A new family of route formulations for split delivery vehicle routing problems

Isaac Balster[1], Teobaldo Bulhões[2], Pedro Munari[3], Artur A. Pessoa[4], and Ruslan Sadykov[5]

[1]Inria Centre at the University of Bordeaux, 200 Avenue de la Vieille Tour, Talence 33405, France, isaac.balster@inria.fr
[2]Universidade Federal da Paraíba, Centro de Informática, Departamento de Computação Científica, Rua dos Escoteiros s/n, Mangabeira, 58055-000, João Pessoa, Brazil, tbulhoes@ci.ufpb.br
[3]Federal University of São Carlos, Production Engineering Department, Rod. Washington Luís Km 235, 13565-905, São Carlos-SP, Brazil, munari@dep.ufscar.br
[4]Universidade Federal Fluminense, Engenharia de Produção, Rua Passo da Pátria 156, Niterói - RJ - Brasil - 24210-240, arturpessoa@id.uff.br
[5]Inria Centre at the University of Bordeaux, 200 Avenue de la Vieille Tour, Talence 33405, France, ruslan.sadykov@inria.fr

Abstract

We propose a new family of formulations with route-based variables for the split delivery vehicle routing problem with and without time windows. Each formulation in this family is characterized by the maximum number of different demand quantities that can be delivered to a customer during a vehicle visit. As opposed to previous formulations in the literature, the exact delivery quantities are not always explicitly known in this new family. The validity of these formulations is ensured by an exponential set of non-robust constraints. Additionally, we explore a property of optimal solutions that enables us to determine a minimum delivery quantity based on customer demand and vehicle capacity, and this number is often greater than one. We use this property to reduce the number of possible delivery quantities in our formulations, improving the solution times of the computationally strongest formulation in the family. Furthermore, we propose new variants of non-robust cutting planes that strengthen the formulations, namely limited-memory subset-row covering inequalities and limited-memory strong k-path inequalities. Finally, we develop a branch-cut-and-price (BCP) algorithm to solve our formulations enriched with the proposed valid inequalities, which resorts to state-of-the-art algorithmic enhancements. We show how to effectively manage the non-robust cuts when solving the pricing problem that dynamically generates route variables. Numerical results indicate that our formulations and BCP algorithm establish new state-of-the-art results for the variant with time windows, as many benchmark instances with 50 and 100 customers are solved to optimality for the first time. Several instances of the variant without time windows are solved to proven optimality for the first time.

Keywords. vehicle routing, time windows, split delivery, branch-cut-and-price, non-robust cuts

1 Introduction

The applications of the vehicle routing problem (VRP) are ubiquitous and play an important role in promoting effective logistics operations that contribute to economic, environmental and sustainable goals (Shapiro 2007, Bektaş and Laporte 2011). The success of these applications rests on the results of intensive research developments made by the VRP community over more than 60 years. These efforts have involved the design of a number of mathematical formulations and solution approaches for both theoretical and applied variants, continuously pushing the boundaries of the size and type of problems that one can expect to solve in practice (Toth and Vigo 2014, Braekers, Ramaekers, and Nieuwenhuyse 2016).

In the traditional variants of the capacitated VRP, one must design a set of least-cost routes in a way to visit each customer exactly once using a homogeneous fleet of vehicles available in a single depot such that the total demand delivered in each route does not exceed the vehicle capacity. In certain variants, such as the VRP with time windows (VRPTW), customers might only be available for service during a certain period of time throughout the day, known as a time window, and service times might vary among customers. In this paper, we are mainly interested in an extension of this latter variant, known in the literature as the split delivery VRPTW (SDVRPTW), in which customers may be visited more than once, if beneficial, so that their demands are split between two or more vehicles.

The SDVRPTW adds a degree of operational flexibility by relaxing the VRPTW and encompassing the decision of how much to deliver to each customer. Multiple visits allow us to include customers with demands that are larger than the vehicle capacity. Additionally, multiple visits can be beneficial even if a customer's demand fits in a single vehicle since these split deliveries can promote significant savings by increasing the utilization of the vehicles' capacity. As pointed out originally by Dror and Trudeau (1989, 1990) and Archetti, Savelsbergh, and Speranza (2006) for the variant without time windows (SDVRP), the savings can reach up to 50% of routing costs. However, the benefits of multiple visits come in exchange for increased difficulty in modeling and solving these variants with respect to their nonsplit counterparts, especially regarding exact approaches (Bianchessi and Irnich 2019, Munari and Savelsbergh 2022, Gouveia, Leitner, and Ruthmair 2023).

1.1 Related literature

Despite its practical benefits and theoretical relevance, the SDVRPTW has received relatively little attention in the literature. Mullaseril and Dror (1996) presented the first attempt to model the SDVRPTW using a column generation scheme. Their formulation relies on the replication of customers, followed by the definition of split configurations in advance so that the problem becomes an instance of the VRPTW. A first standard branch-and-price scheme tailored for the SDVRPTW and with no initial assumptions on the number of splits was presented by Feillet et al. (2006), in which the decision on how much to deliver is addressed at the master problem level, whereas feasible routes are determined in the pricing subproblem, which consists of an elementary shortest path problem with resource constraints (ESPPRC). Desaulniers (2010) proposed an innovative branch-cut-and-price (BCP) algorithm based on extreme delivery patterns, which are determined in the pricing subproblem together with their corresponding routes, and the actual delivery quantities are determined through convex combinations of these extreme patterns at the master problem level. Archetti, Bouchard, and Desaulniers (2011) enhanced this approach by implementing acceleration techniques on the subproblem by means of a tabu search heuristic, as well as presenting novel valid inequalities. Luo et al. (2017) presented a BCP algorithm that extends the extreme delivery pattern concept to the SDVRPTW with linear weight-related costs and takes advantage of acceleration techniques in their label-setting pricing algorithm.

More recently, tailored branch-and-cut (BC) algorithms have shown superior performance for the SDVRPTW variant and have become the state-of-the-art exact approaches. Bianchessi and Irnich (2019) proposed a tailored BC method for the SDVRPTW based on a relaxed commodity flow formulation. The authors presented new types of valid inequalities, which, together with other well-known cuts from the literature, are used in their BC to strengthen the linear relaxation of the relaxed model, as well as cut off integer solutions that are infeasible for the SDVRPTW. Munari and Savelsbergh (2022) introduced three novel compact formulations for the SDVRP and SDVRPTW and proposed a BC algorithm based on a relaxation of their best-performing formulation. Different from previous approaches, their BC algorithm locally extends the relaxed model by inserting new variables and the so-called regularity property constraints every time an infeasible integer solution is found in the BC tree. The same BC algorithm was used by Munari and Savelsbergh (2020) to develop a column generation-based heuristic that consists of adding to the relaxed formulation a set of time-feasible routes that are generated in advance. Other heuristics have also been proposed specifically for the SDVRPTW and related variants, employing different strategies. Frizzell and Giffin (1995) developed a construction heuristic using a look-ahead approach, along with improvement heuristics based on moving and exchanging customers between routes. Mullaseril, Dror, and Leung (1997) adapted the construction and improvement heuristics proposed by Dror and Trudeau (1990) to solve a variant applied to livestock feed distribution that was modeled as a split-delivery capacitated rural postman problem with time windows on arcs. Belfiore and Yoshizaki (2009) also addressed a real-life variant, modeled as a heterogeneous fleet SDVRPTW, by proposing constructive heuristics and a scatter search algorithm. Finally, Ho and Haugland (2004) developed a tabu search heuristic for the SDVRPTW, using traditional move operators (relocate, exchange and 2-opt*) as well as a new move operator called relocate split. The authors conducted computational experiments using modified Solomon's instances. Except for Munari and Savelsbergh (2020), none of the mentioned heuristics have been tested on the exact same benchmark instances considered in the experiments performed with the recent exact approaches (Archetti, Bouchard, and Desaulniers 2011, Bianchessi and Irnich 2019, Munari and Savelsbergh 2022).

The aforementioned state-of-the-art exact approaches can effectively solve most benchmark instances with up to 50 customers, but they often become ineffective if the number of customers increases. For example, Bianchessi and Irnich (2019) reported proven optimal solutions for 104 of 168 instances with 50 customers, while this number decreased to 5 of 168 for instances with 100 customers. For the same instances, Munari and Savelsbergh (2020, 2022) presented proven optimal solutions for 123 50-customer instances, whereas they reported relatively large integrality gaps for most 100-customer instances. This behavior has not been observed in the results reported in the literature for 100-customer instances of traditional VRP variants, such as the VRPTW, when they are solved using a BCP method (see, e.g., Sadykov, Uchoa, and Pessoa (2021)), suggesting that there could be room for improvement in the computational solution of the SDVRPTW.

All of the aforementioned algorithms can also be applied for the SDVRP, i.e., the variant without time windows. However, only Munari and Savelsbergh (2022) presented results for both the SDVRPTW and the SDVRP. Exact approaches proposed in the literature specifically for the SDVRP include those by Jin, Liu, and Eksioglu (2008), Moreno, De Aragao, and Uchoa (2010), Archetti, Bianchessi, and Speranza (2011, 2014), Ozbaygin, Karasan, and Yaman (2018), Gouveia, Leitner, and Ruthmair (2023). These approaches were not extended to include time windows or verified on SDVRPTW instances. Again, tailored branch-and-cut algorithms dominate the state of the art for the SDVRP. The current best algorithms by Gouveia, Leitner, and Ruthmair (2023) and by Munari and Savelsbergh (2022) are able to solve to optimality most of the literature instances with 50 customers and a small proportion of instances having between 64 and 100 customers.

1.2 Contributions

The main contributions of this paper are summarized as follows.

- We explore a property that holds true for at least one optimal solution of the problem, applicable to both the SDVRP and the SDVRPTW. This property allows us to establish a minimum delivery quantity when visiting a customer on a route. The efficiency of our solution approach depends on the ratio between the minimum delivery quantity and average customer demand. The larger this ratio, the better our algorithm performs.

- We introduce a new family of route-based formulations for the problem, as well as a BCP algorithm for solving them. In contrast to previous column generation-based approaches in the literature, the exact delivery quantities are not always explicitly known, neither in the pricing problem nor in the master problem. Covering of customer demand is ensured through an exponential family of constraints separated dynamically using a maximum flow-based algorithm. These constraints are supported by a flow graph representation of a solution, which provides both theoretical and practical advantages, as we show in this paper.

- We propose variants of non-robust valid inequalities designed to improve the strength of our formulations, which are limited-memory subset-row covering inequalities and limited-memory strong k-path inequalities. Additionally, we show how to effectively manage novel non-robust valid inequalities when solving the pricing problem of the column generation procedure.

- We numerically compare the strength and the solution time of the formulations in the newly proposed family using instances of the SDVRPTW with different characteristics.

- Finally, we show that our BCP algorithm outperforms the state-of-the-art exact approaches for the SDVRPTW. We achieve optimality for numerous benchmark instances for the first time, including the majority of instances with 50 customers and many instances with 100 customers. Additionally, we solve to optimality a few SDVRP instances for the first time.

1.3 Organization of the paper

The remainder of this paper is organized as follows. In Section 2, we define the problem and state its known properties of optimal solutions. A new property is introduced in Section 3. In Section 4, a new family of formulations is presented, as well as known and novel valid inequalities for this family. Our BCP algorithm is described in Section 5. The results of computational experiments are shown in Section 6, in which we numerically compare the formulations from the proposed family and test our BCP algorithm on benchmark instances for both the SDVRPTW and the SDVRP. Section 7 outlines the major contributions and future research directions.

2 Problem definition and known properties

We define the SDVRPTW over a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V} = \{0, n+1\} \cup \mathcal{C}$ and $\mathcal{A} = \{(i,j) : i,j \in \mathcal{V},\ i \neq n+1,\ j \neq 0,\ i \neq j\}$. Nodes 0 and $n+1$ are the source and sink representations of the depot, respectively. Set $\mathcal{C} = \{1, 2, \ldots, n\}$ represents customer nodes. For each arc $a \in \mathcal{A}$, a cost $c_a \geq 0$ and a travel time $t_a \geq 0$ is defined. For the sake of simplicity, when $a = (i,j)$, we may drop the parenthesis and the comma, and replace $c_{(i,j)}$ and $t_{(i,j)}$ with c_{ij} and t_{ij}, respectively. We suppose that the triangle inequality holds for both costs and travel times. In practice, triangle inequality always holds for travel times. Thus, in the case in which arc costs are equal to arc travel times, this assumption comes without loss of generality. Each customer $i \in \mathcal{C}$ has a rational positive demand d_i to be fulfilled by one or more vehicles, and this customer is available for service within a nonempty interval $[e_i, l_i]$, where e_i and l_i stand for the earliest and latest times for starting service, respectively. The time window of the depot $[e_0, l_0] = [e_{n+1}, l_{n+1}]$ defines the planning horizon. In the case of an early arrival at a customer, waiting before starting the service is allowed. The service time for each customer $i \in \mathcal{C}$ is assumed to be constant and is embedded into the travel times of all arcs (i,j) leaving i. A homogeneous fleet of H vehicles with rational, positive capacity Q is available. A route $r = (v_0^r = 0, v_1^r, \ldots, v_{n_r}^r = n+1)$ visiting n_r nodes in graph $\mathcal{G}$, starting at time $t_0^r = e_0$ and delivering amount $d_k^r \geq 0$ of demand to every customer v_k^r, $1 \leq k < n_r$, is feasible if the total delivered demand $\sum_{k=1}^{n_r-1} d_k^r$ does not exceed Q, and the service start time t_k^r at every node is within its time window. The service start time t_k^r, $1 \leq k \leq n_r$, is recursively determined as $t_k^r = \max\{t_{k-1}^r + t_{a_k^r}, e_{v_k^r}\}$, where a_k^r denotes the arc (v_{k-1}^r, v_k^r). We define q_i^r as the total demand delivered by route r to customer i, which may involve multiple visits. The cost c^r of route r is the total of the costs of the arcs that it traverses: $c^r = \sum_{k=1}^{n_r} c_{a_k^r}$. A feasible solution of the problem consists of a set of at most H feasible routes in which the total delivered quantity to every customer $i \in \mathcal{C}$ is at least equal to d_i. A split customer in a solution is a customer who is visited by two or more vehicles. The aim of the SDVRPTW is to find a feasible solution that minimizes the total cost of the routes. A related variant, the (capacitated) split delivery vehicle routing problem (SDVRP), is the most basic one, in which time windows, service times, and route timings are not defined.

There always exists an optimal solution to the SDVRPTW that satisfies the following properties:

Property 1 *(Dror and Trudeau 1990, Feillet et al. 2006) Two routes in the solution share at most one single split customer;*

Property 2 *(Feillet et al. 2006) Each route is elementary: it visits each customer at most once;*

Property 3 *(Feillet et al. 2006) Each arc between customer nodes is traversed at most once;*

Property 4 *(Desaulniers 2010) For each pair of arcs between two customers, at most one is traversed; and*

Property 5 *(Archetti, Bouchard, and Desaulniers 2011) All delivery quantities are integers if demand and vehicle capacity are integers.*

In the next section, we present a new property that generalizes Property 5.

3 Flow graph solution representation and a useful property

The flow graph solution representation introduced in this section can be used to verify the feasibility of a set of routes with respect to the customer demand. This graph has theoretical and practical importance since we rely on it to prove the validity of our formulations and to separate some of our valid inequalities.

Let $\tilde{\mathcal{R}} = \{r_1, r_2, \ldots, r_{|\tilde{\mathcal{R}}|}\}$ be a set of time-feasible routes (i.e., satisfying time window constraints), in which delivery quantities are not defined. For ease of presentation, we consider that routes r_i and r_j are different whenever $i \neq j$, even if they follow the same sequence of arcs. We now construct the following valued graph $\mathcal{F}(\tilde{\mathcal{R}})$ to check whether set $\tilde{\mathcal{R}}$ defines a feasible SDVRPTW solution. In the case of positive answer, delivery quantities are determined. The values of arcs in $\mathcal{F}(\tilde{\mathcal{R}})$ correspond to their capacities.

The set of nodes in $\mathcal{F}(\tilde{\mathcal{R}})$ is $\{0\} \cup \tilde{\mathcal{R}} \cup \mathcal{C} \cup \{n+1\}$. Nodes 0 and $n+1$ are the source and the sink, respectively. The first set of arcs $\mathcal{A}_1(\tilde{\mathcal{R}})$ connects the source with each of the route nodes in $\tilde{\mathcal{R}}$. The capacity of these arcs is Q. The second set of arcs $\mathcal{A}_2(\tilde{\mathcal{R}})$ connects the route nodes in $\tilde{\mathcal{R}}$ with customer nodes in $\mathcal{C}$: arc (r, i) belongs to $\mathcal{A}_2(\tilde{\mathcal{R}})$ if and only if route $r \in \tilde{\mathcal{R}}$ visits customer $i \in \mathcal{C}$. The capacity of these arcs is ∞. Finally, the third set of arcs $\mathcal{A}_3(\tilde{\mathcal{R}})$ connects each customer node to the sink. The capacity of an arc $(i, n+1) \in \mathcal{A}_3(\tilde{\mathcal{R}})$ is d_i. Figure 1 provides an illustration of graph $\mathcal{F}(\tilde{\mathcal{R}})$, where $\tilde{\mathcal{R}} = \{r_1 = \{0, 1, 2, 3, 6\}, r_2 = \{0, 2, 3, 6\}, r_3 = \{0, 4, 5, 6\}\}$ is a set of routes serving customers $\mathcal{C} = \{1, 2, 3, 4, 5\}$ with demands $d = \{10, 20, 30, 40, 10\}$, respectively. The capacity of the vehicle is $Q = 30$. From the construction of the graph, we deduce the following observation.

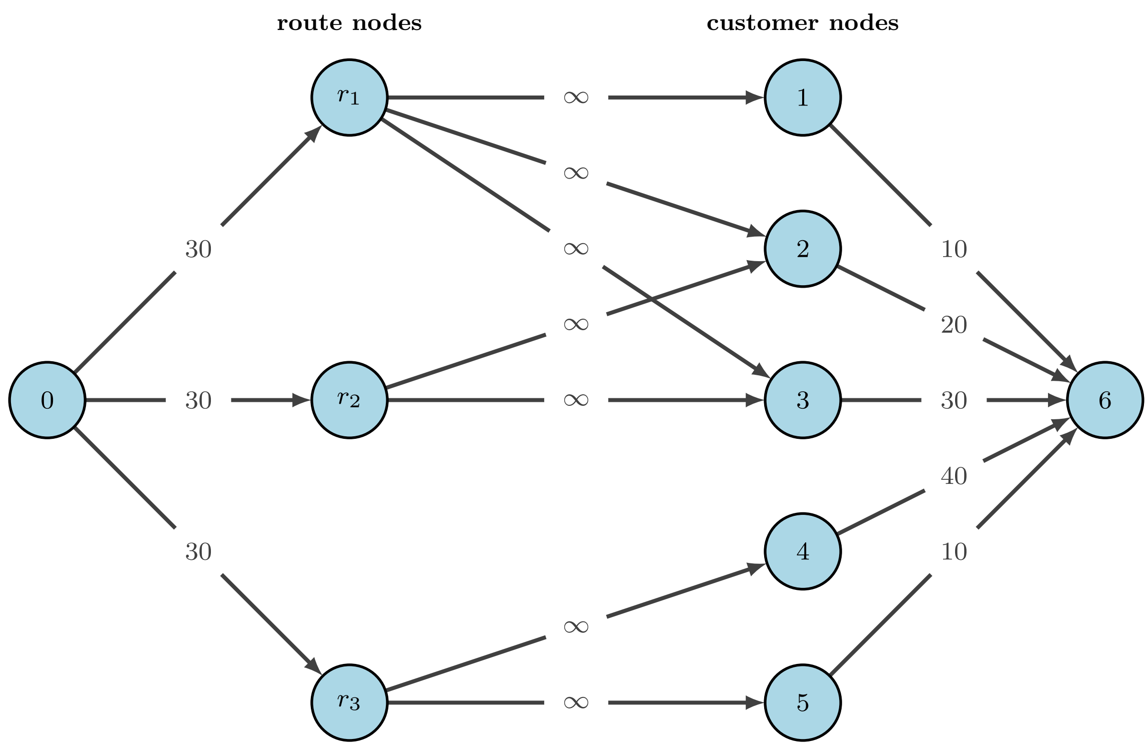

Figure 1: An example of the flow graph $\mathcal{F}(\tilde{\mathcal{R}})$.

Observation 1 *A set $\tilde{\mathcal{R}}$ of time-feasible routes forms a feasible SDVRPTW solution if and only if the maximum flow f in graph $\mathcal{F}(\tilde{\mathcal{R}})$ has a value of $\sum_{i\in\mathcal{C}} d_i$. In such a case, values f_a, $a \in \mathcal{A}_2$, correspond to the delivery quantities for every route in $\tilde{\mathcal{R}}$ to each customer in $\mathcal{C}$.*

The maximum flow f in the example flow graph depicted in Figure 1 has a value of 90, which is less than $\sum_{i\in\mathcal{C}} d_i = 110$. As a result, this particular set of routes does not form a feasible solution.

Let now $\bar{q} = gcd(Q, d_1, d_2, \ldots, d_n)$ be the greatest common divisor of the capacity and all customer demands. Since all these values are rational and positive, value $\bar{q}$ exists (Weil 1983).

Property 6 *There exists an optimal solution in which all delivery quantities in all routes are multiples of $\bar{q}$.*

Proof. Let I be a problem instance. We define a new instance I' derived from I by dividing all demands and capacity by $\bar{q}$. In this case, any solution s' of I' is feasible if, and only if, there is an equivalent feasible solution s of I with the same cost, and the same delivery quantities multiplied by $\bar{q}$. Since, as stated in Property 5, there exists an optimal solution s' of I' in which all delivery quantities are integers, the corresponding solution s of I is also optimal and satisfies Property 6. □

4 Mathematical formulations

In this section, we introduce a new family of formulations based on route variables. Let $\mathcal{R}$ be the set of all feasible SDVRPTW routes satisfying Property 6. Requiring elementarity for routes may make the route generation subproblem difficult to solve. Thus, we do not enforce Property 2 for routes in $\mathcal{R}$.

We define $D_i = \{\bar{q}, 2\bar{q}, \ldots, d_i\}$ as the set of all possible delivery quantities to customer $i \in \mathcal{C}$ via any route in $\mathcal{R}$. Zero delivery is not included in D_i due to the triangle inequalities. For a given route $r \in \mathcal{R}$, customer $i \in \mathcal{C}$, and delivery quantity $q \in D_i$, we define the parameter b^r_{iq} as the number of times r makes a visit to i with a delivery quantity equal to q.

4.1 Base formulation

Let h_{rS} be a binary value that is equal to 1 if and only if route r enters subset $S \subseteq \mathcal{C}$. For every route $r \in \mathcal{R}$, we define a nonnegative integer variable θ_r, which represents the number of vehicles that follow route r. We now state our first formulation, which we denote as (F0).

$$\text{(F0):} \quad \text{Min} \quad \sum_{r\in\mathcal{R}} c^r \theta_r, \tag{1}$$

$$\text{s.t.} \quad \sum_{r\in\mathcal{R}} h_{rS}\theta_r \geq \left\lceil \sum_{i\in S} d_i/Q \right\rceil, \qquad \forall S \subseteq \mathcal{C}, \tag{2}$$

$$\theta_r \in \mathbb{Z}^+, \qquad \forall r \in \mathcal{R}. \tag{3}$$

Objective Function (1) minimizes the sum of routing costs. Constraints (2) correspond to the strong k-path inequalities introduced by Baldacci, Christofides, and Mingozzi (2008) and used for the SDVRPTW in Archetti, Bouchard, and Desaulniers (2011). These inequalities are a strengthened version of well-known rounded capacity cuts, with the same right-hand side. The difference lies in the coefficient h_{rS}, which is equal to one even if route r leaves and enters subset S several times. Finally, the integrality of the θ_r variables is ensured by (3).

We now show that Constraints (2) and (3) suffice to define the set of feasible solutions to our problem. We first denote by $\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}}) = \{r_1, r_2, \dots, r_{|\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}})|}\}$ the set of routes corresponding to an integer solution $\bar{\boldsymbol{\theta}}$: every route $r \in \mathcal{R}$ is added $\bar{\theta}_r$ times to $\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}})$. Again, we consider that $r_i \neq r_j$ are different whenever $i \neq j$, although there may be multiple routes in $\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}})$ representing the same route $r \in \mathcal{R}$ in case $\bar{\theta}_r > 1$.

If an integer solution $\bar{\boldsymbol{\theta}}$ satisfies Constraints (2), then set $\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}})$ of routes forms a feasible SDVRPTW solution.

Proof by contradiction. Suppose that set $\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}})$ does not form a feasible SDVRPTW solution. By Observation 1, it follows that, in the flow graph $\mathcal{F}(\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}}))$, the maximum flow value is strictly less than $\sum_{i\in\mathcal{C}} d_i$. By the min-cut-max-flow theorem, the value of a minimum $0-(n+1)$ cut C in this graph is strictly smaller than $\sum_{i\in\mathcal{C}} d_i$. We denote by $\mathcal{R}(C)$ the set of route nodes $r \in \tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}})$ in graph $\mathcal{F}(\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}}))$ such that arc $(0, r)$ is included in cut C. Set $\mathcal{R}(C)$ is not empty since the cut cannot contain all arcs in $\mathcal{A}_3(\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}}))$ and cannot contain any arc in $\mathcal{A}_2(\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}}))$. We also denote by $\mathcal{C}(C)$ the set of customer nodes $i \in \mathcal{C}$ in graph $\mathcal{F}(\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}}))$ such that arc $(i, n+1)$ is not included in cut C. No arc in $\mathcal{A}_2(\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}}))$ crosses cut C in the direction of the sink since otherwise the value of the cut would be ∞. Therefore, no route in $\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}}) \setminus \mathcal{R}(C)$ enters subset $\mathcal{C}(C)$ of customers. Conversely, all routes in $\mathcal{R}(C)$ enter $\mathcal{C}(C)$; otherwise, C would not be a minimum cut. We therefore have

$$\sum_{r\in\mathcal{R}} Q h_{r,\mathcal{C}(C)} \bar{\theta}_r = |\mathcal{R}(C)| \cdot Q < \sum_{i\in\mathcal{C}(C)} d_i. \tag{4}$$

The last inequality in (4) holds because the value of cut C is $\sum_{i\in\mathcal{C}\setminus\mathcal{C}(C)} d_i + |\mathcal{R}(C)| \cdot Q$, which should be smaller than $\sum_{i\in\mathcal{C}} d_i = \sum_{i\in\mathcal{C}\setminus\mathcal{C}(C)} d_i + \sum_{i\in\mathcal{C}(C)} d_i$. By canceling the term $\sum_{i\in\mathcal{C}\setminus\mathcal{C}(C)} d_i$ in both expressions, we obtain the result. Dividing Inequality (4) by Q yields $\sum_{r\in\mathcal{R}} h_{r,\mathcal{C}(C)} \bar{\theta}_r < \sum_{i\in\mathcal{C}(C)} d_i/Q$, and Constraint (2) for set $\mathcal{C}(C)$ of customers is violated by solution $\bar{\boldsymbol{\theta}}$. Therefore, our assumption is incorrect, and set $\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}})$ forms a feasible SDVRPTW solution. □

Constraints (2) can be exactly separated for integer solutions $\bar{\boldsymbol{\theta}}$ of Formulation (F0) in polynomial time by defining graph $\mathcal{F}(\tilde{\mathcal{R}}(\bar{\boldsymbol{\theta}}))$ and finding a minimum cut in it.

The minimum $0-6$ cut in the example flow graph in Figure 1 includes customer nodes 4 and 5 at the sink side of the cut. The subset consisting of these two customers induces a violated inequality (2). Only one route in $\tilde{\mathcal{R}}$ enters this subset, whereas at least two routes are needed.

Observation 2 *Formulation (F0) is correct even without any information about delivery quantities in routes $r \in \mathcal{R}$.*

Although Observation 2 highlights an interesting theoretical property, it is not always possible to use Formulation (F0) in practice. We define $\mathcal{R}^0$ as the set of routes without specification of delivery quantities. Obviously, $\mathcal{R}^0$ is not smaller than $\mathcal{R}$. Set $\mathcal{R}^0$ is not finite in two cases: i) for the standard SDVRP; and ii) for the SDVRPTW if there exists a cycle with zero total travel time. This is because routes in $\mathcal{R}^0$ are not necessarily elementary and are not constrained by the vehicle capacity. Thus, the length of routes is not restricted in the SDVRP case. This is also the case for the SDVRPTW with zero travel times, as a route may indefinitely circulate along a cycle with zero travel time. Even for the SDVRPTW with only positive travel times, Formulation (F0) is weak, as shown in the computational experiments.

4.2 A family of partially discretized formulations

Formulation (F0) can be strengthened in the following way. Since the triangle inequalities are satisfied, there exists an optimal solution with no zero deliveries. Thus, we can reduce the number of variables in Formulation (F0) by only considering variables θ_r for routes delivering at least $\bar{q}$ at every visit. The minimum delivery of one unit was already imposed by Archetti, Bianchessi, and Speranza (2011). We denote this strengthened formulation as (F1). Moreover, there exists an optimal solution $\bar{\boldsymbol{\theta}}$ to the linear relaxation of (F1) in which every route r such that $\bar{\theta}_r > 0$ delivers exactly $\bar{q}$ at every visit. Consequently, the length of each route is at most $Q/\bar{q}$. We denote the set of all such routes as $\mathcal{R}^1 \subseteq \mathcal{R}$.

There are further ways to strengthen (F0) if we consider the information about full and partial deliveries along routes. Recall that b^r_{iq} is the number of times that a route $r \in \mathcal{R}$ includes a visit to i with delivery quantity equal to q. We denote as $b^r_{i\mathrm{F}} = b^r_{id_i}$ the number of times that route $r \in \mathcal{R}$ delivers full demand to customer $i \in \mathcal{C}$. Additionally, let $b^r_{i\mathrm{P}} = \sum_{q \in D_i \setminus \{d_i\}} b^r_{iq}$ be the number of times that route $r \in \mathcal{R}$ delivers partial demand to customer $i \in \mathcal{C}$. The following constraints are valid for the SDVRPTW:

$$\sum_{r \in \mathcal{R}} (2b^r_{i\mathrm{F}} + b^r_{i\mathrm{P}})\theta_r \geq 2, \quad \forall i \in \mathcal{C}. \tag{5}$$

These constraints are a special case of the strong minimum number of vehicles (SVM) inequalities used by Archetti, Bouchard, and Desaulniers (2011).

We denote as (F2) the formulation with Objective Function (1) and Constraints (2), (3) and (5). Let $\mathcal{R}^2 = \{r \in \mathcal{R} : b^r_{iq} = 0,\ \forall i \in \mathcal{C},\ \forall q \in D_i \setminus \{\bar{q}, d_i\}\}$ be the set of routes in which the delivery quantity in every visit to customer $i \in \mathcal{C}$ is equal either to $\bar{q}$ or to d_i.

Observation 3 *There exists an optimal solution $\bar{\boldsymbol{\theta}}$ to the linear relaxation of Formulation (F2) such that $\bar{\theta}_r = 0$ for all $r \notin \mathcal{R}^2$.*

This observation is derived from the fact that the coefficients of variables in Formulation (F2) depend only on whether a delivery is full or partial, and not on the exact delivery quantities. As a result, any route $r' \notin \mathcal{R}^2$ is dominated, i.e., can be replaced by a route $r \in \mathcal{R}^2$ in any optimal solution of the relaxation without losing feasibility and optimality of this solution. Thus, we can restrict the set of routes in Formulation (F2) to $\mathcal{R}^2$ without compromising the validity of the dual bound provided by its linear relaxation.

Restricting the number of different delivery quantities to a customer is useful to speed up the dynamic generation of route variables, as will be shown in Section 5. We now extend the case with at most two different delivery quantities per customer to any integer $K \geq 2$. As will be shown by computational experiments, increasing the value of K renders the formulation stronger, possibly at the expense of slower generation of route variables. We start with the observation that the following constraints are valid for the SDVRPTW:

$$\sum_{r \in \mathcal{R}} \sum_{q \in D_i} q b^r_{iq} \theta_r \geq d_i, \quad \forall i \in \mathcal{C}. \tag{6}$$

For some customers $i \in \mathcal{C}$, we derive new inequalities as follows. Let us define $\mathcal{C}(K) = \{i \in \mathcal{C} : K\bar{q} < d_i\}$ as the set containing each customer $i \in \mathcal{C}$ to which K deliveries of size $\bar{q}$ are not enough to satisfy the demand d_i. For a given customer $i \in \mathcal{C}(K)$, we multiply Inequality (6) by $(K-1)/(d_i - \epsilon)$, where $\epsilon > 0$ is a constant significantly smaller than $\bar{q}$. Next, we apply Chvátal-Gomory rounding on both sides of the resulting inequality and obtain

$$\sum_{r\in\mathcal{R}}\sum_{q\in D_i}\left\lceil\frac{(K-1)qb^r_{iq}}{d_i-\epsilon}\right\rceil\theta_r \geq \left\lceil\frac{(K-1)d_i}{d_i-\epsilon}\right\rceil. \tag{7}$$

Since $d_i/(d_i-\epsilon)$ is slightly greater than 1, the right-hand side of (7) is equal to K. Additionally, given that b^r_{iq} is integer, the following holds for the coefficient in the left-hand side of these inequalities:

$$b^r_{iq}\left\lceil\frac{(K-1)q}{d_i-\epsilon}\right\rceil \geq \left\lceil\frac{(K-1)qb^r_{iq}}{d_i-\epsilon}\right\rceil. \tag{8}$$

Hence, we can rewrite (7) as

$$\sum_{r\in\mathcal{R}}\sum_{q\in D_i}b^r_{iq}\left\lceil\frac{(K-1)q}{d_i-\epsilon}\right\rceil\theta_r \geq K. \tag{9}$$

For a given value of K and delivery quantity $q \in D_i$ of a route r, the rounded-up coefficient in the left-hand side of these inequalities is a step function that assumes integer values k from 1 to K, depending on the value of q. For example, for $K = 2$, the possible values of $\lceil q/(d_i - \varepsilon)\rceil$ with $q \in D_i$ are 1, if $q < d_i$; and 2, if $q = d_i$. In this case, Inequalities (9) are the same as (5). For $K = 3$, $\lceil 2q/(d_i - \varepsilon)\rceil$ results in 1, if $q < d_i/2$; in 2, if $d_i/2 \leq q < d_i$; and in 3, if $q = d_i$. The first two cases correspond to partial deliveries, while the last is a full delivery. Extending this analysis to an arbitrary value of K, we have that for a given delivery quantity $q \in D_i$, $\left\lceil\frac{(K-1)q}{(d_i-\varepsilon)}\right\rceil$ is equal to k, if $\frac{(k-1)d_i}{K-1} \leq q < \frac{kd_i}{K-1}$, for $k = 1, \ldots, K$. Using this observation, we define the binary value g^k_{iq} that assumes the value of 1 if and only if $\frac{(k-1)d_i}{K-1} \leq q < \frac{kd_i}{K-1}$, and use it to rewrite Inequality (9) as follows:

$$\sum_{r\in\mathcal{R}}\sum_{q\in D_i}\sum_{k=1}^{K}b^r_{iq}g^k_{iq}k\theta_r \geq K. \tag{10}$$

We denote as (FK) the formulation with Objective Function (1), Constraints (2) and (3), Constraints (10) for $i \in \mathcal{C}(K)$, and Constraints (6) for $i \in \mathcal{C} \setminus \mathcal{C}(K)$. We define set $\mathcal{R}^K$ of routes in which the delivery quantity of every visit to a customer $i \in \mathcal{C}(K)$ is the minimum nonzero value in $\left[\frac{(k-1)d_i}{K-1}, \frac{kd_i}{K-1}\right)$ that is a multiple of $\bar{q}$, for some $k \in \{1, \ldots, K\}$; and the delivery quantity of every visit to a customer $i \in \mathcal{C} \setminus \mathcal{C}(K)$ is a nonzero multiple of $\bar{q}$. Let $D_i(K)$ be the set of such delivery quantities, defined as follows:

$$D_i(K) = \bigcup_{k=1}^{K}\left\{\min\{l\bar{q}\} \;:\; l\bar{q} \in \left[\frac{(k-1)d_i}{K-1}, \frac{kd_i}{K-1}\right),\; l \in \mathbb{N}\right\} \quad \text{if } i \in \mathcal{C}(K), \tag{11}$$

$$D_i(K) = \{l\bar{q} : \forall l = 1, \ldots, d_i/\bar{q}\} \quad \text{if } i \in \mathcal{C} \setminus \mathcal{C}(K). \tag{12}$$

Then, $\mathcal{R}^K = \left\{r \in \mathcal{R} : b^r_{iq} = 0,\ \forall i \in \mathcal{C},\ \forall q \notin D_i(K)\right\}$. Note that for customers in $\mathcal{C} \setminus \mathcal{C}(K)$, the set $D_i(K)$ includes all nonzero multiples of $\bar{q}$, ranging from $\bar{q}$ to d_i. For this reason, we call it a full discretization of d_i. Conversely, for customers in $\mathcal{C}(K)$, there is at least one multiple of $\bar{q}$ that is not included in $D_i(K)$, and hence we have a partial discretization.

Figure 2 illustrates the set of delivery quantities $D_i(K)$ for different values of K, considering $\bar{q} = 5$ and a customer i with demand $d_i = 40$. Since $d_i/\bar{q} = 8$, we observe partial discretizations for $K = 2$ to 7, and full discretization for $K = 8$. For instance, in the partial discretization with $K = 4$, the possible delivery quantities are 5, 15, 30 and 40, which are the minimum multiples of $\bar{q}$ inside the ranges $[0, 13.33)$, $[13.33, 26.67)$, $[26.67, 40)$ and $[40, 53.33)$, obtained from the expression $\left[\frac{(k-1)d_i}{K-1}, \frac{kd_i}{K-1}\right)$ for each $k = 1, 2, 3$ and 4. As indicated in the figure, each of these intervals has size 13.33 $(= d_i/(K-1))$.

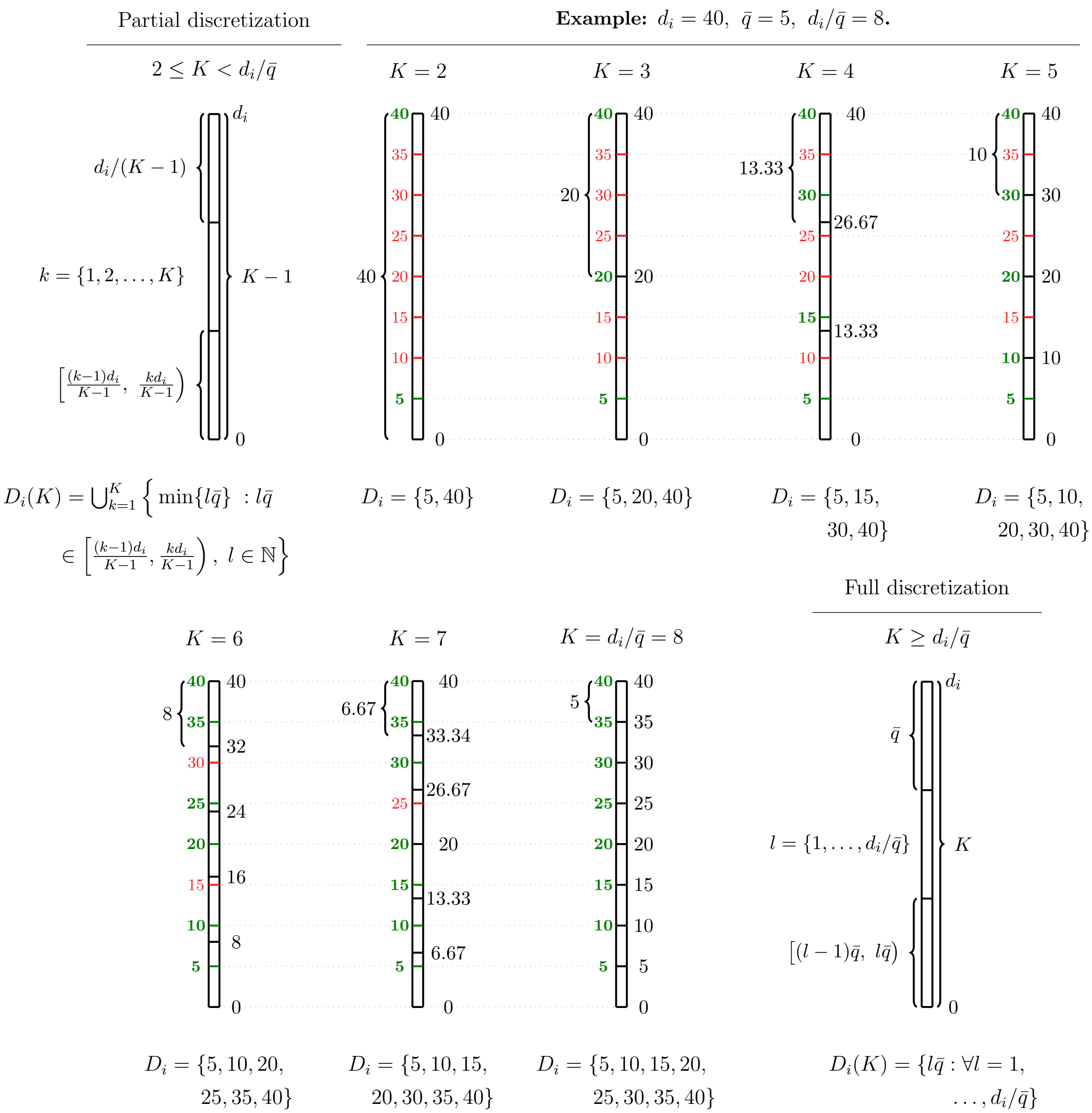

Figure 2: From partial to full discretization – a numerical example.

Observation 4 *There exists an optimal solution* $\bar{\boldsymbol{\theta}}$ *to the linear relaxation of Formulation (FK) such that* $\bar{\theta}_r = 0$ *for all* $r \notin \mathcal{R}^K$.

Let $K_{\max} = \max_{i\in\mathcal{C}} d_i/\bar{q}$. We call Formulation (F$K_{\max}$) the fully discretized formulation since all possible delivery quantities according to Property 6 are considered and $\mathcal{R}^K = \mathcal{R}$. Strong k-path inequalities (2) are redundant for this formulation but still useful as cutting planes. Formulations (FK) with $K > K_{\max}$ are equivalent to (F$K_{\max}$). Formulations (FK) with $K < K_{\max}$ are partially discretized. For such formulations, the exact separation of strong k-path inequalities (2) is necessary to ensure the feasibility of integer solutions.

4.3 Valid inequalities

To further strengthen the proposed formulations, we present well-known valid inequalities that are adapted to the split delivery variants and propose novel families of limited-memory subset-row inequalities and limited-memory strong k-path inequalities.

4.3.1 Rounded capacity inequalities.

Let x^r_{ij} be the number of times route $r \in \mathcal{R}$ traverses arc $(i,j) \in \mathcal{A}$. Constraints

$$\sum_{r\in\mathcal{R}} \sum_{\substack{(i,j)\in\mathcal{A}:\\ |\{i,j\}\cap S|=1}} x^r_{ij}\theta_r \geq 2\left\lceil \sum_{i\in S} d_i/Q \right\rceil, \quad \forall S \subseteq \mathcal{C}, \tag{13}$$

are known in the literature as rounded capacity inequalities (RCIs) or weak k-path inequalities (Desaulniers 2010, Archetti, Bouchard, and Desaulniers 2011). They were introduced by Laporte and Nobert (1983) for the CVRP. RCIs have been separated in virtually all branch-and-cut and branch-cut-and-price algorithms in the literature for the SDVRP and the SDVRPTW due to their importance for obtaining strong lower bounds.

4.3.2 Limited-memory subset-row packing inequalities.

Cuts of the next family are adapted from the subset-row inequalities introduced by Jepsen et al. (2008) for the CVRP. They can be obtained by Chvátal-Gomory rounding of the set packing constraints, stating that every customer can be visited at most once. However, since the split delivery variants do not include this requirement, we modify these constraints to consider only the visits in which the delivery quantity is strictly greater than half of its demand. Hence, for each customer i, the number of such visits must be at most one, i.e.:

$$\sum_{r\in\mathcal{R}} \sum_{\substack{q\in D_i:\\ q>d_i/2}} b^r_{iq}\theta_r \leq 1, \quad \forall i \in \mathcal{C}. \tag{14}$$

By considering subsets $S \subseteq \mathcal{C}$ of size three and applying Chvátal-Gomory rounding of Constraints (14) for customers in S with multiplier $1/2$, we obtain the following subset-row packing inequalities (SRPIs) that are valid for Formulation (FK) with $K \geq 1$:

$$\sum_{r \in \mathcal{R}} \left\lfloor \sum_{i \in S} \sum_{\substack{q \in D_i: \\ q > d_i/2}} \frac{1}{2} b^r_{iq} \right\rfloor \theta_r \leq 1, \quad \forall S \subseteq \mathcal{C},\ |S| = 3. \tag{15}$$

A weaker version of the SRPIs was used for the SDVRPTW by Archetti, Bouchard, and Desaulniers (2011). In their cuts, condition $q > d_i/2$ is restricted to $q = d_i$. Note that any formulation (FK) with $K \geq 1$ still involves only variables θ_r, $r \in \mathcal{R}^K$. Of course, inequalities (15) are just an example of SRPIs that can be derived from (14). In general, one can define a rational multiplier $\mu_i/\eta \in [0, 1), \eta > 0$, for each customer $i \in \mathcal{C}$ and obtain the following inequality by means of Chvátal-Gomory rounding:

$$\sum_{r \in \mathcal{R}} \left\lfloor \sum_{i \in \mathcal{C}} \sum_{\substack{q \in D_i: \\ q > d_i/2}} \frac{\mu_i}{\eta} b^r_{iq} \right\rfloor \theta_r \leq \left\lfloor \sum_{i \in \mathcal{C}} \frac{\mu_i}{\eta} \right\rfloor. \tag{16}$$

The reader is referred to Appendix A for further details on the definition of general SRPIs.

We also use elementarity cuts to enforce Property 2 in any integer solution. These cuts state that every customer can be visited at most once in any route:

$$\sum_{r \in \mathcal{R}} \left\lfloor \sum_{q \in D_i} \frac{1}{2} b^r_{iq} \right\rfloor \theta_r \leq 0, \quad \forall i \in \mathcal{C}. \tag{17}$$

Constraints (17) can be obtained from Inequalities (16) by considering a larger set of values q, and by setting $\eta = 2$, $\mu_i = 1$, $\mu_{i'} = 0$ for all $i' \in \mathcal{C} \setminus \{i\}$. Thus, these constraints are also included in the family of subset-row packing inequalities.

It is known in the literature that a large number of active subset-row cuts can render the dynamic generation of route variables very expensive (Jepsen et al. 2008, Pecin et al. 2017a). Thus, we adopt the following limited memory technique introduced by Pecin et al. (2017a). Each limited-memory SRPI (lm-SRPI) $\sum_{r \in \mathcal{R}} \alpha(r, \boldsymbol{\mu}, \eta, M)\theta_r \leq \left\lfloor \sum_{i \in \mathcal{C}} \frac{\mu_i}{\eta} \right\rfloor$ is associated with an arc memory $M \subseteq \mathcal{A}$. Algorithm 1 shows the computation of function $\alpha(r, \boldsymbol{\mu}, \eta, M)$. From its definition, it follows that

$$\alpha(r, \boldsymbol{\mu}, \eta, M) \leq \left\lfloor \sum_{i \in \mathcal{C}} \sum_{\substack{q \in D_i: \\ q > d_i/2}} \frac{\mu_i}{\eta} b^r_{iq} \right\rfloor \tag{18}$$

for any $M \subseteq \mathcal{A}$. Therefore, lm-SRPIs are valid for the problem.

4.3.3 Limited-memory subset-row covering inequalities.

Novel subset-row covering inequalities (SRCIs) can be obtained by Chvátal-Gomory rounding of constraints stating that, for every customer, there should be at least one nonzero delivery. Again,

Algorithm 1: Function $\alpha(r,\, \boldsymbol{\mu},\, \eta,\, M)$

$\alpha \leftarrow 0$, $\phi \leftarrow 0$
for $k = 1$ **to** $n_r - 1$ **do**
  if $a^r_k = (i,j) \notin M$ **then** $\phi \leftarrow 0$
  if $d^r_k > d_j/2$ **then**
    $\phi \leftarrow \phi + \mu_j$
    if $\phi \geq \eta$ **then** $\alpha \leftarrow \alpha + 1$, $\phi \leftarrow \phi - \eta$
return α

let $\mu_i/\eta \in [0,1), \eta > 0$, be a rational multiplier defined for customer $i \in \mathcal{C}$. Then, the cut is as follows:

$$\sum_{r \in \mathcal{R}} \left\lfloor \sum_{i \in \mathcal{C}} \sum_{\substack{q \in D_i: \\ q>0}} \frac{\mu_i}{\eta} b^r_{iq} \right\rfloor \theta_r \geq \left\lceil \sum_{i \in \mathcal{C}} \frac{\mu_i}{\eta} \right\rceil \tag{19}$$

We developed a computational approach, inspired by the work of Pecin et al. (2017b), to determine which multipliers to consider (see Appendix A) and, again, we rely on the limited-memory technique to reduce the impact of SRCIs on solution time. Each limited-memory SRCI (lm-SRCI) $\sum_{r \in \mathcal{R}} \beta(r, \boldsymbol{\mu}, \eta, M)\theta_r \geq \left\lceil \sum_{i \in \mathcal{C}} \frac{\mu_i}{\eta} \right\rceil$ is associated with an arc memory $M \subseteq \mathcal{A}$. Algorithm 2 shows the computation of function $\beta(r, \boldsymbol{\mu}, \eta, M)$. From its definition, it follows that

$$\beta(r, \boldsymbol{\mu}, \eta, M) \geq \left\lfloor \sum_{i \in \mathcal{C}} \sum_{\substack{q \in D_i: \\ q>0}} \frac{\mu_i}{\eta} b^r_{iq} \right\rfloor \tag{20}$$

for any $M \subseteq \mathcal{A}$. Therefore, lm-SRCIs are valid for the problem.

Algorithm 2: Function $\beta(r,\, \boldsymbol{\mu},\, \eta,\, M)$

$\beta \leftarrow 0$, $\psi \leftarrow 0$
for $k = 1$ **to** $n_r - 1$ **do**
  if $a^r_k = (i,j) \notin M$ **then** $\psi \leftarrow 0$
  if $d^r_k > 0$ **then**
    $\psi \leftarrow \psi - \mu_j$
    if $\psi < 0$ **then** $\beta \leftarrow \beta + 1$, $\psi \leftarrow \psi + \eta$
return β

4.3.4 Limited-memory strong k-path inequalities.

Finally, we adapt the limited-memory technique to inequalities (2). Each limited-memory strong k-path inequality (lm-SKPI) $\sum_{r \in \mathcal{R}} \gamma(r, S, M)\theta_r \geq \left\lceil \sum_{i \in S} d_i/Q \right\rceil$ is associated with a set S of customers and an arc memory $M \subseteq \mathcal{A} \setminus \{(i,j)\}_{\{i,j\} \subseteq S}$. The memory can only include arcs with at least one node outside set S. Algorithm 3 presents the pseudocode to compute $\gamma(r, S, M)$. Based on its definition, we have that $\gamma(r, S, M) \geq h_{rS}$ for any $M \subseteq \mathcal{A}$. Thus, lm-SKPIs are valid for the problem. In Algorithm 3, the value σ is used to "remember" visits to set S. If $\sigma = 0$, set S has not yet been visited or a visit has been already "forgotten" as the memory has been left. If $\sigma = 1/2$, set S has been visited and is still "remembered". These specific values, 0 and 1/2, have been chosen as they are used later in (24).

Algorithm 3: Function $\gamma(r,\, S,\, M)$

```
γ ← 0, σ ← 0
for k = 1 to n_r − 1 do
    if v^r_{k−1} ∈ S and v^r_k ∉ S then σ ← 1/2
    if a^r_k ∉ M then σ ← 0
    if v^r_{k−1} ∉ S and v^r_k ∈ S then
        σ ← σ − 1/2
        if σ = −1/2 then γ ← γ + 1, σ ← 0
return γ
```

5 Branch-cut-and-price algorithm

In this section, we describe the branch-cut-and-price (BCP) algorithm to solve Formulation (FK) for a fixed $K \in \{0, 1, 2, \ldots, K_{\max}\}$, together with the valid inequalities presented in Section 4.3. The linear relaxation of (FK), or the master problem, is solved by the column generation procedure. On every iteration of this procedure, the restricted master problem (RMP) is solved considering a restricted subset of variables $\boldsymbol{\theta}$, others being fixed to zero. Let $(\bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\rho}}, \bar{\boldsymbol{\zeta}}, \bar{\boldsymbol{\xi}}, \bar{\boldsymbol{\tau}})$ be an optimal dual solution of the RMP. The dual value $\bar{\pi}_i \geq 0$ corresponds to Constraint (9) if $i \in \mathcal{C}(K)$ and to Constraint (6) if $i \in \mathcal{C} \setminus \mathcal{C}(K)$. If $K \leq 1$ then Constraints (6) and (9) are not defined, and we assume $\bar{\pi}_i = 0$ for all $i \in \mathcal{C}$. Let O be the set of active RCIs, and let $S^o \subseteq \mathcal{C}$ define rounded capacity inequality $o \in O$ with dual value $\bar{\rho}_o > 0$. Let $\delta(S^o)$ also be the set of arcs in $\mathcal{A}$ which have exactly one node in S^o. Let P be the set of active lm-SRPIs, and tuple $(\boldsymbol{\mu}^p, \eta^p, M^p)$ defines limited-memory subset-row packing inequality $p \in P$ with dual value $\bar{\zeta}_p < 0$. Let U be the set of active lm-SRCIs, and tuple $(\boldsymbol{\mu}^u, \eta^u, M^u)$ defines limited-memory subset-row covering inequality $u \in U$ with dual value $\bar{\xi}_u > 0$. Let W be the set of active lm-SKPIs, and pair (S^w, M^w) defines limited-memory strong k-path inequality $w \in W$ with dual value $\bar{\tau}_w > 0$.

5.1 Pricing problem

To determine whether the current solution to the RMP is optimal for the master problem, we must find the minimum reduced cost among all variables θ_r, $r \in \mathcal{R}^K$. We represent set $\mathcal{R}^K$ of routes as resource-constrained paths in multi-graph $\mathcal{G}'(K) = (\mathcal{V}, \mathcal{A}'(K))$. Every arc (i, j) in the original graph $\mathcal{G}$ is replaced by multiple arcs (i, j, q), $q \in D_j(K)$, between nodes i and j in $\mathcal{G}'(K)$. Sets $D_j(K)$ are defined in (11) and (12) for $K \geq 2$. We set $D_j(0) = \{0\}$ and $D_j(1) = \{\bar{q}\}$ for all $j \in \mathcal{C}$, and $D_{n+1}(K) = \{0\}$ for all $K \geq 0$. We define disposable time and capacity resources with accumulated resource consumption bounds $[e_i, l_i]$ and $[0, Q]$, respectively, for every node $i \in \mathcal{V}$. The time resource consumption of every arc $(i, j, q) \in \mathcal{A}'$ equals t_{ij}, and the capacity resource consumption of this arc equals q. Thus, the set of resource-feasible paths in multigraph $\mathcal{G}'(K)$ corresponds to the set of routes in $\mathcal{R}^K$. The capacity resource is redundant for $K = 0$ and may be skipped.

The reduced cost $\bar{c}_{(i,j,q)}$ of every arc $(i, j, q) \in \mathcal{A}'(K)$ equals

$$\bar{c}_{(i,j,q)} = c_{ij} - \sum_{\substack{o \in O: \\ (i,j) \in \delta(S^o)}} \bar{\rho}_o - \begin{cases} 0, & \text{if } K < 2, \\ \bar{k}(j, q, K) \cdot \bar{\pi}_j, & \text{if } K \geq 2 \text{ and } j \in \mathcal{C}(K), \\ q\bar{\pi}_j, & \text{if } K \geq 2 \text{ and } j \in \mathcal{C} \setminus \mathcal{C}(K), \end{cases} \tag{21}$$

where $\bar{k}(j, q, K)$ is equal to the value k that satisfies $\frac{(k-1)d_j}{K-1} \leq q < \frac{kd_j}{K-1}$.

Let $\mathcal{A}'_r(K)$ be the set of arcs traversed by the resource-constrained path in $\mathcal{G}'(K)$, corresponding to a route $r \in \mathcal{R}^K$. Then, the reduced cost $\bar{c}^r$ of the resource-constrained path r and its related variable θ_r is equal to

$$\bar{c}^r = \sum_{(i,j,q) \in \mathcal{A}'_r(K)} \bar{c}_{(i,j,q)} - \sum_{p \in P} \alpha(r, \boldsymbol{\mu}^p, \eta^p, M^p)\bar{\zeta}_p - \sum_{u \in U} \beta(r, \boldsymbol{\mu}^u, \eta^u, M^u)\bar{\xi}_u - \sum_{w \in W} \gamma(r, S^w, M^w)\bar{\tau}_w. \tag{22}$$

To find the resource-constrained path r in $\mathcal{G}'(K)$ corresponding to the best reduced cost, we use the bucket-graph based bidirectional labeling algorithm proposed by Sadykov, Uchoa, and Pessoa (2021). Every label L represents a partial path $\mathcal{G}'(K)$, which is either forward (starting from node 0) or backward (starting from node $n + 1$). Every label L is characterized by a vector $(\bar{c}^L, v^L, t^L, q^L, \boldsymbol{\phi}^L, \boldsymbol{\psi}^L, \boldsymbol{\sigma}^L)$, where $\bar{c}^L$ is the reduced cost of the partial path, v^L is the terminating node, and t^L and q^L are the accumulated time and capacity resource consumption, respectively. Finally, $\boldsymbol{\phi}^L$, $\boldsymbol{\psi}^L$, and $\boldsymbol{\sigma}^L$ are vectors of states corresponding to active limited-memory cuts. The lengths of these vectors are $|P|$, $|U|$, and $|W|$. These states are computed in the same way as in Functions α, β, and γ presented in Section 4.3. To adapt the bucket-graph labeling algorithm to our problem, we must define label initialization, extension, domination, and concatenation functions.

The initial forward label is defined as $(0, 0, e_0, 0, \mathbf{0}, \mathbf{0}, \mathbf{0})$, and the initial backward label is defined as $(0, 0, l_{n+1}, Q, \mathbf{0}, \mathbf{0}, \mathbf{0})$. In the backward labeling algorithm, the direction of arcs is reversed. The function that extends a label L' in the forward or the backward direction along an arc $a = (i, j, q)$, such that $i = v^{L'}$ to obtain label L, is presented in Algorithm 4. It returns true if extension is feasible. For ease of presentation, we define $\mu^p_0 = \mu^p_{n+1} = 0$ for every $p \in P$ and $\mu^u_0 = \mu^u_{n+1} = 0$ for every $u \in U$.

A label L dominates label L' if $v^L = v^{L'}$, $q^L \leq q^{L'}$, $t^L \leq t^{L'}$ ($q^L \geq q^{L'}$ and $t^L \geq t^{L'}$ for backward labels), and

$$\bar{c}^L - \sum_{\substack{p \in P: \\ \phi^L_p > \phi^{L'}_p}} \bar{\zeta}_p + \sum_{\substack{u \in U: \\ \psi^L_u > \psi^{L'}_u}} \bar{\xi}_u + \sum_{\substack{w \in W: \\ \sigma^L_w > \sigma^{L'}_w}} \bar{\tau}_w \leq \bar{c}^{L'}. \tag{23}$$

For a set of labels $\mathcal{L}$ and a label L', the bucket-graph labeling algorithm uses inequality $\bar{c}^{L'} < \min_{L \in \mathcal{L}} \bar{c}^L$ as a sufficient condition for nondomination of label L' by any label in $\mathcal{L}$. This sufficient condition remains valid in our case since $\bar{\zeta}_p \leq 0$ for all $p \in P$, $\bar{\xi}_u \geq 0$ for all $u \in U$, and $\bar{\tau}_w \geq 0$ for all $w \in W$.

The partial paths represented by a forward label $\vec{L}$ and a backward label $\overleftarrow{L}$ can be concatenated along an arc $(i, j, q) \in \mathcal{A}'$ if $i = v^{\vec{L}}$, $j = v^{\overleftarrow{L}}$, $q^{\vec{L}} + q \leq q^{\overleftarrow{L}}$, $t^{\vec{L}} + t_{ij} \leq t^{\overleftarrow{L}}$. The reduced cost $\bar{c}(\vec{L}, \overleftarrow{L}, i, j, q)$

Algorithm 4: Extension of label L' along arc $a = (i, j, q) \in \mathcal{A}'$ to obtain label L

```
c̄^L ← c̄^{L'} + c̄_a, v^L ← j, φ^L ← φ^{L'}, ψ^L ← ψ^{L'}, σ^L ← σ^{L'}
if forward direction then
    q^L ← q^{L'} + q, t^L ← max{t^L + t_ij, e_j}
    if q^L > Q or t^L > l_j then return false
if L is a backward label then
    q^L ← q^{L'} − q, t^L ← min{t^L − t_ji, l_j}
    if q^L < 0 or t^L < e_j then return false
for p ∈ P do
    if (i, j) ∉ M^p then φ_p^L ← 0
    if q > d_j/2 then
        φ_p^L ← φ_p^L + μ_j^p
        if φ_p^L ≥ η^p then φ_p^L ← φ_p^L − η^p, c̄^L ← c̄^L − ζ̄_p
for u ∈ U do
    if (i, j) ∉ M^u then ψ_u^L ← 0
    if q > 0 then
        ψ_u^L ← ψ_u^L − μ_j^u
        if ψ_u^L < 0 then ψ_u^L ← ψ_u^L + η^u, c̄^L ← c̄^L − ξ̄_u
for w ∈ W do
    if i ∈ S^w and j ∉ S^w then σ_w^L ← 1/2
    if (i, j) ∉ M^w then σ_w^L ← 0
    if i ∉ S^w and j ∈ S^w then
        σ_w^L ← σ_w^L − 1/2
        if σ_w^L = −1/2 then σ_u^L ← 0, c̄^L ← c̄^L − τ̄_w
return true
```

of the path obtained by such concatenation can be computed as

$$\bar{c}(\vec{L}, \overleftarrow{L}, i, j, q) = \bar{c}^{\vec{L}} + \bar{c}_{(i,j,q)} + \bar{c}^{\overleftarrow{L}} - \sum_{\substack{p \in P: \\ (i,j) \in M^p, \\ \phi_p^{\vec{L}} + \phi_p^{\overleftarrow{L}} \geq \eta^p}} \bar{\zeta}_p + \sum_{\substack{u \in U: \\ (i,j) \in M^u, \\ \psi_u^{\vec{L}} + \psi_u^{\overleftarrow{L}} \geq \eta^u}} \bar{\xi}_u + \sum_{\substack{w \in W: \\ (i,j) \in M^w, \\ \sigma_p^{\vec{L}} + \sigma_p^{\overleftarrow{L}} = 1}} \bar{\tau}_w + \sum_{\substack{w \in W: \\ \{i,j\} \subseteq S^w}} \bar{\tau}_w. \tag{24}$$

The bucket-graph labeling algorithm uses a lower bound on the reduced cost of any path obtained by concatenation along an arc $a = (i, j, q)$ of a forward label $\vec{L}$ and any backward label in a given set $\overleftarrow{\mathcal{L}}$. We use the following lower bound in our case:

$$\min_{\overleftarrow{L} \in \overleftarrow{\mathcal{L}}} \bar{c}(\vec{L}, \overleftarrow{L}, i, j, q) = \bar{c}^{\vec{L}} + \bar{c}_{(i,j,q)} + \min_{\overleftarrow{L} \in \overleftarrow{\mathcal{L}}} \bar{c}^{\overleftarrow{L}} + \sum_{\substack{w \in W: \\ \{i,j\} \subseteq S^w}} \bar{\tau}_w. \tag{25}$$

Bound (25) is valid since the part related to limited-memory cuts in (24) is nonnegative.

5.2 Cut separation

To separate the RCIs, we use four algorithms presented in Lysgaard, Letchford, and Eglese (2004): i) the connected components heuristic; ii) the max-flow-based algorithm, which separates fractional capacity inequalities; iii) the greedy construction heuristic; and iv) the heuristic which inspects the pool of previously generated inequalities and performs a fast local search for each of them.

Our approach for separating the strong k-path inequalities is as follows. First, we use a greedy construction heuristic similar to that for the separation of RCIs. Then, the separation of fractional strong k-path inequalities is performed by finding the minimum cut in the flow graph, similar to that described in Section 3. This graph is based on the current fractional solution $\bar{\boldsymbol{\theta}}$ and the set of routes $\bar{\mathcal{R}} = \{r \in \mathcal{R}^K : \bar{\theta}_r > 0\}$. The capacity of each arc $(0, r)$ connecting the source to route node r is set to $Q\bar{\theta}_r$. After finding the minimum cut in the flow graph, the candidate set $S \subseteq \mathcal{C}$ of customers is constructed according to the proof of Theorem 4.1. As shown by the theorem, this separation algorithm is exact for integer solutions $\bar{\boldsymbol{\theta}}$. Finally, we use a variant of the connected components heuristic called the route-based algorithm proposed by Archetti, Bouchard, and Desaulniers (2011).

We separate only 3-row subset-row packing and covering cuts. Preliminary experiments have indicated that cuts with more rows do not significantly improve the quality of the linear relaxation on average, but they do have a noticeable impact on the computation time. The 3-row cuts are separated by the enumeration of all triples of customers. Elementarity cuts (17) are also separated by enumeration.

The limited memory for strong k-path inequalities and subset-row cuts is obtained in the same way as proposed by Pecin et al. (2017b). For each violated cut, a minimal memory is generated such that the coefficients of the route variables $\bar{\theta}_r > 0$ in the limited-memory cut are equal to coefficients of these variables in the full-memory cut.

5.3 Other algorithmic components

Our BCP algorithm incorporates the following algorithmic enhancements.

- In the labeling algorithm, labels are stored in buckets according to their consumption of the capacity resource (SDVRP) or the time resource (SDVRPTW). To reduce the number of dominance checks, we employ the bucket-graph based labeling algorithm (Sadykov, Uchoa, and Pessoa 2021). Before using the exact labeling algorithm, we utilize the labeling heuristic, which retains only the best label (according to reduced cost) in each bucket.

- As previously mentioned, elementarity constraints are not enforced in the labeling algorithm. Instead, we impose partial elementarity by using the ng-path relaxation introduced by Baldacci, Mingozzi, and Roberti (2011). For a given customer $i \in \mathcal{C}$, its ng-neighborhood consists of the eight closest customers, including i itself.

- To improve the convergence of column generation, we use the automatic dual price smoothing stabilization technique. The dual solution provided to the pricing problem is a convex combination of the optimal dual solution of the current RMP and the dual solution that has yielded the best Lagrangian bound thus far. The convex combination multiplier is dynamically adjusted using the approach proposed by Pessoa et al. (2018).

- Following each convergence of column generation, the bucket graph used in the labeling algorithm is filtered based on a reduced cost argument. A bucket arc, defined by a bucket and an outgoing arc, is eliminated if it can be proven that there does not exist an improving solution that includes any route passing through this bucket arc.

- After bucket arc elimination, we run the elementary route enumeration procedure. This procedure, initially developed by Baldacci, Christofides, and Mingozzi (2008) for classic VRPs, involves enumerating all elementary routes with reduced cost smaller than the current primal-dual gap, i.e. all routes which may participate in an improving solution. After successful enumeration, the pricing problem is solved by inspection instead of the labeling algorithm. If the number of enumerated routes is less than three thousand, the node is finished by solving the master problem as a MIP.

Additional details regarding these enhancements can be found in Sadykov, Uchoa, and Pessoa (2021). The labeling procedure is a dynamic-programming-based algorithm that keeps the best partial route for each customer visit pattern (and each endpoint) as a state. Since the number of possible patterns can grow exponentially with n, it uses a reduced cost argument to prune a large number of states. In split delivery variants, the maintained states depend not only on the visited customers but also on the corresponding delivery quantities. This is because partial routes that visit the same set of customers but with different delivery quantities cannot dominate each other. While at most 2^n visit patterns are possible in classic VRPs, this number increases to $(K+1)^n$ in SDVRPs when using the proposed formulations. Therefore, the route enumeration procedure becomes impractical for $K > 2$, and it is only used for formulations (F0), (F1), and (F2).

Finally, we employ the ILS-based matheuristic proposed by Alvarez and Munari (2022) to generate initial upper bounds for the optimal values of the instances. Before launching our BCP algorithm, we run the matheuristic with a time limit set to $t = 8\lceil 4^{\log_2(n/16)} \rceil$ seconds. For example, the time limits are 24, 80, and 320 seconds for instances with 25, 50, and 100 customers, respectively. The value of the best solution (plus a small epsilon) is then used as the initial upper bound in the BCP algorithm. A high-quality upper bound is extremely useful for bucket arc elimination and elementary route enumeration.

5.4 Branching

If no violated cutting planes are found for the current fractional solution, or the tailing off condition is met, we proceed to branching. The tailing off condition is satisfied under two conditions: 1) when the primal-dual gap decreases by less than 1.5% in three rounds of cut separation, which do not have to be consecutive; or 2) when the average exact pricing time exceeds 10 seconds.

We use two branching strategies, described as follows. Let $\bar{x}_{ij} = \sum_{r \in \mathcal{R}^K} x_{ij}^r \bar{\theta}_r$. First, we branch on expressions $\bar{x}_{ij} + \bar{x}_{ji}$ for all $\{i,j\} \subset \mathcal{C}$ (i.e., edges between customers), $\bar{x}_{0,i} + \bar{x}_{i,n+1}$ for all $i \in \mathcal{C}$ (i.e., edges between the depot and customers), and $\sum_{i \in \mathcal{C}} \frac{1}{2}(\bar{x}_{0,i} + \bar{x}_{i,n+1})$ (i.e., the number of used vehicles). The best branching expression is determined using the multiphase strong branching scheme described by Sadykov, Uchoa, and Pessoa (2021).

If none of the above expressions is fractional for the current solution $\bar{\boldsymbol{\theta}}$, we perform the following Ryan&Foster (Ryan and Foster 1981, Desrochers and Soumis 1989) branching. We find a pair $\{i,j\} \subset \mathcal{C}$ such that $\sum_{r \in \mathcal{R}^K :\, h_{ri} + h_{rj} = 2} \bar{\theta}_r$ is fractional and impose the constraint that customers i and j should be on the same route in one branch and on different routes in another branch. These constraints are imposed in the pricing problem by introducing additional binary resources. Desaulniers (2010) showed that the combination of branching on edges and Ryan&Foster branching is sufficient to fulfill the integrality requirements for route variables $\boldsymbol{\theta}$. Again, the best Ryan&Foster branching pair is determined using the multiphase strong branching.

To enhance the quality of branching candidates, we employ the multiphase strong branching procedure. In phase zero, we select a maximum of 100 candidates from the most fractional pairs $\{i, j\}$ and from the branching history. Then, in phase one, for each candidate and each branch, we only resolve the restricted master problem without conducting any column generation. Up to five best candidates (according to the product rule) are chosen for phase two, where only heuristic column generation is performed. Finally, the best candidate is selected, also considering the product rule. Further information on the multiphase strong branching can be found in Pecin et al. (2017b), Sadykov, Uchoa, and Pessoa (2021).

6 Computational results

In this section, we numerically investigate the strength of the proposed family of formulations and verify the performance of our BPC algorithm using benchmark instances with and without time windows. The algorithm is coded in the C++ programming language on top of the generic BCP library BaPCod (Sadykov and Vanderbeck 2021) with its VRPSolver extension (Pessoa et al. 2020), containing the implementation of the labeling algorithm for the SPPRC. We use IBM CPLEX version 20.1, as the general-purpose LP and MIP solver. We performed the numerical evaluation of our BCP algorithm on a server with nodes having the 2.6GHz Cascade Lake Intel Xeon Skylake Gold 6240 processor with 36 cores and 196 GB of RAM. Up to 36 instances were run in parallel on each node, each run using a single core.

6.1 Instances

We benchmark our algorithm on standard literature test instances of the SDVRPTW and SDVRP. For the SDVRPTW, we use the same instances that were previously tested in Desaulniers (2010), Archetti, Bouchard, and Desaulniers (2011), Bianchessi, Drexl, and Irnich (2019), Munari and Savelsbergh (2022). These instances are derived from the classic Solomon's VRPTW instances with 50 and 100 customers (56 instances of each size). To create the SDVRPTW instances, we modify the original instances to allow for split deliveries and impose a vehicle capacity constraint of 30, 50, and 100 units. Therefore, there are 168 instances for each size.

We also use modified SDVRPTW instances. First, we generate instances with 75 customers by removing the last 25 customers from instances with 100 customers, resulting in 168 instances. Secondly, we create new instances by modifying instances with $\bar{q} = 10$ (these are all instances in class C, and instances in class RC with 50 customers). For this, we add a random integer value in the range of $[-3, 3]$ to the demand of each customer $i \in \mathcal{C}$ such that $d_i \in [1.2 d_{\min}, 0.8 d_{\max}]$. Then we verify whether $\bar{q} = 1$ and that the value $\lceil \sum_{i \in \mathcal{C}} d_i / 30 \rceil$ remains the same as in the original instance. We regenerated each instance until these two conditions are verified. We denote these new instances as 50P, 75P, and 100P, depending on the instance size. There are 201 instances of this kind. Thus, the total number of instances is 705.

Following a similar approach as presented in Bianchessi and Irnich (2019), we preprocess the instances using the following steps. Firstly, we round the travel times to one decimal place. Next, we replace the travel time of each arc with the shortest path between its tail and head nodes. Lastly, we add the service time of the tail node to the travel time of each arc.

For the SDVRP, we use the same four classes of instances tested in Archetti, Bianchessi, and Speranza (2014), Ozbaygin, Karasan, and Yaman (2018), Gouveia, Leitner, and Ruthmair (2023), Munari and Savelsbergh (2022). First class SD contains 21 instances, having from 8 to 288 customers. Second class S contains 14 instances, having from 50 to 100 customers. The third class p contains 42 instances, having from 50 to 199 customers. Finally, fourth class eil contains 11 instances having from 21 to 100 customers. Each instance has four variations depending on whether distances (and travel costs) are rounded and whether the number of vehicles is fixed to the minimum possible number. Instances with unlimited and limited fleet are referred to as UF and LF, respectively, if distances are considered to be Euclidean distances. Similarly, we use UF-r and LF-r to refer to instances with rounded Euclidean distances.

6.2 Strength of formulations (FK) for different values K

First, we numerically investigate the dependency of the quality of the lower bound and the solution time of the master problem on value K. As a reminder, the master problem is the linear relaxation of Formulation (FK) enriched by cutting planes. The master problem is solved by applying the column and cut generation procedure. It terminates either when no violated cutting planes are found or when the tailing-off criterion is satisfied. In this experiment, we set the initial upper bound equal to the optimal value of the instance. We do so to exclude the random impact of improving the primal solution during the column and cut generation procedure.

This investigation is performed using the SDVRPTW test instances from the literature with 50 customers. All optimal solutions for these instances were obtained by us during preliminary experiments. The instances are categorized into two groups based on the value of $K_{\max}$. The first group contains instances in classes C and RC, for which $K_{\max} = 4$. We tested Formulations (F0), (F1), (F2), and (F4) for these instances. The second group consists of instances in class R for which $K_{\max} = 36$. We tested Formulations (F0), (F1), (F2), (F4), (F8), (F16), and (F36) for these instances.

In Figure 3, we show the plots for different instance groups and different values of the vehicle capacity. In these plots, *Gap* is the average difference between the optimal value and the lower bound of the master problem, expressed as a percentage of the former (right scale), and *Time* is the average solution time in seconds of the column and cut generation procedure (left scale).

As expected, the strength of the master problem increases (the optimality gap decreases) with increasing value K. However, this decrease in the gap is steeper for instances with a smaller vehicle capacity ($Q = 30$ and $Q = 50$) and less steep for instances with longer routes ($Q = 100$). The optimality gap is almost nullified for $K = K_{\max}$ for instances with small $K_{\max}$. For instances in class R, the gap remains significant even for $K = K_{\max}$.

The decrease of the solution time with the increase of value K for instances in the first group is less intuitive. This outcome occurs because the linear relaxation of (F$K_{\max}$) becomes stronger with the increase in value K and thus a smaller number of cut generation rounds is necessary to attain convergence. In addition, the size of graph $\mathcal{G}'(K)$ remains reasonable since $K_{\max}$ is still small. Thus, Formulation (F$K_{\max}$) is a clear choice for instances in the first group.

The selection of the most suitable formulation for instances in the second group is not as obvious. It is evident that Formulations (F0) and (F1) are dominated by (F2) in terms of both the optimality gap and solution time. However, when K increases beyond the value of 2, the solution time increases due to a larger size of graph $\mathcal{G}'(K)$, while the optimality gap decreases. A slight increase in the gap between (F16) and (F36) for $Q = 100$ is attributed to the termination criterion at the root, which depends on the time taken by the route generation subproblem.

There is a clear trade-off between the root primal-dual gap and the root solution time. However, the slope gradient varies depending on the vehicle capacities, i.e., for instances with different average

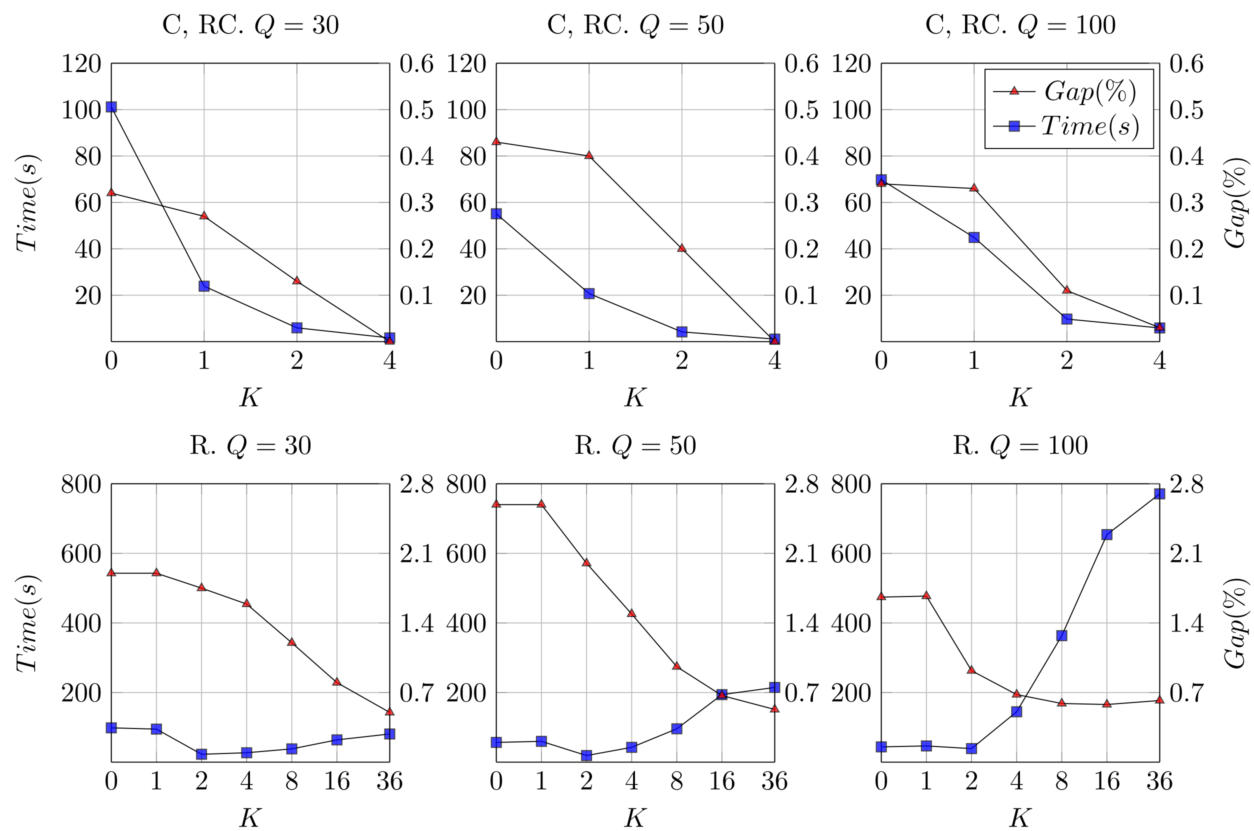

Figure 3: Dependency of the optimality gap and the solution time of master problem on value K

route lengths. When routes are short on average, the decrease in the optimality gap is steep as the value of K increases, while the increase in solution time is not as significant. The opposite outcome occurs for instances with relatively long routes ($Q = 100$): the gain in terms of the optimality gap for large values of K is small compared to $K = 2$, whereas the loss in terms of the solution time is very large and reaches one order of magnitude for large values of K. This observation may indicate that Formulation (F$K_{\max}$) should be chosen for instances with smaller vehicle capacity, while Formulation (F2) is more suitable for instances with larger vehicle capacity. This intuition is confirmed by the next round of experiments, in which we solve the benchmark and novel SDVRPTW instances using our BCP algorithm on Formulations (F2) and (F$K_{\max}$).

6.3 BCP performance for the SDVRPTW instances

We first test the impact of new families of cutting planes on the BCP performance for the SD-VRPTW instances. In all tested BCP variants, we use rounded capacity cuts and limited-memory subset-row cuts with three rows, as these cuts are shown to be beneficial in previous studies. Thus, we compare the following variants of the BCP algorithm.

- BCPmax$_{\text{all}}$ — BCP over formulation (F$K_{\max}$) with all cutting planes.
- BCPmax$_{\text{all}-\text{SKPI}}$ — BCP over formulation (F$K_{\max}$) with all cutting planes except limited-memory strong k-path inequalities.

- $\text{BCPmax}_{\text{all}-\text{SRCI}}$ — BCP over formulation (F$K_{\max}$) with all cutting planes except limited-memory subset-row covering inequalities.

- $\text{BCPmax}_{\text{all}-\text{SKPI}-\text{SRCI}}$ — BCP over formulation (F$K_{\max}$) with all cutting planes except limited-memory subset-row covering inequalities and strong k-path inequalities.

- BCP2_{all} — BCP over formulation (F2) with all cutting planes.

- $\text{BCP2}_{\text{all}-\text{SRCI}}$ — BCP over formulation (F2) with all cutting planes except limited-memory subset-row covering inequalities.

We do not consider variants of BCP2 without strong k-path inequalities as these inequalities are necessary for the validity of formulation (F2).

We would like to note that variant $\text{BCPmax}_{\text{all}-\text{SKPI}-\text{SRCI}}$ can be obtained equivalently by defining an appropriate VRPSolver model (Pessoa et al. 2020). In this model, two vertices are defined for each customer: one representing deliveries exceeding half the demand and another representing smaller deliveries. For each customer, its packing set includes only the first vertex, but its elementarity set includes both vertices. Multiple arcs are defined between pairs of vertices, with one arc for each possible delivery quantity, corresponding to the head vertex of the arc. Ryan&Foster branching is defined on elementarity sets, and not on packing sets as usual. Definition of resources is the same as for the CVRP and VRPTW. Enumeration should be turned off as the sufficient condition defined in Pessoa et al. (2020) is not satisfied.

Table 1 presents the results obtained from different BCP variants, aggregated across all SDVRPTW instances. We use the standard time limit of one hour for the combined matheuristic and BCP execution. The columns show the BCP variant used, the total number of instances solved to optimality, the geometric mean BCP time (which excludes the matheuristic time), the arithmetic mean BCP time, the average primal-dual gap in the root node, and the average final primal-dual gap. The geometric mean times are included as, compared with arithmetic means, they are less affected by extreme values, which in this case correspond to large computation times for the open instances.

Table 1: Aggregated results for different BCP variants for all 705 SDVRPTW instances (one-hour time limit)

BCP variant	Opt	Geom. time(s)	Time(s)	Root gap(%)	Final gap(%)
$\text{BCPmax}_{\text{all}-\text{SKPI}-\text{SRCI}}$	361	360.40	1837.16	0.87	0.72
$\text{BCPmax}_{\text{all}-\text{SKPI}}$	358	362.56	1865.11	0.86	0.72
$\text{BCPmax}_{\text{all}-\text{SRCI}}$	368	354.53	1820.09	0.84	0.70
$\text{BCPmax}_{\text{all}}$	363	349.39	1828.03	0.82	0.69
$\text{BCP2}_{\text{all}-\text{SRCI}}$	273	734.92	2243.48	2.42	2.00
BCP2_{all}	273	730.12	2238.13	2.39	1.93

We can see from Table 1 that both new families of cuts contribute to reducing the average root and final gaps. Moreover, the inclusion of strong k-path inequalities leads to an increased number of instances solved to optimality. However, subset-row covering inequalities do not allow us to solve more instances to optimality. Thus, we do not use these cuts in the subsequent experiments.

Concerning formulations (F2) and (F$K_{\max}$), it is evident that the latter is more efficient. However, it does not dominate the former as there are instances, and even instance classes, that are better solved by (F2). Thus, Table 2 presents a detailed comparison between the variants $\text{BCPmax}_{\text{all}-\text{SRCI}}$ and $\text{BCP2}_{\text{all}-\text{SRCI}}$. The results in this table are aggregated by instance class, instance size, and vehicle capacity. The columns show the instance class, the number of customers (n), the vehicle capacity (Q), and the number of instances in the group (#). The instances in groups "P" are the modified ones. Then, for both BCP variants, the table displays the average number of nodes in the BCP tree, the average final optimality gap, the arithmetic mean pure BCP time in seconds, and the number of instances solved to optimality. The last two columns indicate (for the literature instances) the number of instances for which the best literature lower bounds are improved, and the number of instances solved to optimality for the first time. The last two lines in Table 2 aggregate statistics for all instances (Total) and for a selected subset of instances (F$K_{\max}$-Hard). These selected instances have $Q = 100$ and $K_{\max} > 10$, specifically instances 50P, 75P and 100P in class C, instances 50P, 75 and 100 in class RC, and all instances in class R. They are particularly hard for the fully discretized formulation (F$K_{\max}$), as the size of multi-graph $\mathcal{G}'(K_{\max})$ is large and routes are long.

Although variant $\text{BCPmax}_{\text{all}-\text{SRCI}}$ is significantly more efficient than variant $\text{BCP2}_{\text{all}-\text{SRCI}}$, the latter performs better for instances in this subset. Variant $\text{BCP2}_{\text{all}-\text{SRCI}}$ solves 16 more instances to optimality in this subset, and is on average 14.48% faster for it.

Comparing our results with the literature, our best configuration solves 214 instances within one hour, surpassing the 127 instances solved by Munari and Savelsbergh (2022), the 109 instances solved by Bianchessi, Drexl, and Irnich (2019), and the 94 instances solved by Archetti, Bouchard, and Desaulniers (2011). In total, 84 instances are solved to optimality for the first time by at least one of the configurations.

We emphasize the exceptional results for instances in class C with 100 customers by configuration $K = K_{\max}$. All 51 instances are solved to optimality in a relatively short time, whereas only seven of them were previously solved in the literature within a one-hour time limit and all of them for $Q = 100$. These results allow us to suggest that instances with small $K_{\max}$ are easier to solve than instances with large $K_{\max}$, such as instances in class R with 50 and 100 customers and in class RC with 100 customers. Note that the difficulty of instances in class RC depends on their size: $K_{\max} = 40$ for instances with 100 customers and $K_{\max} = 4$ for instances with 50 customers. The detailed results are presented in the supplementary materials (E-Companion).

If one is interested in understanding how the parameter K affects the number of split customers in the solutions and the average number of visits each split customer receives, Appendix C provides aggregated results in a format similar to Table 2. In summary, the percentage of split customers remains below 30% for all tested instances, even with small capacities. The number of splits per split customer is rarely greater than one, indicating that split customers generally receive a maximum of 2 visits. Additionally, information about the quantity of inequalities of each class added to the master problem is also provided.

6.4 BCP performance for the SDVRP instances

We evaluate three variants of our BCP algorithm using the complete test set of SDVRP instances. These variants are $\text{BCPmax}_{\text{all}-\text{SRCI}}$, $\text{BCP2}_{\text{all}-\text{SRCI}}$, and $\text{BCP10}_{\text{all}-\text{SRCI}}$, with the latter corresponding to a BCP implementation based on formulation (F10) and incorporating all cutting planes except limited-memory subset-row covering inequalities. As is common in the literature, a time limit of two hours is imposed for these instances, which includes the initial matheuristic time.

Table 2: Aggregated results for two BCP variants for SDVRPTW instances (one-hour time limit)

Class	n	Q	#	BCP2$_{\text{all}-\text{SRCI}}$				BCPmax$_{\text{all}-\text{SRCI}}$				Improvement	
				Nodes	Gap(%)	Time(s)	Opt	Nodes	Gap(%)	Time(s)	Opt	LB*	Opt*
C	50	30	17	7.9	0.03	566.68	16	1.0	0.00	1.72	17	0	0
		50	17	18.8	0.00	53.54	17	1.0	0.00	1.36	17	0	0
		100	17	3.5	0.00	20.31	17	1.9	0.00	7.24	17	0	0
	50P	30	17	138.8	0.15	2179.03	10	121.0	0.10	1384.95	12	-	-
		50	17	145.6	0.59	1817.09	9	21.1	0.00	571.65	17	-	-
		100	17	70.5	0.04	797.56	16	32.9	0.21	1598.33	13	-	-
	75	30	17	235.8	0.60	3392.34	3	5.1	0.00	13.85	17	-	-
		50	17	159.6	0.44	1706.02	9	3.7	0.00	14.22	17	-	-
		100	17	104.4	0.32	1531.32	11	8.5	0.00	65.51	17	-	-
	75P	30	17	52.3	1.29	3425.55	0	100.2	0.53	3424.89	0	-	-
		50	17	58.1	1.89	3424.65	0	37.0	0.98	3348.80	1	-	-
		100	17	48.5	1.52	3365.44	2	7.0	1.57	3429.93	1	-	-
	100	30	17	131.5	0.62	3281.85	0	3.6	0.00	24.76	17	15	15
		50	17	168.9	0.69	3280.58	0	10.1	0.00	79.29	17	17	17
		100	17	32.9	0.02	750.23	16	2.5	0.00	39.60	17	14	14
	100P	30	17	36.5	5.18	3282.42	0	29.1	1.32	3284.38	0	-	-
		50	17	31.5	2.68	3282.01	0	8.3	1.50	3290.51	0	-	-
		100	17	24.1	1.47	2890.75	5	2.9	1.90	3363.90	0	-	-
R	50	30	23	311.8	2.25	3520.69	0	157.5	0.23	2028.55	18	11	7
		50	23	350.5	1.47	2621.81	8	51.5	0.00	894.13	23	12	12
		100	23	51.1	0.00	499.80	23	27.5	0.10	1212.18	20	17	17
	75	30	23	61.4	5.50	3425.53	0	43.8	2.05	3427.79	0	-	-
		50	23	94.7	4.20	3425.25	0	30.5	1.63	3365.12	1	-	-
		100	23	96.0	1.39	2586.37	9	11.3	1.04	2811.71	6	-	-
	100	30	23	45.1	8.34	3282.81	0	22.5	2.39	3285.66	0	23	0
		50	23	44.0	6.04	3281.88	0	8.2	2.40	3310.98	0	23	0
		100	23	40.0	2.58	2886.52	3	3.5	2.08	3045.91	3	22	2
RC	50	30	16	1.3	0.00	730.15	16	1.0	0.00	2.02	16	0	0
		50	16	1.0	0.00	3.49	16	1.0	0.00	1.07	16	0	0
		100	16	1.0	0.00	1.61	16	1.0	0.00	1.34	16	0	0
	50P	30	16	2.5	0.00	70.61	16	247.8	0.01	689.98	13	-	-
		50	16	6.5	0.00	67.58	16	8.9	0.00	260.60	16	-	-
		100	16	4.8	0.00	27.19	16	5.3	0.03	667.98	15	-	-
	75	30	16	64.4	1.31	3426.25	0	78.5	0.33	3142.91	3	-	-
		50	16	119.4	1.33	3424.76	0	41.5	0.46	3292.30	2	-	-
		100	16	166.6	1.09	3032.95	3	6.9	1.51	3118.69	3	-	-
	100	30	16	29.6	7.29	3283.99	0	17.1	0.52	3285.50	0	16	0
		50	16	27.9	6.35	3283.95	0	7.1	0.77	3293.45	0	16	0
		100	16	59.0	6.35	3312.55	0	2.5	1.81	3388.61	0	14	0
Total			705			2243.48	273			1820.09	368	200	84
F$K_{\max}$-Hard			168			2138.39	77			2500.46	61	92	19

Table 3 presents a summary of the results obtained from testing the three BCP variants. Recall that we consider four versions of the original test set of 88 SDVRP instances, obtained by the combination of unlimited- and limited-fleet and Euclidean and rounded Euclidean distances (as described in Section 6.1). The results in the table are aggregated based on each version, instance class, and BCP variant. The table includes the average number of nodes processed in the branch-and-bound tree, the average final primal-dual gap, the arithmetic mean of the pure BCP time and the number of instances solved to optimality. For the $\text{BCPmax}_{\text{all}-\text{SRCI}}$ variant, there is an additional column indicating the number of instances for which the column generation procedure of the root node did not terminate within the time limit. For other variants, column generation always terminated successfully.

Table 3: Results of three BCP variants for the SDVRP instances (two-hour time limit)

Version	Class	#	$\text{BCP2}_{\text{all}-\text{SRCI}}$				$\text{BCP10}_{\text{all}-\text{SRCI}}$				$\text{BCPmax}_{\text{all}-\text{SRCI}}$				
			Nodes	Gap(%)	Time(s)	Opt	Nodes	Gap(%)	Time(s)	Opt	Nodes	Gap(%)	Time(s)	Opt	CG limit
LF-r	eil	11	49.9	0.90	3804.82	6	31.7	0.29	3368.29	8	9.5	0.15	3448.62	4	5
	p	42	33.1	51.27	6439.91	1	39.6	13.71	6352.99	3	11.0	2.98	6232.28	3	2
	S	14	64.6	3.96	5903.60	3	90.6	1.97	5796.95	3	35.1	1.94	5468.34	3	0
	SD	21	28.1	4.28	3836.35	9	111.6	0.66	2726.69	12	111.8	0.66	2713.10	12	0
LF	eil	11	53.5	0.99	3819.96	5	37.7	0.34	3367.02	7	4.5	0.73	3890.60	3	4
	p	42	35.5	45.80	6437.22	1	33.0	12.34	6273.83	2	11.6	3.09	6489.90	3	0
	S	14	78.7	4.18	5738.95	3	98.7	2.03	5742.96	4	49.9	2.05	5619.90	5	0
	SD	21	34.2	6.17	3984.20	9	27.0	0.68	2493.28	13	28.8	0.68	2489.94	13	0
UF-r	eil	11	62.3	1.10	3826.63	5	42.6	0.26	3330.40	9	25.5	0.62	3812.57	5	3
	p	42	33.4	53.48	6443.65	1	37.1	12.97	6300.30	2	12.5	2.71	6314.80	3	1
	S	14	67.4	3.54	6030.97	2	102.3	1.79	5723.07	3	23.0	1.19	5466.32	3	1
	SD	21	30.2	4.85	4010.23	8	41.3	0.51	2711.24	12	45.4	0.51	2717.05	12	0
UF	eil	11	70.5	1.12	3821.47	5	42.3	0.38	3356.50	7	29.5	0.73	3464.70	4	4
	p	42	37.2	45.70	6437.30	1	38.8	12.16	6426.74	2	10.3	2.78	6403.94	3	0
	S	14	81.1	3.71	5778.64	3	113.6	1.73	5669.97	4	36.3	1.36	5376.13	4	2
	SD	21	31.5	6.04	4009.51	8	127.6	0.62	3170.29	11	144.2	0.62	3173.21	11	0
Total		352	43.0	25.49	5428.46	70	57.0	6.59	5019.03	102	33.0	1.92	5026.39	91	22

The results presented in Table 3 clearly indicate that the best performing BCP variant for the SDVRP instances is $\text{BCP10}_{\text{all}-\text{SRCI}}$. The detailed results obtained with this variant can be found in the supplementary materials (E-Companion). Variant $\text{BCPmax}_{\text{all}-\text{SRCI}}$ is ranked second overall. However, its robustness is limited as it fails to obtain even a lower bound for certain instances with very large values of $K_{\max}$. Variant $\text{BCP10}_{\text{all}-\text{SRCI}}$ solved to optimality 102 instances, including fourteen instances for which optimality was proved for the first time. We compare these numbers with the state-of-the-art branch-and-cut algorithms proposed by Munari and Savelsbergh (2022) and Gouveia, Leitner, and Ruthmair (2023). We observe that the latter achieved optimality for 106 instances out of a total of 352 instances. Our algorithm improved the best known lower bounds for 116 instances, accounting for nearly 50% of the previously unsolved instances. For the smaller test set of 224 instances considered by Munari and Savelsbergh (2022), our algorithm achieved optimality for 96 instances, surpassing the results presented in the aforementioned paper by 11 instances. Furthermore, our algorithm improved the best known lower bounds for 51 instances in the reduced set of 108 previously open instances.

Table 4 provides aggregated results for variant $BCP10_{all-SRCI}$. For each class of instances, we show the number of instances solved to optimality, the number of improved lower bounds compared to the literature, and the number of instances solved to optimality for the first time. Appendix B contains aggregated statistics on the solutions, including the percentage of split customers, and information on the number of cuts added to the master problem.

Table 4: Aggregated results of variant $BCP10_{all-SRCI}$ for the SDVRP instances (two-hour time limit)

Class	#	LF-r			LF			UF-r			UF		
		Opt	LB*	Opt*	Opt	LB*	Opt*	Opt	LB*	Opt*	Opt	LB*	Opt*
eil	11	8	5	2	7	6	2	9	5	3	7	6	2
p	42	3	16	1	2	9	0	2	16	0	2	7	0
S	14	3	2	0	4	1	0	3	5	0	4	2	0
SD	21	12	9	1	13	9	1	12	9	1	11	9	1
Total	88	26	32	4	26	25	3	26	35	4	24	24	3

7 Conclusions

We have introduced a new family of partially discretized route-based formulations, denoted as (FK), for split delivery vehicle routing problems. Here, K represents the maximum number of different delivery quantities allowed when visiting a customer. We have shown experimentally that as K increases, the formulation becomes stronger but might become more challenging to solve. In the strongest fully discretized formulation (F$K_{\max}$), all possible delivery quantities are considered.

The proposed formulations rely on a new property that holds true for at least one optimal solution of the problem. This property provides a minimum delivery quantity derived from the customer demand and vehicle capacity, enabling a reduction in $K_{\max}$ for certain instances. Consequently, this reduction improves the computational efficiency of the strongest formulation. This property has the potential to benefit other formulations, as well as other exact and heuristic approaches in the literature.

To effectively solve the formulations, we have designed a BCP algorithm that resorts to new and state-of-the-art algorithmic improvements. Specifically, we have developed the limited-memory variant of subset-row covering inequalities and strong k-path inequalities, mitigating the influence of non-robust valid inequalities. Moreover, we have shown how to consider them while solving the pricing problem that dynamically generates the route variables.

Experimental results on the benchmark instances of the SDVRPTW highlight the excellent performance of our BCP algorithm. In total, 84 instances are solved to optimality for the first time, including many instances with 50 customers and all instances with 100 customers and a small value of $K_{\max}$. Formulation (F$K_{\max}$) demonstrates the best average performance. However, formulation (F2) is more efficient for instances with long routes and a large value of $K_{\max}$. Formulation (F10) stands as the best choice for the SDVRP instances (i.e., instances without time windows). Several SDVRP instances were solved to optimality for the first time, and the best known lower bounds were improved for many or them. In comparison with the literature, our BCP algorithm is especially efficient for instances with longer routes. Based on these findings, the proposed formulations and BCP algorithm establish a new state-of-the-art for the SDVRPTW, and are highly competitive with the best approaches in the literature for the SDVRP. These results indicate that column generation-based approaches can offer comparable or superior performance compared to pure branch-and-cut approaches.

We believe there are interesting future research topics related to improving and extending our solution approach. For example, the master problem of our strongest formulation (F$K_{\max}$) may take a long time to be solved, especially for instances with long routes. One possible way to make it faster is to avoid the discretization of delivery quantities since it reduces the size of the graph in the pricing problem. This can be done, e.g., by inserting load flow variables into the master problem, as proposed by Munari and Savelsbergh (2020), or by defining extreme delivery patterns together with the generation of routes, as introduced by Desaulniers (2010). Another possible improvement to our BCP algorithm concerns the insufficient strength of Formulation (F$K_{\max}$) for instances with a relatively large value of $K_{\max}$. Root optimality gaps may still be high for such instances, even after adding non-robust cuts. One should search for other families of valid inequalities. Separation of cuts based on Chvátal-Gomory rounding of demand covering constraints with different multipliers might be useful to reduce optimality gaps. Thus, developing efficient separation algorithms for such cuts is a promising research direction.

The proposed solution approach could be extended to other variants with additional attributes, such as multiple depots (Gouveia, Leitner, and Ruthmair 2023), heterogeneous fleet (Belfiore and Yoshizaki 2009), pickup and delivery (Casazza, Ceselli, and Wolfler Calvo 2021), and others. Such extensions could be made following the generic modeling approach by Pessoa et al. (2020). Of course, the numerical efficiency of these extensions remains to be seen for each variant separately. Another important extension concerns the case in which the service time for a customer depends on delivery quantity. This dependence might come from non-negligible loading or unloading times (Li et al. 2020).

Acknowledgments

The experiments presented in this paper were performed using the PlaFRIM experimental testbed, supported by Inria, CNRS (LABRI and IMB), Université de Bordeaux, Bordeaux INP and Conseil Régional d'Aquitaine (see https://www.plafrim.fr/).

Pedro Munari is supported by the São Paulo Research Foundation (FAPESP) [grant numbers 19/23596-2, 16/01860-1, 13/07375-0]; and the National Council for Scientific and Technological Development (CNPq-Brazil) [grant number 313220/2020-4].

Teobaldo Bulhões is supported by the National Council for Scientific and Technological Development (CNPq-Brazil) [grant number 314088/2021-0].

Artur Pessoa is supported by the National Council for Scientific and Technological Development (CNPq-Brazil) [grant number 306033/2019-4]

Isaac Balster is supported by the French region Nouvelle Aquitaine [project AAPR2020A-2020-8601810].

We thank Eduardo Uchoa for fruitful discussions, which helped us to start this work.

We would like to thank the associated editor and referees who helped us to improve the quality of the manuscript.

Appendix

A Subset-row inequalities

In this appendix we provide further details on the subset-row cuts defined in Section 4.

A.1 Subset-row packing inequalities

Inequalities (15) can be referred to as 3SRPIs since they were derived from exactly three packing inequalities in (14), i.e., from a set of multipliers in which there are exactly three nonzero numerators μ_i. To find strong SRPIs, Pecin et al. (2017b) performed a computational study of the complete set partitioning polytope $CSPP_=(p)$, defined below together with the related complete set packing polytope $CSPP_\leq(p)$ and complete set covering polytope $CSPP_\geq(p)$.

$$CSPP_=(p) = \text{Conv}\left\{B_p x = \mathbb{1}, x \in \{0,1\}^{2^p-1}\right\}. \tag{26}$$

$$CSPP_\leq(p) = \text{Conv}\left\{B_p x \leq \mathbb{1}, x \in \{0,1\}^{2^p-1}\right\}. \tag{27}$$

$$CSPP_\geq(p) = \text{Conv}\left\{B_p x \geq \mathbb{1}, x \in \{0,1\}^{2^p-1}\right\}. \tag{28}$$

In the definition of these polytopes, B_p is a binary matrix with p rows and all distinct $2^p - 1$ nonzero columns, and $\mathbb{1}$ represents the p-dimensional all-ones vector. Due to the combinatorial explosion of their approach, the authors managed to study the $CSPP_=(p)$ only for $p \leq 5$, and they concluded that the SRPIs associated with the following set of multipliers are facet inducing:

- 3 rows: $\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$;
- 4 rows: $\left(\frac{2}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$;
- 5 rows: $\left(\frac{2}{4}, \frac{2}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$, $\left(\frac{3}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$, $\left(\frac{3}{5}, \frac{2}{5}, \frac{2}{5}, \frac{1}{5}, \frac{1}{5}\right)$, $\left(\frac{2}{3}, \frac{2}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$, $\left(\frac{3}{4}, \frac{3}{4}, \frac{2}{4}, \frac{2}{4}, \frac{1}{4}\right)$, $\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$, $\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$.

Later, Bulhões et al. (2018) determined new families of multipliers that induce facets of $CSPP_=(p)$ for arbitrarily large values of p, and they showed that, with very few exceptions, every facet-inducing inequality for $CSPP_\leq(p)$ is also facet inducing for $CSPP_=(p)$, and vice versa.

Observation 5 *The existence of routes $r \in \mathcal{R}$ with nonbinary coefficients in* (14) *has no impact on the strength of the multipliers found by Pecin et al. (2017b) and Bulhões et al. (2018).*

This follows from such routes not being part of an integer solution.

A.2 Subset-row covering inequalities

We developed a computational approach to analyze the complete set covering polytope $CSPP_\geq(p)$ (see (28)) in the spirit of the work by Pecin et al. (2017b). Despite the inferior scalability of this approach for $CSPP_{\geq}(p)$ due to a larger number of extreme points, we managed to find the following multipliers, the associated SRCIs of which are facet-inducing:

- 3 rows: $\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$;
- 4 rows: $\left(\frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{1}{3}\right), \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$;
- 5 rows: $\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right), \left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{2}{3}, \frac{2}{3}\right), \left(\frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}, \frac{2}{3}\right), \left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right), \left(\frac{1}{4}, \frac{1}{4}, \frac{2}{4}, \frac{2}{4}, \frac{3}{4}\right)$, $\left(\frac{1}{4}, \frac{3}{4}, \frac{3}{4}, \frac{3}{4}, \frac{3}{4}\right), \left(\frac{2}{4}, \frac{2}{4}, \frac{3}{4}, \frac{3}{4}, \frac{3}{4}\right), \left(\frac{1}{5}, \frac{1}{5}, \frac{3}{5}, \frac{3}{5}, \frac{3}{5}\right), \left(\frac{1}{5}, \frac{2}{5}, \frac{2}{5}, \frac{2}{5}, \frac{4}{5}\right), \left(\frac{2}{5}, \frac{3}{5}, \frac{3}{5}, \frac{4}{5}, \frac{4}{5}\right)$.

Observation 6 *In contrast to the packing case, one might find stronger SRCIs by analyzing a generalization of $CSPP_{\geq}(p)$ in which the extreme points are not necessarily binary.*

Preliminary experiments showed that the large number of extreme points of this generalized polytope renders our computational approach impractical.

B Aggregated statistics for benchmark SDVRP tests

Table 5 aggregates statistics on the benchmark tests with formulation $\text{BCP10}_{\text{all}-\text{SRCI}}$. For each test configuration and instance class, we list the percentage of partially discretized customers in column "Part. Disc.(%)", the percentage of customers with splits in the column "Split cust.(%)", the average number of splits per split customer in the column "Splits", the total of cuts added to the master in the column "# Cuts", as well as for each inequality, the specific quantity of cuts added in columns "# R1C", "# CAP" and "# SKP". At last, column "Opt" counts the number of optimal solutions found.

C Aggregated statistics for benchmark SDVRPTW tests

Table 6 aggregates statistics on the benchmark tests and the results obtained with formulations $\text{BCP2}_{\text{all}-\text{SRCI}}$ and $\text{BCPmax}_{\text{all}-\text{SRCI}}$. For each class of instances, we list $\bar{q}$, the percentage of partially discretized customers in column "Part. Disc.(%)" (only for variant $\text{BCP2}_{\text{all}-\text{SRCI}}$), the percentage of customers with splits in the column "Split cust.(%)", the average number of splits per split customer in the column "Splits", the total of cuts added to the master in the column "# Cuts", as well as for each inequality, the specific quantity of cuts added in columns "# R1C", "# CAP" and "# SKP". At last, column "Opt" counts the number of optimal solutions found.

Table 5: Aggregated statistics on the number of partially discretized customers, splits per split customer and cuts for the SDVRP (two-hours time limit)

Configuration	Class	#	$BCP10_{all-SRCI}$							
			Part. Disc.(%)	Split cust.(%)	# Splits	# Cuts	# R1C	# CAP	# SKP	Opt
LF-r	eil	11	73.4	3.3	1.2	9664	4369	1945	3349	8
	p	42	91.9	28.2	1.2	7969	850	4238	2880	3
	S	14	85.7	18.3	1.2	15419	1564	7971	5884	3
	SD	21	0.0	53.6	1.1	6062	309	3425	2329	12
LF	eil	11	73.4	3.1	1.0	10085	4799	1775	3512	7
	p	42	91.9	27.4	1.1	7623	576	4467	2580	2
	S	14	85.7	17.0	1.2	17206	1557	10057	5592	4
	SD	21	0.0	53.8	1.0	5037	116	3159	1761	13
UF-r	eil	11	73.4	2.5	1.1	9941	5039	1955	2947	9
	p	42	91.9	24.8	1.3	7814	748	4227	2839	2
	S	14	85.7	15.0	1.3	15362	1573	8831	4958	3
	SD	21	0.0	51.4	1.1	7618	262	4516	2840	12
UF	eil	11	73.4	2.2	1.0	8160	4161	1636	2363	7
	p	42	91.9	23.9	1.2	7677	607	4527	2543	2
	S	14	85.7	16.1	1.2	17399	1666	10677	5056	4
	SD	21	0.0	50.3	1.1	26057	298	18958	6801	11

Table 6: Aggregated statistics on the number of partially discretized customers, splits per split customer and cuts for the SDVRPTW (one-hour time limit)

Class	n	Q	$\bar{q}$	#	BCP2$_{\text{all-SRCI}}$								BCPmax$_{\text{all-SRCI}}$						
					Part. Disc.(%)	Split cust.(%)	# Splits	# Cuts	# R1C	# CAP	# SKP	Opt	Split cust.(%)	# Splits	# Cuts	# R1C	# CAP	# SKP	Opt
C	50	30	10	17	18.0	9.9	1.0	3922	6	3277	640	16	9.9	1.0	545	0	545	0	17
		50	10	17	18.0	5.8	1.0	3003	76	2348	579	17	5.8	1.0	267	0	267	0	17
		100	10	17	18.0	0.5	1.0	693	147	384	162	17	0.5	1.0	512	170	224	117	17
	50P	30	1	17	100.0	26.5	1.1	56938	507	33080	23351	10	26.1	1.1	17071	165	11317	5588	12
		50	1	17	100.0	11.5	1.1	27087	541	18179	8367	9	15.8	1.0	2282	223	1191	868	17
		100	1	17	100.0	4.0	1.0	8336	2806	2300	3230	16	4.0	1.0	4025	2112	752	1161	13
	75	30	10	17	21.3	11.8	1.0	90396	624	53322	36450	3	11.9	1.0	1850	5	1733	112	17
		50	10	17	21.3	2.3	1.0	44788	1405	34793	8590	9	2.9	1.0	1044	44	887	113	17
		100	10	17	21.3	0.7	1.0	14696	4228	5704	4764	11	1.2	1.0	1806	705	638	463	17
	75P	30	1	17	100.0	24.2	1.1	51929	650	29152	22128	0	25.1	1.1	23329	464	15920	6946	0
		50	1	17	100.0	8.8	1.1	34985	867	22267	11851	0	9.6	1.1	8183	817	4926	2440	1
		100	1	17	100.0	4.2	1.1	16472	3197	6031	7243	2	3.1	1.1	2220	831	664	725	1
	100	30	10	17	22.0	10.0	1.0	83362	785	46822	35755	0	10.4	1.0	1790	2	1681	108	17
		50	10	17	22.0	2.8	1.0	55313	1635	42492	11186	0	4.1	1.0	2526	74	1991	461	17
		100	10	17	22.0	1.1	1.0	7155	1379	4355	1420	16	1.1	1.0	838	112	582	145	17
	100P	30	1	17	100.0	25.9	1.2	38957	1300	20136	17521	0	25.9	1.2	10695	223	7510	2962	0
		50	1	17	100.0	12.6	1.1	27848	965	15731	11152	0	12.6	1.1	2986	304	1830	853	0
		100	1	17	100.0	6.8	1.0	11267	1520	6543	3204	5	5.4	1.0	1092	313	483	297	0
R	50	30	1	23	96.0	23.3	1.0	96912	1230	64388	31294	0	25.0	1.0	23115	706	15408	7001	18
		50	1	23	96.0	9.2	1.2	47886	2412	27137	18337	8	10.4	1.2	5599	620	2654	2325	23
		100	1	23	96.0	4.1	1.0	5872	2306	1908	1659	23	3.7	1.0	3586	2059	498	1028	20
	75	30	1	23	96.0	24.6	1.2	47306	942	25686	20678	0	24.6	1.2	10564	387	6359	3818	0
		50	1	23	96.0	9.2	1.1	34100	1254	21683	11164	0	9.4	1.1	4664	735	2132	1798	1
		100	1	23	96.0	3.3	1.1	16155	5636	5912	4606	9	3.2	1.1	2621	1465	509	646	6
	100	30	1	23	95.0	26.2	1.2	33304	1229	16904	15170	0	26.2	1.2	4649	220	2833	1595	0
		50	1	23	95.0	10.6	1.1	24960	1018	14836	9106	0	10.6	1.1	1671	257	814	600	0
		100	1	23	95.0	2.1	1.0	11353	3729	3909	3715	3	2.1	1.0	787	422	181	183	3
RC	50	30	10	16	26.0	13.1	1.0	1386	2	1288	97	16	13.0	1.0	662	0	662	0	16
		50	10	16	26.0	5.5	1.0	381	0	381	0	16	5.0	1.0	197	0	197	0	16
		100	10	16	26.0	1.3	1.0	86	1	85	0	16	1.3	1.0	81	20	59	2	16
	50P	30	1	16	100.0	27.4	1.1	4644	36	3404	1204	16	27.5	1.1	2433	26	1787	620	13
		50	1	16	100.0	12.8	1.0	2969	26	2612	331	16	12.5	1.0	1572	72	1096	404	16
		100	1	16	100.0	6.5	1.2	785	127	453	205	16	6.8	1.1	973	296	306	372	15
	75	30	1	16	100.0	15.5	1.0	66187	1355	36440	28391	0	15.8	1.0	34366	380	24474	9512	3
		50	1	16	100.0	4.3	1.0	38643	1145	23396	14102	0	4.9	1.0	6684	479	3842	2363	2
		100	1	16	100.0	2.3	1.0	20113	5815	7435	6863	3	2.1	1.0	1238	689	242	308	3
	100	30	1	16	99.0	17.7	1.1	33571	1703	16931	14937	0	17.7	1.1	11579	182	7534	3863	0
		50	1	16	99.0	6.7	1.1	23164	812	13665	8687	0	6.7	1.1	2330	158	1501	671	0
		100	1	16	99.0	1.7	1.1	13632	3187	6478	3967	0	1.5	1.1	737	371	179	188	0

References

Alvarez A, Munari P, 2022 Heuristic approaches for split delivery vehicle routing problems. Technical report, number 8790, Operations Research Group, Production Engineering Department, Federal University of Sao Carlos - Brazil.

Archetti C, Bianchessi N, Speranza MG, 2011 A column generation approach for the split delivery vehicle routing problem. Networks 58(4):241–254.

Archetti C, Bianchessi N, Speranza MG, 2014 Branch-and-cut algorithms for the split delivery vehicle routing problem. European Journal of Operational Research 238(3):685–698.

Archetti C, Bouchard M, Desaulniers G, 2011 Enhanced branch and price and cut for vehicle routing with split deliveries and time windows. Transportation Science 45:285–298.

Archetti C, Savelsbergh MWP, Speranza MG, 2006 Worst-case analysis for split delivery vehicle routing problems. Transportation Science 40:226–234.

Baldacci R, Christofides N, Mingozzi A, 2008 An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. Mathematical Programming 115:351–385.

Baldacci R, Mingozzi A, Roberti R, 2011 New route relaxation and pricing strategies for the vehicle routing problem. Operations Research 59(5):1269–1283.

Bektaş T, Laporte G, 2011 The pollution-routing problem. Transportation Research Part B: Methodological 45:1232–1250.

Belfiore P, Yoshizaki HTY, 2009 Scatter search for a real-life heterogeneous fleet vehicle routing problem with time windows and split deliveries in brazil. European Journal of Operational Research 199(3):750–758.

Bianchessi N, Drexl M, Irnich S, 2019 The split delivery vehicle routing problem with time windows and customer inconvenience constraints. Transportation Science 53:1067–1084.

Bianchessi N, Irnich S, 2019 Branch-and-cut for the split delivery vehicle routing problem with time windows. Transportation Science 53:442–462.

Braekers K, Ramaekers K, Nieuwenhuyse IV, 2016 The vehicle routing problem: State of the art classification and review. Computers & Industrial Engineering 99:300–313.

Bulhões T, Pessoa A, Protti F, Uchoa E, 2018 On the complete set packing and set partitioning polytopes: Properties and rank 1 facets. Operations Research Letters 46(4):389–392.

Casazza M, Ceselli A, Wolfler Calvo R, 2021 A route decomposition approach for the single commodity split pickup and split delivery vehicle routing problem. European Journal of Operational Research 289(3):897–911.

Desaulniers G, 2010 Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. Operations Research 58:179–192.

Desrochers M, Soumis F, 1989 A column generation approach to the urban transit crew scheduling problem. Transportation Science 23:1–13.

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

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

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

Frizzell PW, Giffin JW, 1995 The split delivery vehicle scheduling problem with time windows and grid network distances. Computers & Operations Research 22(6):655–667.

Gouveia L, Leitner M, Ruthmair M, 2023 Multi-depot routing with split deliveries: Models and a branch-and-cut algorithm. Transportation Science 57:512–530.

Ho SC, Haugland D, 2004 A tabu search heuristic for the vehicle routing problem with time windows and split deliveries. Computers & Operations Research 31(12):1947–1964.

Jepsen M, Petersen B, Spoorendonk S, Pisinger D, 2008 Subset-row inequalities applied to the vehicle-routing problem with time windows. Operations Research 56(2):497–511.

Jin M, Liu K, Eksioglu B, 2008 A column generation approach for the split delivery vehicle routing problem. Operations Research Letters 36(2):265–270.

Laporte G, Nobert Y, 1983 A branch and bound algorithm for the capacitated vehicle routing problem. Operations-Research-Spektrum 5(2):77–85.

Li J, Qin H, Baldacci R, Zhu W, 2020 Branch-and-price-and-cut for the synchronized vehicle routing problem with split delivery, proportional service time and multiple time windows. Transportation Research Part E: Logistics and Transportation Review 140:101955.

Luo Z, Qin H, Zhu W, Lim A, 2017 Branch and price and cut for the split-delivery vehicle routing problem with time windows and linear weight-related cost. Transportation Science 51:668–687.

Lysgaard J, Letchford AN, Eglese RW, 2004 A new branch-and-cut algorithm for the capacitated vehicle routing problem. Mathematical Programming 100:423–445.

Moreno L, De Aragao MP, Uchoa E, 2010 Improved lower bounds for the split delivery vehicle routing problem. Operations Research Letters 38(4):302–306.

Mullaseril P, Dror M, 1996 A set covering approach for directed node and arc routing problems with split deliveries and time windows. Technical report, MIS department, University of Arizona, Tucson, Arizona.

Mullaseril PA, Dror M, Leung J, 1997 Split-delivery routeing heuristics in livestock feed distribution. Journal of the Operational Research Society 48(2):107–116.

Munari P, Savelsbergh M, 2020 A column generation-based heuristic for the split delivery vehicle routing problem with time windows. SN Operations Research Forum 1(4):1–24.

Munari P, Savelsbergh M, 2022 Compact formulations for split delivery routing problems. Transportation Science 56:1022–1043.

Ozbaygin G, Karasan O, Yaman H, 2018 New exact solution approaches for the split delivery vehicle routing problem. EURO Journal on Computational Optimization 6:85–115.

Pecin D, Pessoa A, Poggi M, Uchoa E, 2017a Improved branch-cut-and-price for capacitated vehicle routing. Mathematical Programming Computation 9(1):61–100.

Pecin D, Pessoa A, Poggi M, Uchoa E, Santos H, 2017b Limited memory rank-1 cuts for vehicle routing problems. Operations Research Letters 45(3):206–209.

Pessoa A, Sadykov R, Uchoa E, Vanderbeck F, 2018 Automation and combination of linear-programming based stabilization techniques in column generation. INFORMS Journal on Computing 30(2):339–360.

Pessoa A, Sadykov R, Uchoa E, Vanderbeck F, 2020 A generic exact solver for vehicle routing and related problems. Mathematical Programming 183:483–523.

Ryan DM, Foster BA, 1981 An integer programming approach to scheduling. Computer scheduling of public transport: Urban passenger vehicle and crew scheduling 269–280.

Sadykov R, Uchoa E, Pessoa A, 2021 A bucket graph–based labeling algorithm with application to vehicle routing. Transportation Science 55(1):4–28.

Sadykov R, Vanderbeck F, 2021 BaPCod — a generic Branch-And-Price Code. Technical report HAL-03340548, Inria Bordeaux — Sud-Ouest.

Shapiro JF, 2007 Modeling the supply chain, volume 2 (Thomson Brooks/Cole).

Toth P, Vigo D, 2014 Vehicle Routing: Problems, Methods and Applications (Society for Industrial and Applied Mathematics), second edition.

Weil A, 1983 Number Theory: an approach through history. From Hammurapi to Legendre. (Boston/Basel/Stuttgart: Birkhäuser).