s \in F, \quad (14) \\
\delta_i \in \{0, 1\}, \quad \forall i \in N. \quad (15)
\end{align*}
\]

The objective function (1) minimizes the total fixed costs, travel costs, and outsourcing costs. Constraint (2) ensures that a customer node that is not outsourced is visited exactly once by a vehicle. Constraint (3) is the flow conservation constraint of the vehicles. Constraint (4) ensures that the number of vehicles in use is less than the fleet sizes. Constraint (5) is the flow conservation constraint of the customer demands. Constraint (6) ensures that vehicle capacity is respected. Constraints (7), (8), and (12) ensure that the customer time windows are respected. Constraints (9)–(11) formulate the piecewise linear outsourcing costs.

### 4. The Set-Partitioning Formulation: SP1

To derive the set-partitioning formulation (SP1) for the RVRPPC, we define \(\Theta^k\) as the set of feasible private paths corresponding to vehicle type \(k \in K\) and \(\Omega\) as the set of customer order subsets subcontracted to the common carrier. A private path is feasible for vehicle type \(k\) if it satisfies its customers delivery time windows and capacity constraints. Furthermore, we define \(\Omega = \bigcup_{k \in K} \Theta^k\) as the set of all feasible private paths. For each private path \(p \in \Omega\), \(c_p\) denotes its cost (i.e., including a fixed cost for using the vehicle and the traveling cost), and for each subcontracted subset \(\sigma \in \Theta\), \(c_\sigma\) denotes its cost that can be calculated based on the common carrier’s tariff sheet. Let \(\sigma_p\) be a constant that counts the number of times node \(i\) is visited by path \(p\) and \(\rho_\sigma\) be a constant that counts the number of times node \(i\) is in subset \(\sigma\). Furthermore, let \(y_p\) be a binary variable that takes value one if and only if path \(p\) is included in the solution and \(w_\sigma\) be a binary variable that takes value one if and only if segment \(s\) is selected and \(q_s\) as a continuous variable indicating the quantity within the range \([L_s, U_s]\) of segment \(s\) to be subcontracted. Let \(a_i\) be a continuous variable representing a vehicle’s arrival time at customer \(i\), and \(b_j\) be a binary variable that takes value one if and only if customer \(i\) is assigned to the common carrier. The objective is to determine the vehicle routes, minimizing the total fixed costs, travel costs, and outsourcing costs, subject to the following requirements:

- All routes start and end at the same depot 0.
- Each route is performed by exactly one private vehicle.
- The total demand in a route does not exceed the vehicle’s capacity.
- A customer is either visited by a private vehicle or subcontracted to the common carrier.
- The customers that are not outsourced must be visited exactly once by one vehicle.
- A vehicle visits a customer node within its corresponding time window.

The RVRPPC can be formulated as the following mixed-integer programming model:

\[
\begin{align*}
\min \sum_{k \in K} \sum_{(i,j) \in A} f_{ij} x_{ij}^k + \sum_{k \in K} \sum_{(i,j) \in A} \sum_{s \in F} c_{ij} x_{ij}^k + \sum_{s \in F} (M_s q_s + C_s z_s) \quad (1) \\
\text{s.t.} \quad \sum_{k \in K} \sum_{(i,j) \in A} x_{ij}^k + \sum_{k \in K} \sum_{(j,i) \in A} x_{ji}^k = 1, \quad \forall j \in N_0, \quad (2) \\
\sum_{(i,j) \in A} x_{ij}^k - \sum_{(j,i) \in A} x_{ji}^k = 0, \quad \forall k \in K, \quad \forall i \in N, \quad (3) \\
\sum_{(i,j) \in A} y_{ij} - \sum_{(j,i) \in A} y_{ji} = d_i (1 - \delta_i), \quad \forall i \in N_0, \quad (5) \\
y_{ij} \leq \sum_{k \in K} (Q_k - d_i) x_{ij}^k, \quad \forall k \in K, (i, j) \in A, \quad (6) \\
a_j - a_i \geq s_i + t_j - M \left(1 - \sum_{k \in K} x_{ij}^k\right), \quad \forall (i, j) \in A: i \neq 0, \quad (7) \\
a_j \geq t_0 \sum_{k \in K} x_{ij}^k, \quad \forall j \in N: j \neq 0, \quad (8) \\
\sum_{s \in F} d_i b_i \leq \sum_{s \in F} q_s, \quad \forall i \in N, \quad (9) \\
L_s z_s \leq q_s \leq U_s z_s, \quad \forall s \in F, \quad (10) \\
\sum_{s \in F} z_s = 1, \quad (11) \\
e_i \leq a_i \leq l_i, \quad \forall i \in N, \quad (12) \\
y_{ij} q_s \geq 0, \quad \forall s \in F, (i, j) \in A, \quad (13) \\
x_{ij}^k z_s \in \mathbb{B}, \quad \forall k \in K, (i, j) \in A, s \in F, \quad (14) \\
\delta_i \in \{0, 1\}, \quad \forall i \in N. \quad (15)
\end{align*}
\]
if and only if subset \( o \) is subcontracted to the common carrier. The RVRPPC is formulated as the following set-partitioning problem:

\[
(SPI) : \min \sum_{p \in \Omega} c_p y_p + \sum_{o \in \Theta} c_o w_o
\]

s.t. \( \sum_{p \in \Omega} g_p y_p + \sum_{l \in \mathcal{L}} \rho_{i,l} w_o = 1, \quad \forall i \in \mathcal{N}_o, \quad (17) \]

\[
\sum_{p \in \Omega} y_p \leq m_k, \quad \forall k \in K, \quad (18) \]

\[
y_p, w_o \in \mathbb{R}, \quad \forall p \in \Omega, o \in \Theta. \quad (19)
\]

The objective function (16) minimizes the cost of the chosen private routes and the cost charged for subcontracting orders to the common carrier. Constraint (17) guarantees that each node is either visited once by a private vehicle or is in a subset of orders subcontracted to the common carrier. Constraint (18) ensures that the number of used private vehicles doesn’t exceed the available number \( m_k \) of each vehicle type. Constraint (19) imposes binary conditions on the decision variables. We use column generation to solve the linear programming (LP) relaxation of (16)–(19). Starting with a variable representing the case in which all orders are outsourced, we generate additional variables (i.e., private paths and subcontracted orders) for the master problem by solving two separate pricing subproblems: First is, for each vehicle type, a pricing subproblem that searches for private paths with negative reduced cost. When the heuristic fails to generate a new label \( L' \) along an arc \( (i(L'), j) \) to a node \( j \in \mathcal{N}\backslash V(L') \) to a new label \( L \). We update the resources for the new label as follows:

\[
i(L) = j, \quad V(L) = V(L') \cup \{j\}, \quad t(L) = t(L') + t_{i(L'), j},
\]

\[
d(L) = d(L') + d_j, \quad c(L) = c(L') + c^k_{i(L'), j} - \pi_j.
\]

The reduced cost of a complete path \( p \) (i.e., starting and ending at the depot) that is performed by a type–\( k \) vehicle is calculated as

\[
\tau_p = f_k + \lambda_k + \sum_{(i,j) \in p} (c^k_{i,j} - \pi_i) x_{ijp},
\]

where \( x_{ijp} \) is an integer variable that counts the number of times arc \( (i, j) \) is used in path \( p \). In the labeling algorithm, for every label, all possible extensions are derived and stored. It ends when all labels are calculated. However, the number of labels can be very large. To reduce the number of labels, a dominance test is usually introduced. Let \( E(L) \) denote the set of feasible extensions of label \( L \) to the end depot with respect to time windows, vehicle capacity, and elementarity. However, when comparing two labels \( L_1 \) and \( L_2 \), it is not straightforward to evaluate all feasible extensions of both labels \( L_1 \) and \( L_2 \). Consequently, usually sufficient dominance criteria that are computationally less expensive are desirable. Therefore, we dominate \( L_2 \) by \( L_1 \) if \( i(L_1) = i(L_2), \ d(L_1) \leq d(L_2), \ t(L_1) \leq t(L_2), \ c(L_1) \leq c(L_2), \) and \( V(L_1) \subseteq V(L_2). \)

To speed up the solution of the private pricing problem, we first use heuristics to generate paths with negative reduced cost. The exact procedure is called only when the heuristics fail to find any paths with negative reduced cost. In our branch-and-cut-and-price framework, we use two different heuristics. First is a heuristic that runs on a trimmed graph that is generated by keeping, for each node in the original graph, at most \( k \) outgoing arcs. We choose to keep \( k \) arcs with the smallest reduced cost. When the heuristic fails, the number of arcs kept is increased to two and then to four. Second, we implemented a truncated labeling heuristic in which only a limited number of labels (with the best cost) is kept and considered for a possible extension. The number of stored labels can be increased each time when the heuristic fails to find paths with negative reduced cost (e.g., we start with 250 and then increase the number of labels to 500, 1,000, and finally to 2,000 labels).
4.2. The Common Pricing Problem

The CPP generates subsets of customer orders with negative reduced cost to be subcontracted to the common carrier. If the quantity to be subcontracted is $q$, the reduced cost of the corresponding subcontracted subset $i$ is

$$F(q) = F(q) - \sum_{i \in \mathcal{N}_0} \pi_i \delta_i,$$  \hspace{1cm} (21)

where $F(q)$ is the cost of subcontracting the quantity $q$ calculated based on the tariff sheet imposed by the common carrier. The CPP can then be formulated as follows:

$$\begin{align*}
\max & \sum_{i \in \mathcal{N}_0} \pi_i \delta_i - F(q) \\
\text{s.t.} & \sum_{i \in \mathcal{N}_0} d_i \delta_i \leq q, \\
& \delta_i \in \mathbb{B}, \quad \forall i \in \mathcal{N}_0, \\
& q \geq 0.
\end{align*}$$  \hspace{1cm} (22-25)

It is worth noting that when the common carrier doesn’t charge any cost (i.e., $F(q) = 0$ for all $q$) and $q$ is fixed, the CPP is exactly a knapsack problem with $q$ as the knapsack capacity, and customer orders are the items to pack in the knapsack. Hence, the CPP is a generalization of the knapsack problem in which the knapsack capacity itself is a decision variable and in which costs are incurred for “buying” more capacity. The function $F(q)$ is a piecewise linear function as defined in Section 3. If $\mathcal{F}$ is the set of segments belonging to the function $F(q)$, $z_s$ for all $s \in \mathcal{F}$ is the binary variable that takes value one if and only if segment $s$ is selected, and $q_s$ is the continuous variable indicating the quantity within the range $[L_s, U_s]$ of segment $s$ to be subcontracted. The CPP can be reformulated as

$$\begin{align*}
\max & \sum_{i \in \mathcal{N}_0} \pi_i \delta_i - \sum_{s \in \mathcal{F}} (M_s q_s + C_s z_s) \\
\text{s.t.} & \sum_{i \in \mathcal{N}_0} d_i \delta_i \leq \sum_{s \in \mathcal{F}} q_s, \\
& L_s z_s \leq q_s \leq U_s z_s, \quad \forall s \in \mathcal{F}, \\
& \sum_{s \in \mathcal{F}} z_s = 1, \\
& \delta_i, z_i \in \mathbb{B}, \quad \forall i \in \mathcal{N}_0, s \in \mathcal{F}, \\
& q_s \geq 0, \quad \forall s \in \mathcal{F}.
\end{align*}$$  \hspace{1cm} (26-31)

The objective function (26) minimizes the reduced cost of the subcontracted subset of orders. Constraint (27) guarantees that the total demand of the subcontracted orders is less than or equal to the quantity to be subcontracted. Constraint (28) ensures that the subcontracted quantity is within the range of the selected segment. Constraint (29) ensures that exactly one segment is selected. Constraints (30) and (31) set the domain for the decision variables.

4.3. Valid Inequalities

Jepsen et al. (2008) introduced SR inequalities for the VRPTW. The SR inequalities as introduced by Jepsen et al. (2008) are still valid for the RVRPPC. However, they will not be as strong as they are for the VRPTW because they don’t capture the fact that some customer orders may be subcontracted to the common carrier. Similarly to Jepsen et al. (2008), we can derive SR inequalities for the RVRPPC defined on a subset of nodes $S \subseteq \mathcal{N}$ and an integer $0 < \kappa \leq |S|$ as

$$\sum_{p \in \Omega} \left[ \frac{1}{|\mathcal{F}|} \sum_{i \in \mathcal{F}} a_{ip} \right] y_p + \sum_{i \in \mathcal{F}} \left( \frac{1}{|\mathcal{F}|} \sum_{p \in \Omega} \rho_{ip} \right) w_p \leq \frac{|S|}{\kappa}. \hspace{1cm} (32)$$

As emphasized by Jepsen et al. (2008), SR inequalities are nonrobust inequalities, meaning that adding them to the relaxation of the master problem destroys the structure of the pricing subproblems and, hence, complicates their solution by the label-setting algorithm.

Consider an active valid SR inequality $I$ of the form (32), defined by the subset of nodes $S_I$ and integer $\kappa_I$, and let $\xi_I \in \mathbb{R}$ ($\xi_I < 0$) be its corresponding dual variable. The dual variable $\xi_I$ is negative and, hence, will be acting as a penalty when subtracted from a column’s reduced cost. When generating new variables for the master problem, we must take $\xi_I$ into account. If a master variable $r \in \Omega \cup \Theta$ that contributes to the violation of $I$ is regenerated, its reduced cost must be penalized by $\xi_I$. Hence,

$$\xi_I = \begin{cases} 
\sum_{(i,j) \in \mathcal{F}} (c_{ij} - \pi_i) x_{ij} - \frac{1}{\kappa_I} \sum_{i \in \mathcal{F}} a_{i} \xi_I & \text{if } r \in \Omega, \\
F(q) - \sum_{i \in \mathcal{N}_0} \pi_i \delta_i - \frac{1}{\kappa_I} \sum_{i \in \mathcal{F}} \rho_{i} \xi_I & \text{if } r \in \Theta.
\end{cases} \hspace{1cm} (33)$$

However, we only know such a column is regenerated when it is complete (e.g., when a private path reaches the end node). Therefore, the standard dominance test cannot be directly applied in the labeling algorithm used to solve the PPP because partial paths that will be hit by $\xi_I$ when they reach the end node might erroneously dominate other partial paths that will not contribute to the violation of $I$. In our labeling algorithm, we handle the additional complexity stemming from adding the SR inequalities (32) as described in Jepsen et al. (2008).

SR inequalities change the structure of the CPP as well as we must account for the resulting dual variables when calculating the reduced cost of a subcontracted subset of customer orders. In the next section, the modified CPP accounting for adding SR inequalities is presented.

4.4. The Modified CPP

In this section, we present how we capture dual variables resulting from adding the SR inequalities (32) in the CPP.

Let $\mathcal{F}$ be the set of active SR inequalities. For each SR inequality $I \in \mathcal{F}$ defined by the subset of nodes $S_I$ and integer $\kappa_I$, let $\xi_I < 0$ be its dual variable. Furthermore, we
define $u_l \in \mathbb{N}$ as an integer variable that increments by one each time $\kappa_l$ new customers from $S_l$ are subcontracted to the common carrier. The modified CPP can be formulated as follows:

$$\max \sum_{i \in N} \pi_i \delta_i - \sum_{s \in \mathcal{F}} (M_i q_s + C_i z_s) + \sum_{l \in \overline{L}} \xi_l u_l$$

s.t. \(\sum_{i \in N} d_{i} \delta_i \leq \sum_{s \in \mathcal{F}} q_s,\)

\(L_i z_s \leq q_s \leq U_i z_s, \quad \forall s \in \mathcal{F},\)

\(\sum_{s \in \mathcal{F}} z_s = 1,\)

\(\frac{1}{\kappa_l} \sum_{i \in \overline{L}} \delta_i = 1 < u_l \leq \frac{1}{\kappa_l} \sum_{i \in \overline{L}} \delta_i, \quad \forall \mathcal{I} \in \mathcal{F} \mathcal{R},\)

\(\delta_i, z_s \in \mathbb{B}, \quad \forall i \in \overline{N}, s \in \mathcal{F},\)

\(q_s \geq 0, \quad \forall s \in \mathcal{F},\)

\(u_l \in \mathbb{N}, \quad \forall \mathcal{I} \in \mathcal{F} \mathcal{R}.\)

The objective function (34) now includes the dual variables resulting from including SR inequalities. The modified CPP has the additional set of constraint (38) (one for each included SR inequality) that set the coefficient of each of the included SR inequalities in the subcontracted subset of orders. Constraint (41) sets the domain for the additional decision variables $u_l$.

Before calling the mixed-integer problem (34)-(41), we first use a simple label-setting algorithm heuristic that, in each column-generation iteration, generates multiple subsets of orders with negative reduced cost that are added to the master problem. In the labeling-setting algorithm, a label represents a subset of orders, $o \in \Theta$, that may be outsourced to the common carrier. The label consists of only three resources, that is, the set of customers in $o$; the sum of the dual variables $\pi_i, i \in \overline{N},$ corresponding to customers in $o$; and the total demand of customers in $o$. Labels are extended by adding new orders to them, and a dominance test is performed such that, from two labels with the same total quantity, only the label with the highest sum of dual variables is kept. The labeling-setting algorithm considers the dual variables originating for the SR inequalities when checking whether the reduced cost of a label is negative.

### 4.5. Branching Rules

The branch-and-bound tree is explored using a best-bound strategy. First, the algorithm branches on the number of vehicles $\sum_{k \in K} \sum_{s \in \mathcal{F}} x^{k}_{ij}$ over all vehicle types. It creates two branches: $\sum_{k \in K} \sum_{s \in \mathcal{F}} x^{k}_{ij} \leq \sum_{s \in \mathcal{F}} \sum_{k \in K} x^{k}_{ij}$ and $\sum_{k \in K} \sum_{s \in \mathcal{F}} x^{k}_{ij} \geq \sum_{k \in K} \sum_{s \in \mathcal{F}} x^{k}_{ij}$ ($\pi_i$ is the current fractional solution expressed in the arc variables). If the number of vehicles for all types is integer, the algorithm branches on the number of vehicles per type. It looks for the vehicle type $k$ with the most fractional number of vehicles and creates two branches: $\sum_{s \in \mathcal{F}} x^{k}_{ij} \leq \sum_{s \in \mathcal{F}} x^{k}_{ij}$ and $\sum_{s \in \mathcal{F}} x^{k}_{ij} \geq \sum_{s \in \mathcal{F}} x^{k}_{ij}$. If for all vehicle types the number of vehicles is integer, the algorithm branches on the arc variables $x^{k}_{ij}$. It looks for pairs $(i,j), i,j \in \overline{N},$ and vehicle $k \in K$ such that $x^{k}_{ij} + x^{k}_{ij}$ is close to 0.5 and imposes two branches $x^{k}_{ij} + x^{k}_{ij} \leq 0$ and $x^{k}_{ij} + x^{k}_{ij} \geq 1$. If $x^{k}_{ij} + x^{k}_{ij}$ is integer for all pairs $(i,j), i,j \in \overline{N}$ and a vehicle of type $k \in K$, then the algorithm looks for an arc $(i,j) \in \mathcal{A}$ and a vehicle of type $k \in K$ for which $x^{k}_{ij}$ is fractional and branches on that instead. Strong branching is used; that is, the impact of branching on several candidates is investigated every time a branching decision has to be made. For each branch candidate, we estimate the lower bound in the two child nodes by solving the associated LP relaxation using the quick-pricing heuristic described in Section 4.1. The branch that maximizes the lower bound in the weakest of the two child nodes is chosen. We consider 30 branch candidates in the first 20 nodes of the branch-and-bound tree and 20 candidates in the rest.

### 5. The Set-Partitioning Formulation: SP2

The set-partitioning formulation (SP1) of the RVRPPC necessitates the solution of two separate pricing subproblems (i.e., the PPP and the CPP). In both subproblems, customers carrying large dual variables will probably be included, in the same column-generation iteration, in both private paths as well as in subcontracted order subsets leading to a highly symmetric formulation. Intuitively, customer orders that are not included in any private route must be subcontracted to the common carrier. Hence, we don’t need to generate order subsets to be subcontracted as long as we can impose that a customer order must either be included in a private route or subcontracted to the common carrier. In the following, we propose another set-partitioning formulation including only variables corresponding to private paths. Although this formulation is expected to have a weaker lower bound than formulation SP1, it only requires solving one pricing subproblem that generates new private paths with negative reduced cost. We call this new formulation SP2, and it is presented in the following:

$$\min \sum_{p \in \Omega} c_{p} y_{p} + \sum_{s \in \mathcal{F}} (M_i q_s + C_i z_s)$$

s.t. \(\sum_{p \in \Omega} q_p y_{p} + \delta_i = 1, \quad \forall i \in \overline{N},\)

\(\sum_{p \in \Omega} y_{p} \leq m_k, \quad \forall k \in K,\)

\(\sum_{s \in \mathcal{F}} d_{ih} \leq \sum_{s \in \mathcal{F}} q_{s}, \quad \forall i \in \overline{L} \mathcal{R},\)

\(L_i z_s \leq q_s \leq U_i z_s, \quad \forall s \in \mathcal{F},\)

\(\sum_{s \in \mathcal{F}} z_s = 1,\)

\(y_{p}, z_s \in \mathbb{B}, \quad \forall p \in \Omega, s \in \mathcal{F},\)

\(\delta_i, q_s \geq 0, \quad \forall i \in \overline{N}, s \in \mathcal{F}.\)
The objective function (42) minimizes the cost of the chosen private routes and the cost charged for subcontracting orders to the common carrier. Constraint (43) guarantees that each node is either visited once by a private truck or is subcontracted to the common carrier. Constraint (44) ensures that the number of used private vehicles doesn’t exceed the available number \( m_t \) of each vehicle type. Constraints (45)–(47) are the constraints (27)–(29) of the CPP moved now to the master problem. Constraints (48) and (49) set the domain of the decision variables. We use column generation to solve the LP relaxation of (42)–(49). Starting with a small subset of variables, we generate additional variables for the master problem. This new formulation requires solving only one subproblem that searches for new private paths with negative reduced cost, which is similar to the PPP presented in Section 4.1. Note that the integrality constraint for the \( \delta \) variables is relaxed. In fact, when the \( y \) variables are integer, the \( \delta \) variables must be integer as well. Therefore, we only impose integrality on the \( y \) variables.

### 5.1. Valid Inequalities

The SR inequalities introduced by Jepsen et al. (2008) can easily be shown to be valid for the RVRPPC when formulated as the set partitioning SP2. However, these inequalities do not capture the features of the RVRPPC, namely the fact that demand can be outsourced to the common carrier. So we expect these inequalities to be tightened by reducing the right-hand side (RHS) by a quantity that reflects the outsourced customer demand.

For a subset of nodes \( S \subseteq \mathcal{N}_0 \), we denote \( d(S) = \sum_{i \in S} d_i \) as the total demand delivered to customers included in \( S \). If we split this demand into packages with \( \kappa \) units, then you expect in an integer solution that the packages delivered using private routes should be less than the total packages reduced by the packages of each of the customers in \( S \) outsourced to the common carrier. Based on this intuition and on formulation SP2, we introduce the generalized subset–row (GSR) inequalities for the RVRPPC, defined by the subset of nodes \( S \subseteq \mathcal{N}_0 \) and integer \( 0 < \kappa \leq d(S) \) as

\[
\sum_{p \in \mathcal{S}_I} \frac{1}{\kappa} \sum_{i \in S} d_i \sigma_{ip} y_p + \sum_{i \in S} \frac{d_i}{\kappa} \delta_i \leq \frac{d(S)}{\kappa}.
\]

(50)

Clearly, the SR inequalities as introduced by Jepsen et al. (2008) are a special case of the GSR inequalities in (50), when all customers have a demand of one unit, and \( \kappa = 1 \). Moreover, GSR inequalities in (50) capture the fact that demand can be subcontracted to the common carrier in the term \( \sum_{i \in S} \frac{d_i}{\kappa} \delta_i \) and are, therefore, expected to be stronger for the RVRPPC. In fact, both private delivery routes and subcontracted customer demand contribute to the left-hand side (LHS) of an inequality. Note that these contributions depend rather on the demand (delivered/subcontracted) than on the number of visits (in opposition to the classical SR inequalities) as a route visiting only one customer involved in the cut and delivering more than \( \kappa \) units to that customer already contributes to the cut’s violation. In the following, an example of a fractional solution shows a violated GSR inequality.

**Example 1.** Let’s consider the fractional solution in Tables 1 and 2 obtained after solving the master problem, for instance, RC103a with 25 customers (see description instances in Section 7). Table 1 reports the fraction of demand outsourced to the common carrier. The first column shows the customers’ index, the second column shows the customers’ demand, and the third column shows the fraction of demand outsourced. Table 2 reports the private routes. The first column shows the path’s index, the second column corresponds to the path’s weight in the solution, the third column shows the truck type performing the route, the fourth column represents the path’s load, and the fifth column shows the sequence of the path. Let’s consider a GSR inequality \( I \) of the form (50) that is defined by the set of nodes \( S_I = \{19, 22, 25\} \) with \( \{40, 40, 20\} \) as their respective demands and integer \( \kappa_I = 6 \). The implied GSR inequality can be expressed as

\[
10y_{4} + 13y_{10} + 13y_{13} + 10y_{14} + 6y_{19} + 6y_{22} + 3y_{25} \leq 16.
\]

(51)

The LHS of inequality (51) sums up to 16.3, and the RHS is equal to 16. This implies that the inequality is violated. We also note that the outsourced demand of customer 25 contributed with \( 3 \times 0.4 = 1.2 \) to the LHS of inequality (51).

### 5.2. The Modified Dominance Criteria

Consider the PPP for a type–\( k \) vehicle; the GSR inequalities (50) destroy the structure of the PPP. In fact, the dominance criteria used in the PPP in the case of no inequalities added becomes invalid and must be modified to account for the dual variables originating from the added GSR inequalities. Consider some GSR inequality \( I \in \mathcal{F} \) of the form (50) defined by subset \( S_I \subseteq \mathcal{N}_0 \) and

| Table 1. Fractional Demand Outsourced |
|---|---|---|
| \( i \) | \( d_i \) | \( \delta_i \) |
| 13 | 10 | 0.2 |
| 14 | 10 | 0.2 |
| 24 | 10 | 0.1 |
| 25 | 20 | 0.4 |
null
The resources of label $L_2$ are such that $i(L_2) = 26, d(L_2) = 17, c(L_2) = 518.7, t(L_2) = 94, V(L_2) = \{19, 26\}, \text{ and } V(L_2) = \{2, 12, 15, 19, 21, 23, 26\}$. Let’s further consider the GSR cut $I$ defined by the subset of nodes $S_I = \{2, 4, 15, 17\}$ with their corresponding demands $\{7, 19, 8, 2\}$ and integer $\kappa_I = 13$. Label $L_1$ visits customer 2 from the cut; therefore, $m(L_1) = 7$. Label $L_2$ does not visit any customer from the cut; therefore, $m(L_2) = 0$. Using Proposition 1, we should penalize $L_1$ with the dual of $I$ ($\xi_i = -10.3$) but not $L_2$. Hence, $L_2$ cannot be dominated by $L_1$ because $c(L_1) - \xi_i = 515.8 + 10.3 = 526.1 > c(L_2) - \xi_i = 518.7$.

Let’s now consider Proposition 2. The set of nodes in the cut that can still be visited by an extension of $L_2$ is $S_I \setminus V(L_2) = \{4, 17\}$. We have $m(L_1) + d(S_I \setminus V(L_2)) = 7 + 19 + 2 = 28 > 13$. Hence, the cut $I$ may be active in an extension of $L_1$. Solving the subset sum problems (52)–(54) gives $z' = 19$ as at least node 4 needs to be in the extension of $L_1$ to exceed the threshold $\kappa_I$. However, including four in the extension of $L_2$ will also result in exceeding the threshold $\kappa_2$ (i.e., $m(L_2) + z' = 0 + 19 > 13$). Consequently, $L_2$ must also be penalized by the dual variable $\xi_2$ and, hence, can be dominated by $L_1$ because we now have that $c(L_1) - \xi_i = 526.1 < c(L_2) - \xi_i = 529.0$.

Example 3. The following example is taken when running the algorithm with instance R109a with 25 customers (see Section 7 for the description of instances). Consider two (to be compared) labels $L_1$ and $L_2$. The resources of $L_1$ are such that $i(L_1) = 16, d(L_1) = 45, c(L_1) = 404.5, t(L_1) = 52, V(L_1) = \{5, 16\}$, and $V(L_1) = \{5, 11, 12, 16, 21, 23\}$. The resources of label $L_2$ are such that $i(L_2) = 16, d(L_2) = 47, c(L_2) = 412.8, t(L_2) = 92, V(L_2) = \{14, 15, 16\}$, and $V(L_2) = \{2, 3, 5, 7, 8, 9, 11, 12, 14, 15, 16, 18, 19, 20, 21, 23\}$. Let’s further consider the GSR cut $I$ defined by the subset of nodes $S_I = \{5, 8, 14, 17\}$ with their corresponding demands $\{26, 9, 20, 2\}$ and integer $\kappa_I = 29$. Label $L_1$ visits customer 5 from the cut; therefore, $m(L_1) = 26$. Label $L_2$ visits customer 14 from the cut; therefore, $m(L_2) = 20$. Using Proposition 1, we should penalize $L_1$ with the dual of $I$ ($\xi_i = -12.7$) but not $L_2$. Hence, $L_2$ cannot be dominated by $L_1$.

Let’s now consider Proposition 2. The set of nodes in the cut that can still be visited by an extension of $L_2$ is $S_I \setminus V(L_2) = \{17\}$. We have $m(L_1) + d(S_I \setminus V(L_2)) = 26 + 2 = 28 < 29$. Hence, the cut $I$ will never be active in an extension of $L_1$. Consequently, $L_1$ must not be penalized by the dual variable $\xi_2$ and, hence, can dominate $L_2$.

5.3. Branching Rules

The branch-and-bound tree is explored using a best-bound strategy. The branching decisions are similar to those described in Section 4.5. However, given the new master problem formulation, we first start by branching on the selected segment from the outsourcing cost function. The algorithm looks for the segment $s \in \mathcal{S}$ from the cost function with the most fractional $z_s$ variable and creates the two branches $z_s \leq 0$ and $z_s \geq 1$. When the variables $z_s$ are integer for all $s \in \mathcal{S}$, the algorithm proceeds as in Section 4.5.

6. Implementation Features

In this section, we present some of the futures that we implemented in our framework but that did not result in a clear improvement of the algorithm’s performance. We implemented the $ng$-path relaxation and cover inequalities based on the $\delta$ variables. As the results were disappointing, we did not include them the computational results section.

In the $ng$-path relaxation introduced by Baldacci, Mingozzi, and Roberti (2011), a private path $p$ is allowed to have cycles but can only be extended to nodes that are not in a set of prohibited extensions. The size of this prohibited set of extensions (a parameter that can be tuned) determines the complexity of the resulting pricing problem and the quality of the lower bound.

Cover inequalities are well-known valid inequalities for the knapsack problem that were first discovered separately by Balas (1975) and Wolsey (1975). These inequalities can be strengthened to obtain the lifted cover inequalities (see, e.g., Zonghao et al. (1998) and Kaparis and Letchford 2008). For the set-partitioning formulation SP2 and after branching on the $z$ variables, the RHS of constraint (46) is fixed for the branch $z_s \geq 1, s \in \mathcal{S}$, hence defining an inequality for a knapsack problem in which the knapsack capacity is set to $U_s$ and the items to put in the knapsack are the orders to be subcontracted to the common carrier. We implemented both regular and lifted cover inequalities. The separation of the cover inequalities can be formulated as a knapsack problem and solved by dynamic programming as the one in Pisinger (1997).

7. Computational Results

The branch-and-cut-and-price algorithm is implemented in Java on an Intel Core i7 CPU, 2.8 GHz. For all experiments, we use a time limit of one hour. The LP relaxation of the master problems are solved using commercial solver Gurobi 6.5.1. For our numerical study, we use the modified Solomon’s data sets (Solomon 1987) introduced by Liu and Shen (1999) for the heterogeneous vehicle routing problem with time windows. Instances with 25, 50, and 100 customers are tested. For each instance size, six categories of instances are tested, $R1, R2, C1, C2, RC1, \text{ and } RC2$, according to the geographical distribution and the tightness of time windows. The geographical distribution of the customers are randomly generated for sets $R1$ and $R2$, are clustered for sets $C1$ and $C2$, and are a mix of random and clustered for sets $RC1$ and $RC2$. The time windows are tight in sets $R1, C1$, and $RC1$ and wide in sets $R2, C2$.
Table 3. Tariff Sheet for R Instances

<table>
<thead>
<tr>
<th>Quantity interval</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0–200</td>
<td>5.00</td>
</tr>
<tr>
<td>200–400</td>
<td>4.50</td>
</tr>
<tr>
<td>400–600</td>
<td>4.00</td>
</tr>
<tr>
<td>≥600</td>
<td>3.50</td>
</tr>
</tbody>
</table>

Table 4. Tariff Sheet for C Instances

<table>
<thead>
<tr>
<th>Quantity interval</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0–200</td>
<td>2.00</td>
</tr>
<tr>
<td>200–400</td>
<td>1.50</td>
</tr>
<tr>
<td>400–600</td>
<td>1.00</td>
</tr>
<tr>
<td>≥600</td>
<td>0.50</td>
</tr>
</tbody>
</table>

Table 5. Tariff Sheet for RC Instances

<table>
<thead>
<tr>
<th>Quantity interval</th>
<th>Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>0–200</td>
<td>3.50</td>
</tr>
<tr>
<td>200–400</td>
<td>3.00</td>
</tr>
<tr>
<td>400–600</td>
<td>2.50</td>
</tr>
<tr>
<td>≥600</td>
<td>2.00</td>
</tr>
</tbody>
</table>

7.1. Comparing Formulations SP1 and SP2

Table 6 compares performance of SP1 and SP2 (with cuts and strong dominance as described in Sections 4 and 5) on the instances with 25 customers. The table shows, respectively, the average CPU time for solving the instances to optimality (Time), the number of instances solved to optimality (#Optimal), the average value of the LP relaxation computed at the root node (Root LB), and the number of nodes in the branch-and-bound search tree (Tree) for SP2 without cuts and with cuts, respectively. In both cases, the strong dominance as described in Section 5.2 is used. Instances for which the root node could not be solved within the time limit by both algorithms (i.e., SP2 with and without cuts) are excluded from the root lower bounds calculation, and the instances that cannot be solved to optimality by both algorithms (i.e., SP2 with and without cuts) within the time limit are excluded from the CPU times calculation. The detailed results per instance can be found in Tables 1–9, reported in the online appendix. Additionally, in the online appendix, statistics on the number of added columns and cuts and time to solve the master and subproblems are reported for each instance in Tables 10–18.

The results shown in Table 7 reveal the effectiveness of the cuts described in Section 5. The SP2 formulation is strengthened by the cuts for all the instance sizes, resulting in improved lower bounds and a smaller branch-and-bound tree. In particular, almost 80% of the 25 customer instances as well as about 50% and 30% of those with 50 and 100 customers are solved to optimality when cuts are used, and the computing times are reduced by more than one third with respect to the algorithm without cuts.

Table 6. Comparison of Formulations SP1 and SP2 on Instances with 25 Customers

<table>
<thead>
<tr>
<th></th>
<th>SP1</th>
<th>SP2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time(s)</td>
<td>158</td>
<td>50</td>
</tr>
<tr>
<td>#Optimal</td>
<td>124</td>
<td>131</td>
</tr>
<tr>
<td>Root LB</td>
<td>676</td>
<td>673</td>
</tr>
<tr>
<td>Tree</td>
<td>4.2</td>
<td>6.5</td>
</tr>
</tbody>
</table>

However, solving two separate pricing subproblems is clearly computationally more demanding. From the results reported in Table 6, we can see that the SP2 formulation outperforms SP1 because seven additional optimal solutions can be obtained (i.e., about 6% more) within much smaller CPU times.

7.2. Impact of Valid Inequalities

Table 7 shows the average CPU time in seconds for solving the instances to optimality (Time), the number of instances solved to optimality (#Optimal), the average value of the LP relaxation computed at the root node (Root LB), and the average number of nodes in the branch-and-bound search tree (Tree) for SP2 with and without cuts, respectively. In both cases, the strong dominance as described in Section 5.2 is used. Instances for which the root node could not be solved within the time limit by both algorithms (i.e., SP2 with and without cuts) are excluded from the root lower bounds calculation, and the instances that cannot be solved to optimality by both algorithms (i.e., SP2 with and without cuts) within the time limit are excluded from the CPU times calculation. The detailed results per instance can be found in Tables 1–9, reported in the online appendix. Additionally, in the online appendix, statistics on the number of added columns and cuts and time to solve the master and subproblems are reported for each instance in Tables 10–18.
7.3. Impact of the Dominance

Table 8 shows the CPU times (in seconds) on solving a selection of the 25-customer instances using SP2 with and without the strong dominance conditions described in Section 5. Table 9 summarizes the total number of instances that are solved to optimality within the time limit and the average CPU times spent for obtaining the optimal solutions.

As shown in Tables 8 and 9, with the proposed strong dominance conditions, the pricing subproblems can be solved more efficiently. As a result, more instances can be solved to optimality, and the optimal solutions can be obtained using about one third of the CPU times. Strong dominance improves the overall performance as shown in Table 9. However, there are some instances in which weak dominance outperforms strong dominance, for example, C202b, C207b, and RC202b. This is due to the extra computation effort required by the strong dominance conditions, which sometimes does not lead to the desired effect.

7.4. Managerial Insights

By closely looking into the structure of the optimal solutions visualized in Figures 2–8, some interesting insights can be derived. In this section, we describe, through some illustrative examples, these insights and provide an intuitive explanation for them.

Table 8. Impact of Strong Dominance on CPU Times for Selected Instances with 25 Customers

<table>
<thead>
<tr>
<th>Instance</th>
<th>SP2 (weak dominance)</th>
<th>SP2 (strong dominance)</th>
</tr>
</thead>
<tbody>
<tr>
<td>R205a</td>
<td>—</td>
<td>131</td>
</tr>
<tr>
<td>R209a</td>
<td>—</td>
<td>956</td>
</tr>
<tr>
<td>C207a</td>
<td>—</td>
<td>68</td>
</tr>
<tr>
<td>RC201a</td>
<td>261</td>
<td>59</td>
</tr>
<tr>
<td>RC205a</td>
<td>—</td>
<td>3,096</td>
</tr>
<tr>
<td>R110b</td>
<td>3,209</td>
<td>5</td>
</tr>
<tr>
<td>C103b</td>
<td>740</td>
<td>166</td>
</tr>
<tr>
<td>C107b</td>
<td>2,809</td>
<td>9</td>
</tr>
<tr>
<td>R205b</td>
<td>3,246</td>
<td>617</td>
</tr>
<tr>
<td>R207b</td>
<td>—</td>
<td>656</td>
</tr>
<tr>
<td>R205c</td>
<td>232</td>
<td>80</td>
</tr>
<tr>
<td>R209c</td>
<td>—</td>
<td>2,911</td>
</tr>
<tr>
<td>C202b</td>
<td>5</td>
<td>88</td>
</tr>
<tr>
<td>C207b</td>
<td>10</td>
<td>167</td>
</tr>
<tr>
<td>RC202b</td>
<td>2</td>
<td>51</td>
</tr>
</tbody>
</table>

Impact of vehicle fixed cost: Figure 2 shows the optimal solutions of instances R109a and R109c. These two instances are similar in all characteristics except for vehicle fixed costs. In fact, instance R109a has higher fixed cost than instance R109c. Note that customers in these two instances are randomly spread in the geographical area. Clearly, customers located in more remote areas tend to be outsourced (in Figure 2(b), these are customers 17 and 24). Moreover, these customers have relatively smaller demands, that is, two and three, respectively. When the vehicle fixed costs increase, it is expected that more customers are outsourced. However, in Figure 2(a) in which vehicle fixed costs are higher than in Figure 2(b), only one additional customer is outsourced, that is, customer 25 with demand equal to six. This shows that R-type instances are not sensitive to vehicle fixed cost. For R instances, the variable traveling costs are also significant and even seem to be dominant. This is opposite to C (clustered) instances in which the variable traveling cost for serving customers in the same cluster is low, and therefore, it is expected that when a customer in a cluster is outsourced the whole cluster tends to be outsourced. This is reflected in Figure 3 in which we see that, for instance, C105, when the vehicle fixed cost increases, entire clusters are outsourced.
Similar behavior is observed in many other randomly distributed and clustered instances.

Impact of outsourcing: In this section, we discuss the impact of outsourcing on the structure of the optimal routes. Figures 4 and 5 show a visualization of the optimal routes for instances R109a and R111a, respectively. In particular, Figures 4(b) and 5(b) show the optimal routes when no outsourcing is allowed and all orders must be delivered by the private fleet. Furthermore, Figures 4(a) and 5(a) show the optimal routes when outsourcing is allowed. For these two instances, we can see that, by outsourcing some of the customers, better routes (i.e., with fewer crossings) can be constructed. This might be due to the exclusion of inconvenient customers (e.g., because of their time windows) from the private fleet routing plan. As in Euclidean instances, crossings in a route are only caused by the existence of time windows; the outsourcing option allows for the construction of routes with fewer crossings even in instances with tight time windows as is the case for instances R109 and R111.

7.4.1. Impact of Quantity Discounts. In this section, we discuss the impact of quantity discounts on the optimal routes. We solved several instances without considering quantity discounts. We used the highest cost in the common carrier’s tariff sheet; that is, regardless of the quantity outsourced, the unit outsourcing cost is 5.00 for R instances, 3.50 for RC instances, and 2.00 for C instances. For all instance types, discarding quantity discounts increases, as expected, the optimal values. For the random instances, discarding discounts did not have a clear impact on the structure of the optimal routes as shown in Figure 6, for instance, RC105 with 100 customers. In this case, only one additional customer (namely, customer 53 with demand five) is subcontracted when quantity discounts are offered. Therefore, from the common carrier’s point of view, it may not be beneficial to offer discounts when delivery locations are randomly dispersed over the shipper’s operating area as the shipper outsources the same quantity regardless of whether quantity discounts are present or not. For clustered instances, we instead observe a significant change in the structure of optimal routes as depicted in Figures 7 and 8 for instances RC107a and C105b, both with 50 customers. The more clustered the instance is, the more significant the impact of discounts on the optimal routes is. For clustered instances, the common carrier has more incentive to offer discounts as the shipper outsources more when quantity discounts are offered.

8. Conclusion and Future Work
In this paper, we presented the first exact approach for the RVRPPC, which is a new variant of the VRPPC with heterogeneous fleet and time windows and includes cost quantity discounts for the outsourced customers. To the best of our knowledge, this is the first exact approach for a problem of the VRPPC family.

The exact algorithm is based on the branch-and-cut-and-price paradigm and considers two different set-partitioning formulations. The first one separately considers the outsourced customers and those belonging to private fleet routes, for which two specialized pricing procedures are derived, called private and common pricing procedures. Although the linear relaxation of this first formulation is very good, overall it leads to high symmetry, thus proving not very efficient in the exact solution of the problem. Therefore, we defined a second formulation in which only variables for private paths are considered and that has a better overall efficacy. The performance of the algorithm is
further enhanced by using new valid inequalities of the subset-row type and a novel and strong dominance procedure that mitigates the negative effect of nonrobust inequalities. In particular, the procedure can be extended to many other shortest path–based algorithms and has, therefore, a wide practical use.

Figure 6. (Color online) Solution of RC105 with 100 Customers with and Without Discount

Figure 7. Solution of RC107 with 50 Customers with and Without Discount

Figure 8. Solution of C105 with 50 Customers with and Without Discount

An extensive testing of the exact method on different instance classes obtained by extending known benchmarks from the literature allowed for a thorough comparison of the two formulations and of the effect of the various original components of the approach. In particular, the new subset-row inequalities allowed for the solution of some additional instances, and the new dominance reduced considerably the overall computing times. As a consequence, the exact algorithm was able to solve a large number of instances with up to 100 customers (i.e., 272 out of 504 instances and more than 80% of those with 25 customers). The results are integrated by an analysis of the managerial insights that can be derived from the optimal solutions obtained with the algorithm, showing, for example, that the outsourcing option is sensitive to vehicle fixed costs only in some cases related to a customer’s spatial distribution. Moreover, the outsourcing option allows for a more compact shaping of the private fleet routes.

As this is the first exact algorithm for this family of problems, there is clearly some room for improvement and future research. In particular, it would be strongly
beneficial to reduce the impact of solution symmetry incurred when using the first set-partitioning formulation, which has a stronger linear relaxation. Moreover, the developed exact method can be the base of effective matheuristics for the approximate solution of the problem.

References


