Submitted to *Transportation Science*
manuscript TS-2019-0423

Compact Formulations for Split Delivery Routing Problems

Pedro Munari
Production Engineering Department, Federal University of Sao Carlos, Sao Carlos, SP, 13561-353, Brazil
munari@dep.ufscar.br

Martin Savelsbergh
H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 30332
martin.savelsbergh@isye.gatech.edu

Split delivery routing problems are concerned with serving the demand of a set of customers with a fleet of capacitated vehicles at minimum cost, where a customer can be served by more than one vehicle if beneficial. They generalize traditional variants of routing problems and have applications in commercial as well as humanitarian logistics. Previously, formulations involving only commonly used arc-based variables have provided only relaxations for split delivery variants, as the possibility of visiting customers more than once introduces modeling challenges. The only known compact formulations are based on variables indexed by vehicle or by visit number and perform poorly when using general-purpose integer programming software. We present compact formulations that avoid the use of these types of variables and that can model split delivery routing problems with and without time windows. Computational experiments demonstrate their superior performance over existing compact formulations. We also develop a branch-and-cut algorithm that balances the efficiency derived from a relaxed formulation with the strength derived from one of the proposed formulations, and demonstrate its efficacy on a large set of benchmark instances. The algorithm solves 95 instances to proven optimality for the first time and improves the best known lower and/or upper bound for many other instances.

Key words: vehicle routing; split delivery; compact models; branch-and-cut

1. Introduction

Split delivery routing problems are concerned with serving the demand of a set of customers with a fleet of capacitated vehicles at minimum cost. A customer, contrary to what is assumed in classical routing problems, can be served by more than one vehicle, if beneficial. Split delivery routing problems have attracted interest of researchers since the late eighties of the last century, e.g., Dror and Trudeau (1989, 1990), Dror, Laporte, and Trudeau (1994), Belenguer, Martinez, and Mota (2000),

Archetti, Speranza, and Hertz (2006), Archetti, Savelsbergh, and Speranza (2008a,b), Moreno, De Aragao, and Uchoa (2010), Archetti and Speranza (2012), Archetti, Bianchessi, and Speranza (2014, 2015), Silva, Subramanian, and Ochi (2015), Chen et al. (2017), Ozbaygin, Karasan, and Yaman (2018). Variants in which customers need to be visited during a given time window have also received attention, but much less, e.g., Gendreau et al. (2006), Desaulniers (2010), Archetti, Bouchard, and Desaulniers (2011), Bianchessi and Irnich (2019). Many properties of split delivery routing problems have been derived and many exact and heuristic solution approaches have been proposed. Recently, the focus has been on exact solution approaches, which is also the topic of our research.

Let $\mathcal{C} = \{1, \dots, n\}$ be a set of n customers, each with demand $d_i > 0$ for $i \in \mathcal{C}$ and $\mathcal{M} = \{1, \dots, m\}$ be a set of homogeneous vehicles, each with capacity Q. When visiting customer $i \in \mathcal{C}$, a vehicle $k \in \mathcal{M}$ can deliver any quantity less than or equal to $\min\{d_i, Q\}$. Consequently, customers may be visited by more than one vehicle. It is this feature that differentiates split delivery routing problems from classical routing problems.

We assume a single depot, represented by 0 and $n+1$, and require that each vehicle starts at 0 and ends its route at $n+1$. The problem can be represented on a graph $G = (\mathcal{N}, \mathcal{A})$ where the set of nodes $\mathcal{N} = \mathcal{C} \cup \{0, n+1\}$ and the set of arcs $\mathcal{A} = \{(i,j) : i, j \in \mathcal{C}, i \neq j\} \cup \{(0,i) : i \in \mathcal{C}\} \cup \{(i, n+1) : i \in \mathcal{C}\}$. A cost c_{ij} is associated with each arc $(i,j) \in \mathcal{A}$; we assume costs c_{ij} satisfy the triangle inequality. The split delivery vehicle routing problem (SDVRP) seeks a least-cost set of routes serving all customer demands. It generalizes the vehicle routing problem (VRP) and can provide solutions that reduce routing costs by up to 50% (Archetti, Savelsbergh, and Speranza 2006).

A closely-related variant, the SDVRP with time windows (SDVRPTW), generalizes the VRP with time windows (VRPTW) and requires that a customer $i \in \mathcal{C}$ is served within a time window $[w_i^a, w_i^b]$. A vehicle can arrive at a customer $i \in \mathcal{C}$ before w_i^a, but it has to wait until w_i^a to serve the customer. In this setting, a travel time t_{ij} is associated with each arc $(i,j) \in \mathcal{A}$; we assume travel times t_{ij} satisfy the triangle inequality. Furthermore, we assume that serving customer $i \in \mathcal{C}$ takes s_i and that vehicles depart from the depot no earlier than w_0^a and return to the depot no later than w_{n+1}^b.

The possibility to visit a customer more than once creates modeling as well as algorithmic challenges. Compact formulations for the SDVRP and SDVRPTW use variables indexed by arcs and vehicles (Dror and Trudeau 1990, Ozbaygin, Karasan, and Yaman 2018). These formulations can only be used to solve small instances because of the inherent symmetry. Recently, an alternative compact formulation involving variables capturing the number of visits at a customer has been proposed (Hernández-Pérez and Salazar-González 2019). This formulation too can only be used to solve small instances because of its weak linear programming relaxation. As a consequence, the

most successful exact algorithms for the SDVRP and SDVRPTW use *relaxed* formulations, i.e., formulations that do not capture all aspects of feasibility. The most common relaxed formulation allows the transfer of product between vehicles at a customer location, i.e., when two vehicles visit the same customer, product in one vehicle is allowed to be transferred to the other vehicle. Thus, it may be possible that a vehicle making a delivery at a customer arrives at that customer with less product than when it departs from that customer. If an optimal solution to the relaxed formulation is found to be infeasible, then an attempt is made to convert it to a feasible solution, and, if that fails, the relaxed formulation is modified to make that solution infeasible for the relaxed formulation as well, e.g., by adding a *feasibility* cut, by locally extending the formulation, or by adjusting the underlying network.

The contributions of our research can be summarized as follows:

- We present three new compact formulations for the SDVRP that use variables indexed by arcs only; the formulations can be adapted easily to the SDVRPTW. They can be viewed as extended formulations, in the sense that a variable associated with an arc also identifies a subsequent arc. The first formulation uses Miller-Tucker-Zemlin (MTZ) constraints (Miller, Tucker, and Zemlin 1960), while the other two use commodity-flow constraints. To the best of our knowledge, these are the first compact formulations that avoid the use of variables indexed by vehicle or visit number. Computational experiments using benchmark instances demonstrate their superior performance over existing compact formulations. We provide further insights by analytically comparing the strength of the two commodity-flow formulations. The new compact formulations will benefit OR practitioners needing or wanting to solve split delivery routing problems using general-purpose integer programming software.
- We develop a branch-and-cut (BC) algorithm that *locally* extends a relaxed formulation to prevent transfer of demand at a customer. The algorithm combines the efficiency that comes from using a relaxed formulation with the strength that comes from using an extended formulation (similar ideas are presented by Ozbaygin, Karasan, and Yaman (2018)). These ideas may also be explored in other VRP variants, e.g., with multiple customer visits (Berbeglia et al. 2007, Bruck and Iori 2017), with variables indexed by more than one arc (Munari, Dollevoet, and Spliet 2016), or even in other settings in which extended formulations have been used, e.g., machine scheduling (Pessoa et al. 2010) and lot sizing (Fragkos, Degraeve, and De Reyck 2016).
- We enhance the effectiveness of our BC algorithm by incorporating well-known valid inequalities (using cuts available in CVRPSep (Lysgaard, Letchford, and Eglese 2004)) and demonstrate its efficacy on a large set of benchmark instances (for both SDVRP and SDVRPTW); for several open instances optimal solutions are found and for several other open instances tighter lower and/or upper bounds are found.

We want to emphasize that the excellent computational performance is due primarily to the use of a new formulation. The implementation of the BC algorithm uses only basic functionality provided in any (commercial) integer programming solver and standard cuts available in CVRPSep.

The remainder of the paper is organized as follows. Section 2 presents several properties of optimal solutions of the split delivery variants and illustrates the modeling challenges that arise when customers can be visited multiple times. Section 3 reviews relevant literature and highlights the state-of-the-art solution approaches. Section 4 presents the three new compact formulations. Section 5 details the BC algorithm based on a relaxed formulation that is gradually extended to one of the proposed formulations. Computational experiments using benchmark instances of the SDVRP and SDVRPTW are presented in Section 6. Finally, Section 7 gives final remarks and discusses ideas for future work.

2. Properties

In search for effective algorithms for the solution of split delivery routing problems, researchers have been able to establish useful structural properties (Dror and Trudeau 1990, Gendreau et al. 2006, Desaulniers 2010, Archetti, Bianchessi, and Speranza 2011, 2014). Most importantly, it is known that for a feasible instance with integer demands and vehicle capacity as well as costs and travel times that satisfy the triangle inequality, there exists an optimal solution with the following properties:

1. Routes are elementary, i.e., a route never visits a node more than once;
2. Two routes have at most one node in common;
3. Arcs between customers are traversed at most once;
4. If arc (i, j) is traversed, then arc (j, i) is not; and
5. Delivery quantities are positive integers.

This knowledge can be exploited to significantly reduce the solution space.

However, the fact that customers may receive deliveries from more than one vehicle creates modeling challenges, particularly for formulations involving only two-index variables, as there is no easy way to prevent the transfer of product from one vehicle to another at a customer (which is not allowed in split delivery routing problems). The difficulty is illustrated in Figure 1. Customer 3 with a demand of 2 is visited by two vehicles, one arriving from customer 1 with 6 units of product left in the vehicle and another arriving from customer 2 with 8 units of product left in the vehicle. One issue is that with only two-index variables, we do not know which of the two vehicles departs to customer 4 and which of the two vehicles departs to customer 5. The more important issue, however, is that product may be transferred from one vehicle to another. Even though both vehicles arrive with less than or equal to 8 units of product, one of them departs to customer 5 with 9 units of product.

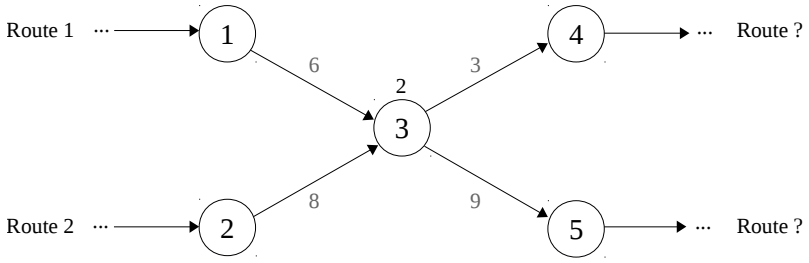


Figure 1 **Example with two different routes visiting node 3, illustrating the difficulty of uniquely determining the set of routes by using two-index variables only.**

As a consequence, existing compact exact formulations for split delivery routing problems involve variables indexed by vehicle or by visit number. State-of-the-art special-purpose branch-and-cut (BC) algorithms for solving split delivery routing problems, on the other hand, use relaxed formulations involving arc-indexed variables, and have to (1) determine whether a solution to the relaxed problem is feasible for the SDVRP, and, if not, (2) modify the formulation to eliminate that solution from the relaxed problem. In particular, w.r.t. (1), it has been shown, e.g., Archetti, Bianchessi, and Speranza (2014) and Ozbaygin, Karasan, and Yaman (2018), that a feasible solution to a relaxed commodity-flow formulation that satisfies the following regularity property is also feasible for the SDVRP:

Regularity Property. *A solution has the regularity property if, for each route, the vehicle load flow on arc* (i, j) *traversed by the route is greater than or equal to the vehicle load flow in the subsequent arc* (j, k) *traversed by the route.*

Moreover, verifying whether a solution to a relaxed commodity-flow formulation satisfies the regularity property can be done using a polynomial-time algorithm, which seeks to assign arcs to routes in such a way that the regularity property is satisfied.

3. Relevant Literature

Dror, Laporte, and Trudeau (1994) were the first to propose a BC algorithm for the SDVRP using a vehicle-indexed formulation and several classes of valid inequalities (subtour elimination, connectivity, and fractional cycle elimination). They explore the benefits of the valid inequalities by examining the value of the LP relaxation at the root node. Their computational experiments, using instances with up to 20 customers, demonstrate that the inequalities can significantly improve the root node bound. Belenguer, Martinez, and Mota (2000) were the first to explore a formulation involving only arc-indexed variables. They conducted a polyhedral study of the formulation and derived several classes of facet-inducing inequalities, which were used in a cutting-plane algorithm

to provide lower bounds for the SDVRP. Their cutting plane approach was able to obtain provably optimal solutions to some instances with 50 customers.

Archetti, Bianchessi, and Speranza (2014) presented two BC algorithms for the SDVRP, one based on the formulation of Belenguer, Martinez, and Mota (2000), the other based on a commodity-flow formulation; both are relaxed formulations. To verify the feasibility of candidate solutions, a separate optimization problem has to be solved, which may also identify an alternative solution that is feasible for the SDVRP when the candidate is infeasible. In the case that the candidate solution is obtained using the commodity-flow formulation, their BC algorithm relies on the polynomial-time procedure `MatchArcs` to check feasibility, but still resorts to the optimization problem when this procedure cannot verify feasibility. The computational study in Archetti, Bianchessi, and Speranza (2014) shows that both BC algorithms outperform earlier approaches; 17 instances were solved to optimality (with up to 100 customers) and new lower and upper bounds were obtained for many other instances.

Ozbaygin, Karasan, and Yaman (2018) proposed a solution approach based on a new compact, vehicle-indexed flow formulation for the SDVRP. Rather than solving this formulation directly, the authors work with a relaxation obtained by aggregating the vehicle-indexed variables over the vehicles. A solution to this relaxation may not satisfy the regularity property and, thus, be SDVRP infeasible. The authors cut-off such solutions, in a novel way, either by extending the formulation locally with vehicle-indexed variables or by modifying the graph representation of the instance by node splitting. Thus, the approach solves a sequence of relaxations (where each relaxation is itself an integer program) until the optimal solution to the relaxation satisfies the regularity property and is feasible for the SDVRP. A computational study shows that the approach is competitive with existing approaches.

For the SDVRPTW, Bianchessi and Irnich (2019) proposed a BC algorithm based on a new, relaxed commodity flow formulation in which time is propagated using commodity-flow constraints. Their BC algorithm extends the one proposed by Archetti, Bianchessi, and Speranza (2014) and also relies on a residual optimization problem to verify the feasibility of solutions. The authors also propose new classes of valid inequalities, which strengthen the relaxed formulation and cut off (some) infeasible solutions. Their computational study, using benchmark instances, shows that the proposed BC is competitive with state-of-the-art methods; 23 instances were solved to optimality for the first time.

Branch-and-price (BP) algorithms have also been developed for split delivery routing problems and, different from what is observed in several other VRP variants, they are not superior to BC algorithms. Indeed, the possibility to visit a customer more than once also creates challenges

for algorithms using path-based formulations. Gendreau et al. (2006) propose a BP algorithm for the SDVRPTW, in which routes are generated by solving a resource constrained elementary shortest path problem (RCESPP) using a labeling algorithm, and where the delivery quantities are determined in the master problem. The advantage is that the pricing problem is similar to those for other VRP variants, such as the CVRP and VRPTW. The disadvantage is that the master problem requires one constraint for each route, leading to a formulation with an exponential number of constraints (in addition to an exponential number of variables). Ceselli, Righini, and Salani (2009) and Moreno, De Aragao, and Uchoa (2010) propose different path-based formulations, with a polynomial number of constraints, and use column and cut generation to provide tight(er) lower bounds for both the SDVRP and the SDVRPTW.

Desaulniers (2010) proposes a BP method for the SDVRPTW that relies on a pricing problem that generates routes associated with "extreme" delivery patterns. Hence, in the master problem, the variables correspond to routes and their corresponding extreme delivery patterns, and, for a given route, non-extreme delivery quantities are obtained as convex combinations of the extreme delivery patterns associated with the considered route. The pricing problem is more challenging than the traditional RCESPP arising for the VRPTW, but the number of constraints in the master problem is polynomial. Archetti, Bouchard, and Desaulniers (2011) enhanced this approach by incorporating a Tabu Search heuristic to solve the RCESPP, and adding new classes of valid inequalities to the master problem. Archetti, Bianchessi, and Speranza (2011) use a similar formulation for solving the SDVRP. Because there are no time windows to restrict the search space, the pricing problem becomes even more difficult to solve. To speed up the solution of the pricing problem, the requirement that routes have to be elementary is dropped.

Several heuristics and hybrid methods have been proposed as well for the split delivery routing problems, e.g., Ho and Haugland (2004), Archetti, Speranza, and Hertz (2006), Archetti, Savelsbergh, and Speranza (2008a), Jin, Liu, and Eksioglu (2008), Silva, Subramanian, and Ochi (2015), and Chen et al. (2017), some with focus on solving real-world instances, e.g., Sierksma and Tijssen (1998), Belfiore and Yoshizaki (2009), Sepúlveda, Escobar, and Adarme-Jaimes (2014), and Chen, Golden, and Wasil (2007). For comprehensive reviews, see Archetti and Speranza (2012) and Irnich, Schneider, and Vigo (2014).

Finally, it is worth mentioning a few other VRP variants that also allow multiple visits to customers and, hence, share similarities with the split delivery variants. For example, pickup and delivery problems (PDPs) may require multiple visits to customers in settings that involve delivering new as well as collecting old products, or involve relocating items among facilities in order to achieve

balance (see Berbeglia et al. (2007) for a comprehensive review of problems of this type). One interesting application of PDPs arises in bike sharing systems (Chemla, Meunier, and Calvo 2013, Casazza et al. 2018) for which several authors have proposed novel modeling techniques and effective solution strategies that can benefit split delivery variants (Bruck and Iori 2017, Bruck et al. 2019, Casazza, Ceselli, and Wolfler Calvo 2021, Wolfinger and Salazar-Gonzalez 2021). Salazar-González and Santos-Hernández (2015) introduced the split-demand one-commodity pickup and delivery traveling salesman problem that generalizes the SDVRP as the depot node can be visited multiple times as well. They proposed a new formulation and a branch-and-cut method that was later used as the basis of the solution approach proposed by Hernández-Pérez and Salazar-González (2019) to solve SDVRP instances.

4. Exact Models for Split Delivery Routing Problems

In this section, we propose three new compact exact integer programming formulations for the SDVRP and SDVRPTW. They use the following vehicle-flow decision variables:

$$x_{hij} = \begin{cases} 1 \text{ if there is a route that traverses arcs } (h,i) \text{ and } (i,j) \text{ consecutively for } (h,i),(i,j) \in \mathcal{A} \\ 0 \text{ otherwise.} \end{cases}$$

The resulting formulations can be viewed as extended (arc-based) formulations. They do not suffer from the symmetry inherent in formulations involving vehicle-indexed variables, although require $O(n^3)$ binary variables. For the ease of notation, we redefine the travel costs as

$$\bar{c}_{hij} = \begin{cases} c_{hi} & \text{if } j \in \mathcal{C} \\ c_{hi} + c_{i(n+1)} & \text{otherwise,} \end{cases} \tag{1}$$

for all $(h,i),(i,j) \in \mathcal{A}$. Additionally, let $\bar{d}_i = \min\{d_i, Q\}$ be the maximum quantity that can be delivered at customer $i \in C$ in a route. Given a node $i \in \mathcal{N}$, let $\delta_i^- = \{(j,i) \in \mathcal{A}\}$ denote the set of its incoming arcs, $\delta_i^+ = \{(i,j) \in \mathcal{A}\}$ denote the set of its outgoing arcs, and $\delta_i = \delta_i^- \cup \delta_i^+$ denote the set of all its incident arcs. Similarly, for any subset of nodes $\mathcal{S} \subset \mathcal{N}$, let $\delta_{\mathcal{S}}^- = \{(j,i) \in \mathcal{A} \mid i \in \mathcal{S},\ j \notin \mathcal{S}\}$, $\delta_{\mathcal{S}}^+ = \{(i,j) \in \mathcal{A} \mid i \in \mathcal{S},\ j \notin \mathcal{S}\}$ and $\delta_{\mathcal{S}} = \delta_{\mathcal{S}}^- \cup \delta_{\mathcal{S}}^+$. Finally, let $\mathcal{A}(\mathcal{C}) = \{(i,j) \in \mathcal{A} \mid i,j \in \mathcal{C}\}$.

The proposed formulations differ from each other in the way the vehicle load and the route timing are modelled. In all formulations, we assume that the vehicle is empty when it departs from the depot and that its load increases after a customer visit. In this way, the load and time flow variables behave similarly which simplifies models presentation and discussion. As the load flow can be interpreted as the residual capacity in the vehicle, the formulations can model a pickup as well as a delivery setting.

4.1. MTZ Formulation

For the MTZ-based formulation, in addition to the binary x_{hij} variables, we introduce the following continuous decision variables:

$q_{ij} =$ the quantity picked up at customer j on a route that traverses arc (i,j) for $(i,j) \in \mathcal{A}(\mathcal{C})$;
$q^0_{ij} =$ the quantity picked up at customer i on a route that traverses arcs $(0,i)$ and (i,j) consecutively for $(0,i),(i,j) \in \mathcal{A}$;
$y_{ij} =$ the load of a vehicle that traverses arc (i,j) upon arrival at j for $(i,j) \in \mathcal{A}$, $i \in \mathcal{C}$.

Observe that the variables q_{ij} are defined only for arcs between two customers and the variables q^0_{ij} are defined only for arcs out of the depot. Using the defined variables, we propose the following formulation for the SDVRP, referred to hereafter as `SD-MTZ`:

$$\min \quad \sum_{\substack{(h,i)\in\mathcal{A}\\(i,j)\in\mathcal{A}}} \bar{c}_{hij} x_{hij} \tag{2}$$

$$\text{s.t.} \quad \sum_{(h,i)\in\delta_i^-} x_{hij} \le 1, \qquad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{3}$$

$$\sum_{(h,i)\in\delta_i^-} x_{hij} = \sum_{(j,h)\in\delta_j^+} x_{ijh}, \quad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{4}$$

$$\sum_{(j,h)\in\delta_j^+} q^0_{jh} + \sum_{\substack{(i,j)\in\delta_j^-:\\ i\in\mathcal{C}}} q_{ij} = d_j \quad j \in \mathcal{C}, \tag{5}$$

$$q_{ij} \le \bar{d}_j \sum_{(j,h)\in\delta_j^+} x_{ijh}, \qquad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{6}$$

$$q^0_{jh} \le \bar{d}_j x_{0jh}, \qquad (j,h) \in \mathcal{A} : j \in \mathcal{C} \tag{7}$$

$$y_{ij} \ge y_{hi} + q_{hi} + Q(x_{hij} - 1), \ (h,i),(i,j) \in \mathcal{A} : h,i \in \mathcal{C}, \tag{8}$$

$$y_{ij} \ge q^0_{ij} + Q(x_{0ij} - 1), \qquad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{9}$$

$$0 \le y_{ij} \le Q, \qquad (i,j) \in \mathcal{A} : i \in \mathcal{C}, \tag{10}$$

$$q_{ij} \ge 0, \qquad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{11}$$

$$q^0_{ij} \ge 0, \qquad (i,j) \in \mathcal{A} : i \in \mathcal{C}, \tag{12}$$

$$x_{0i(n+1)} \in \mathbb{Z}_+, \qquad i \in \mathcal{C}, \tag{13}$$

$$x_{hij},\ x_{0ij},\ x_{hi(n+1)} \in \{0,1\}, \ (h,i),(i,j) \in \mathcal{A}(\mathcal{C}). \tag{14}$$

The objective function (2) seeks to minimize the total travel cost. Constraints (3) enforce that an arc between two customer nodes is traversed at most once. These constraints can be strengthened by exploiting the fact that if arc (i,j) is used, the reverse arc (j,i) will not be used, i.e.,

$$\sum_{(h,i)\in\delta_i^-} x_{hij} + \sum_{(h,j)\in\delta_j^-} x_{hji} \le 1.$$

This is the form used in our computational experiments. Constraints (4) ensure the vehicle flow balance, i.e., an arc (i,j) is traversed only if there is a vehicle that arrives at i and departs from j. Constraints (5) ensure that the demand of customer $j \in \mathcal{C}$ is met by the pickup quantities from vehicles arriving from the depot (q^0_{jh}) or from other customers (q_{ij}). Constraints (6) and (7) ensure that pickup quantities do not exceed the customer demand and the vehicle capacity. Constraints (8) and (9) are the MTZ constraints that prevent subtours and set the vehicle load along a route. Since the arcs that depart from the depot can be traversed more than once, we need to consider these arcs in separate constraints, (9), to ensure the uniqueness of the load in these arcs. Finally, constraints (10)–(14) specify the domains of the decision variables. Observe that constraints (8), which "force" the value of a variable y_{ij}, are only defined for consecutive arcs (h,i) and (i,j) with both h and i customers. Consequently, they only force the value of variable y_{ij} when (i,j) is traversed as part of a multi-customer route. Thus, a single-customer route visiting i does not force a value of variable $y_{i,n+1}$. This implies that the value of variable $y_{i,n+1}$ is determined by the greatest load among the vehicles that traverse $(i,n+1)$ and visit more than one customer.

Formulation `SD-MTZ` can be extended to model the SDVRPTW. To do so, we introduce continuous variables w_{ij} representing the time that service starts at node i when it is visited by a (multi-customer) route that traverses arc $(i,j) \in \mathcal{A}$. Hence, by adding the following inequalities to `SD-MTZ`, we obtain a formulation for the SDVRPTW:

$$w_{ij} \geq w_{hi} + (s_h + t_{hi})x_{hij} + M_{hi}(x_{hij} - 1), \ (h,i),(i,j) \in \mathcal{A} : i \in \mathcal{C}, \text{ with } h \in \mathcal{C} \text{ or } j \in \mathcal{C}, \tag{15}$$

$$w^a_i \leq w_{ij} \leq \overline{w}^b_{ij}, \qquad (i,j) \in \mathcal{A} : w^a_i \leq \overline{w}^b_{ij}, \tag{16}$$

$$x_{hij} = 0, \qquad (h,i),(i,j) \in \mathcal{A} : i \in \mathcal{C}, \text{ with } w^a_h > \overline{w}^b_{hi} \text{ or } w^a_i > \overline{w}^b_{ij}, \tag{17}$$

where M_{hi} is sufficiently large, e.g., $M_{hi} = \max\{0, w^b_h - w^a_i\}$ for $(h,i) \in \mathcal{A}$, and $\overline{w}^b_{ij} = \min\{w^b_i, w^b_j - (s_i + t_{ij})\}$, for all $(i,j) \in \mathcal{A}$. Constraints (15) ensure the service at customer i begins only after the start of the service at node h plus the service time at h and the travel time between h and i, if there is a route that visits arcs (h,i) and (i,j) consecutively. As with constraints (8) these constraints only affect multi-customer routes. The time feasibility of single-customer routes is ensured by constraints (17). Constraints (16) enforce the time window for the start of the service at each customer i. Since variables w_{ij} are defined for arcs, we can define a tighter time window for customer i using $\overline{w}^b_{ij}$ which also takes the time window of customer j into account. These constraints are only valid if $w^a_i \leq \overline{w}^b_{ij}$, as otherwise the model becomes infeasible. Finally, constraints (17) fix to zero the vehicle flow variables corresponding to a sequence of visits that is time infeasible.

4.2. Commodity-Flow Formulation

Models based on the MTZ constraints are well-known to have weak linear programming relaxations (Gouveia and Pires 1999, Bektas and Gouveia 2014). Hence, we propose an alternative formulation that controls the vehicle load along a route using commodity-flow constraints. In addition to decision variables x_{hij}, q_{ij}, and q^0_{ij}, the model uses the following continuous variable:

$f_{hij} =$ the vehicle load on arc (i,j) for a route that traverses arcs (h,i) and (i,j) consecutively for $(h,i),(i,j) \in \mathcal{A}$, $i \in \mathcal{C}$.

We can ensure that customer demand is served, that vehicle capacity limits are respected, and that there will be no subtours by the following constraints:

$$\sum_{(j,h)\in\delta^+_j} f_{ijh} - \sum_{(h,i)\in\delta^-_i} f_{hij} = q_{ij},\ (i,j) \in \mathcal{A}(\mathcal{C}), \tag{18}$$

$$0 \le f_{hij} \le Q x_{hij}, \qquad (h,i),(i,j) \in \mathcal{A} : h,i \in \mathcal{C}, \tag{19}$$

$$f_{0ij} = q^0_{ij}, \qquad (i,j) \in \mathcal{A} : i \in \mathcal{C}. \tag{20}$$

Constraints (18) ensure the flow balance between consecutive arcs. Constraints (19) enforce vehicle capacity and relate variables f_{hij} and x_{hij} to guarantee that there is a positive load on these arcs only if they belong to the same route. Constraints (20) serve the same purpose for the arcs leaving the depot, and, additionally, handle the demand at the first node served on the route, q^0_{ij}.

By replacing constraints (8)–(10) with (18)–(20) in formulation `SD-MTZ`, we obtain a commodity-flow formulation of the SDVRP, referred to hereafter as `SD-CF`:

$$\min \quad \sum_{\substack{(h,i)\in\mathcal{A}\\(i,j)\in\mathcal{A}}} \bar{c}_{hij} x_{hij} \tag{21}$$

$$\text{s.t.} \quad \sum_{(h,i)\in\delta^-_i} x_{hij} \le 1, \qquad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{22}$$

$$\sum_{(h,i)\in\delta^-_i} x_{hij} = \sum_{(j,h)\in\delta^+_j} x_{ijh}, \quad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{23}$$

$$\sum_{(j,h)\in\delta^+_j} q^0_{jh} + \sum_{\substack{(i,j)\in\delta^-_j:\\ i\in\mathcal{C}}} q_{ij} = d_j \quad j \in \mathcal{C}, \tag{24}$$

$$q_{ij} \le \bar{d}_j \sum_{(j,h)\in\delta^+_j} x_{ijh}, \qquad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{25}$$

$$q^0_{jh} \le \bar{d}_j x_{0jh}, \qquad (j,h) \in \mathcal{A} : j \in \mathcal{C}, \tag{26}$$

$$\sum_{(j,h)\in\delta^+_j} f_{ijh} - \sum_{(h,i)\in\delta^-_i} f_{hij} = q_{ij},\ (i,j) \in \mathcal{A}(\mathcal{C}), \tag{27}$$

$$f_{hij} \le Q x_{hij}, \qquad (h,i),(i,j) \in \mathcal{A} : h,i \in \mathcal{C}, \tag{28}$$

$$f_{0ij} = q^0_{ij}, \qquad (i,j) \in \mathcal{A} : i \in \mathcal{C}, \tag{29}$$

$$f_{hij} \geq 0, \qquad (h,i),(i,j) \in \mathcal{A} : i \in \mathcal{C}, \tag{30}$$

$$q_{ij} \geq 0, \qquad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{31}$$

$$q^0_{ij} \geq 0, \qquad (i,j) \in \mathcal{A} : i \in \mathcal{C}, \tag{32}$$

$$x_{0i(n+1)} \in \mathbb{Z}_+, \qquad i \in \mathcal{C}, \tag{33}$$

$$x_{hij},\ x_{0ij},\ x_{hi(n+1)} \in \{0,1\}, \quad (h,i),(i,j) \in \mathcal{A}(\mathcal{C}). \tag{34}$$

Formulation SD-CF can be extended to model the SDVRPTW as described in the previous section, using constraints (15)–(17). Alternatively, we can consider time as an additional commodity and enforce time windows constraints similar to (18)–(20). To do so, we need to add decision variable

$v_{hij} =$ the start time of service at node h in a route that traverses arcs (h,i) and (i,j) consecutively for $(h,i),(i,j) \in \mathcal{A}$, $i \in \mathcal{C}$,

and add the following commodity flow constraints

$$\sum_{(j,h)\in\delta^+_j} v_{ijh} - \sum_{(h,i)\in\delta^-_i} v_{hij} \geq \sum_{(h,i)\in\delta^-_i} (s_h + t_{hi}) x_{hij},\ (i,j) \in \mathcal{A}(\mathcal{C}), \tag{35}$$

$$w^a_h x_{hij} \leq v_{hij} \leq \overline{w}^b_{hi} x_{hij}, \qquad (h,i),(i,j) \in \mathcal{A} : i \in \mathcal{C}, \tag{36}$$

$$v_{hi(n+1)} + (s_h + t_{hi}) x_{hi(n+1)} \leq \overline{w}^b_{i(n+1)} x_{hi(n+1)}, \quad (h,i) \in \mathcal{A} : i \in \mathcal{C}, \tag{37}$$

where $\overline{w}^b_{ij}$ is set as defined before at the end of Section 4.1. Constraints (35) ensure the time flow balance between consecutive arcs, imposing that the service at customer i cannot start earlier than the start of service at customer h plus the related service and travel times in a route that traverses arcs (h,i) and (i,j) consecutively. Only one route can traverse this sequence of arcs and thus the right-hand side corresponds to the service and travel times associated with at most one node h. Note that these constraints are defined only when both i and j are customer nodes and hence they do not affect single-customer routes. Constraints (36) enforce the time window at node h for a route that traverses arcs (h,i) and (i,j) consecutively, considering also the time window at node i. Constraints (37) ensure that the time window is respected at the depot node $n+1$. These constraints account for the service and travel times related to nodes h and i, since $(i,n+1)$ is not covered by constraints (35). Finally, note that constraints (36) and (37) imply in $x_{hij} = 0$ if the corresponding sequence of arcs is time infeasible, thus serving a similar purpose to constraints (17) used in formulation SD-MTZ. For example, if $w^a_h > \overline{w}^b_{hi}$ for a given pair of arcs $(h,i),(i,j) \in \mathcal{A}$ with $i \in \mathcal{C}$, then constraints (36) imply in $x_{hij} = 0$. Even though redundant, adding constraints (17) to SD-CF may improve the performance of general-purpose integer-programming solvers.

4.3. Alternative Commodity-Flow Formulation

Next, we exploit the regularity property to simplify model SD-CF as follows. Recall that we assume the vehicle load increases along the route. Thus, a route satisfies the regularity property if and only if the vehicle load flow on an arc (j,k) that immediately follows an arc (i,j) in the route is greater than or equal to the vehicle load flow on arc (i,j). Let f_{ij} represent the load on arc $(i,j) \in \mathcal{A}$. When $i,j \in \mathcal{C}$, we may assume that arc (i,j) is traversed by at most one route, and, thus, f_{ij} represents the load in the vehicle of that route. For $j = n+1$, f_{ij} represents the sum of the loads in the vehicles assigned to the routes that traverse (i,j). Additionally, we assume the vehicle is empty when it departs from the depot and, hence, $f_{ij} = 0$ for $i = 0$. We can relate variables f_{ij} with arc flow variables x_{hij} and enforce the initial vehicle load using the constraints

$$f_{ij} \leq Q \sum_{(h,i)\in\delta_i^-} x_{hij},\ (i,j) \in \mathcal{A} : i \in \mathcal{C}, \tag{38}$$

$$f_{0j} = 0, \qquad j \in \mathcal{C}, \tag{39}$$

and thus we can guarantee the regularity property by the following constraints

$$f_{ij} \geq f_{hi} + Q(x_{hij} - 1),\ (h,i),(i,j) \in \mathcal{A} : i,j \in \mathcal{C}. \tag{40}$$

Since satisfying the regularity property implies that a solution to the (two-index) single commodity-formulation is SDVRP feasible, we derive the following exact formulation for the SDVRP, referred to as SD-CFRP:

$$\min \quad \sum_{\substack{(h,i)\in\mathcal{A}\\(i,j)\in\mathcal{A}}} \bar{c}_{hij} x_{hij} \tag{41}$$

$$\text{s.t.} \quad \sum_{(h,i)\in\delta_i^-} x_{hij} \leq 1, \qquad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{42}$$

$$\sum_{(h,i)\in\delta_i^-} x_{hij} = \sum_{(j,h)\in\delta_j^+} x_{ijh}, \quad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{43}$$

$$\sum_{(i,j)\in\delta_i^+} f_{ij} - \sum_{(h,i)\in\delta_i^-} f_{hi} = d_i, \quad i \in \mathcal{C}, \tag{44}$$

$$f_{ij} \leq Q \sum_{(h,i)\in\delta_i^-} x_{hij}, \qquad (i,j) \in \mathcal{A} : i \in \mathcal{C}, \tag{45}$$

$$f_{ij} \geq f_{hi} + Q(x_{hij} - 1), \quad (h,i),(i,j) \in \mathcal{A} : i,j \in \mathcal{C}, \tag{46}$$

$$f_{0j} = 0, \qquad j \in \mathcal{C}, \tag{47}$$

$$f_{ij} \geq 0, \qquad (i,j) \in \mathcal{A}, \tag{48}$$

$$x_{0i(n+1)} \in \mathbb{Z}_+, \qquad i \in \mathcal{C}, \tag{49}$$

$$x_{hij},\ x_{0ij},\ x_{hi(n+1)} \in \{0,1\},\ (h,i),(i,j) \in \mathcal{A}(\mathcal{C}). \tag{50}$$

We can extend this formulation to the SDVRPTW by using a similar idea to guarantee the correct time flow. Let v_{ij} represent the starting time of service at node i in the route that traverses arc $(i,j) \in \mathcal{A}$. To model the time flow through the network and ensure that the customer time windows are satisfied, we can impose:

$$\sum_{(i,j)\in\delta_i^+} v_{ij} - \sum_{(h,i)\in\delta_i^-} v_{hi} \geq \sum_{(h,i)\in\delta_i^-} \sum_{(i,j)\in\delta_i^+} (t_{hi} + s_h) x_{hij}, \;\; i \in \mathcal{C}, \tag{51}$$

$$\sum_{(h,i)\in\delta_i^-} w_i^a x_{hij} \leq v_{ij} \leq \sum_{(h,i)\in\delta_i^-} \overline{w}_{ij}^b x_{hij}, \qquad (i,j) \in \mathcal{A} : i \in \mathcal{C}, \tag{52}$$

$$\sum_{(i,j)\in\delta_i^+} w_h^a x_{hij} \leq v_{hi} \leq \sum_{(i,j)\in\delta_i^+} \overline{w}_{hi}^b x_{hij}, \qquad (h,i) \in \mathcal{A} : i \in \mathcal{C}, \tag{53}$$

where $\overline{w}_{ij}^b$ is set as defined in Section 4.1. We can fix $v_{0i} = w_0^a$ in the model without loss of optimality. Constraints (51) ensure the time flow balance between the incoming and outgoing arcs at each customer i, while constraints (52) and (53) enforce the time windows at nodes. Observe that constraints (53) are necessary only for arcs $(h,i) \in \mathcal{A}$ with $h = 0$, but by defining them as presented we relate the vehicle flow variables x_{hij} also with the time flow variables v_{hi}.

In addition to constraints (51)–(53), we can guarantee that the regularity property is satisfied by adding the following constraints:

$$v_{ij} \geq v_{hi} + (s_h + t_{hi}) x_{hij} + M_{hi}(x_{hij} - 1), \; (h,i),(i,j) \in \mathcal{A} : i,j \in \mathcal{C}, \tag{54}$$

where M_{hi} is a sufficiently large value that can be set as defined at the end of Section 4.1. Constraints (54) are similar to (40) used for load flow, but must further include service and travel times. These constraints are, in fact, the same as the MTZ-based constraints (15). Alternatively, instead of (51)–(54), we can rely on constraints (35)–(37) based on the three-index time flow variables v_{hij}, or even on the MTZ-based constraints (15)–(17).

4.4. Analysis

Let $\mathcal{LR}(\texttt{F})$ denote the linear relaxation of a given formulation F.

Theorem 1. $\mathcal{LR}(\texttt{SD-CF}) \subseteq \mathcal{LR}(\texttt{SD-CFRP})$ and $\mathcal{LR}(\texttt{SD-CF}) \subsetneq \mathcal{LR}(\texttt{SD-CFRP})$ when $d_j < Q$ for at least one customer $j \in \mathcal{C}$.

Proof. Given a feasible solution $(\bar{x}, \bar{f}, \bar{q}^0, \bar{q})$ of $\mathcal{LR}(\texttt{SD-CF})$, let $(\hat{x}, \hat{f})$ with $\hat{x} \in \mathbb{R}^{n^3}$ and $\hat{f} \in \mathbb{R}^{n^2}$, be defined as follows:

$$\hat{x}_{hij} := \bar{x}_{hij}, \qquad (h,i),(i,j) \in \mathcal{A},$$

$$\hat{f}_{ij} := \sum_{(\ell,i)\in\delta_i^-} \bar{f}_{\ell ij}, \; (i,j) \in \mathcal{A} : i \in \mathcal{C},$$

$$\hat{f}_{0j} := 0, \qquad j \in \mathcal{C}.$$

We show now that $(\hat{x}, \hat{f})$ is a feasible solution of $\mathcal{LR}(\texttt{SD-CFRP})$. First, notice it has the same objective value as $\mathcal{LR}(\texttt{SD-CF})$. Constraints (42), (43) and (47) are trivially satisfied, while (23) and (28) imply that $(\hat{x}, \hat{f})$ satisfies also (45). From (27) and (24) we have that, for each $i \in \mathcal{C}$,

$$\begin{aligned}
\sum_{(i,j)\in\delta_i^+} \hat{f}_{ij} - \sum_{(h,i)\in\delta_i^-} \hat{f}_{hi} &= \sum_{(i,j)\in\delta_i^+} \sum_{(\ell,i)\in\delta_i^-} \bar{f}_{\ell ij} - \sum_{\substack{(h,i)\in\delta_i^- \\ h\in\mathcal{C}}} \sum_{(\ell,h)\in\delta_h^-} \bar{f}_{\ell hi} \\
&= \sum_{(i,j)\in\delta_i^+} \sum_{(\ell,i)\in\delta_i^-} \bar{f}_{\ell ij} + \sum_{\substack{(h,i)\in\delta_i^- \\ h\in\mathcal{C}}} \left(\bar{q}_{hi} - \sum_{(i,\ell)\in\delta_i^+} \bar{f}_{hi\ell} \right) \\
&= \sum_{\substack{(\ell,i)\in\delta_i^- \\ \ell\in\mathcal{C}}} \sum_{(i,j)\in\delta_i^+} \bar{f}_{\ell ij} + \sum_{(i,j)\in\delta_i^+} \bar{f}_{0ij} + \sum_{\substack{(h,i)\in\delta_i^- \\ h\in\mathcal{C}}} \bar{q}_{hi} - \sum_{\substack{(h,i)\in\delta_i^- \\ h\in\mathcal{C}}} \sum_{(i,\ell)\in\delta_i^+} \bar{f}_{hi\ell} \\
&= \sum_{(i,j)\in\delta_i^+} \bar{q}^0_{ij} + \sum_{\substack{(h,i)\in\delta_i^- \\ h\in\mathcal{C}}} \bar{q}_{hi} \\
&= d_i
\end{aligned}$$

and hence (44) holds. From (22) and (28) we have that

$$\sum_{(i,\ell)\in\delta_i^+} Q\bar{x}_{hi\ell} \le Q \;\Rightarrow Q\bar{x}_{hij} + \sum_{\substack{(i,\ell)\in\delta_i^+ \\ \ell\neq j}} Q\bar{x}_{hi\ell} \le Q \;\Rightarrow Q\bar{x}_{hij} + \sum_{\substack{(i,\ell)\in\delta_i^+ \\ \ell\neq j}} \bar{f}_{hi\ell} \le Q,$$

for any $(h,i),(i,j) \in \mathcal{A}$ with $i,j \in \mathcal{C}$. Furthermore, since $(\bar{x}, \bar{f}, \bar{q}^0, \bar{q}) \ge 0$, we have

$$\sum_{\substack{(\ell,i)\in\delta_i^- \\ \ell\neq h}} \bar{f}_{\ell ij} + \bar{q}_{hi} \ge 0,$$

which together with (27) leads to

$$\begin{aligned}
\sum_{\substack{(\ell,i)\in\delta_i^- \\ \ell\neq h}} \bar{f}_{\ell ij} &\ge Q - Q - \bar{q}_{hi} \\
&\ge \sum_{\substack{(i,\ell)\in\delta_i^+ \\ \ell\neq j}} \bar{f}_{hi\ell} + Q\bar{x}_{hij} - Q - \bar{q}_{hi} + \bar{f}_{hij} - \bar{f}_{hij} \\
&\ge \sum_{(\ell,h)\in\delta_h^-} \bar{f}_{\ell hi} + Q\bar{x}_{hij} - Q - \bar{f}_{hij}
\end{aligned}$$

and hence $\hat{f}_{ij} \ge \hat{f}_{hi} + Q\bar{x}_{hij} - Q$ for all $(h,i),(i,j) \in \mathcal{A}$ with $i,j \in \mathcal{C}$, showing that (46) holds. Therefore, any feasible solution of $\mathcal{LR}(\texttt{SD-CF})$ is also feasible for $\mathcal{LR}(\texttt{SD-CFRP})$.

Now, consider $(\hat{x}, \hat{f})$ defined as follows:

$$\hat{x}_{hij} = \begin{cases} d_i/Q, \text{ if } i \in \mathcal{C}, h = 0 \text{ and } j = n+1, \\ 0, \quad \text{otherwise;} \end{cases} \quad \text{and} \quad \hat{f}_{ij} = \begin{cases} d_i, \text{ if } i \in \mathcal{C}, j = n+1, \\ 0, \text{ otherwise.} \end{cases}$$

It is easy to verify that $(\hat{x}, \hat{f})$ satisfies (42) – (48) and hence is a feasible solution for $\mathcal{LR}(\texttt{SD-CFRP})$. The corresponding solution $(\bar{x}, \bar{f}, \bar{q}, \bar{q}^0)$ in $\mathcal{LR}(\texttt{SD-CF})$, using $\bar{x}_{hij} := \hat{x}_{hij}$, must have $\bar{q}_{ij} = 0$ because of constraints (25), and, therefore, $\bar{q}^0_{i,n+1} = d_i$ because of constraints (24). Because $\bar{d}_i \bar{x}_{0,i,n+1} = \min\{d_i, Q\} d_i/Q$, constraints (26) are satisfied if and only if $d_i = Q$ for $i \in \mathcal{C}$. Thus, when $d_i < Q$ for at least one customer $i \in \mathcal{C}$, we have that $(\bar{x}, \bar{f}, \bar{q}, \bar{q}^0) \notin \mathcal{LR}(\texttt{SD-CF})$. □

Consequently, the bound provided by the value of an optimal solution to the linear programming relaxation of SD-CF is as good as or better than the bound provided by the value of an optimal solution to the linear programming relaxation of SD-CFRP.

5. A Branch-and-Cut Algorithm

Special-purpose BC algorithms based on relaxed two-index formulations represent the current state-of-the-art for solving the SDVRP and SDVRPTW (Archetti, Bianchessi, and Speranza 2014, Bianchessi and Irnich 2019). In addition to using (feasibility) cuts to eliminate optimal solutions to the relaxed formulation that are infeasible, these algorithms use various classes of valid inequalities to strengthen the linear programming relaxations of the relaxed formulations, e.g., capacity and connectivity cuts. Since the number of variables and constraints in these relaxed formulations is smaller than in the new compact exact formulations proposed in Section 4, they have certain computational advantages in a BC algorithm. On the other hand, the proposed formulations enforce the regularity property in a natural way without resorting to vehicle-indexed variables. Therefore, in this section, we propose a BC algorithm that exploits both features. For the sake of completeness, we first present the relaxed, single commodity-flow formulation of the SDVRP (Archetti, Bianchessi, and Speranza 2014):

$$\min \quad \sum_{(i,j)\in\mathcal{A}} c_{ij} x_{ij} \tag{55}$$

$$\text{s.t.} \quad \sum_{(i,j)\in\delta_i^+} x_{ij} = z_i, \qquad i \in \mathcal{C} \cup \{0\}, \tag{56}$$

$$\sum_{(i,j)\in\delta_i^+} x_{ij} = \sum_{(j,i)\in\delta_i^-} x_{ji}, \qquad i \in \mathcal{C}, \tag{57}$$

$$\sum_{(i,j)\in\delta_i^+} f_{ij} = \sum_{(j,i)\in\delta_i^-} f_{ji} + d_i, \quad i \in \mathcal{C}, \tag{58}$$

$$f_{0i} = 0, \qquad i \in \mathcal{N}, \tag{59}$$

$$f_{ij} \le Q x_{ij}, \qquad (i,j) \in \mathcal{A}, \tag{60}$$

$$\sum_{i\in\mathcal{C}} z_i \le n + m - 1, \tag{61}$$

$$\overline{m} \le z_0 \le m, \tag{62}$$

$$1 \le z_i \le m, \qquad i \in \mathcal{C}, \tag{63}$$

$$f_{ij} \geq 0, \qquad (i,j) \in \mathcal{A}, \tag{64}$$

$$z_i \in \mathbb{Z}_+, \qquad i \in \mathcal{N}, \tag{65}$$

$$x_{0i}, x_{i(n+1)} \in \mathbb{Z}_+, \qquad i \in \mathcal{C}, \tag{66}$$

$$x_{ij} \in \{0,1\}, \qquad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{67}$$

where x_{ij} represents the number of times arc (i,j) is traversed in the solution; z_i represents the number of visits to node i; and f_{ij} is the (total) load of the vehicles that traverse arc (i,j). We will refer to x_{ij} as vehicle flow variables, similar to variables x_{hij} in Section 4. This should not cause confusion as they have a different number of indices. The objective function (55) seeks to minimize the travel cost. Constraints (56) relate variables x_{ij} and z_i. Constraints (57) enforce vehicle flow balance, while constraints (58) enforce load flow balance accounting for customer demand. (As before, we take a pickup perspective when modeling vehicle load; vehicles depart empty from the depot – enforced by constraints (59)). Constraints (60) partially enforce vehicle capacity and allow a positive load flow on arc (i,j) only if there is a vehicle that traverses the arc. Constraints (61)–(63) enforce lower and upper bounds on the number of customer visits, where m is the number of vehicles available at the depot and $\overline{m} = \lceil \sum_{i \in \mathcal{C}} d_i/Q \rceil$ is the minimum number of vehicles needed to meet customer demands. Finally, constraints (64)–(67) specify the domains of the variables. Similar to Archetti, Bianchessi, and Speranza (2014) and Bianchessi and Irnich (2019), we include the redundant pairing inequalities $x_{ij} + x_{ji} \leq 1$, $\forall (i,j) \in \mathcal{A}(\mathcal{C})$ in the formulation as they strengthen its linear relaxation.

The proposed BC algorithm differs in the way it handles integer solutions to the relaxed formulation that are not feasible for the actual problem. Rather than using feasibility cuts to eliminate such solutions, our proposed BC algorithm uses constraints that enforce the regularity property, similar to those in `SD-CFRP`, to eliminate such solutions. That is, we locally change to an extended formulation.

For the SDVRPTW, we add the following constraints to the relaxed formulation to partially enforce the time windows:

$$\sum_{(i,j) \in \delta_i^+} v_{ij} \geq \sum_{(j,i) \in \delta_i^-} v_{ji} + \sum_{(j,i) \in \delta_i^-} (s_j + t_{ji}) x_{ji}, \quad i \in \mathcal{C}, \tag{68}$$

$$w_i^a x_{ij} \leq v_{ij} \leq \overline{w}_{ij}^b x_{ij}, \qquad (i,j) \in \mathcal{A}, \tag{69}$$

$$v_{0i} = w_0^a, \qquad i \in \mathcal{C}, \tag{70}$$

$$v_{ij} \geq 0, \qquad (i,j) \in \mathcal{A}. \tag{71}$$

Constraints (68) impose that the total time flow in the routes departing from node i is equal to the total time flow arriving at node i plus the service and travel times corresponding to nodes that

immediately precede node i in the routes. Constraints (69) enforce the time windows at the nodes. However, a solution to the relaxed model (with time flows satisfying the time windows) may not correspond to a time feasible solution to the SDVRPTW because the relaxed model allows vehicles to "transfer" time at split nodes (similar to the transfer of load at split nodes). Finally, constraints (70) fix the start of the service at the depot, and constraints (71) impose the nonnegativity of the decision variables. Bianchessi and Irnich (2019) do not include load flow constraints (58)–(60) in their formulation, but only time flow constraints similar to (68)–(71). As a consequence, to obtain feasible solutions for the SDVRPTW, Bianchessi and Irnich (2019) resort to a path-based model defined on a graph induced by a feasible solution to their relaxed formulation. Another difference to their approach is that we extend the `MatchArcs` procedure for separating feasibility cuts to the time windows variant and, thus, can identify an infeasible solution in polynomial time for both SDVRP and SDVRPTW.

Further details on handling feasibility as well as on algorithmic enhancements will be provided next.

5.1. Handling Feasibility

For the SDVRP, similarly to Archetti, Bianchessi, and Speranza (2014) and Ozbaygin, Karasan, and Yaman (2018), we check the feasibility of a candidate incumbent using variables f_{ij} and the `MatchArcs` procedure, which verifies the regularity property in polynomial time. If the candidate incumbent is feasible, then it is accepted as incumbent and the BC method continues. Otherwise, we use the infeasible solution to locally extend the formulation as follows. Given a solution $(\bar{x}, \bar{z}, \bar{f})$ to formulation (55)–(67), let $i \in \mathcal{C}$ such that $z_i > 1$ and for which the regularity property is not satisfied, i.e., it is impossible to find an assignment of loads to vehicles such that for every vehicle the load entering node i is less than or equal to the load leaving node i (it is impossible to find combinations $h-i-j$ such that $\bar{f}_{hi} \leq \bar{f}_{ij}$ for the arcs with $\bar{x}_{hi} > 0$ and $\bar{x}_{ij} > 0$). Then we add the following set of constraints to the relaxed model:

$$x_{ij} = \sum_{(h,i)\in\delta_i^-} x_{hij}, \qquad j \in \delta_i^+ : \bar{x}_{ij} > 0, \tag{72}$$

$$x_{hi} = \sum_{(i,j)\in\delta_i^+} x_{hij}, \qquad h \in \delta_i^- : \bar{x}_{hi} > 0, \text{ and} \tag{73}$$

$$f_{ij} \geq f_{hi} - Q(1 - x_{hij}), \;\; j \in \delta_i^+, \; h \in \delta_i^- : \bar{x}_{ij}, \bar{x}_{hi} > 0. \tag{74}$$

These constraints are derived from (43) and (45)–(46) and cut-off $(\bar{x}, \bar{z}, \bar{f})$ as they enforce the regularity property at node i.

This strategy can be used for the SDVRPTW as well when we adapt `MatchArcs` to verify both load and time flow feasibility. This has the added benefit that we can check if a solution to the

relaxed model is infeasible in polynomial time. Given a solution $(\bar{x}, \bar{z}, \bar{f}, \bar{v})$ of the relaxed model (55)–(71), the procedure first verifies the feasibility of the time flow using the value of $\bar{v}_{ij}$. If for a given node $i \in \mathcal{C}$ such that $z_i > 1$, there is one pair of incoming and outgoing arcs (h,i) and (i,j), with $\bar{x}_{hi} > 0$ and $\bar{x}_{ij} > 0$, for which the procedure cannot find a feasible time flow, then we add the following set of constraints to the relaxed model and terminate the procedure:

$$x_{ij} = \sum_{(h,i) \in \delta_i^-} x_{hij}, \qquad j \in \delta_i^+ : \bar{x}_{ij} > 0, \tag{75}$$

$$x_{hi} = \sum_{(i,j) \in \delta_i^+} x_{hij}, \qquad h \in \delta_i^- : \bar{x}_{hi} > 0, \tag{76}$$

$$v_{ij} \geq v_{hi} + (s_h + t_{hi}) x_{hij} - M_{hi}(1 - x_{hij}),\ j \in \delta_i^+,\ h \in \delta_i^- : \bar{x}_{ij}, \bar{x}_{hi} > 0. \tag{77}$$

Otherwise, if it was possible to successfully pair all incoming and outgoing arcs (with positive vehicle flow) in a way to guarantee a feasible time flow, then the procedure verifies the load flow based on the values of $\bar{f}_{ij}$ as described in the previous paragraph, but using the same pairing of arcs as defined for the time flow. If this is not feasible, we add constraints (72)–(74) and terminate the procedure.

The complete procedure for checking for violation of the constraints that ensure the regularity property for the SDVRPTW is described in Algorithm 1. Similarly to the notation used in the `MatchArcs` procedure of Archetti, Bianchessi, and Speranza (2014), $A^+(i)$ denotes the list of outgoing arcs $(i,j) \in \mathcal{A}$ such that $\bar{x}_{ij} > 0$; $A^-(i)$ denotes the list of incoming arcs $(h,i) \in \mathcal{A}$ such that $\bar{x}_{hi} > 0$; and $\mathcal{I}_{ij}$ denotes the incoming arc that is paired with the outgoing arc (i,j).

If no constraints are generated by Algorithm 1, then $(\bar{x}, \bar{z}, \bar{f}, \bar{v})$ satisfies the regularity property and is a feasible solution for the SDVRPTW. Note that the proposed algorithm always identifies a solution that is infeasible for the SDVRPTW, as no feasible pairing of arcs is possible in such case. However, differently from `MatchArcs`, the algorithm may fail to find a feasible matching of arcs when the solution is feasible, as it first pairs the arcs based on the time flow, and then uses this same pairing to verify the load flow. This does not invalidate the procedure, but may result in the addition of more cuts than needed. It is worth mentioning that, since we check the values of variables x_{hij} already inserted into the model (see line 4 of Algorithm 1), the sets of constraints (72)–(74) and (75)–(77) are never added more than once.

5.2. Algorithmic Enhancements

Locally extending the formulation, i.e., adding constraints that enforce the regularity property, suffices to guarantee that the proposed BC algorithm converges to an optimal solution. We can speed up its convergence by additional, yet simple algorithmic enhancements. First, to strengthen the linear programming relaxation, we add capacity cuts (using the separation routines provided

Algorithm 1: Separation for constraints ensuring the regularity property for SDVRPTW

Input: Solution $(\bar{x}, \bar{z}, \bar{f}, \bar{v})$ to the relaxed formulation (55)–(71).

Output: New violated constraints or a feasible matching of arcs.

foreach $i \in \mathcal{C}$ such that $\bar{z}_i > 1$ **do**
  $A^+(i) \leftarrow \varnothing$;
  $A^-(i) \leftarrow \varnothing$;
  if there are $(h,i),(i,j) \in \mathcal{A}$ such that x_{hij} is in the model and $x_{hij} = 1$ **then**
    Insert (i,j) into $A^+(i)$ and (h,i) into $A^-(i)$;
  else
    foreach $(i,j) \in \mathcal{A}$ such that $\bar{x}_{ij} > 0$ **do** insert (i,j) into $A^+(i)$, $\bar{x}_{ij}$ times;
    foreach $(h,i) \in \mathcal{A}$ such that $\bar{x}_{hi} > 0$ **do** insert (h,i) into $A^-(i)$, $\bar{x}_{hi}$ times;
  Sort the arcs in $A^+(i)$ in non-ascending order of $\bar{v}^+_{ij}$, where $\bar{v}^+_{ij} \leftarrow \bar{v}_{ij}/\bar{x}_{ij}$ if $j < n+1$, and $\bar{v}^+_{i(n+1)} \leftarrow w^b_{n+1}$;
  Sort the arcs in $A^-(i)$ in non-ascending order of $\bar{v}^-_{hi} \leftarrow (\bar{v}_{hi} + (s_h + t_{hi})\bar{x}_{hi})/\bar{x}_{hi}$;
  foreach $(i,j) \in A^+(i)$ **do**
    Get the first arc in $A^-(i)$ such that $\bar{v}^-_{hi} \leq \bar{v}^+_{ij}$;
    if no such arc exists **then**
      Add constraints (75)–(77) to the relaxed model and **stop**.
    $\mathcal{I}_{ij} \leftarrow (h,i)$;
  foreach $(i,j) \in A^+(i)$ **do**
    $(h,i) \leftarrow \mathcal{I}_{ij}$;
    if $f_{hi} > f_{ij}$ **then** add constraints (72)–(74) to the relaxed model and **stop**.

in the CVRPSep library (Lysgaard, Letchford, and Eglese 2004)) and connectivity cuts, presented in both Archetti, Bianchessi, and Speranza (2014) and Bianchessi and Irnich (2019):

$$\sum_{(i,j) \in \delta^+_{\mathcal{S}}} x_{ij} \geq z_k, \quad \forall \mathcal{S} \subseteq \mathcal{C},\ |\mathcal{S}| \geq 2,\ k \in \mathcal{S}. \tag{78}$$

Every time a candidate incumbent is found, and before checking if it satisfies the regularity property, we check if it satisfies these valid inequalities, and, if not, add any violated cuts and discard the incumbent solution. We adopted this strategy because these cuts are simpler and have less impact on the linear programming relaxation than the regularity property constraints.

If a candidate incumbent does not satisfy the regularity property, we use it to define a residual optimization problem that seeks to find a feasible solution. More specifically, we use the compact model `SD-CFRP` proposed in Section 4.3 modified as follows: for each $i,j \in \mathcal{C}$ such that $\bar{x}_{ij} = 0$ in the candidate incumbent with vehicle flow $\bar{x}$, we set $x_{hij} = x_{ijh} = 0$, for all $h \in \mathcal{N}$, and we add a constraint that forces the objective function value to be less than or equal to the value of the

(actual) incumbent. Thus, we look for a feasible solution that uses the same arcs between customers as the candidate incumbent (or a subset of them) and has an objective value not greater than the current upper bound. In many cases, when a candidate incumbent is rejected, it does not satisfy the regularity property, i.e., the load and/or time flows are infeasible, but does have feasible vehicle flow. (Often, a candidate incumbent is identified by a general-purpose heuristic embedded in the integer programming solver.) Note that the vehicle flow in the arcs incident to the depot nodes is not limited in the residual problem, thus single-customer routes and new departures/returns from/to the depot can be created in the solution to the residual problem. If there exists a feasible solution to the residual problem with a value that strictly improves upon the value of the (actual) incumbent, then it will be set as the (new) incumbent. Otherwise, if the residual problem is infeasible, then we add the following *feasibility* cut:

$$\sum_{\substack{(i,j)\in\mathcal{A}:\\ \bar{x}_{ij}>0}} x_{ij} \leq \sum_{i\in\mathcal{C}\cup\{0\}} z_i - 1. \tag{79}$$

This cut only involves arcs with positive vehicle flow in the solution to the relaxed model and hence it is often sparser than the feasibility cut used by Archetti, Bianchessi, and Speranza (2014) and Bianchessi and Irnich (2019), which is based on arcs without flow. If the incumbent is not optimal, then there is at least one optimal solution to the SDVRP that satisfies (79).

Theorem 2. Given a feasible solution $(\bar{x}, \bar{z}, \bar{f})$ to the relaxed formulation (55)–(67), if the residual optimization problem based on the `SD-CFRP` is infeasible, then there is at least one optimal solution to the SDVRP that satisfies (79).

Proof. Let $(\bar{x}, \bar{z}, \bar{f})$ be a feasible solution to the relaxed formulation (55)–(67) such that the residual optimization problem induced by vehicle flow $\bar{x}$ is infeasible. Based on this solution, we define the sets $\bar{A} = \{(i,j) : \bar{x}_{ij} > 0\}$, $\bar{A}(\mathcal{C}) = \{(i,j) \in A(\mathcal{C}) : \bar{x}_{ij} > 0\}$ and $\bar{F} = \{(i,j) \in A(\mathcal{C}) : \bar{x}_{ij} = 0\}$. Set $\bar{F}$ represents the forbidden arcs in the residual problem, i.e., we fix at zero all variables related to arcs $(i,j) \in \bar{F}$. All the remaining variables are left unchanged in the residual problem. From (56), we have that any solution to the relaxed formulation satisfies

$$\sum_{(i,j)\in A} x_{ij} = \sum_{i\in\mathcal{C}\cup\{0\}} \sum_{(i,j)\in\delta_i^+} x_{ij} = \sum_{i\in\mathcal{C}\cup\{0\}} z_i. \tag{80}$$

Hence, using the notation defined above, we obtain

$$\sum_{(i,j)\in\bar{A}} x_{ij} = \sum_{i\in\mathcal{C}\cup\{0\}} z_i - \sum_{\substack{j\in\mathcal{C}:\\ \bar{x}_{0j}=0}} x_{0j} - \sum_{\substack{i\in\mathcal{C}:\\ \bar{x}_{i(n+1)}=0}} x_{i(n+1)} - \sum_{(i,j)\in\bar{F}} x_{ij}. \tag{81}$$

Since the residual problem induced by $\bar{x}$ is infeasible, and we allowed additional flow on any arc incident to the depots, we can only obtain a feasible SDVRP solution that is better than the incumbent (if any exists) if we allow one additional unit of flow in at least one arc $(i,j) \in \bar{F}$, i.e.:

$$\sum_{(i,j)\in\bar{F}} x_{ij} \geq 1. \tag{82}$$

From (82) and (81), we obtain the inequalities

$$\sum_{\substack{(i,j)\in A:\\ \bar{x}_{ij}>0}} x_{ij} \;\leq\; \sum_{i\in\mathcal{C}\cup\{0\}} z_i - \sum_{\substack{j\in\mathcal{C}:\\ \bar{x}_{0j}=0}} x_{0j} - \sum_{\substack{i\in\mathcal{C}:\\ \bar{x}_{i(n+1)}=0}} x_{i(n+1)} - 1 \;\leq\; \sum_{i\in\mathcal{C}\cup\{0\}} z_i - 1 \tag{83}$$

which completes de proof. □

Additionally, when the residual problem has an optimal solution and the value of that solution, θ_{res}, is strictly greater than the value of the candidate incumbent, then we can add the following *optimality* cut:

$$\sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} \geq \theta_{res}\left(\sum_{\substack{(i,j)\in\mathcal{A}:\\ \bar{x}_{ij}>0}} x_{ij} - \sum_{i\in\mathcal{C}\cup\{0\}} z_i + 1\right). \tag{84}$$

Note that θ_{res} being strictly greater than value of $\bar{x}$ in the relaxed model means that the candidate incumbent is infeasible for the SDVRP (as otherwise θ_{res} would be equal to the value of the candidate incumbent). Hence, the optimal solution of the residual problem is an alternative solution that uses the same residual network, but is feasible for the SDVRP (e.g. by adding a new single-customer route). In this way, cut (84) helps to "correct" the (wrong) assessment of the value of the (infeasible) candidate incumbent in the relaxed formulation, instead of cutting-off this solution using the feasibility cut (79). It is a disjunctive constraint that is considered only when the solution (regarding the vehicle flow) uses at least the same arcs as the residual network defined by a previous candidate solution.

The flowchart in Fig. 2 illustrates the steps taken when a new candidate incumbent is found.

6. Computational Study

In this section, we present and discuss the results of a set of computational experiments with the proposed compact models and the proposed BC algorithm. The goal of the computational study is (*i*) to show that the proposed compact models can be solved effectively by a general-purpose integer programming solver and that these models outperform previously proposed compact models, (*ii*) to demonstrate the efficacy of the proposed BC algorithm (thus establishing that locally extending formulations can be effective) and compare its performance to other BC algorithms, and (*iii*) to find new optimal solutions and improved lower and upper bounds for instances of the SDVRP and SDVRPTW.

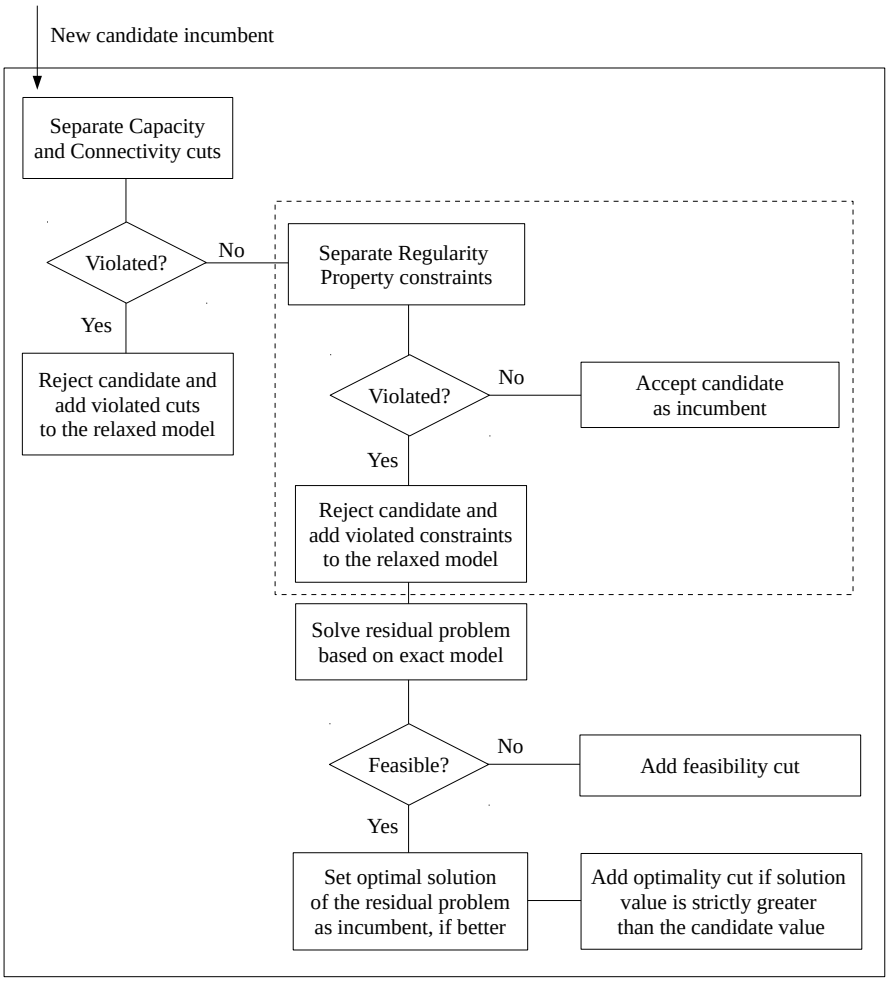


Figure 2 Flowchart of steps applied every time a candidate incumbent solution is found.

For the SDVRP, we use the same benchmark instances as Archetti, Bianchessi, and Speranza (2014). There are four sets of instances. Sets 1 and 2 have 11 and 14 instances, respectively, and were proposed by Belenguer, Martinez, and Mota (2000) based on instances from the TSPLIB repository. Set 3 has 18 instances and was introduced in Archetti, Speranza, and Hertz (2006) and Set 4 has 13 instances and was introduced in Chen, Golden, and Wasil (2007). Each instance specifies the coordinates of the depot and the coordinates of each of the customers. We consider limited- and unlimited-fleet variants of the SDVRP. For the limited-fleet variant, commonly referred to as SDVRP-LF, we set $m = \overline{m}$, i.e., the fleet size is equal to the minimum number of vehicles needed to meet customer demands. For the unlimited-fleet variant, commonly referred to as SDVRP-UF, we set $m = n$. Distances are taken to be the Euclidean distances, and travel times and costs are set equal to the distances. Similar to Archetti, Bianchessi, and Speranza (2014), we consider the original instances, but also modified instances with integer distances. To obtain integer distances, the distances in the original instances are first rounded to the nearest integer. Then, to ensure that the triangle inequality holds, the distances are replaced by the lengths of the shortest paths

obtained using the Floyd-Warshall algorithm. Instances with rounded distances are referred to as SDVRP-LF-r and SDVRP-UF-r for limited- and unlimited-fleet variants, respectively.

For the SDVRPTW, we use 56 benchmark instances of Solomon (1987), originally proposed for the VRPTW. The instances have 100 customers and are classified according to the spatial distribution of the customers: classes C1 and C2 (clustered), R1 and R2 (random), RC1 and RC2 (mixed). Similar to Desaulniers (2010), Archetti et al. (2012) and Bianchessi and Irnich (2019), we replicate the instances three times and set vehicle capacities equal to 30, 50, and 100, respectively, resulting in 168 instances. We focus on the instances with 50 customers, obtained by taking only the first 50 customers of the original instances. As with the SDVRP instances, each instance specifies the coordinates of the depot and the coordinates of each of the customers, and distances are taken to be the Euclidean distances. However, in the context of the SDVRPTW, distance are truncated after the first decimal place. This, unfortunately, does not guarantee that the triangle inequality is satisfied, and, thus, as is done also in Bianchessi and Irnich (2019), distances are replaced by the lengths of the shortest paths obtained by the Floyd-Warshall algorithm. As before, travel times and costs are set equal to the distances. No fleet-size limits are assumed (i.e., we use $m = n$).

The models and the BC algorithm are coded in C++ using the Concert Library of the IBM CPLEX Optimization Studio v.12.9. We use the default settings of CPLEX in the experiments with the compact models, while for the BC algorithm we turn on the Local Branching heuristic (parameter `LBHeur = 1`) and change the frequency of heuristic calls to 100 (parameter `HeurFreq = 100`), as preliminary experiments revealed that the Local Branching heuristic is often successful in finding feasible solutions. To add valid inequalities and to check the feasibility of candidate incumbent solutions, we use the generic callback procedures of CPLEX. The number of threads available to CPLEX was not restricted, but the use of callback procedures may (negatively) impact the use of parallelism by CPLEX. Furthermore, we use the CVRPSep package to generate capacity cuts and to solve the maximum flow problems required in the separation of connectivity cuts. Using other classes of cuts available in CVRPSep has not shown a significant improvement, which is in line with the findings of Archetti, Bianchessi, and Speranza (2014). We sort capacity and connectivity cuts by their violation and include the 150 most violated cuts in a single call of the separation procedure. Regularity property cuts were generated using `MatchArcs` (Archetti, Bianchessi, and Speranza 2014) for the SDVRP, and Algorithm 1 for the SDVRPTW (see Section 5.1). Feasibility and optimality cuts are generated as discussed in Section 5.2, and all violated cuts found are added.

6.1. Compact Models

To assess the performance of the proposed compact models, we use the SDVRP instances with up to 50 customers, unlimited fleet and original distances (SDVRP-UF). In addition to the proposed

compact models, i.e., `SD-MTZ`, `SD-CF`, and `SD-CFRP`, we use two compact models found in the literature, both with vehicle-indexed variables. The first one, `VI-MTZ`, is the three-index vehicle flow formulation based on the one proposed by Dror, Laporte, and Trudeau (1994). The second one, `VI-CF`, is the three-index commodity-flow formulation proposed by Ozbaygin, Karasan, and Yaman (2018). We use two different settings. In the first, we simply provide the compact model to CPLEX and try to solve the instances. In the second, we add capacity cuts to the models to strengthen the linear relaxations. More specifically, we rely on the CVRPSep package and CPLEX callbacks to add violated capacity cuts at any node of the search tree. All experiments reported in this section were run on a Linux PC with Intel Xeon Skylake 2.1 GHz processor (16 cores) and 128 GB of memory.

Table 1 shows the results obtained with a time limit of 1 hour, where the results are grouped based on the classes of instances. We report the class of instances (*Instances*) with the number of instances in the class in parentheses (all instances with up to 50 customers), the model used to solve the instances (*Model*), the number of instances in a class solved to proven optimality (*Opt*), the average objective value of the solutions obtained (*UB*), the average optimality gap (*Gap*), the average computing time (*Time*), and the average number of nodes in the search tree (*Nodes*). For the setting in which capacity cuts are added, we also report the average number of capacity cuts generated (*Cuts*). We highlight in bold the best values for *Opt*, *UB*, *Gap*, and *Time* for each class of instances (considering both settings). Detailed results of these experiments can be found in Tables EC.1 and EC.2 in the e-companion.

The results in Table 1 clearly indicate the superior performance of the proposed compact models with `SD-CFRP` providing the smallest *UB* and *Gap* values for classes *S51*, *p01*, and *SD*. It is interesting to observe that for these three classes of instances, the best results are obtained without the use of capacity cuts. For the class *eil*, the use of capacity cuts is beneficial (for all models), especially for `SD-MTZ` where the average gap (after one hour) is reduced from 14.19% to 0.24% and where four instances are solved to optimality rather than just one (the best overall performance for this class). The performance of `SD-CFRP` with capacity cuts for this class is similar, with the average gap equal to 0.45% and four instances solved to optimality.

We observe too that identifying and adding violated capacity cuts significantly increases the time required to process a node in the search tree, often more than an order of magnitude. This limits, and sometimes eliminates, the benefit of stronger linear relaxations.

A somewhat similar trade-off can be observed when comparing the results for `SD-CF` and `SD-CFRP`. As we have shown in Section 4.4, the linear relaxation of `SD-CF` is stronger than the linear relaxation of `SD-CFRP`, but `SD-CF` has more variables and constraints than `SD-CFRP`. This is reflected in the time required to process a node, which, in turn, is reflected in the number of nodes that can be

Table 1 Average results of compact models for instances with up to 50 customers, with and without capacity cuts (time limit 3600 sec).

		Model only					Model with capacity cuts					
Instances	*Model*	*Opt*	*UB*	*Gap*	*Time*	*Nodes*	*Opt*	*UB*	*Gap*	*Time*	*Nodes*	*Cuts*
eil (5)	VI-MTZ	0	744.45	34.12%	3600.0	771554.8	**4**	919.75	15.70%	1132.0	10427.0	496.8
	VI-CF	1	634.43	18.65%	3502.4	56548.4	3	924.93	18.09%	1880.0	2535.2	121.8
	SD-MTZ	1	578.37	14.19%	3144.7	1516916.8	**4**	**562.11**	**0.24%**	1018.8	2652.8	1180.6
	SD-CF	2	571.16	4.63%	2329.6	79722.4	3	564.96	1.06%	1447.4	158.4	406.2
	SD-CFRP	2	568.92	3.92%	2168.8	116815.8	**4**	563.14	0.45%	**935.0**	2008.2	1179.8
S51 (6)	VI-MTZ	0	2155.21	74.91%	3600.0	34405.5	0	3009.63	67.75%	3600.0	1180.7	1030.7
	VI-CF	0	1548.18	41.54%	3600.0	13905.8	0	3847.47	61.56%	3600.0	1255.3	141.8
	SD-MTZ	**1**	1337.41	34.19%	**2400.0**	44300.2	**1**	1503.45	30.45%	3047.1	932.2	2471.2
	SD-CF	0	1235.68	9.46%	3600.0	2015.2	**1**	1241.45	8.27%	3340.6	1244.5	1020.8
	SD-CFRP	**1**	**1219.07**	**5.54%**	3090.8	6105.5	**1**	1246.10	7.92%	3083.4	1393.7	1321.0
p01 (6)	VI-MTZ	0	2343.01	76.38%	3600.0	33475.3	0	2918.13	66.42%	3600.0	1071.3	1142.3
	VI-CF	0	1627.43	42.13%	3600.0	13292.5	0	3842.49	59.96%	3600.0	1223.2	224.3
	SD-MTZ	0	1351.45	33.29%	3600.0	41296.5	**1**	1518.90	29.50%	3082.6	1337.3	2575.7
	SD-CF	0	1283.01	9.74%	3600.0	1797.3	0	1309.42	13.93%	3600.0	1266.2	867.5
	SD-CFRP	**1**	**1250.61**	**5.69%**	3091.8	5654.0	**1**	1270.87	7.61%	**3010.1**	1565.2	1222.5
SD (9)	VI-MTZ	1	1939.47	37.10%	3200.3	1231789.8	4	2302.27	17.37%	2047.7	6651.6	420.1
	VI-CF	1	1699.83	7.60%	3200.4	247875.4	4	1724.24	7.48%	2079.8	260.6	249.9
	SD-MTZ	1	1769.01	25.37%	3200.5	2534410.8	3	1839.66	11.50%	2405.5	15310.2	1063.0
	SD-CF	3	1676.11	3.37%	2438.1	240066.8	4	1685.10	2.91%	2018.8	10341.2	484.9
	SD-CFRP	**5**	**1669.68**	**2.38%**	2105.0	107141.7	**5**	1678.50	2.51%	**1834.7**	12321.6	634.8

processed in an hour. The fact that with SD-CFRP more nodes can be processed in an hour gives it an edge over SD-CF.

6.2. BC Algorithm

We compare the proposed BC algorithm to the state-of-the-art algorithms found in the literature, using both SDVRP and SDVRPTW instances. As was done in recent papers (Archetti, Bianchessi, and Speranza 2014, Bianchessi and Irnich 2019), we impose a time limit of 2 hours for the experiments with SDVRP instances and 1 hour for the experiments with SDVRPTW instances. The experiments reported in this section were run on a Linux PC with Intel Xeon E5-2680 2.7 GHz processor (16 cores) and 192 GB of memory.

6.2.1. Results for SDVRP Instances. Table 2 summarizes the results for SDVRP instances considering four different variants: unlimited-fleet and non-rounded distances (UF), unlimited-fleet and rounded distances (UF-r), limited-fleet and non-rounded distances (LF), and limited-fleet and rounded distances (LF-r). We first report the variant (*Variant*), the instance set (*Set*) and the number of instances in the set (*Inst*). Next, we report the number of instances for which at least one feasible solution has been found (*Feas*), the number of instances solved to proven optimality (*Opt*), the number of instances solved to proven optimality for the first time (*Opt**), the number of instances for which an improved lower bound has been found (*LB**), and the number of instances for which an improved upper bound (*UB**) has been found. Finally, we report statistics averaged over the instances in a set. Specifically, we report the optimality gap (*Gap*), computed using the

Table 2 Summary of the results obtained with the proposed BC algorithm on the SDVRP benchmark instances considering unlimited- and limited-fleet with non-rounded and rounded distances (time limit 7200 sec).

Variant	*Set*	*Inst*	*Feas*	*Opt*	*Opt**	*LB**	*UB**	*Gap*	T_{total}	T_{best}^{MIP}	T_{best}^{CPL}	*Nodes*	*Check*	*Cap*	*Conn*	*Reg*	*F/O*
UF	1	11	11	5	5	7		4.06%	3954.9	1020.9	1180.8	8664.1	59.5	16804.0	2629.6	943.9	9.1
	2	14	14	4	4	8	3	4.34%	5888.7	1564.9	2593.9	60105.1	63.2	17303.3	2052.2	1744.6	9.6
	3	18	18	2	2	9	7	4.31%	6708.1	2880.5	3184.4	52303.1	75.6	16909.6	1252.3	1834.3	8.9
	4	13	13	11	8	10		0.36%	1632.9	593.4	596.3	11283.5	72.9	2967.2	40.3	570.8	3.3
UF-r	1	11	11	5		11		4.19%	3944.8	1138.6	455.0	8458.0	36.4	15232.2	2517.2	763.1	6.3
	2	14	14	5	5	14	4	3.59%	5426.7	2194.8	1650.9	62528.4	45.9	13613.2	3061.1	1980.7	10.5
	3	18	18	2	2	18	18	3.99%	6764.2	2447.2	2272.8	90708.7	47.2	15079.9	2560.3	1773.4	8.9
	4	13	13	11	11	13	13	0.28%	1620.9	274.8	268.6	26332.2	50.8	2837.7	50.5	454.7	2.5
LF	1	11	6	5				1.69%	3983.4	1137.7	1094.2	5843.2	32.7	6505.2	1348.4	4423.5	19.3
	2	14	7	2		2		0.56%	6234.7	1233.1	3070.6	55848.6	26.2	10949.6	809.1	2328.1	8.6
	3	18	9	1		3	2	1.58%	6803.6	1997.1	3984.8	45321.4	43.3	13541.6	986.3	2758.8	8.7
	4	13	11	11	1	1		0.00%	2205.8	656.8	1059.0	6359.7	30.8	1595.9	10.8	560.0	2.0
LF-r	1	11	7	5				2.23%	3968.9	946.3	1846.7	3735.1	37.6	5444.1	1878.7	3856.4	28.1
	2	14	7	3		2	1	0.50%	6186.7	2890.0	3386.3	41194.8	40.2	8626.1	928.4	2436.2	12.9
	3	18	8	2	1	14	3	2.93%	6799.2	1981.0	2427.4	61858.1	31.3	10808.5	1341.6	2526.6	17.4
	4	13	11	11	1	3		0.00%	2061.5	551.9	703.3	6429.8	22.2	1724.9	10.7	592.8	1.9

global lower and upper bound found by the BC algorithm and only using instances for which both bounds are available, the total computing time (T_{total}), the last time at which an incumbent was found by solving the auxiliary compact model (T_{best}^{MIP}), the last time at which an incumbent (satisfying the regularity property) was found by CPLEX (T_{best}^{CPL}), the number of evaluated nodes (*Nodes*), the number of candidate solutions checked for feasibility (*Check*), the number of capacity cuts added (*Cap*), the number of connectivity cuts added (*Conn*), the number of regularity property cuts added (*Reg*), and the number of feasibility and optimality cuts added (*F/O*). Detailed results can be found in Tables EC.3 through EC.9 in the e-companion. To determine whether an improved lower or upper bound is found, we used the best known lower and upper bounds from the papers by Hernández-Pérez and Salazar-González (2019), Ozbaygin, Karasan, and Yaman (2018), Chen et al. (2017), Silva, Subramanian, and Ochi (2015), Archetti, Bianchessi, and Speranza (2014), and Belenguer, Martinez, and Mota (2000).

The results show that the proposed BC algorithm provides new provably optimal solutions as well as improved lower and upper bounds for several benchmark instances. The BC algorithm finds feasible solutions to all instances with an unlimited-fleet (UF and UF-r) and for the majority of instances with a limited-fleet (LF and LF-r). It is interesting to observe that for several unlimited-fleet instances, the optimal solution uses the minimum possible number of vehicles, $\overline{m}$, but that no feasible solution is found for the companion limited-fleet instance (Tables EC.3 and EC.5 in the e-companion include the number of vehicles/routes (nR)). This suggests that imposing a limit on the number of vehicles increases the difficulty of finding feasible solutions, even in situations in which a solution with $\overline{m}$ vehicles would be obtained naturally. For the UF and UF-r variants,

the BC algorithm solves 37 instances to proven optimality for the first time and improves the best known lower bounds and the best known upper bounds 90 and 45 times, respectively. For the LF and LF-r variants, the results are not as impressive, but the BC algorithm is still able to solve three instances to optimality for the first time, and improve the best known lower bounds and the best known upper bounds 25 and 6 times, respectively.

The average optimality gap is less than 5% for all variants and instance sets, and less than 0.4% for instances in Set 4. Instances in Sets 2 and 3 are the most challenging as they require the largest average total computation time (T_{total}) and result in the largest average number of processed nodes ($Nodes$) for all variants. Instances in Set 1 are challenging for the variants with an unlimited fleet. For most variants and instance sets T_{best}^{CPL} is greater than T_{best}^{MIP}, indicating that most of the optimal or feasible solutions returned when the BC algorithm terminates are found by CPLEX' general-purpose heuristics. Finally, we observe that many capacity cuts are generated (many more than other types of cuts), typically more than 10,000, but sometimes even more than 16,000, where we recall that cuts are generated at every node of the search tree. Even though relatively few feasibility and optimality cuts are generated, they appear to make the BC algorithm more robust.

Next, in Figure 3, we compare the performance of the proposed BC algorithm with the performance of the branch-and-price (BP) algorithm of Archetti, Bianchessi, and Speranza (2011) for the variant with unlimited-fleet and non-rounded distances (UF). To the best of our knowledge, this BP algorithm produces the best results for this variant (Archetti, Bianchessi, and Speranza (2014) considered only variants with a limited fleet). We use the results as they appear in the paper, where we note that these results are obtained using a computer that is different from the one that we use in our experiments, but that a time limit of six hours was imposed as opposed to our time limit of two hours. Given the differences in computing environments, the goal of the comparison is to assess the quality of the bounds produced by the proposed BC algorithm. The bars in the plot in Figure 3 represent the percentage increase (or decrease) in the lower (or upper) bounds obtained by the BC algorithm with respect to the bounds obtained by the BP algorithm, for each of the instances in Sets 1, 2, and 3 – Archetti, Bianchessi, and Speranza (2011) use an old version of instance Set 4 and therefore no results are included for this set. For lower bounds (LB), the percentage increase is computed as $(LB_{BC} - LB_{BP})/LB_{BP}$, where LB_{BC} and LB_{BP} are the lower bounds obtained by the proposed BC algorithm and the BP algorithm, respectively; for upper bounds (UB), the percentage decrease is computed as $(UB_{BP} - UB_{BC})/UB_{BP}$. Thus, a positive percentage implies that the BC algorithm obtained an improved lower or upper bound, while a negative percentage implies the opposite. When the BP algorithm does not produce a lower or upper bound, we set the percentage to 100%, and, similarly, when the BC algorithm does not produce a lower or upper

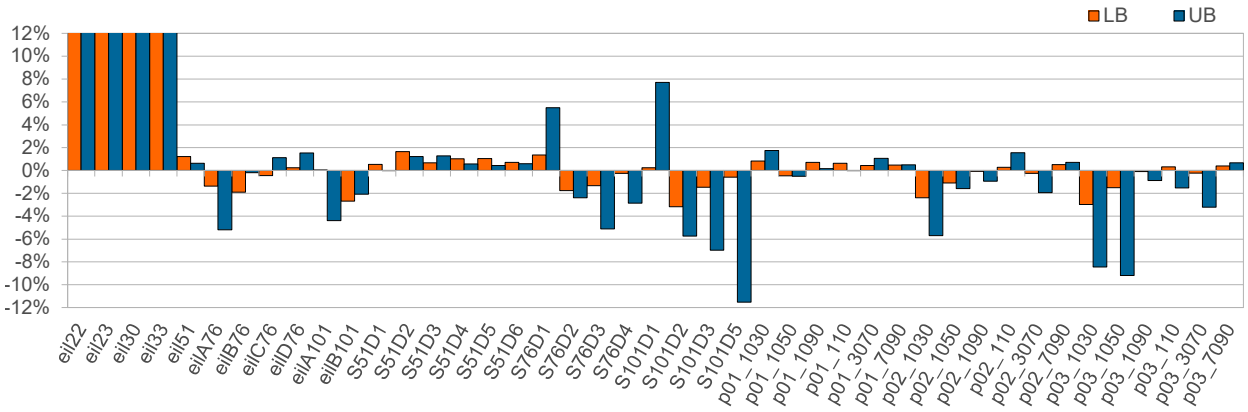
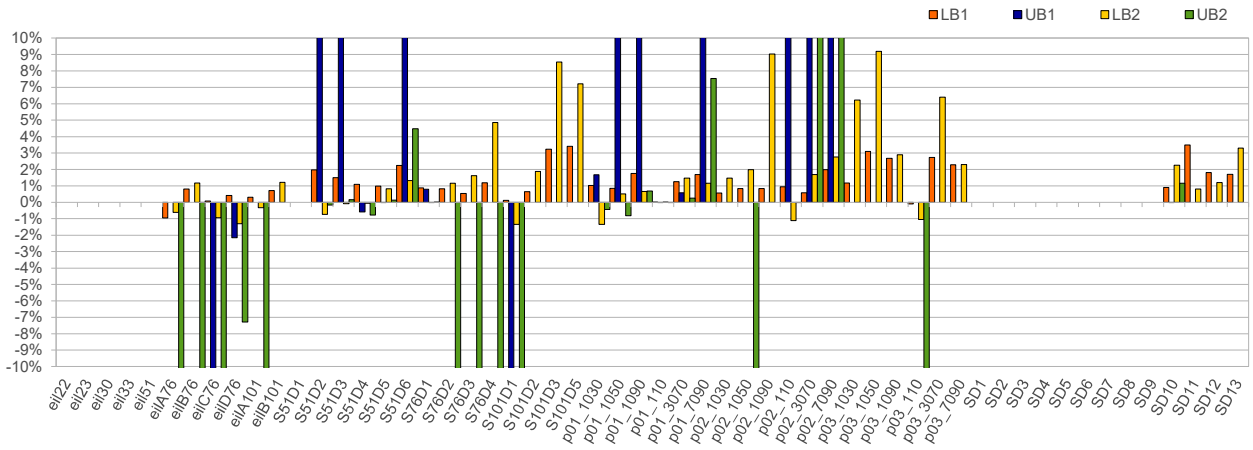


Figure 3 **Percentage of increase (or decrease) in the lower (or upper) bounds obtained by the BC algorithm with respect to the BP method of Archetti, Bianchessi, and Speranza (2011) on SDVRP instances of Sets 1 to 3, with unlimited fleet and non-rounded instances (SDVRP-UF).**

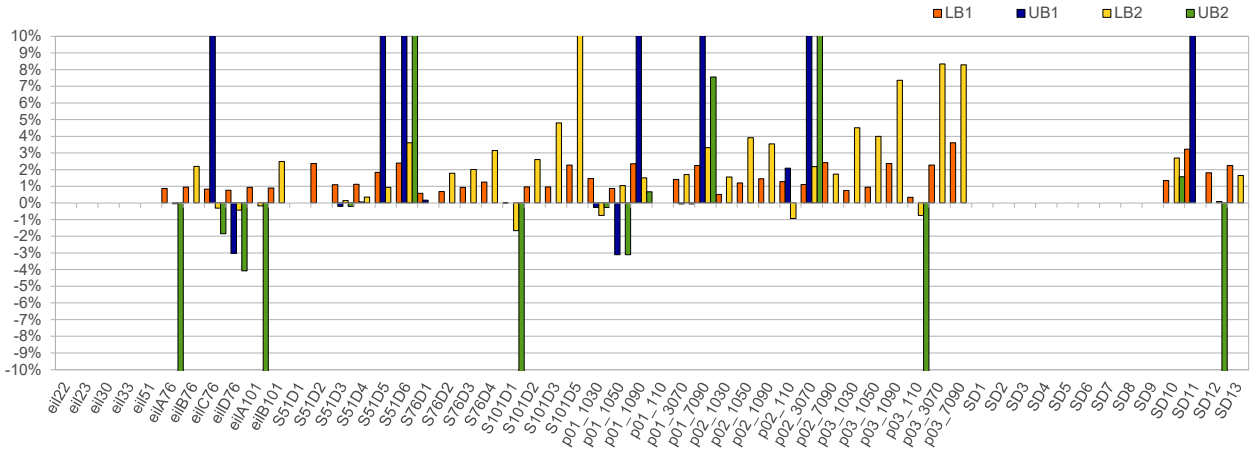
bound, we set the percentage to -100%. The plot shows only percentages between -12% and 12%; percentages of 100% have been truncated for visual clarity. Detailed results of the performance comparison can be found in Table EC.4 in the e-companion.

The plot in Figure 3 shows that the performance of the proposed BC algorithm, in terms of quality, is comparable to that of state-of-the-art exact approaches. It solves instances *eil22, eil23, eil30*, and *eil33*, which cannot be solved by the BP algorithm, and it improves the lower and upper bound of many other instances. Furthermore, when it does not improve the lower bound of an instance, the lower bound it does produce is never more than 3.2% worse than the one produced by the BC algorithm. Its performance is not as strong when it comes to producing upper bounds. However, Archetti, Bianchessi, and Speranza (2011) used bespoke heuristics to provide initial feasible solutions. Embedding these (or other) primal heuristics in the BC algorithm would likely improve its performance, but is beyond the scope of this research. Interestingly, instances where the BC algorithm does not produce a high-quality upper bound, are also the instances where it does not produce the strongest possible lower bounds (e.g. *S101D5* and *p03_1050*), which suggests that these instances are challenging for the proposed BC algorithm. Of course, it may be that providing a good initial feasible solution for these instances may also result in better lower bounds, as more nodes may be pruned early during the search.

Figure 4 shows similar plots for the variant with a limited fleet and for non-rounded and rounded distances. The performance of the BC algorithm is compared to the performance of the two best BC algorithms of Archetti, Bianchessi, and Speranza (2014), the first based on a single-commodity flow formulation and the second based on a two-index vehicle flow formulation. LB1 and UB1 represent the percentage increase and decrease with respect to the BC algorithm based on the first formulation, while LB2 and UB2 represent the percentage increase and decrease with respect to



(a) Limited-fleet, non-rounded distances (SDVRP-LF).



(b) Limited-fleet, rounded distances (SDVRP-LF-r).

Figure 4 **Relative differences between lower and upper bounds obtained by the two best BC algorithms of Archetti, Bianchessi, and Speranza (2014) and the proposed BC algorithm on instances of the SDVRP with limited-fleet and (a) non-rounded distances; (b) rounded distances.**

the BC algorithm based on the second formulation. The percentages are computed as described for the unlimited fleet variant. The detailed results of this experiment can be found in Tables EC.7 and EC.9 in the e-companion.

In both plots, we see that most of the bars appear in the upper part of the plot, indicating that the proposed BC algorithm produces better lower and upper bounds for most instances. The lower bound is better by more than 5% in several instances, and for the few instances where the lower bound is worse, it is never worse than about 2%. The proposed BC algorithm also produces better upper bounds than the other two BC algorithms for many instances. However, it fails to produce an upper bound for some instances in which one or both of the other BC algorithms do.

6.2.2. Results for SDVRPTW Instances. Table 3 presents the results of the BC algorithm on instances grouped according to class and vehicle capacity. We first report the class name (*Class*),

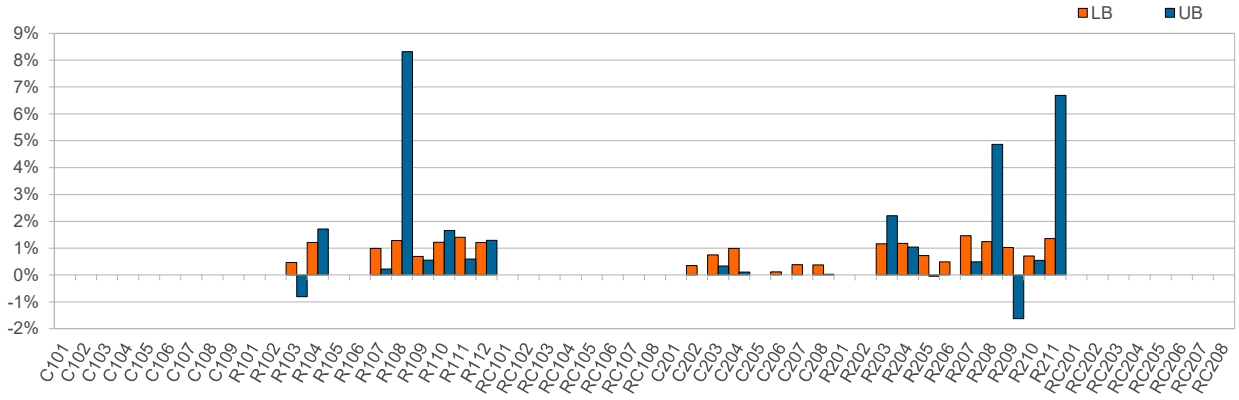
Table 3 Summary of the results obtained with the proposed BC algorithm on the SDVRPTW benchmark instances with $n = 50$ (time limit: 3600 sec).

Class	*Q*	*Inst*	*Opt*	*Opt**	*LB**	*UB**	*Gap*	T_{total}	T_{best}^{MIP}	T_{best}^{CPL}	*Nodes*	*Check*	*Cap*	*Conn*	*Reg*	*F/O*
C1	30	9	9				0.00%	92.7	67.0	63.6	4189.3	28.7	406.8	17.0	5802.0	21.6
	50	9	9				0.00%	161.0	142.1	125.0	8849.4	35.6	794.8	65.6	4552.1	19.7
	100	9	7				0.12%	900.6	92.8	708.0	41680.9	41.3	2451.6	1787.1	5278.7	22.0
R1	30	12	6	2	8	7	0.61%	2773.9	1422.9	1433.9	59884.7	59.0	18376.8	354.2	9441.0	38.0
	50	12	5	2	7	4	1.39%	2721.9	1592.7	1273.2	35997.9	36.3	18923.7	505.1	3696.6	17.6
	100	12	2		7	1	7.82%	3009.1	446.9	410.9	27157.2	21.3	16274.7	3495.0	1187.2	4.6
RC1	30	8	8				0.00%	162.7	71.9	94.1	27445.4	24.9	67.4	110.4	3918.1	15.1
	50	8	8				0.00%	503.1	262.3	313.2	133254.4	43.5	42.1	909.5	5739.6	19.3
	100	8	8				0.00%	446.3	230.9	378.7	63162.1	35.5	695.6	4292.3	3937.8	14.8
C2	30	8	8	6	6	3	0.00%	216.9	127.2	93.3	12639.0	18.4	917.4	86.8	2685.0	11.5
	50	8	8	6	6	3	0.00%	920.3	468.2	493.3	53921.6	87.3	3042.9	356.0	10005.4	73.8
	100	8	8				0.00%	584.2	447.4	473.1	30849.4	45.6	4414.1	698.1	4626.3	18.1
R2	30	11	6	4	9	6	0.54%	2635.9	1879.5	1743.7	56119.2	50.8	15539.4	281.9	6869.6	33.4
	50	11	3	2	7	4	1.62%	2870.7	1506.1	1322.6	43009.6	29.2	19594.1	573.5	3301.8	12.5
	100	11	4	2	8	2	3.41%	2890.9	847.2	542.5	36761.7	41.5	21710.6	2487.7	3082.5	9.7
RC2	30	8	8				0.00%	164.5	94.8	68.4	18652.5	24.9	101.8	143.0	2751.1	12.9
	50	8	8				0.00%	553.2	215.2	241.5	172357.1	34.3	37.3	685.4	3902.1	14.8
	100	8	8				0.00%	157.6	97.0	120.4	15137.6	53.4	178.0	1717.0	3243.9	16.0

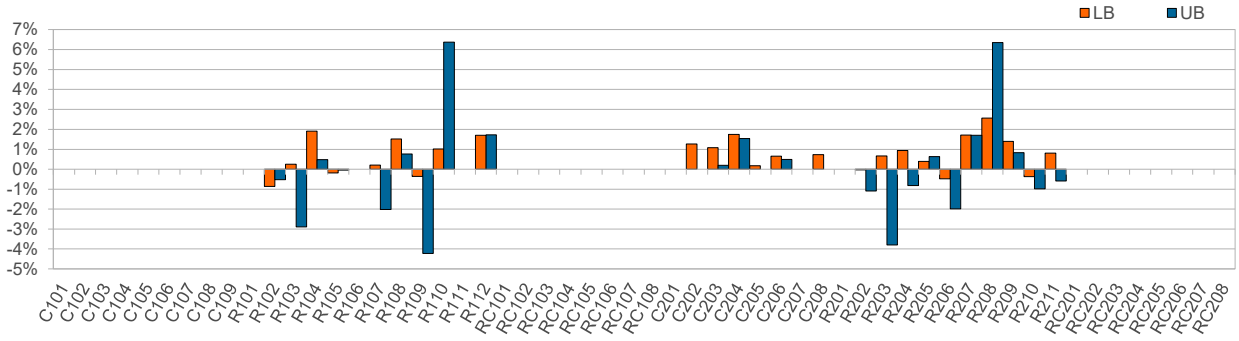
the vehicle capacity (Q) and the number of instances in the group (*Inst*). Next, we report the same statistics as for the SDVRP instances in Table 2, where we omit the column *Feas* as the proposed BC algorithm finds feasible solutions to all instances. Detailed results can be found in Tables EC.10 – EC.12 in the e-companion.

The results show that the proposed BC algorithm is also effective for SDVRPTW instances: it solves most instances to optimality, it proves optimality for the first time for 24 instances, and it finds improved best known lower and upper bounds for 58 and 30 instances, respectively. The average optimality gap is small for all instance groups, except for R1 and R2 with $Q = 100$ (7.82% and 3.41%, respectively). Overall, instances in classes R1 and R2 are the most challenging, as fewer than half are solved to optimality and these instances result in the largest average computational times. The average number of solutions checked for feasibility is of the same magnitude as for the SDVRP instances. The average number of capacity cuts generated, however, is much smaller than for the SDVRP instances, relative to the average number of other cuts generated. The number of feasibility and optimality cuts generated is slightly larger for instances with time windows, suggesting that the enhancement may be more beneficial in this setting.

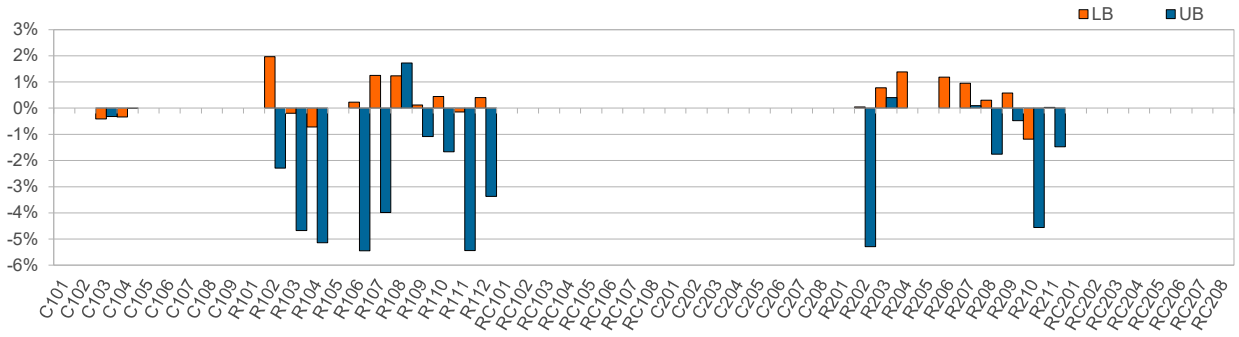
Next, in Figure 5, we compare the performance of the proposed BC algorithm with the performance of the state-of-the-art BC algorithm of Bianchessi and Irnich (2019). Specifically, the figure shows three plots with percentage increases and decreases in lower and upper bounds, considering vehicle capacity $Q = 30$, 50, and 100, respectively, based on the detailed results presented in Tables



(a) $Q = 30$.



(b) $Q = 50$.



(c) $Q = 100$.

Figure 5 **Relative differences between lower and upper bounds obtained by the BC method of Bianchessi and Irnich (2019) and the proposed BC algorithm on instances of the SDVRPTW with (a) $Q = 30$; (b) $Q = 50$; (c) $Q = 100$.**

EC.10 – EC.12 in the e-companion. The percentage increase and decrease are computed as before. Again, the primary goal of the comparison is to assess the quality of the bounds produced by the proposed BC algorithm. (Bianchessi and Irnich (2019) also use a time limit of 1 hour, but a different computing environment, which makes a more thorough comparison difficult).

The plots indicate that the performance of the algorithms is comparable, with a slight edge for the proposed BC algorithm when it comes to producing lower bounds. With vehicle capacity $Q = 30$,

the proposed BC algorithm produces a lower bound that is either equal or better than the lower bound produced by the BC algorithm of Bianchessi and Irnich (2019), while with vehicle capacity $Q = 50$ and $Q = 100$, the proposed BC algorithm produces a lower bound that is slightly worse for a few instances – the percentage decrease is always less than 1.2%. This is quite an achievement as the method of Bianchessi and Irnich (2019) incorporates several classes of sophisticated valid inequalities specifically designed for the SDVRPTW. The results are somewhat mixed when it comes to producing upper bounds. The proposed BC algorithm seems to do a bit better when the vehicle capacity is small ($Q = 30$), but worse when the vehicle capacity is large ($Q = 100$). We note that the BC algorithm of Bianchessi and Irnich (2019) incorporates a simple, but special-purpose heuristic for generating an initial feasible solution.

6.3. New Optimal Solutions and Overall Performance

As reported in Tables 2 and 3, the proposed BC algorithm has proven optimality for several benchmark instances for the first time and has produced new best lower and upper bounds for several other benchmark instances. A closer look reveals that the optimality gap is quite small for many benchmark instances, especially SDVRP instances with unlimited-fleet and non-rounded distances (UF) and SDVRPTW instances with $Q = 30$ and 50. Therefore, we have conducted one additional experiment, in which we solve these instances once more, but now with a time limit of 24 hours. With the additional computing time, the proposed BC algorithm solves to proven optimality six more SDVRP-UF instances (all instances with up to 64 customers, except for *p01_7090*); and 18 more SDVRPTW instances. Tables EC.13 and EC.14 in the e-companion present detailed statistics for the instances that are solved to proven optimality for the first time. In summary, the proposed BC algorithm provides proven optimal solutions for 95 instances for the first time: for 25 of the SDVRP-UF instance, for 18 of the SDVRP-UF-r instance, for 8 of the SDVRP-LF instances, for 2 of the SDVRP-LF-r instances, and for 42 of the SDVRPTW instances.

Tables 4 and 5 summarize the overall performance of the proposed algorithm and the state-of-the-art methods for the SDVRP and the SDVRPTW, respectively, regarding the number of optimal solutions (*Opt*), average relative gap (*Gap*) and average total computational time (T_{total}). The results are grouped by variant and set, for the SDVRP; and by class and vehicle capacity, for the SDVRPTW. In particular for the latter, the table shows also the average lower and upper bounds, as both solution methods found lower and upper bounds for all instances in the experiments (for the SDVRP, there were missing lower and/or upper bounds for some instances in any of the methods). The results presented for the other algorithms need some clarification, as follows. For the SDVRP-UF, the BP algorithm of Archetti, Bianchessi, and Speranza (2011) achieved the time limit of 6 hours for all instances of Sets 1 to 3, and the authors did not include the same instances we used

Table 4 Summary of the state-of-the-art and proposed BC algorithms on the SDVRP benchmark instances considering unlimited- and limited-fleet with non-rounded and rounded distances.

			Other algorithms[1]			Proposed BC algorithm[2]		
Variant	*Set*	*Inst*	*Opt*	*Gap*	T_{total}	*Opt*	*Gap*	T_{total}
UF	1	11	0	4.91%	–	5	4.06%	3954.9
	2	14	0	3.45%	–	4	4.34%	5888.7
	3	18	0	2.82%	–	2	4.31%	6708.1
	4	13				11	0.36%	1632.9
UF-r	1	11				5	4.19%	3944.8
	2	14	2	3.79%		5	3.59%	5426.7
	3	18				2	3.99%	6764.2
	4	13				11	0.28%	1620.9
LF	1	11	5	2.18%	3929.8	5	1.69%	3983.4
	2	14	4	4.79%	5369.2	2	0.56%	6234.7
	3	18	2	2.59%	6450.9	1	1.58%	6803.6
	4	13	9	0.39%	2273.2	11	0.00%	2205.8
LF-r	1	11	5	2.34%	3931.9	5	2.23%	3968.9
	2	14	4	0.66%	5425.4	3	0.50%	6186.7
	3	18	3	2.71%	6576.8	2	2.93%	6799.2
	4	13	10	0.77%	2254.9	11	0.00%	2061.5

[1] UF: BP algorithm of Archetti, Bianchessi, and Speranza (2011) with a time limit of 6 hours, using a PC with Intel Dual Core Pentium IV 2.4 GHz processor and 3 GB of memory; UF-r: cutting plane algorithm of Belenguer, Martinez, and Mota (2000) (computer characteristics and time limit were not specified by the authors); LF and LF-r: BC algorithm of Archetti, Bianchessi, and Speranza (2014) based on the two-index vehicle flow formulation, with a time limit of 2 hours, using a Windows PC with Intel Xeon W3680 3.33 GHz processor and 12 GB of memory.

[2] Time limit of 2 hours, using a Linux PC with Intel Xeon E5-2680 2.7 GHz processor (16 cores) and 192 GB of memory.

Table 5 Summary of the state-of-the-art and proposed BC algorithms on the SDVRPTW benchmark instances with $n = 50$ (time limit: 3600 sec).

			Bianchessi and Irnich (2019)[1]					Proposed BC algorithm[2]				
	Q	*Inst*	*Opt*	*LB*	*UB*	*Gap*	T_{total}	*Opt*	*LB*	*UB*	*Gap*	T_{total}
C1	30	9	9	1598.97	1598.97	0.00%	492.4	9	1598.97	1598.97	0.00%	92.7
	50	9	9	1013.40	1013.40	0.00%	119.3	9	1013.40	1013.40	0.00%	161.0
	100	9	9	584.37	584.37	0.00%	308.9	7	583.88	584.59	0.12%	900.6
R1	30	12	4	1544.53	1583.49	2.55%	2830.6	6	1560.59	1570.10	0.61%	2773.9
	50	12	4	1074.24	1098.95	2.38%	2708.8	5	1079.60	1094.94	1.39%	2721.9
	100	12	2	768.87	814.59	6.33%	3000.9	2	771.76	835.05	7.82%	3009.1
RC1	30	8	8	2739.51	2739.51	0.00%	14.4	8	2739.51	2739.51	0.00%	162.7
	50	8	8	1699.59	1699.59	0.00%	3.0	8	1699.59	1699.59	0.00%	503.1
	100	8	8	940.45	940.45	0.00%	292.0	8	940.45	940.45	0.00%	446.3
C2	30	8	2	1771.76	1779.34	0.43%	3325.0	8	1778.30	1778.30	0.00%	216.9
	50	8	2	1149.12	1160.49	1.00%	3118.4	8	1157.21	1157.21	0.00%	920.3
	100	8	8	686.09	686.09	0.00%	981.1	8	686.09	686.09	0.00%	584.2
R2	30	11	2	1538.33	1582.15	2.87%	3271.2	6	1551.86	1560.36	0.54%	2635.9
	50	11	3	1054.89	1080.20	2.45%	2979.4	3	1062.03	1079.59	1.62%	2870.7
	100	11	2	719.77	739.27	2.78%	3251.5	4	722.31	748.18	3.41%	2890.9
RC2	30	8	8	2739.50	2739.50	0.00%	12.4	8	2739.50	2739.50	0.00%	164.5
	50	8	8	1699.63	1699.63	0.00%	3.3	8	1699.63	1699.63	0.00%	553.2
	100	8	8	934.81	934.81	0.00%	12.9	8	934.81	934.81	0.00%	157.6

[1] Linux PC with Intel Xeon E5-1650v3 3.5 GHz processor and 64 GB of memory.

[2] Linux PC with Intel Xeon E5-2680 2.7 GHz processor (16 cores) and 192 GB of memory.

in Set 4 in their computational experiments. To the best of our knowledge, Belenguer, Martinez, and Mota (2000) were the only authors to report results for the SDVRP-UF-r, and considering instances of Set 2 only. For the limited fleet variants, we present the results of the BC algorithm of Archetti, Bianchessi, and Speranza (2014) with the best overall performance. Finally, for the SDVRPTW, we present the results of Bianchessi and Irnich (2019). These two tables highlight the superior performance of proposed BC algorithm with respect to number of optimal solutions and optimality gap, in comparison to the state-of-the-art approaches, for several classes of instances and different variants.

7. Final Remarks

We have introduced three new compact formulations for split delivery routing problems. The formulations are *extended* formulations in the sense that more detail is embedded in the variables – variables represent two arcs that appear consecutively in a route. Computational experiments show that that these formulations (even though they have more variables and constraints) are more effective when used to solve instances of the SDVRP and SDVRPTW with a commercial integer programming solver compared to existing compact formulations. The formulations introduce novel ideas for modeling commodity-flow and MTZ constraints, which may have applications in other vehicle routing variants.

To achieve even better performance, we have proposed a BC algorithm, which starts from a two-index commodity-flow formulation and locally extends that formulation when needed to guarantee that the regularity property is satisfied. The BC algorithm has been enhanced by incorporating well-known valid inequalities (i.e., capacity and connectivity cuts), and has been shown to be quite effective, proving optimality for several benchmark instances for the first time.

The proposed BC algorithm can likely be enhanced by adding more sophisticated valid inequalities, e.g., the one proposed by Bianchessi and Irnich (2019). However, a more promising and interesting future research direction is to develop a hybrid BC algorithm that embeds metaheuristics to quickly find high-quality feasible as well as improve feasible solutions found during the search (as done in Álvarez and Munari (2017)). Finally, exploring whether locally extending formulations of other routing problems can lead to effective solution approaches is of interest too.

Acknowledgments

The authors thank the anonymous reviewers for their valuable comments. This work was supported by the Sao Paulo Research Foundation (FAPESP) [grant numbers 18/23555-1, 16/01860-1 and 13/07375-0] and the National Council for Scientific and Technological Development (CNPq) [grant number 304601/2017-9].

References

Álvarez A, Munari P, 2017 *An exact hybrid method for the vehicle routing problem with time windows and multiple deliverymen. Computers & Operations Research* 83:1–12.

Archetti C, Bertazzi L, Hertz A, Speranza MG, 2012 *A hybrid heuristic for an inventory routing problem. INFORMS Journal on Computing* 24(1):101–116.

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, Bianchessi N, Speranza MG, 2015 *A branch-price-and-cut algorithm for the commodity constrained split delivery vehicle routing problem. Computers & Operations Research* 64:1–10.

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(3):285–298.

Archetti C, Savelsbergh M, Speranza MG, 2006 *Worst-case analysis for split delivery vehicle routing problems. Transportation Science* 40(2):226–234.

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

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

Archetti C, Speranza MG, 2012 *Vehicle routing problems with split deliveries. International Transactions in Operational Research* 19(1-2):3–22.

Archetti C, Speranza MG, Hertz A, 2006 *A tabu search algorithm for the split delivery vehicle routing problem. Transportation Science* 40(1):64–73.

Bektas T, Gouveia L, 2014 *Requiem for the Miller-Tucker-Zemlin subtour elimination constraints? European Journal of Operational Research* 236(3):820–832.

Belenguer JM, Martinez M, Mota E, 2000 *A lower bound for the split delivery vehicle routing problem. Operations Research* 48(5):801–810.

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.

Berbeglia G, Cordeau JF, Gribkovskaia I, Laporte G, 2007 *Static pickup and delivery problems: a classification scheme and survey. TOP* 15(1):1–31.

Bianchessi N, Irnich S, 2019 *Branch-and-cut for the split delivery vehicle routing problem with time windows. Transportation Science* 53(2):442–462.

Bruck BP, Cruz F, Iori M, Subramanian A, 2019 *The static bike sharing rebalancing problem with forbidden temporary operations. Transportation Science* 53(3):882–896.

Bruck BP, Iori M, 2017 *Non-elementary formulations for single vehicle routing problems with pickups and deliveries. Operations Research* 65(6):1597–1614.

Casazza M, Ceselli A, Chemla D, Meunier F, Wolfler Calvo R, 2018 *The multiple vehicle balancing problem. Networks* 72(3):337–357.

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.

Ceselli A, Righini G, Salani M, 2009 *Column generation for the split delivery vehicle routing problem. Technical report, University of Milan-DTI-Note del Polo n. 118* .

Chemla D, Meunier F, Calvo RW, 2013 *Bike sharing systems: Solving the static rebalancing problem. Discrete Optimization* 10(2):120–146.

Chen P, Golden B, Wang X, Wasil E, 2017 *A novel approach to solve the split delivery vehicle routing problem. International Transactions in Operational Research* 24(1-2):27–41.

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

Desaulniers G, 2010 *Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. Operations Research* 58(1):179–192.

Dror M, Laporte G, Trudeau P, 1994 *Vehicle routing with split deliveries. Discrete Applied Mathematics* 50(3):239–254.

Dror M, Trudeau P, 1989 *Savings by split delivery routing. Transportation Science* 23(2):141–145.

Dror M, Trudeau P, 1990 *Split delivery routing. Naval Research Logistics (NRL)* 37(3):383–402.

Fragkos I, Degraeve Z, De Reyck B, 2016 *A horizon decomposition approach for the capacitated lot-sizing problem with setup times. INFORMS Journal on Computing* 28(3):465–482.

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

Gouveia L, Pires JM, 1999 *The asymmetric travelling salesman problem and a reformulation of the Miller-Tucker-Zemlin constraints. European Journal of Operational Research* 112(1):134–146.

Hernández-Pérez H, Salazar-González JJ, 2019 *Optimal solutions for the vehicle routing problem with split demands. International Conference on Computational Logistics*, 189–203 (Springer).

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.

Irnich S, Schneider M, Vigo D, 2014 *Chapter 9: Four variants of the vehicle routing problem. Vehicle Routing: Problems, Methods, and Applications, Second Edition*, 241–271 (SIAM).

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.

Lysgaard J, Letchford AN, Eglese RW, 2004 *A new branch-and-cut algorithm for the capacitated vehicle routing problem. Mathematical Programming* 100(2):423–445.

Miller CE, Tucker AW, Zemlin RA, 1960 *Integer programming formulation of traveling salesman problems. Journal of the ACM (JACM)* 7(4):326–329.

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.

Munari P, Dollevoet T, Spliet R, 2016 *A generalized formulation for vehicle routing problems. arXiv preprint arXiv:1606.01935* .

Ozbaygin G, Karasan O, Yaman H, 2018 *New exact solution approaches for the split delivery vehicle routing problem. EURO Journal on Computational Optimization* 6(1):85–115.

Pessoa A, Uchoa E, de Aragão MP, Rodrigues R, 2010 *Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. Mathematical Programming Computation* 2(3):259–290.

Salazar-González JJ, Santos-Hernández B, 2015 *The split-demand one-commodity pickup-and-delivery travelling salesman problem. Transportation Research Part B: Methodological* 75:58–73.

Sepúlveda J, Escobar JW, Adarme-Jaimes W, 2014 *An algorithm for the routing problem with split deliveries and time windows (SDVRPTW) applied on retail SME distribution activities. Dyna* 81(187):223–231.

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

Silva MM, Subramanian A, Ochi LS, 2015 *An iterated local search heuristic for the split delivery vehicle routing problem. Computers & Operations Research* 53:234 – 249.

Solomon MM, 1987 *Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research* 35(2):254–265.

Wolfinger D, Salazar-Gonzalez JJ, 2021 *The pickup and delivery problem with split loads and transshipments: A branch-and-cut solution approach. European Journal of Operational Research* 289(2):470–484.

E-Companion: Detailed results of computational experiments

In this electronic companion, we present detailed results of the computational experiments reported in Section 6 of the paper. The instances and meaning of the column headers are the same as in the body of the paper. In particular, Tables EC.13 and EC.14 present the results for the instances that are solved to proven optimality for the first time. In Table EC.14, the number of routes (nR) in the optimal solutions is equal to the minimum possible, $\overline{m}$, for all instances except *eil30*. Hence, the optimal values presented for those instances are also optimal for the SDVRP-LF.

Table EC.1 Results of the compact models for instances with up to 50 customers (time limit: 3600 sec).

			VI-MTZ				VI-CF				SD-MTZ				SD-CF				SD-CFRP			
Instance	n	Q	UB	Gap	Time	Nodes	UB	Gap	Time	Nodes	UB	Gap	Time	Nodes	UB	Gap	Time	Nodes	UB	Gap	Time	Nodes
eil22	21	6000	375.28	13.61	–	1296593	375.28	2.88	–	74267	375.28	3.04	–	3473559	375.28	0.00	73.03	25933	375.28	0.00	41.43	15119
eil23	22	4500	568.56	13.65	–	1550388	568.56	0.00	3111.95	118845	568.56	0.00	1323.34	2484731	568.56	0.00	774.72	282938	568.56	0.00	2.66	4752
eil30	29	4500	565.32	36.96	–	527476	505.01	10.11	–	11526	505.01	25.63	–	1325842	505.01	8.49	–	73392	505.01	6.07	–	466195
eil33	32	8000	1342.51	56.08	–	434608	845.48	7.06	–	7594	881.77	25.12	–	263895	844.10	4.15	–	16094	839.63	4.88	–	89313
eil51	50	160	870.55	50.28	–	48709	877.83	73.22	–	70510	561.22	17.13	–	36557	562.87	10.48	–	255	556.12	8.66	–	8700
S51D1	50	160	1386.73	69.43	–	62360	1213.75	86.26	–	13545	461.87	4.09	–	79897	516.72	14.15	–	42	459.50	0.00	544.58	13604
S51D2	50	160	1414.46	68.13	–	44861	1189.04	54.34	–	6144	841.70	36.65	—	36341	727.31	9.17	–	683	714.30	6.72	–	5036
S51D3	50	160	2131.00	77.59	–	38232	1345.17	41.14	–	4985	1114.77	40.59	–	37659	989.29	11.25	–	1267	968.58	7.67	–	4383
S51D4	50	160	2745.69	80.65	–	24782	1709.27	22.51	–	13881	1766.31	44.03	–	37164	1605.12	8.56	–	3315	1607.51	8.27	–	3951
S51D5	50	160	1749.87	70.64	–	27305	1550.55	30.47	–	18815	1467.19	40.76	–	37111	1370.79	8.42	–	2473	1361.37	5.72	–	5144
S51D6	50	160	3503.51	83.05	–	8893	2281.31	14.49	–	26065	2372.59	39.03	–	37629	2204.87	5.21	–	4311	2203.18	4.84	–	4515
p01-1030	50	160	1486.61	69.08	–	52819	1282.61	54.45	–	4947	877.11	34.76	–	36735	798.22	10.93	–	728	792.02	9.52	–	4675
p01-1050	50	160	2104.23	77.09	–	39650	1455.38	42.99	–	10732	1125.92	39.15	–	37031	1059.16	11.93	–	1376	1043.01	8.13	–	4696
p01-1090	50	160	2940.05	82.08	–	19431	1847.80	27.90	–	6621	1644.15	41.48	–	39602	1565.37	10.06	–	2179	1496.74	4.08	–	5981
p01-110	50	160	1274.10	66.73	–	57952	1159.22	87.33	–	9349	462.91	4.59	–	65423	479.41	7.02	–	65	459.50	0.00	551.08	9582
p01-3070	50	160	2549.36	79.34	–	20696	1816.93	27.79	–	6002	1611.24	40.42	–	36674	1558.21	10.87	–	2607	1521.30	7.36	–	4035
p01-7090	50	160	3703.72	83.97	–	10304	2202.61	12.34	–	42104	2387.36	39.35	–	32314	2237.66	7.64	–	3829	2191.10	5.08	–	4955
SD1	8	100	228.28	0.00	2.98	15487	228.28	0.00	3.86	4431	228.28	0.00	4.13	43096	228.28	0.00	0.92	226	228.28	0.00	0.68	204
SD2	16	100	714.14	21.73	–	4302680	708.28	7.39	–	1221442	708.28	12.50	–	9951991	708.28	3.63	–	1052156	708.28	0.00	290.03	343799
SD3	16	100	430.58	3.86	–	5277945	430.58	1.45	–	874453	430.58	4.39	–	10333620	430.58	0.00	46.30	87833	430.58	0.00	13.75	4301
SD4	24	100	676.89	19.60	–	1140307	631.05	2.25	–	30511	645.48	12.49	–	1326344	631.05	0.00	295.93	158453	631.05	0.00	1217.04	55189
SD5	32	100	1450.50	42.80	–	105426	1418.62	13.12	–	46425	1414.56	26.73	–	444521	1390.57	5.43	–	88301	1390.57	3.91	–	84101
SD6	32	100	894.17	30.07	–	172807	853.36	8.02	–	16525	893.68	20.40	–	506849	831.24	0.06	–	722200	831.24	0.00	3023.95	335446
SD7	40	100	3857.41	72.56	–	39254	3700.00	10.81	–	17049	3798.31	54.18	–	107475	3640.00	7.42	–	32300	3640.00	6.21	–	92664
SD8	48	100	6037.28	79.46	–	22965	5200.00	10.00	–	7321	5640.00	60.78	–	37693	5160.00	8.35	–	4459	5100.00	6.58	–	43044
SD9	48	100	3165.93	63.79	–	9237	2128.24	15.38	–	12722	2161.95	36.88	–	58108	2065.02	5.44	–	14673	2067.10	4.73	–	5527

– Solver reached the time limit of 3600 seconds

Table EC.2 Results of the compact models with capacity cuts, for instances with up to 50 customers (time limit: 3600 sec).

	VI-MTZ					VI-CF					SD-MTZ					SD-CF					SD-CFRP				
Instance	UB	Gap	Time	Nodes	Cuts	UB	Gap	Time	Nodes	Cuts	UB	Gap	Time	Nodes	Cuts	UB	Gap	Time	Nodes	Cuts	UB	Gap	Time	Nodes	Cuts
eil22	375.28	0.00	55.78	4543	364	375.28	0.00	105.41	0	38	375.28	0.00	9.03	144	339	375.28	0.00	6.50	0	46	375.28	0.00	4.89	0	54
eil23	568.56	0.00	13.94	1487	113	568.56	0.00	515.42	4520	28	568.56	0.00	4.04	0	16	568.56	0.00	3.80	0	16	568.56	0.00	1.17	0	8
eil30	505.01	0.00	91.76	4332	207	505.01	0.00	1579.42	290	32	505.01	0.00	27.34	105	140	505.01	0.00	26.77	0	28	505.01	0.00	17.78	17	62
eil33	837.06	0.00	1898.72	40824	272	837.06	0.32	–	4641	232	837.06	0.00	1453.74	9181	1235	837.06	0.23	–	767	1307	837.06	0.00	1051.02	8802	461
eil51	2312.85	78.51	–	949	1528	2338.76	90.12	–	3225	279	524.63	1.21	–	3834	4173	538.87	5.06	–	25	634	529.79	2.24	–	1222	5314
S51D1	1567.91	70.85	–	6094	614	1009.80	87.27	–	2755	74	459.50	0.00	282.46	686	484	459.50	0.00	2043.49	177	257	459.50	0.00	500.15	506	1186
S51D2	2194.62	72.51	–	120	2208	1731.21	74.57	–	3935	105	936.43	35.68	–	426	3771	731.91	9.58	–	193	719	753.37	11.52	–	327	1868
S51D3	2617.81	69.57	–	122	1820	13297.27	94.80	–	173	47	1354.35	39.49	–	368	3494	1021.44	14.12	–	691	1665	1033.52	13.90	–	668	1766
S51D4	3492.66	63.70	–	262	447	2324.05	41.96	–	227	353	2010.08	40.96	–	607	2693	1619.38	9.40	–	1764	899	1620.48	8.32	–	1747	1110
S51D5	3288.33	66.33	–	276	648	2328.58	53.08	–	254	143	1876.10	40.77	–	843	2591	1389.90	9.75	–	1578	523	1373.68	6.90	–	1860	803
S51D6	4896.44	63.54	–	210	447	2393.91	17.68	–	188	129	2384.27	25.83	–	2663	1794	2226.60	6.73	–	3064	2062	2236.04	6.89	–	3254	1193
p01-1030	2314.49	72.03	–	101	2408	1921.91	74.56	–	2090	68	1104.73	40.17	–	451	4228	802.36	11.51	–	232	970	822.03	12.87	–	583	2649
p01-1050	2435.65	64.93	–	169	1792	13297.27	94.44	–	626	117	1362.14	39.82	–	597	3836	1056.46	11.61	–	823	1811	1072.08	11.11	–	1051	1872
p01-1090	3156.24	60.00	–	152	977	2169.57	41.69	–	175	399	1759.51	29.35	–	1021	2087	1556.86	9.69	–	1857	574	1547.28	7.81	–	1386	851
p01-110	2280.76	79.96	–	4965	657	1079.36	87.53	–	3971	143	459.50	0.00	495.73	688	1520	671.98	33.18	–	0	2	459.50	0.00	60.40	36	71
p01-3070	3165.34	61.14	–	167	548	2221.45	44.01	–	177	446	2096.33	43.97	–	814	1998	1544.70	10.20	–	1691	651	1541.37	8.60	–	2107	985
p01-7090	4156.30	60.47	–	874	472	2365.36	17.56	–	300	173	2331.18	23.68	–	4453	1785	2224.16	7.36	–	2994	1197	2182.93	5.28	–	4228	907
SD1	228.28	0.00	0.57	0	21	228.28	0.00	0.58	0	11	228.28	0.00	0.18	0	10	228.28	0.00	0.26	0	6	228.28	0.00	0.17	0	9
SD2	708.28	0.00	63.36	41490	91	708.28	0.00	38.62	98	80	708.28	0.00	34.11	4352	68	708.28	0.00	43.77	5638	118	708.28	0.00	235.99	39206	225
SD3	430.58	0.00	11.01	849	249	430.58	0.00	19.66	67	131	430.58	0.00	14.78	2388	351	430.58	0.00	6.75	1067	70	430.58	0.00	3.93	0	7
SD4	631.05	0.00	354.09	3293	694	631.05	0.00	659.43	0	69	645.19	5.11	–	80685	804	631.05	0.00	118.86	6338	22	631.05	0.00	405.24	7234	35
SD5	1431.60	3.39	–	5919	354	1414.56	10.14	–	75	571	1415.26	5.61	–	24362	1057	1390.57	2.20	–	14782	836	1390.57	2.70	–	23137	532
SD6	883.24	12.90	–	2084	1209	901.80	17.52	–	106	724	856.94	7.10	–	20728	750	831.24	0.46	–	58077	215	831.24	0.00	1467.10	16402	334
SD7	4506.79	20.01	–	4605	66	3700.00	10.67	–	1937	165	3834.30	23.27	–	3263	2747	3640.00	8.00	–	4514	509	3680.00	5.82	–	17781	446
SD8	8874.77	82.14	–	1410	233	5260.00	9.75	–	25	46	6114.04	36.06	–	768	1546	5220.00	8.90	–	993	1129	5120.00	7.49	–	3819	2697
SD9	3025.81	37.93	–	214	864	2243.62	19.25	–	37	452	2324.07	26.32	–	1246	2234	2085.85	6.67	–	1662	1459	2086.46	6.59	–	3315	1428

– Solver reached the time limit of 3600 seconds

Table EC.3 Detailed results of the proposed BC algorithm on the SDVRP benchmark instances with non-rounded distances and unlimited fleet (SDVRP-UF) – time limit: 7200 sec.

				Best known		BC algorithm												
Instance	n	Q	$\overline{m}$	LB	UB	LB	UB	Gap	T_{total}	T_{best}^{MIP}	T_{best}^{CPL}	$Nodes$	$Check$	Cap	$Conn$	Reg	F/O	nR
eil22	21	6000	4	–	375.28	**375.28**	375.28	⋆	2.67	2.47	2.61	0	29	111	17	146	7	**4**
eil23	22	4500	3	451.80	568.56	**568.56**	568.56	⋆	1.86	1.60	1.77	0	37	34	43	44	4	**3**
eil30	29	4500	3	218.92	505.01	**505.01**	505.01	⋆	3.88	3.80	3.83	0	49	76	311	290	2	4
eil33	32	8000	4	–	837.06	**837.05**	837.06	⋆	102.26	87.38	98.97	14797	115	1212	738	2627	40	**4**
eil51	50	160	5	518.23	524.61	**524.61**	524.61	⋆	115.62	40.90	94.92	6718	97	2950	461	253	6	**5**
eilA76	75	140	10	809.58	823.89	798.43	875.96	8.85%	–	2119.79	299.24	11349	31	30905	3018	1878	8	11
eilB76	75	100	14	984.13	1009.04	965.42	1049.19	7.98%	–	2849.33	2835.72	14317	44	24756	1040	1230	5	15
eilC76	75	180	8	721.39	738.67	718.19	756.78	5.10%	–	5067.64	5023.31	10568	48	37076	3964	1806	13	**8**
eilD76	75	220	7	672.34	687.60	**673.97**	694.38	2.94%	–	420.93	4335.58	30748	132	52378	15333	600	7	**7**
eilA101	100	200	8	804.27	826.14	**804.65**	891.54	9.75%	–	500.88	76.70	2375	34	16860	3226	800	2	10
eilB101	100	112	14	1055.59	1076.26	1027.42	1142.33	10.06%	–	134.97	215.66	4433	39	18486	775	709	6	**14**
S51D1	50	160	3	457.08	459.50	**459.50**	459.50	⋆	14.46	5.37	7.71	118	28	255	190	100	1	**3**
S51D2	50	160	9	697.00	709.29	**708.41**	**708.42**	⋆	2459.86	1323.06	1371.25	84492	46	15815	526	2545	17	**9**
S51D3	50	160	15	933.97	948.06	**940.38**	**947.97**	0.80%	–	845.44	3582.93	113163	94	19539	524	1909	14	**15**
S51D4	50	160	27	1545.19	1562.01	**1560.87**	**1560.88**	⋆	6779.78	5927.21	6589.23	177407	132	23361	116	4613	31	**27**
S51D5	50	160	23	1316.93	1333.67	**1330.61**	1333.67	0.23%	–	5177.11	5878.78	176889	137	13059	62	2931	18	**23**
S51D6	50	160	41	2149.55	2169.10	**2164.39**	2169.10	0.22%	–	8.72	3330.55	173227	113	14042	1	203	1	**41**
S76D1	75	160	4	590.92	598.94	**598.94**	598.94	⋆	1111.36	397.45	916.88	11865	47	17361	10520	1873	6	**4**
S76D2	75	160	15	1066.88	1087.40	1048.24	1130.87	7.31%	–	39.00	2841.07	15723	30	24612	1020	684	7	16
S76D3	75	160	23	1406.85	1427.86	1388.19	1508.52	7.98%	–	2417.63	2351.84	28204	73	24234	572	1447	8	24
S76D4	75	160	37	2053.66	2079.76	2048.48	2166.93	5.47%	–	1969.10	1955.20	41058	39	20481	344	2274	12	39
S101D1	100	160	5	714.50	726.59	**716.17**	730.11	1.91%	–	829.31	6596.96	2847	42	21709	13405	600	3	**5**
S101D2	100	160	20	1356.78	1378.43	1313.63	1508.09	12.89%	–	903.75	830.63	4952	38	22710	706	1808	5	21
S101D3	100	160	31	1845.07	1874.81	1818.19	2044.71	11.08%	–	1043.11	46.65	5554	47	17146	647	1015	6	35
S101D5	100	160	48	2758.21	2791.22	2742.40	3149.42	12.92%	–	1022.30	14.89	5972	19	7922	98	2422	6	55
p01_1030	50	160	11	750.45	770.19	**756.70**	**756.71**	⋆	5446.83	5270.76	5356.94	110712	160	19509	707	2923	23	**11**
p01_1050	50	160	16	996.72	1017.18	992.10	1022.47	2.97%	–	414.53	1184.59	104332	59	26604	1152	2680	18	**16**
p01_1090	50	160	26	1471.54	1489.37	**1481.96**	**1487.18**	0.35%	–	6305.77	7200.10	180792	121	17651	107	914	10	**26**
p01_110	50	160	3	456.63	459.50	**459.50**	459.50	⋆	12.24		3.90	54	17	349	119	0	0	**3**
p01_3070	50	160	26	1466.34	1499.29	**1472.57**	**1483.41**	0.73%	–	1543.89	6990.69	177861	106	30935	145	1538	15	**26**
p01_7090	50	160	41	2134.96	2166.30	**2145.05**	**2155.80**	0.50%	–	10.57	6623.59	181091	251	12674	0	203	2	**41**
p02_1030	75	140	16	1095.65	1121.82	1069.55	1185.60	9.79%	–	2579.02	2563.32	20418	26	25736	1240	1820	9	17
p02_1050	75	140	24	1482.50	1514.39	1466.18	1538.53	4.70%	–	2897.70	2816.46	16973	43	17035	346	2667	10	25
p02_1090	75	140	40	2272.05	2318.28	2270.29	2339.77	2.97%	–	3631.24	4258.40	44395	43	15952	216	2502	10	43
p02_110	75	140	5	604.70	652.93	**606.41**	**642.74**	5.65%	–	3028.78	1269.88	6376	44	32889	9137	1578	9	**5**
p02_3070	75	140	39	2195.44	2237.19	2190.29	2280.40	3.95%	–	3102.10	3148.87	34329	57	18426	162	1991	13	40
p02_7090	75	140	61	3177.20	3258.15	**3193.43**	**3235.51**	1.30%	–	377.68	304.51	31191	73	10992	2	607	4	**61**
p03_1030	100	200	22	1437.78	1477.35	1395.03	1602.12	12.93%	–	6917.61	96.17	5409	60	19147	856	3208	5	24
p03_1050	100	200	33	1971.34	2040.92	1941.94	2228.47	12.86%	–	1379.40	19.19	5677	43	12171	592	2426	6	36
p03_1090	100	200	56	3042.93	3127.06	3039.99	3154.43	3.63%	–	1251.84	1123.20	5820	24	9396	171	1721	7	57
p03_110	100	200	6	743.06	788.23	**745.39**	800.25	6.86%	–	6187.78	5868.28	4064	27	21141	7542	2206	6	7
p03_3070	100	200	53	2945.42	3030.66	2938.92	3128.27	6.05%	–	3083.10	2749.04	5536	49	9416	47	3127	9	54
p03_7090	100	200	82	4319.16	4467.59	**4335.39**	**4437.15**	2.29%	–	986.13	5742.52	6425	158	4349	0	907	5	83
SD1	8	100	6	228.28	228.28	228.28	228.28		0.34		0.30	0	27	28	0	0	0	**6**
SD2	16	100	12	708.28	708.28	708.28	708.28		0.63		0.56	0	73	36	3	0	0	**12**
SD3	16	100	12	430.40	430.58	**430.58**	430.58	⋆	0.67		0.65	0	68	48	0	0	0	**12**
SD4	24	100	18	630.62	631.05	**631.05**	631.05	⋆	1.98	1.16	1.92	0	152	101	0	50	1	**18**
SD5	32	100	24	1373.25	1390.57	**1390.56**	1390.57	⋆	72.23	8.02	57.10	4102	161	777	29	476	9	**24**
SD6	32	100	24	830.86	831.24	**831.24**	831.24	⋆	7.85	3.29	7.04	0	134	258	0	194	2	**24**
SD7	40	100	30	3638.45	3640.00	**3639.96**	3640.00	⋆	33.21	5.65	16.78	871	32	451	9	240	2	**30**
SD8	48	100	36	5068.28	5068.28	5068.28	5068.28		102.11	84.72	97.77	1983	30	1134	22	965	6	**36**
SD9	48	100	36	2028.08	2044.20	**2044.18**	2044.20	⋆	322.30	10.91	177.03	12639	75	3191	37	388	4	**36**
SD10	64	100	48	2678.83	2684.88	**2684.86**	2684.88	⋆	4066.66	430.87	2579.66	69810	102	9915	61	641	5	**48**
SD11	80	100	60	13111.11	13280.00	**13279.88**	13280.00	⋆	2211.45	2142.96	1556.85	8494	21	5277	113	2343	7	**60**
SD12	80	100	60	7089.27	7270.87	**7136.02**	7279.97	1.98%	–	86.30	71.88	35330	35	9263	166	968	3	**60**
SD13	96	100	72	9969.93	10110.58	**10000.23**	10272.13	2.65%	–	3160.25	3184.03	13457	38	8094	84	1156	4	75

⋆ Instance solved to proven optimality for the first time

Table EC.4 Results of the BP method of Archetti et al. (2011) and the proposed BC algorithm on the SDVRP benchmark instances with non-rounded distances and unlimited fleet (SDVRP-UF) – time limit: 7200 sec.

			Archetti, Bianchessi, and Speranza (2011)			Proposed BC		
Instance	n	Q	*LB*	*UB*	*Gap*	*LB*	*UB*	*Gap*
eil22	21	6000	–	–	–	**375.28**	**375.28**	⋆
eil23	22	4500	451.80	–	–	**568.56**	**568.56**	⋆
eil30	29	4500	218.92	–	–	**505.01**	**505.01**	⋆
eil33	32	8000	–	–	–	**837.05**	**837.06**	⋆
eil51	50	160	518.23	527.98	1.88%	**524.61**	**524.61**	⋆
eilA76	75	140	**809.58**	**832.71**	**2.86%**	798.43	875.96	8.85%
eilB76	75	100	**984.13**	**1047.17**	**6.41%**	965.42	1049.19	7.98%
eilC76	75	180	**721.39**	765.30	6.09%	718.19	**756.78**	**5.10%**
eilD76	75	220	672.34	705.26	4.90%	**673.97**	**694.38**	**2.94%**
eilA101	100	200	804.27	**854.03**	**6.19%**	**804.65**	891.54	9.75%
eilB101	100	112	**1055.59**	**1119.17**	**6.02%**	1027.42	1142.33	10.06%
S51D1	50	160	457.08	459.50	0.53%	**459.50**	459.50	⋆
S51D2	50	160	697.00	717.18	2.89%	**708.41**	**708.42**	⋆
S51D3	50	160	933.97	960.38	2.83%	**940.38**	**947.97**	**0.80%**
S51D4	50	160	1545.19	1569.92	1.60%	**1560.87**	**1560.88**	⋆
S51D5	50	160	1316.93	1339.38	1.70%	**1330.61**	**1333.67**	0.23%
S51D6	50	160	2149.55	2182.13	1.52%	**2164.39**	**2169.10**	**0.22%**
S76D1	75	160	590.92	633.83	7.26%	**598.94**	**598.94**	⋆
S76D2	75	160	**1066.88**	**1104.56**	**3.53%**	1048.24	1130.87	7.31%
S76D3	75	160	**1406.85**	**1435.10**	**2.01%**	1388.19	1508.52	7.98%
S76D4	75	160	**2053.66**	**2106.64**	**2.58%**	2048.48	2166.93	5.47%
S101D1	100	160	714.50	791.11	10.72%	**716.17**	**730.11**	**1.91%**
S101D2	100	160	**1356.78**	**1426.15**	**5.11%**	1313.63	1508.09	12.89%
S101D3	100	160	**1845.07**	**1911.40**	**3.60%**	1818.19	2044.71	11.08%
S101D5	100	160	**2758.21**	**2824.16**	**2.39%**	2742.40	3149.42	12.92%
p01_1030	50	160	750.45	770.19	2.63%	**756.70**	**756.71**	⋆
p01_1050	50	160	**996.72**	**1017.18**	**2.05%**	992.10	1022.47	2.97%
p01_1090	50	160	1471.54	1489.37	1.21%	**1481.96**	**1487.18**	0.35%
p01_110	50	160	456.63	459.50	0.63%	**459.50**	459.50	⋆
p01_3070	50	160	1466.34	1499.29	2.25%	**1472.57**	**1483.41**	**0.73%**
p01_7090	50	160	2134.96	2166.30	1.47%	**2145.05**	**2155.80**	**0.50%**
p02_1030	75	140	**1095.65**	**1121.82**	**2.39%**	1069.55	1185.60	9.79%
p02_1050	75	140	**1482.50**	**1514.39**	**2.15%**	1466.18	1538.53	4.70%
p02_1090	75	140	**2272.05**	**2318.28**	**2.03%**	2270.29	2339.77	2.97%
p02_110	75	140	604.70	652.93	7.98%	**606.41**	**642.74**	**5.65%**
p02_3070	75	140	**2195.44**	**2237.19**	**1.90%**	2190.29	2280.40	3.95%
p02_7090	75	140	3177.20	3258.15	2.55%	**3193.43**	**3235.51**	**1.30%**
p03_1030	100	200	**1437.78**	**1477.35**	**2.75%**	1395.03	1602.12	12.93%
p03_1050	100	200	**1971.34**	**2040.92**	**3.53%**	1941.94	2228.47	12.86%
p03_1090	100	200	**3042.93**	**3127.06**	**2.76%**	3039.99	3154.43	3.63%
p03_110	100	200	743.06	**788.23**	**6.08%**	**745.39**	800.25	6.86%
p03_3070	100	200	**2945.42**	**3030.66**	**2.89%**	2938.92	3128.27	6.05%
p03_7090	100	200	4319.16	4467.59	3.44%	**4335.39**	**4437.15**	**2.29%**

⋆ Instance solved to proven optimality for the first time

– Not available

Table EC.5 Detailed results of the proposed BC algorithm on the SDVRP benchmark instances with rounded distances and unlimited fleet (SDVRP-UF-r) – time limit: 7200 sec.

				Best known		BC algorithm												
Instance	n	Q	$\overline{m}$	LB	UB	LB	UB	Gap	T_{total}	T_{best}^{MIP}	T_{best}^{CPL}	*Nodes*	*Check*	*Cap*	*Conn*	*Reg*	F/O	nR
eil22	21	6000	4	–	375.00	**375.00**	375.00		3.64	3.32	3.36	0	45	102	12	170	4	**4**
eil23	22	4500	3	–	569.00	**569.00**	569.00		2.48	2.40	2.45	0	45	39	42	88	6	**3**
eil30	29	4500	3	–	503.00	**503.00**	503.00		2.19		1.32	0	7	84	400	0	0	4
eil33	32	8000	4	–	835.00	**835.00**	835.00		23.64	18.72	12.81	75	48	321	276	741	16	**4**
eil51	50	160	5	–	521.00	**521.00**	521.00		86.13	53.22	68.38	2349	45	2373	793	400	6	**5**
eilA76	75	140	10	–	818.00	**791.96**	849.00	6.72%	–	337.68	284.48	15394	30	31949	4501	918	9	11
eilB76	75	100	14	–	1002.00	**958.01**	1055.00	9.19%	–	2330.58	2466.92	19562	53	24928	2565	1287	8	**14**
eilC76	75	180	8	–	733.00	**711.89**	776.00	8.26%	–	79.23	45.58	11687	40	33681	5443	1200	5	**8**
eilD76	75	220	7	–	682.00	**668.87**	684.00	2.21%	–	1404.39	1506.98	37050	35	44773	9255	378	6	**7**
eilA101	100	200	8	–	814.00	**792.68**	895.00	11.43%	–	7046.55	568.93	2129	30	13130	2976	2806	6	9
eilB101	100	112	14	–	1061.00	**1014.62**	1106.00	8.26%	–	109.72	43.50	4792	22	16174	1426	406	3	**14**
S51D1	50	160	3	454.00	458.00	**458.00**	458.00	⋆	11.00		4.26	73	13	213	216	0	0	**3**
S51D2	50	160	9	676.63	703.00	**703.00**	703.00	⋆	1091.55	645.55	677.31	27858	66	8742	1111	3207	27	**9**
S51D3	50	160	15	905.22	943.00	**933.76**	**942.00**	0.87%	–	5929.89	6061.55	195326	63	21971	1508	2264	19	**15**
S51D4	50	160	27	1520.67	1553.00	**1551.00**	**1551.00**	⋆	5282.80	2057.76	3735.23	112399	60	8930	377	2628	18	**27**
S51D5	50	160	23	1272.86	1328.00	**1320.73**	1328.00	0.55%	–	584.32	1590.22	175192	104	10357	855	2368	15	**23**
S51D6	50	160	41	2113.03	2163.00	**2153.00**	**2153.00**	⋆	4609.30	5.53	1716.58	174658	81	9245	42	200	2	**41**
S76D1	75	160	4	584.87	592.00	**592.00**	592.00	⋆	144.11	38.05	69.05	2180	27	2762	1039	300	2	**4**
S76D2	75	160	15	1020.32	1082.00	**1040.67**	1129.00	7.82%	–	2362.02	2250.54	23148	41	24643	1997	1581	8	17
S76D3	75	160	23	1346.29	1420.00	**1378.11**	1467.00	6.06%	–	324.51	1606.95	31808	47	17962	1349	1128	10	**23**
S76D4	75	160	37	2011.64	2073.00	**2029.93**	2149.00	5.54%	–	1409.44	211.31	56244	34	12748	1646	2501	14	39
S101D1	100	160	5	700.56	716.00	**711.12**	717.00	0.82%	–	1627.19	5135.51	55400	60	39626	28700	2000	6	**5**
S101D2	100	160	20	1270.97	1366.00	**1295.68**	1494.00	13.27%	–	6691.96	42.62	5483	25	21114	1479	2626	12	22
S101D3	100	160	31	1739.66	1864.00	**1806.08**	1990.00	9.24%	–	614.12	2.65	6786	8	8784	1604	2821	6	34
S101D5	100	160	48	2630.43	2770.00	**2712.19**	2888.00	6.09%	–	6241.96	9.46	8842	13	3488	933	4106	8	51
p01_1030	50	160	11	–	–	**752.00**	**753.00**	0.13%	–	1067.06	4323.64	330544	64	21067	2456	3322	19	**11**
p01_1050	50	160	16	–	–	**986.83**	**999.00**	1.22%	–	5307.98	4510.38	168333	81	20320	1788	5123	28	**16**
p01_1090	50	160	26	–	–	**1480.00**	**1480.00**	⋆	6480.17	1065.01	1720.69	303336	56	11872	829	1062	13	**26**
p01_110	50	160	3	–	–	**458.00**	**458.00**	⋆	21.41	10.58	11.19	52	19	334	212	200	2	**3**
p01_3070	50	160	26	–	–	**1466.98**	**1473.00**	0.41%	–	3095.18	4353.14	252771	99	11088	435	1774	15	**26**
p01_7090	50	160	41	–	–	**2135.38**	**2143.00**	0.36%	–	13.83	377.47	241957	63	12388	19	161	0	**41**
p02_1030	75	140	16	–	–	**1060.60**	**1144.00**	7.29%	–	4463.38	6501.74	27793	34	25888	1816	1728	9	**16**
p02_1050	75	140	24	–	–	**1453.40**	**1571.00**	7.49%	–	255.63	243.95	35834	44	17360	1654	987	6	25
p02_1090	75	140	40	–	–	**2256.86**	**2296.00**	1.70%	–	2305.21	2292.93	56871	36	9773	687	2430	11	41
p02_110	75	140	5	–	–	**604.00**	**612.00**	1.31%	–	157.65	6942.07	43156	59	57392	17416	750	3	**5**
p02_3070	75	140	39	–	–	**2173.93**	**2277.00**	4.53%	–	1095.36	1159.93	67894	35	12938	866	1358	10	**39**
p02_7090	75	140	61	–	–	**3171.67**	**3245.00**	2.26%	–	271.81	1567.47	55139	65	11047	2	755	4	**61**
p03_1030	100	200	22	–	–	**1376.64**	**1558.00**	11.64%	–	1398.51	10.05	5592	12	23377	1269	703	9	24
p03_1050	100	200	33	–	–	**1926.50**	**2163.00**	10.93%	–	6886.99	597.05	6116	60	6718	835	2728	7	36
p03_1090	100	200	56	–	–	**3009.58**	**3213.00**	6.33%	–	3987.88	289.58	10316	30	4309	633	2801	4	59
p03_110	100	200	6	–	–	**736.11**	**783.00**	5.99%	–	3924.57	554.68	5465	26	18336	14602	2509	9	7
p03_3070	100	200	53	–	–	**2908.23**	**3144.00**	7.50%	–	4253.43	695.80	10324	20	3162	544	2321	6	56
p03_7090	100	200	82	–	–	**4283.43**	**4408.00**	2.83%	–	4490.32	4759.08	11264	46	4069	22	1209	5	**82**
SD1	8	100	6	–	–	**228.00**	**228.00**	⋆	0.27		0.24	0	19	28	0	0	0	**6**
SD2	16	100	12	–	–	**708.00**	**708.00**	⋆	3.59	3.29	3.55	0	26	55	0	32	0	**12**
SD3	16	100	12	–	–	**432.00**	**432.00**	⋆	0.81		0.78	0	61	44	0	0	0	**12**
SD4	24	100	18	–	–	**630.00**	**630.00**	⋆	3.99	3.24	3.87	0	48	68	0	52	1	**18**
SD5	32	100	24	–	–	**1392.00**	**1392.00**	⋆	28.80	5.04	5.25	445	154	602	12	128	3	**24**
SD6	32	100	24	–	–	**832.00**	**832.00**	⋆	6.15	2.93	5.37	0	116	151	0	64	1	**24**
SD7	40	100	30	–	–	**3640.00**	**3640.00**	⋆	55.08	38.72	46.92	3700	36	1113	26	769	5	**30**
SD8	48	100	36	–	–	**5068.00**	**5068.00**	⋆	40.03	9.02	22.83	329	14	268	7	200	3	**36**
SD9	48	100	36	–	–	**2046.00**	**2046.00**	⋆	221.03	58.48	69.39	7261	83	2570	43	347	3	**36**
SD10	64	100	48	–	–	**2688.00**	**2688.00**	⋆	5174.09	124.70	608.19	277824	57	10609	176	962	6	**48**
SD11	80	100	60	–	–	**13280.00**	**13280.00**	⋆	1131.10	807.46	840.94	7554	12	4453	25	961	4	**60**
SD12	80	100	60	–	–	**7135.64**	**7227.00**	1.26%	–	1754.40	1665.90	33845	23	10055	257	565	4	**60**
SD13	96	100	72	–	–	**10002.80**	**10246.00**	2.37%	–	215.92	218.20	11360	12	6874	110	1831	3	74

⋆ Instance solved to proven optimality for the first time

Table EC.6 Detailed results of the proposed BC algorithm on the SDVRP benchmark instances with non-rounded distances and limited fleet (SDVRP-LF) – time limit: 7200 sec.

			Best known		BC algorithm											
Instance	n	Q	LB	UB	LB	UB	Gap	T_{total}	T_{best}^{MIP}	T_{best}^{CPL}	$Nodes$	$Check$	Cap	$Conn$	Reg	F/O
eil22	21	6000	375.28	375.28	375.28	375.28		3.92	3.53	3.78	0	10	129	10	240	5
eil23	22	4500	568.56	568.56	568.56	568.56		4.56	4.37	4.49	0	42	31	76	119	8
eil30	29	4500	512.72	512.72	512.72	512.72		59.39	2.53	33.09	23094	60	522	2621	2743	48
eil33	32	8000	837.06	837.06	837.05	837.06		100.35	37.41	81.18	17394	87	1248	512	2921	38
eil51	50	160	524.61	524.61	524.61	524.61		419.19	34.40	404.81	3370	56	3121	537	3618	21
eilA76	75	140	809.67	823.89	796.87	–	–	–			4291	19	14605	1224	6015	17
eilB76	75	100	985.42	1009.04	966.92	–	–	–			6402	14	14160	826	6238	12
eilC76	75	180	724.62	738.67	717.79	–	–	–			4314	32	17432	3447	7599	31
eilD76	75	220	677.32	687.60	667.36	742.49	10.12%	–	6744.23	6037.55	5410	27	17987	5065	7744	19
eilA101	100	200	810.06	826.14	806.95	–	–	–			0	7	1320	376	3005	7
eilB101	100	112	1055.40	1076.26	1029.06	–	–	–			0	6	1002	138	8416	6
S51D1	50	160	459.50	459.50	459.50	459.50		12.89		6.07	96	3	262	222	0	1
S51D2	50	160	708.42	708.42	703.27	709.52	0.88%	–	830.76	1527.72	117715	76	21351	1214	7123	45
S51D3	50	160	940.82	947.97	940.09	949.89	1.03%	–	481.11	6544.62	138475	39	17976	360	3005	16
S51D4	50	160	1553.47	1560.88	1552.53	1573.03	1.30%	–		2133.87	116837	34	18698	246	2139	10
S51D5	50	160	1318.90	1333.67	**1327.84**	1333.67	0.44%	–	5848.11	7176.84	222726	93	18481	303	2122	9
S51D6	50	160	2154.70	2169.10	**2163.43**	2169.11	0.26%	–	190.96	3765.06	123865	64	13268	0	867	7
S76D1	75	160	598.94	598.94	598.93	598.94		846.22	48.44	340.25	14990	21	14101	5824	1356	6
S76D2	75	160	1071.30	1087.40	1046.94	–	–	–			6149	5	13611	551	1278	5
S76D3	75	160	1407.54	1427.86	1386.38	–	–	–			10732	12	11976	608	6770	11
S76D4	75	160	2059.80	2079.76	2049.40	–	–	–			30295	13	19130	641	1740	5
S101D1	100	160	726.59	726.59	716.84	–	–	–			0	4	929	1230	1299	3
S101D2	100	160	1358.90	1378.43	1315.50	–	–	–			0	3	1274	74	4894	3
S101D3	100	160	1853.10	1874.81	1818.81	–	–	–			0	0	1378	55	0	0
S101D5	100	160	2767.60	2791.22	2737.76	–	–	–			0	0	859	0	0	0
p01_1030	50	160	756.71	756.71	746.51	759.99	1.77%	–	2462.33	3146.14	85789	77	31153	2404	6714	48
p01_1050	50	160	996.93	1005.75	992.21	1013.94	2.14%	–	3015.23	2962.66	87898	50	34800	1040	10002	38
p01_1090	50	160	1472.58	1487.41	**1482.30**	**1487.18**	0.33%	–	927.30	6826.00	159012	162	11931	113	1264	7
p01_110	50	160	459.50	459.50	459.50	459.50		15.85	5.73	10.66	48	12	371	140	148	1
p01_3070	50	160	1467.09	1481.71	**1475.40**	1481.72	0.43%	–	554.31	1023.89	195962	84	22043	102	2781	9
p01_7090	50	160	2133.94	2162.58	**2145.83**	**2155.80**	0.46%	–	35.33	4804.56	158088	244	14813	0	202	2
p02_1030	75	140	1093.56	1122.91	1068.58	–	–	–			8934	18	14467	606	7417	18
p02_1050	75	140	1483.17	1509.79	1462.82	–	–	–			11515	6	13812	658	4593	6
p02_1090	75	140	2270.44	2372.22	2264.29	–	–	–			14730	1	12159	270	451	1
p02_110	75	140	616.58	617.85	609.77	617.85	1.31%	–	219.21	6806.42	40278	36	57043	11655	2328	8
p02_3070	75	140	2192.25	2235.61	2179.93	2294.49	4.99%	–	5773.70	5485.70	17811	6	11523	197	831	3
p02_7090	75	140	3177.69	3259.36	3176.27	3267.80	2.80%	–	4980.46	4797.18	35721	74	14529	0	1136	6
p03_1030	100	200	1435.23	1491.82	1396.94	–	–	–			0	4	1220	44	7701	4
p03_1050	100	200	1971.43	2018.09	1943.27	–	–	–			0	0	1271	5	0	0
p03_1090	100	200	3043.27	3136.29	3029.04	–	–	–			0	0	614	0	0	0
p03_110	100	200	753.12	762.40	744.47	–	–	–			0	4	1134	516	2601	4
p03_3070	100	200	2945.76	3044.92	2923.82	–	–	–			0	0	618	4	0	0
p03_7090	100	200	4316.42	4452.55	4308.69	–	–	–			0	2	247	0	1489	1
SD1	8	100	228.28	228.28	228.28	228.28		0.37		0.35	0	25	25	0	13	1
SD2	16	100	708.28	708.28	708.28	708.28		1.94	1.91	1.92	0	9	28	0	32	0
SD3	16	100	430.58	430.58	430.58	430.58		0.57		0.55	0	26	4	0	0	0
SD4	24	100	631.05	631.05	631.05	631.05		1.37		1.25	0	11	54	0	30	1
SD5	32	100	1390.57	1390.57	1390.57	1390.57		35.07	27.02	29.83	920	39	497	5	478	6
SD6	32	100	831.24	831.24	831.24	831.24		5.46		4.76	0	34	160	0	0	0
SD7	40	100	3640.00	3640.00	3640.00	3640.00		90.51	77.90	88.02	1	15	215	0	290	2
SD8	48	100	5068.28	5068.28	5068.28	5068.28		235.63		227.82	49	16	277	1	0	0
SD9	48	100	2044.20	2044.20	2044.20	2044.20		207.35		63.80	8146	82	1597	14	0	0
SD10	64	100	2660.83	2684.88	**2684.88**	2684.88	⋆	6976.94	2520.43	4965.02	69916	133	9614	92	1743	8
SD11	80	100	13280.00	13280.00	13280.00	13280.00		6716.26		6265.55	1348	5	2860	22	1192	3
SD12	80	100	7175.80	7213.61	7068.49	–	–	–			2296	3	4982	4	1192	3
SD13	96	100	10053.60	10110.58	9927.70	–	–	–			0	2	434	2	2310	2

⋆ Instance solved to proven optimality for the first time

Table EC.7 Results of the state-of-the-art and proposed BC algorithms on the SDVRP benchmark instances with non-rounded distances and limited fleet (SDVRP-LF) – time limit: 7200 sec.

			SCF-based BC (Archetti et al. 2014)				2VF-based BC (Archetti et al. 2014)				Proposed BC			
Instance	n	Q	*LB*	*UB*	*Gap*	T_{total}	*LB*	*UB*	*Gap*	T_{total}	*LB*	*UB*	*Gap*	T_{total}
eil22	21	6000	375.28	375.28		0.10	375.28	375.28		0.00	375.28	375.28		3.92
eil23	22	4500	568.56	568.56		0.20	568.56	568.56		0.00	568.56	568.56		4.56
eil30	29	4500	512.72	512.72		230.60	512.72	512.72		8.50	512.72	512.72		59.39
eil33	32	8000	837.06	837.06		111.90	837.06	837.06		2.10	837.05	837.06		100.35
eil51	50	160	524.61	524.61		590.20	524.61	524.61		16.70	524.61	524.61		419.19
eilA76	75	140	804.48	–	–	–	**801.74**	**827.23**	**3.18%**	–	796.87	–	–	–
eilB76	75	100	959.21	–	–	–	955.75	**1035.79**	**8.37%**	–	**966.92**	–	–	–
eilC76	75	180	717.20	820.11	14.35%	–	**724.62**	**739.69**	**2.08%**	–	717.79	–	–	–
eilD76	75	220	664.65	726.89	9.36%	–	**676.16**	**692.01**	**2.34%**	–	667.36	742.49	10.12%	–
eilA101	100	200	804.48	–	–	–	**809.60**	**856.40**	**5.78%**	–	806.95	–	–	–
eilB101	100	112	1021.75	–	–	–	1016.75	–	–	–	**1029.06**	–	–	–
S51D1	50	160	459.50	459.50		3.10	459.50	459.50		0.50	459.50	459.50		12.89
S51D2	50	160	689.70	–	–	–	708.42	708.42		368.00	703.27	709.52	0.88%	–
S51D3	50	160	926.20	–	–	–	**940.82**	951.43	1.13%	–	940.09	**949.89**	**1.03%**	–
S51D4	50	160	1535.75	1563.94	1.84%	–	**1553.47**	1560.88	**0.48%**	–	1552.53	1573.03	1.30%	–
S51D5	50	160	1314.81	1333.67	1.43%	–	1317.00	1335.50	1.41%	–	**1327.84**	**1333.67**	**0.44%**	–
S51D6	50	160	2115.95	–	–	–	2135.21	2270.71	6.35%	–	**2163.43**	**2169.11**	**0.26%**	–
S76D1	75	160	593.78	603.69	1.67%		598.94	598.94		164.30	598.93	598.94		846.22
S76D2	75	160	1038.34	–	–	–	1034.96	**1111.00**	**7.35%**	–	**1046.94**	–	–	–
S76D3	75	160	1378.98	–	–	–	1364.35	**1591.08**	**16.62%**	–	**1386.38**	–	–	–
S76D4	75	160	2025.19	–	–	–	1954.46	**2333.53**	19.40%	–	**2049.40**	–	–	–
S101D1	100	160	715.97	770.20	7.57%	–	**726.59**	**726.59**		2635.90	716.84	–	–	–
S101D2	100	160	1307.14	–	–	–	1291.17	–	–	–	**1315.50**	–	–	–
S101D3	100	160	1761.95	–	–	–	1675.73	–	–	–	**1818.81**	–	–	–
S101D5	100	160	2647.53	–	–	–	2553.61	–	–	–	**2737.76**	–	–	–
p01_1030	50	160	739.01	772.89	4.59%	–	**756.71**	**756.71**		915.30	746.51	759.99	1.77%	–
p01_1050	50	160	983.89	–	–	–	987.18	**1005.75**	**1.88%**	–	**992.21**	1013.94	2.14%	–
p01_1090	50	160	1456.76	–	–	–	1472.58	1497.41	1.01%	–	**1482.30**	**1487.18**	**0.33%**	–
p01_110	50	160	459.50	459.50		2.90	459.50	459.50		0.30	459.50	459.50		15.85
p01_3070	50	160	1457.01	1490.46	2.30%	–	1453.95	1485.46	2.17%	–	**1475.40**	**1481.72**	**0.43%**	–
p01_7090	50	160	2109.96	–	–	–	2121.22	2331.58	9.92%	–	**2145.83**	**2155.80**	**0.46%**	–
p02_1030	75	140	1062.59	–	–	–	1053.04	–	–	–	**1068.58**	–	–	–
p02_1050	75	140	1450.61	–	–	–	1434.43	**1531.58**	**6.77%**	–	**1462.82**	–	–	–
p02_1090	75	140	2245.50	–	–	–	2076.61	–	–	–	**2264.29**	–	–	–
p02_110	75	140	604.01	–	–	–	**616.58**	**617.85**	**0.21%**	–	609.77	617.85	1.31%	–
p02_3070	75	140	2167.25	–	–	–	2143.89	–	–	–	**2179.93**	**2294.49**	**4.99%**	–
p02_7090	75	140	3114.30	–	–	–	3090.88	–	–	–	**3176.27**	**3267.80**	**2.80%**	–
p03_1030	100	200	1380.59	–	–	–	1315.01	–	–	–	**1396.94**	–	–	–
p03_1050	100	200	1884.91	–	–	–	1779.59	–	–	–	**1943.27**	–	–	–
p03_1090	100	200	2950.05	–	–	–	2943.83	–	–	–	**3029.04**	–	–	–
p03_110	100	200	745.24	–	–	–	**752.29**	**762.40**	**1.34%**	–	744.47	–	–	–
p03_3070	100	200	2846.35	–	–	–	2747.80	–	–	–	**2923.82**	–	–	–
p03_7090	100	200	4212.43	–	–	–	4211.88	–	–	–	**4308.69**	–	–	–
SD1	8	100	228.28	228.28		0.00	228.28	228.28		0.00	228.28	228.28		0.37
SD2	16	100	708.28	708.28		0.40	708.28	708.28		0.40	708.28	708.28		1.94
SD3	16	100	430.58	430.58		0.20	430.58	430.58		0.00	430.58	430.58		0.57
SD4	24	100	631.05	631.05		1.40	631.05	631.05		0.10	631.05	631.05		1.37
SD5	32	100	1390.57	1390.57		18.80	1390.57	1390.57		18.80	1390.57	1390.57		35.07
SD6	32	100	831.24	831.24		1.70	831.24	831.24		1.70	831.24	831.24		5.46
SD7	40	100	3640.00	3640.00		90.60	3640.00	3640.00		137.20	3640.00	3640.00		90.51
SD8	48	100	5068.28	5068.28		339.20	5068.28	5068.28		79.10	5068.28	5068.28		235.63
SD9	48	100	2044.20	2044.20		931.90	2044.20	2044.20		514.90	2044.20	2044.20		207.35
SD10	64	100	2660.64	2684.88	0.91%	–	2625.77	2716.26	3.45%	–	**2684.88**	2684.88	⋆	6976.94
SD11	80	100	12831.96	13280.00	3.49%	–	13174.03	13280.00	0.80%	–	**13280.00**	13280.00		6716.26
SD12	80	100	6942.46	–	–	–	6984.07	–	–	–	**7068.49**	–	–	–
SD13	96	100	9760.80	–	–	–	9610.76	–	–	–	**9927.70**	–	–	–

⋆ Instance solved to proven optimality for the first time

– Not available (*UB* or *Gap*); Solver reached the time limit of 7200 seconds (T_{total})

Table EC.8 Detailed results of the proposed BC algorithm on the SDVRP benchmark instances with rounded distances and limited fleet (SDVRP-LF-r) – time limit: 7200 sec.

			Best known		BC algorithm											
Instance	n	Q	*LB*	*UB*	*LB*	*UB*	*Gap*	T_{total}	T_{best}^{MIP}	T_{best}^{CPL}	*Nodes*	*Check*	*Cap*	*Conn*	*Reg*	*F/O*
eil22	21	6000	375.00	375.00	375.00	375.00		3.85	3.58	3.79	0	16	154	10	432	9
eil23	22	4500	569.00	569.00	569.00	569.00		2.80	2.21	2.74	0	23	48	50	176	5
eil30	29	4500	510.00	510.00	510.00	510.00		75.24	65.92	71.67	18968	77	790	5700	2823	54
eil33	32	8000	835.00	835.00	835.00	835.00		13.53	9.10	9.65	91	16	315	273	576	6
eil51	50	160	521.00	521.00	521.00	521.00		339.28	322.09	326.00	1587	50	1888	905	4730	22
eilA76	75	140	807.60	818.00	789.73	–	–	–			4918	27	13665	2158	6836	19
eilB76	75	100	981.40	1002.00	958.07	–	–	–			6279	13	12646	1480	6007	11
eilC76	75	180	717.80	733.00	710.13	772.00	8.01%	–		7004.45	3708	19	13053	2564	4365	17
eilD76	75	220	666.10	682.00	661.69	716.00	7.59%	–	5274.62	5508.82	5535	162	15787	7033	5862	155
eilA101	100	200	799.80	814.00	794.60	–	–	–			0	5	719	393	2197	5
eilB101	100	112	1040.60	1061.00	1015.79	–	–	–			0	6	820	100	8416	6
S51D1	50	160	458.00	458.00	458.00	458.00		356.22	11.47	353.30	94	29	302	412	700	9
S51D2	50	160	703.00	703.00	703.00	703.00		1623.98	919.03	1554.34	46939	52	11028	1014	3353	33
S51D3	50	160	930.70	943.00	927.85	947.00	2.02%	–	5913.56	5893.17	94700	65	18733	1424	4882	24
S51D4	50	160	1547.00	1551.00	1541.71	1551.00	0.60%	–	4434.09	5723.09	103271	67	18870	347	1807	11
S51D5	50	160	1313.40	1328.00	**1318.80**	1328.00	0.69%	–	3479.87	3517.90	129597	54	12707	685	2565	14
S51D6	50	160	2141.70	2163.00	**2149.00**	**2153.00**	0.19%	–	601.45	1353.52	155166	64	8635	17	862	7
S76D1	75	160	592.00	592.00	592.00	592.00		5408.84	4870.87	5308.82	7138	170	8222	4148	3303	40
S76D2	75	160	1061.10	1082.00	1035.40	–	–	–			7788	3	15797	914	0	3
S76D3	75	160	1395.90	1420.00	1371.71	–	–	–			12802	11	14200	1227	6470	10
S76D4	75	160	2046.10	2073.00	2019.03	–	–	–			19232	2	8663	1038	1869	2
S101D1	100	160	716.00	716.00	704.10	–	–	–			0	43	928	1625	3402	25
S101D2	100	160	1337.10	1366.00	1298.07	–	–	–			0	3	960	90	4894	3
S101D3	100	160	1832.20	1864.00	1796.01	–	–	–			0	0	822	50	0	0
S101D5	100	160	2737.10	2770.00	2704.46	–	–	–			0	0	898	7	0	0
p01_1030	50	160	753.00	753.00	747.37	755.00	1.01%	–	541.12	4426.55	252954	66	22759	2664	7659	48
p01_1050	50	160	978.63	998.00	**985.10**	1029.00	4.27%	–	1692.33	1498.07	100256	42	22936	1480	6006	20
p01_1090	50	160	1459.58	1481.00	**1480.00**	**1480.00**	⋆	6765.87	2040.81	2440.38	265966	35	13248	763	1961	12
p01_110	50	160	458.00	458.00	458.00	458.00		379.50	11.26	196.79	79	183	341	1330	648	169
p01_3070	50	160	1444.88	1473.00	**1464.99**	1474.00	0.61%	–	3641.86	3580.04	183851	69	11258	289	3094	12
p01_7090	50	160	2086.36	2212.00	**2131.84**	**2142.00**	0.47%	–	30.52	3087.02	174355	80	15822	3	109	2
p02_1030	75	140	1053.88	1157.00	**1058.51**	–	–	–			8695	18	15190	1232	7346	18
p02_1050	75	140	1432.71	–	**1449.88**	–	–	–			12048	4	11516	989	3462	4
p02_1090	75	140	2217.43	–	**2244.55**	–	–	–			12848	0	10727	521	0	0
p02_110	75	140	612.00	612.00	606.29	612.00	0.93%	–	974.51	1762.96	68626	42	42596	13253	1728	6
p02_3070	75	140	2133.77	–	**2157.13**	**2572.00**	16.13%	–	6915.75		14746	1	14198	393	537	0
p02_7090	75	140	3100.65	–	**3154.33**	–	–	–			19021	3	9257	6	1138	3
p03_1030	100	200	1368.38	–	**1378.60**	–	–	–			0	4	1031	60	7701	4
p03_1050	100	200	1899.06	–	**1916.93**	–	–	–			0	0	1115	10	0	0
p03_1090	100	200	2930.02	–	**2999.02**	–	–	–			0	0	532	7	0	0
p03_110	100	200	740.76	760.00	735.20	–	–	–			0	14	883	1141	2601	14
p03_3070	100	200	2838.22	–	**2898.36**	–	–	–			0	0	908	7	0	0
p03_7090	100	200	4117.68	–	**4266.28**	–	–	–			0	2	236	0	1489	1
SD1	8	100	228.00	228.00	228.00	228.00		0.26		0.24	0	23	24	0	13	1
SD2	16	100	708.00	708.00	708.00	708.00		0.85		0.80	0	7	34	2	32	1
SD3	16	100	432.00	432.00	432.00	432.00		0.41		0.40	0	12	16	0	0	0
SD4	24	100	630.00	630.00	630.00	630.00		1.30		1.22	0	7	42	0	30	1
SD5	32	100	1392.00	1392.00	1392.00	1392.00		20.61	5.02	6.23	399	40	468	1	128	1
SD6	32	100	832.00	832.00	832.00	832.00		4.81		4.64	0	6	145	0	0	0
SD7	40	100	3640.00	3640.00	3640.00	3640.00		28.93	10.63	20.82	264	10	445	1	208	1
SD8	48	100	5068.00	5068.00	5068.00	5068.00		23.39	16.91	4.58	7	4	252	0	294	1
SD9	48	100	2046.00	2046.00	2046.00	2046.00		265.44	12.11	46.01	12511	28	3219	57	339	3
SD10	64	100	2652.45	2688.00	**2688.00**	2688.00	⋆	5153.62	317.69	2207.82	63154	133	6764	17	1487	6
SD11	80	100	13280.00	13280.00	13280.00	13280.00		6895.22	6818.78	6852.30	2545	14	4050	16	1674	5
SD12	80	100	7102.36	7315.00	**7108.34**	–	–	–			4707	3	6517	44	1192	3
SD13	96	100	9779.95	–	**9941.77**	–	–	–			0	2	448	1	2310	2

⋆ Instance solved to proven optimality for the first time

Table EC.9 Results of the state-of-the-art and proposed BC algorithms on the SDVRP benchmark instances with rounded distances and limited fleet (SDVRP-LF-r) – time limit: 7200 sec.

			SCF-based BC (Archetti et al. 2014)				2VF-based BC (Archetti et al. 2014)				Proposed BC			
Instance	n	Q	LB	UB	Gap	T_{total}	LB	UB	Gap	T_{total}	LB	UB	Gap	T_{total}
eil22	21	6000	375.00	375.00		0.10	375.00	375.00		0.00	375.00	375.00		3.85
eil23	22	4500	569.00	569.00		0.10	569.00	569.00		0.00	569.00	569.00		2.80
eil30	29	4500	510.00	510.00		47.10	510.00	510.00		13.50	510.00	510.00		75.24
eil33	32	8000	835.00	835.00		16.10	835.00	835.00		1.60	835.00	835.00		13.53
eil51	50	160	521.00	521.00		990.00	521.00	521.00		35.80	521.00	521.00		339.28
eilA76	75	140	782.94	–	–	–	**790.00**	**841.00**	**6.43%**	–	789.73	–	–	–
eilB76	75	100	949.07	–	–	–	937.49	–	–	–	**958.07**	–	–	–
eilC76	75	180	704.25	–	–	–	**712.33**	**758.00**	**6.41%**	–	710.13	772.00	8.01%	–
eilD76	75	220	656.70	695.00	5.83%	–	**664.54**	**688.00**	**3.53%**	–	661.69	716.00	7.59%	–
eilA101	100	200	787.24	–	–	–	**796.02**	**833.00**	**4.65%**	–	794.60	–	–	–
eilB101	100	112	1006.77	–	–	–	991.24	–	–	–	**1015.79**	–	–	–
S51D1	50	160	458.00	458.00		6.80	458.00	458.00		0.50	458.00	458.00		356.22
S51D2	50	160	686.73	703.00	2.37%	–	703.00	703.00		799.70	703.00	703.00		1623.98
S51D3	50	160	917.89	945.00	2.95%	–	926.52	**945.00**	**1.99%**	–	**927.85**	947.00	2.02%	–
S51D4	50	160	1524.53	1552.00	1.80%	–	1536.28	1551.00	0.96%	–	**1541.71**	1551.00	**0.60%**	–
S51D5	50	160	1295.06	–	–	–	1306.57	1328.00	1.64%	–	**1318.80**	1328.00	**0.69%**	–
S51D6	50	160	2098.76	–	–	–	2074.14	–	–	–	**2149.00**	**2153.00**	0.19%	–
S76D1	75	160	588.67	593.00	0.74%	–	592.00	592.00		30.30	592.00	592.00		5408.84
S76D2	75	160	1028.39	–	–	–	1017.29	–	–	–	**1035.40**	–	–	–
S76D3	75	160	1359.10	–	–	–	1344.67	–	–	–	**1371.71**	–	–	–
S76D4	75	160	1994.04	–	–	–	1957.49	–	–	–	**2019.03**	–	–	–
S101D1	100	160	704.16	–	–	–	**716.00**	**716.00**		3124.80	704.10	–	–	–
S101D2	100	160	1285.58	–	–	–	1265.15	–	–	–	**1298.07**	–	–	–
S101D3	100	160	1778.82	–	–	–	1713.80	–	–	–	**1796.01**	–	–	–
S101D5	100	160	2644.63	–	–	–	2438.15	–	–	–	**2704.46**	–	–	–
p01_1030	50	160	736.63	753.00	2.22%	–	**753.00**	**753.00**		6098.10	747.37	755.00	1.01%	–
p01_1050	50	160	976.62	998.00	2.19%	–	975.00	998.00	2.36%	–	**985.10**	1029.00	4.27%	–
p01_1090	50	160	1446.03	–	–	–	1458.05	1490.00	2.19%	–	**1480.00**	**1480.00**	⋆	6765.87
p01_110	50	160	458.00	458.00		7.70	458.00	458.00		0.60	458.00	458.00		379.50
p01_3070	50	160	1444.68	1473.00	1.96%	–	1440.53	1473.00	2.25%	–	**1464.99**	1474.00	**0.61%**	–
p01_7090	50	160	2085.00	–	–	–	2063.37	2317.00	12.29%	–	**2131.84**	**2142.00**	**0.47%**	–
p02_1030	75	140	1053.04	–	–	–	1042.36	–	–	–	**1058.51**	–	–	–
p02_1050	75	140	1432.71	–	–	–	1395.19	–	–	–	**1449.88**	–	–	–
p02_1090	75	140	2212.35	–	–	–	2167.62	–	–	–	**2244.55**	–	–	–
p02_110	75	140	598.59	625.00	4.41%	–	**612.00**	**612.00**		4284.50	606.29	612.00	0.93%	–
p02_3070	75	140	2133.49	–	–	–	2111.24	–	–	–	**2157.13**	**2572.00**	16.13%	–
p02_7090	75	140	3079.64	–	–	–	3100.65	–	–	–	**3154.33**	–	–	–
p03_1030	100	200	1368.38	–	–	–	1318.99	–	–	–	**1378.60**	–	–	–
p03_1050	100	200	1899.06	–	–	–	1843.26	–	–	–	**1916.93**	–	–	–
p03_1090	100	200	2929.70	–	–	–	2793.74	–	–	–	**2999.02**	–	–	–
p03_110	100	200	732.72	–	–	–	**740.76**	**760.00**	**2.60%**	–	735.20	–	–	–
p03_3070	100	200	2833.93	–	–	–	2675.47	–	–	–	**2898.36**	–	–	–
p03_7090	100	200	4117.36	–	–	–	3939.76	–	–	–	**4266.28**	–	–	–
SD1	8	100	228.00	228.00		0.00	228.00	228.00		0.00	228.00	228.00		0.26
SD2	16	100	708.00	708.00		0.20	708.00	708.00		0.20	708.00	708.00		0.85
SD3	16	100	432.00	432.00		0.10	432.00	432.00		0.10	432.00	432.00		0.41
SD4	24	100	630.00	630.00		1.30	630.00	630.00		0.20	630.00	630.00		1.30
SD5	32	100	1392.00	1392.00		10.70	1392.00	1392.00		21.20	1392.00	1392.00		20.61
SD6	32	100	832.00	832.00		11.00	832.00	832.00		1.20	832.00	832.00		4.81
SD7	40	100	3640.00	3640.00		30.50	3640.00	3640.00		2.00	3640.00	3640.00		28.93
SD8	48	100	5068.00	5068.00		116.40	5068.00	5068.00		390.20	5068.00	5068.00		23.39
SD9	48	100	2046.00	2046.00		3459.70	2046.00	2046.00		927.20	2046.00	2046.00		265.44
SD10	64	100	2652.06	2688.00	1.36%	–	2617.48	2731.00	4.34%	–	**2688.00**	2688.00	⋆	5153.62
SD11	80	100	12864.40	–	–	–	13280.00	13280.00		6370.90	13280.00	13280.00		6895.22
SD12	80	100	6982.30	–	–	–	7102.36	**7453.00**	4.94%	–	**7108.34**	–	–	–
SD13	96	100	9723.60	–	–	–	9779.95	–	–	–	**9941.77**	–	–	–

⋆ Instance solved to proven optimality for the first time

– Not available (UB or Gap); Solver reached the time limit of 7200 seconds (T_{total})

Table EC.10 Results of the state-of-the-art and proposed BC algorithms on the SDVRPTW benchmark instances with $n = 50$ and $Q = 30$ (time limit: 3600 sec).

	Bianchessi and Irnich (2019)				Proposed BC algorithm											
Instance	*LB*	*UB*	*Gap*	T_{total}	*LB*	*UB*	*Gap*	T_{total}	T^{MIP}_{best}	T^{CPL}_{best}	*Nodes*	*Check*	*Cap*	*Conn*	*Reg*	*F/O*	*nR*
C101	1599.50	1599.50		98.00	1599.50	1599.50		58.22	40.63	49.40	1813	14	359	2	2304	11	29
C102	1599.50	1599.50		404.00	1599.50	1599.50		120.44	77.56	71.07	5569	23	392	25	4746	16	29
C103	1598.30	1598.30		1015.00	1598.30	1598.30		65.84	40.04	63.91	513	22	914	8	3322	6	29
C104	1598.30	1598.30		1736.00	1598.30	1598.30		115.65	102.89	99.95	3213	50	520	77	18887	39	29
C105	1599.50	1599.50		164.00	1599.50	1599.50		112.85	67.38	58.77	8066	30	479	0	6366	23	29
C106	1599.50	1599.50		142.00	1599.50	1599.50		141.74	129.69	99.91	10484	35	295	39	4328	28	29
C107	1599.50	1599.50		223.00	1599.50	1599.50		113.98	64.78	32.69	8002	21	205	0	3109	16	29
C108	1598.30	1598.30		404.00	1598.30	1598.30		42.34	39.11	42.20	0	14	213	1	1052	10	29
C109	1598.30	1598.30		246.00	1598.30	1598.30		63.33	41.07	54.62	44	49	284	1	8104	45	29
R101	1618.90	1618.90		963.00	1618.90	1618.90		2226.08	1317.38	2128.24	41609	145	22755	3	9030	54	25
R102	1581.30	1581.30		2704.00	1581.30	1581.30		3204.62	2104.77	2723.08	77105	62	29552	608	9610	40	25
R103	1550.20	**1567.90**	**1.14%**	–	**1557.44**	1580.50	1.46%	–	303.90	3321.59	73306	34	23717	330	7911	20	25
R104	1526.73	1578.90	3.42%	–	**1545.20**	**1551.80**	**0.43%**	–	2293.09	1792.62	84927	32	17155	612	5968	17	25
R105	1580.10	1580.10		453.00	1580.10	1580.10		303.66	223.25	4.67	10268	18	5878	0	1336	10	25
R106	1559.60	1559.60		1047.00	1559.60	1559.60		827.10	780.76	785.72	20091	47	9921	319	4929	25	25
R107	1533.40	1552.00	1.21%	–	**1548.60**	**1548.60**	⋆	3016.97	2419.74	2563.38	64970	42	12234	259	7622	25	25
R108	1518.58	1693.40	11.51%	–	**1538.09**	**1552.60**	**0.93%**	–	3282.33	7.30	59004	34	19537	735	7296	22	25
R109	1551.26	1578.50	1.76%	–	**1561.90**	**1569.70**	**0.50%**	–	712.34	1819.68	100402	200	19863	99	37929	184	25
R110	1527.61	1592.30	4.23%	–	**1546.26**	**1566.00**	**1.26%**	–	1500.37	6.37	70260	38	23541	447	6299	27	25
R111	1531.27	1562.10	2.01%	–	**1552.80**	**1552.80**	⋆	2098.55	1722.85	2050.62	41531	31	17862	369	7503	18	25
R112	1518.38	1599.90	5.37%	–	**1536.78**	**1579.30**	**2.69%**	–	414.45	3.32	75143	25	18506	469	7859	14	26
RC101	2739.50	2739.50		4.00	2739.50	2739.50		120.32	94.88	111.17	8135	48	55	5	8587	39	33
RC102	2739.50	2739.50		10.00	2739.50	2739.50		118.33	98.22	98.46	5608	22	72	72	5113	14	33
RC103	2739.50	2739.50		10.00	2739.50	2739.50		122.68	29.36	105.46	6010	17	23	116	2636	10	33
RC104	2739.50	2739.50		15.00	2739.50	2739.50		111.17	60.57	93.12	6041	32	183	142	3292	12	33
RC105	2739.60	2739.60		6.00	2739.60	2739.60		107.68	96.96	39.66	5434	18	69	43	1260	10	33
RC106	2739.50	2739.50		7.00	2739.50	2739.50		103.95	45.06	99.43	5519	14	41	54	2156	9	33
RC107	2739.50	2739.50		47.00	2739.50	2739.50		112.88	83.06	109.16	9526	27	20	216	4896	14	33
RC108	2739.50	2739.50		16.00	2739.50	2739.50		504.70	67.11	96.70	173290	21	76	235	3405	13	33
C201	1778.30	1778.30		3137.00	1778.30	1778.30		117.39	96.29	91.92	7875	37	642	0	2993	25	29
C202	1772.12	1778.30	0.35%	–	**1778.30**	1778.30	⋆	179.51	90.31	12.35	5484	14	809	68	2497	6	29
C203	1765.09	1784.20	1.08%	–	**1778.30**	**1778.30**	⋆	316.95	230.29	284.63	6393	19	1118	154	4274	12	29
C204	1760.85	1780.30	1.10%	–	**1778.30**	**1778.30**	⋆	530.70	329.76	145.89	53031	15	1793	324	2469	8	29
C205	1778.30	1778.30		1863.00	1778.30	1778.30		104.92	49.46	34.91	7966	14	560	19	2107	7	29
C206	1776.15	1778.30	0.12%	–	**1778.30**	1778.30	⋆	118.02	60.53	8.77	8210	14	603	35	1344	11	29
C207	1771.60	1778.30	0.38%	–	**1778.30**	1778.30	⋆	199.35	88.58	61.21	6496	16	1297	59	2635	10	29
C208	1771.67	1778.70	0.40%	–	**1778.30**	**1778.30**	⋆	168.14	72.68	106.60	5657	18	517	35	3161	13	29
R201	1578.40	1578.40		2064.00	1578.40	1578.40		892.21	637.34	15.02	26638	27	16092	13	2965	18	25
R202	1559.60	1559.60		1519.00	1559.60	1559.60		1319.75	1090.63	1254.44	41966	125	10945	260	18557	114	25
R203	1524.89	1591.60	4.37%	–	**1548.60**	**1548.60**	⋆	3204.27	1012.77	2494.98	73595	47	12138	337	4936	19	25
R204	1520.18	1570.70	3.32%	–	**1538.14**	**1554.40**	**1.05%**	–	3139.23	2974.46	66950	40	12013	500	5672	18	25
R205	1549.56	**1565.40**	1.02%	–	**1560.90**	1566.10	**0.33%**	–	2960.42	3425.49	55121	62	23128	84	9083	50	25
R206	1549.29	1556.90	0.49%	–	**1556.90**	1556.90	⋆	1556.59	1189.73	1313.70	30246	53	10431	290	5467	27	25
R207	1526.29	1556.20	1.96%	–	**1548.60**	**1548.60**	⋆	1130.34	846.88	982.36	35846	49	8401	176	3751	22	25
R208	1520.25	1627.80	7.07%	–	**1539.13**	**1548.60**	**0.61%**	–	3441.76	3428.12	89707	50	19855	360	5075	25	25
R209	1533.22	1571.90	2.52%	–	**1548.92**	1597.50	3.04%	–	752.63	2.84	72743	42	29417	414	9132	31	26
R210	1541.93	1561.30	1.26%	–	**1552.80**	**1552.80**	⋆	2882.48	2875.20	2831.69	63495	37	15476	233	6331	22	25
R211	1518.04	1663.80	9.60%	–	**1538.52**	**1552.50**	**0.90%**	–	2728.21	457.56	61004	27	13037	434	4597	21	25
RC201	2739.50	2739.50		5.00	2739.50	2739.50		89.49	76.56	67.48	8061	31	219	7	1745	14	33
RC202	2739.50	2739.50		8.00	2739.50	2739.50		113.69	65.57	105.97	5916	31	161	96	2941	16	33
RC203	2739.50	2739.50		9.00	2739.50	2739.50		132.57	58.16	22.47	12256	24	97	163	2384	7	33
RC204	2739.50	2739.50		14.00	2739.50	2739.50		138.83	36.85	87.64	19613	22	54	189	1563	9	33
RC205	2739.50	2739.50		26.00	2739.50	2739.50		112.08	101.90	24.99	5534	23	53	40	1062	9	33
RC206	2739.50	2739.50		12.00	2739.50	2739.50		149.53	124.77	84.94	5558	21	45	33	3689	15	33
RC207	2739.50	2739.50		7.00	2739.50	2739.50		155.17	136.83	137.43	17608	29	50	160	3456	18	33
RC208	2739.50	2739.50		18.00	2739.50	2739.50		424.58	158.00	16.54	74674	18	135	456	5169	15	33

⋆ Instance solved to proven optimality for the first time

Table EC.11 Results of the state-of-the-art and proposed BC algorithms on the SDVRPTW benchmark instances with $n = 50$ and $Q = 50$ (time limit: 3600 sec).

	Bianchessi and Irnich (2019)				Proposed BC algorithm											
Instance	*LB*	*UB*	*Gap*	T_{total}	*LB*	*UB*	*Gap*	T_{total}	T^{MIP}_{best}	T^{CPL}_{best}	*Nodes*	*Check*	*Cap*	*Conn*	*Reg*	*F/O*	*nR*
C101	1015.80	1015.80		31.00	1015.80	1015.80		131.28	128.72	13.69	5513	27	758	3	2996	12	18
C102	1013.00	1013.00		96.00	1013.00	1013.00		184.15	182.96	177.82	11955	39	915	150	7027	23	18
C103	1012.30	1012.30		456.00	1012.30	1012.30		323.17	304.71	313.79	11336	48	991	275	10740	30	18
C104	1010.20	1010.20		229.00	1010.20	1010.20		190.73	147.27	171.54	8151	54	720	72	5439	35	18
C105	1015.80	1015.80		15.00	1015.80	1015.80		76.18	42.77	5.67	8105	20	588	0	1509	10	18
C106	1015.80	1015.80		42.00	1015.80	1015.80		55.56	37.64	43.81	4581	24	514	1	1389	8	18
C107	1015.80	1015.80		44.00	1015.80	1015.80		142.26	108.34	79.03	5427	35	695	0	2800	19	18
C108	1011.80	1011.80		88.00	1011.80	1011.80		241.15	238.45	239.00	16404	55	1340	72	8109	35	18
C109	1010.10	1010.10		73.00	1010.10	1010.10		104.53	87.72	80.75	8173	18	632	17	960	5	18
R101	1190.70	1190.70		8.00	1190.70	1190.70		60.27	39.87	18.87	1768	54	2878	0	4146	37	15
R102	**1108.87**	**1114.20**	**0.48%**	–	1099.37	1120.00	1.84%	–	3556.71	3212.17	54800	27	33454	1732	4362	17	16
R103	1067.48	**1081.50**	**1.31%**	–	**1070.13**	1112.70	3.83%	–	1310.14	486.19	38834	34	25498	936	4960	15	16
R104	1034.88	1059.70	2.40%	–	**1054.70**	**1054.70**	⋆	3023.23	1940.84	2043.92	68025	28	13339	306	4034	15	15
R105	**1132.30**	**1132.30**		694.00	1130.21	1132.80	0.23%	–	1375.70	2437.54	78443	45	35753	23	3791	18	15
R106	1080.20	1080.20		226.00	1080.20	1080.20		814.92	612.51	773.25	15373	33	16227	463	5209	25	15
R107	1053.16	**1063.90**	**1.02%**	–	**1055.31**	1085.40	2.77%	–	3571.22	8.35	36525	28	21236	499	4016	15	15
R108	1029.12	1064.00	3.39%	–	**1044.72**	**1055.90**	**1.06%**	–	1357.99	1507.52	29988	51	13951	567	5945	28	15
R109	1074.06	1081.80	0.72%	–	**1081.80**	1081.80	⋆	694.97	67.73	338.16	23376	39	10107	45	196	2	15
R110	1034.78	1184.60	14.48%	–	**1045.29**	**1109.10**	**5.75%**	–	78.22	2.98	29824	17	21664	503	1323	6	16
R111	1061.50	1061.50		2777.00	1061.50	1061.50		2852.29	2505.42	2793.11	35303	41	19243	664	3248	18	15
R112	1023.77	1073.00	4.81%	–	**1041.22**	**1054.50**	**1.26%**	–	2696.59	1656.92	19716	39	13734	323	3129	15	15
RC101	1708.30	1708.30		1.00	1708.30	1708.30		107.86	44.35	89.78	6953	48	32	30	1711	9	20
RC102	1700.50	1700.50		3.00	1700.50	1700.50		184.20	59.82	181.31	17211	40	18	584	5225	16	20
RC103	1696.80	1696.80		3.00	1696.80	1696.80		490.38	383.02	399.52	120802	55	74	1507	4442	17	20
RC104	1696.70	1696.70		3.00	1696.70	1696.70		460.30	159.68	252.68	84499	39	44	1545	8240	25	20
RC105	1700.10	1700.10		3.00	1700.10	1700.10		136.43	12.77	125.49	10663	34	14	102	1255	9	20
RC106	1699.00	1699.00		3.00	1699.00	1699.00		340.79	318.37	315.10	59274	53	42	555	10858	34	20
RC107	1698.60	1698.60		4.00	1698.60	1698.60		460.22	403.11	403.30	77825	56	64	1600	7470	29	20
RC108	1696.70	1696.70		4.00	1696.70	1696.70		1844.45	717.56	738.69	688808	23	49	1353	6716	15	20
C201	1159.40	1159.40		465.00	1159.40	1159.40		125.33	98.59	52.50	8496	34	750	0	2651	22	18
C202	1142.44	1156.90	1.27%	–	**1156.90**	1156.90	⋆	915.94	259.94	734.63	18051	481	3625	210	42050	461	18
C203	1144.51	1159.20	1.28%	–	**1156.90**	**1156.90**	⋆	1173.36	676.27	16.13	74424	23	4622	412	4442	16	18
C204	1137.05	1175.00	3.34%	–	**1156.90**	**1156.90**	⋆	3302.65	1793.17	1943.53	177182	43	6929	939	10983	34	18
C205	1154.91	1156.90	0.17%	–	**1156.90**	1156.90	⋆	209.37	70.58	124.77	8322	29	1265	85	2675	8	18
C206	1149.33	1162.70	1.16%	–	**1156.90**	**1156.90**	⋆	362.57	231.67	258.47	23503	28	1449	178	4918	15	18
C207	1156.90	1156.90		2882.00	1156.90	1156.90		475.10	122.35	116.59	59840	25	2054	699	6393	17	18
C208	1148.45	1156.90	0.74%	–	**1156.90**	1156.90	⋆	798.24	493.42	699.89	61555	35	3649	325	5931	17	18
R201	1107.70	1107.70		392.00	1107.70	1107.70		417.31	317.26	384.13	14757	37	15003	23	959	10	15
R202	**1080.20**	**1080.20**		707.00	1079.86	1092.00	1.11%	–	3477.24	1864.71	71467	31	21213	928	7455	22	15
R203	1045.60	**1059.20**	1.30%	–	**1052.55**	1099.40	4.26%	–	459.88	57.94	32239	20	26617	570	2673	8	16
R204	1026.45	**1068.60**	4.11%	–	**1036.10**	1077.30	3.82%	–	2882.27	2988.32	22554	27	17667	1269	5250	12	16
R205	1072.98	1093.10	1.88%	–	**1077.17**	**1086.20**	**0.83%**	–	2833.44	2785.75	55245	38	29246	284	2399	15	15
R206	**1071.50**	**1071.50**		2874.00	1066.34	1092.80	2.42%	–	464.87	50.49	39438	24	23867	655	2611	16	15
R207	1041.34	1077.50	3.47%	–	**1059.20**	**1059.20**	⋆	1110.80	226.79	436.76	28948	23	11652	306	2138	6	15
R208	1024.27	1125.00	9.83%	–	**1050.56**	**1053.50**	**0.28%**	–	995.48	454.68	86183	22	12662	313	1309	6	15
R209	1039.90	1063.30	2.25%	–	**1054.50**	**1054.50**	⋆	1230.45	788.63	748.39	36640	36	10986	263	1582	11	15
R210	**1066.52**	**1072.90**	**0.60%**	–	1062.64	1083.40	1.92%	–	703.32	3436.10	54098	31	28351	1157	5078	17	15
R211	1027.38	**1063.20**	3.49%	–	**1035.70**	1069.50	**3.16%**	–	3417.84	1341.65	31537	32	18271	541	4866	14	15
RC201	1708.30	1708.30		3.00	1708.30	1708.30		106.36	9.25	84.36	7799	25	33	61	1065	6	20
RC202	1700.50	1700.50		6.00	1700.50	1700.50		303.58	65.60	27.75	79551	36	10	467	1216	10	20
RC203	1696.80	1696.80		3.00	1696.80	1696.80		501.37	451.04	486.09	110625	44	24	913	5266	14	20
RC204	1696.70	1696.70		3.00	1696.70	1696.70		2018.32	350.16	320.64	824340	37	56	1340	5379	17	20
RC205	1700.40	1700.40		2.00	1700.40	1700.40		185.74	71.27	167.39	24199	40	53	504	2736	12	20
RC206	1699.00	1699.00		2.00	1699.00	1699.00		119.15	55.27	104.57	5633	22	15	151	3795	12	20
RC207	1698.60	1698.60		3.00	1698.60	1698.60		226.66	163.94	218.89	38607	31	30	659	3901	18	20
RC208	1696.70	1696.70		4.00	1696.70	1696.70		964.30	554.84	522.29	288103	39	77	1388	7859	29	20

⋆ Instance solved to proven optimality for the first time

Table EC.12 Results of the state-of-the-art and proposed BC algorithms on the SDVRPTW benchmark instances with $n = 50$ and $Q = 100$ (time limit: 3600 sec).

	Bianchessi and Irnich (2019)				Proposed BC algorithm											
Instance	*LB*	*UB*	*Gap*	T_{total}	*LB*	*UB*	*Gap*	T_{total}	T^{MIP}_{best}	T^{CPL}_{best}	*Nodes*	*Check*	*Cap*	*Conn*	*Reg*	*F/O*	*nR*
C101	587.50	587.50		9.00	587.50	587.50		28.19	10.97	22.07	80	21	383	0	716	4	9
C102	584.60	584.60		73.00	584.60	584.60		170.28	115.79	129.56	8438	38	1533	510	601	6	9
C103	**582.10**	**582.10**		909.00	579.70	584.00	0.74%	–	62.64	3577.90	127365	49	6661	6777	11343	34	9
C104	**578.80**	**578.80**		1570.00	576.80	578.90	0.36%	–	189.37	2018.47	203485	57	8244	6549	15092	38	9
C105	587.50	587.50		4.00	587.50	587.50		22.68	9.91	14.54	40	18	339	0	708	4	9
C106	587.50	587.50		9.00	587.50	587.50		41.63	36.59	41.01	177	24	559	1	1203	6	9
C107	587.50	587.50		4.00	587.50	587.50		64.17	52.25	55.62	4709	27	657	0	1970	13	9
C108	584.00	584.00		42.00	584.00	584.00		225.38	113.28	221.63	10432	65	941	352	6032	44	9
C109	579.80	579.80		160.00	579.80	579.80		351.37	244.82	290.92	20402	73	2747	1895	9843	49	9
R101	1043.80	1043.80		0.00	1043.80	1043.80		2.67		1.87	0	4	13	0	0	0	12
R102	842.78	**913.20**	8.36%	–	**859.31**	934.10	**8.01%**	–	154.97	27.68	91779	14	15622	9256	1256	6	12
R103	**740.52**	**804.70**	**9.09%**	–	739.10	842.30	12.25%	–	730.48	5.05	22691	11	17929	3930	0	4	11
R104	**688.04**	**714.80**	**3.89%**	–	683.08	751.60	9.12%	–	51.62	61.13	18546	21	18190	2030	2166	9	9
R105	918.10	918.10		11.00	918.10	918.10		85.90	17.50	24.25	9555	16	2301	39	103	2	9
R106	774.57	**824.70**	**6.47%**	–	**776.31**	869.70	10.74%	–	44.82	14.03	44469	13	21708	6894	3665	8	11
R107	703.28	**765.50**	**8.85%**	–	**712.06**	796.00	10.54%	–	791.79	23.93	20766	22	17124	3791	596	5	9
R108	678.00	732.40	8.02%	–	**686.38**	**719.80**	**4.64%**	–	316.75	89.92	17605	44	19595	2713	3498	8	9
R109	764.48	**810.40**	**6.01%**	–	**765.44**	819.20	6.56%	–	2201.48	1248.28	48187	35	25294	3387	197	1	9
R110	694.89	**762.10**	**9.67%**	–	**698.00**	774.80	9.91%	–	39.85	3338.56	18801	30	18533	3418	198	1	9
R111	**706.35**	**758.40**	**7.37%**	–	705.32	799.70	11.80%	–	928.63	9.34	18995	22	19168	3995	1299	6	10
R112	671.58	**727.00**	**8.25%**	–	**674.29**	751.50	10.27%	–	86.11	86.76	14492	23	19819	2487	1268	5	9
RC101	990.50	990.50		10.00	990.50	990.50		74.64	62.09	73.51	8110	36	215	116	1902	8	10
RC102	960.20	960.20		229.00	960.20	960.20		111.93	72.70	99.26	12435	26	323	2097	1443	6	10
RC103	936.20	936.20		1876.00	936.20	936.20		2774.11	1347.12	2293.60	436656	94	3889	24873	21134	66	10
RC104	915.90	915.90		2.00	915.90	915.90		195.86	161.93	175.60	8391	41	35	2239	4849	19	10
RC105	957.40	957.40		203.00	957.40	957.40		148.09	123.71	138.47	17935	26	651	2602	556	4	10
RC106	936.40	936.40		13.00	936.40	936.40		86.50	44.42	84.70	8126	24	383	827	810	11	10
RC107	915.10	915.10		2.00	915.10	915.10		67.92	9.11	67.69	5538	14	21	664	0	1	10
RC108	911.90	911.90		1.00	911.90	911.90		111.22	26.02	97.07	8106	23	48	920	808	3	10
C201	693.10	693.10		263.00	693.10	693.10		476.36	301.55	452.75	56793	48	5477	0	4343	18	9
C202	686.20	686.20		447.00	686.20	686.20		276.84	209.60	263.16	9861	32	4496	234	3472	14	9
C203	685.40	685.40		2338.00	685.40	685.40		515.58	355.90	375.14	15534	32	7712	688	4659	12	9
C204	684.80	684.80		2759.00	684.80	684.80		453.11	448.05	412.40	24116	30	4949	321	5294	17	9
C205	684.80	684.80		121.00	684.80	684.80		239.90	211.33	230.87	11459	65	2322	801	3204	13	9
C206	684.80	684.80		439.00	684.80	684.80		344.14	301.67	338.79	10942	54	2692	591	4078	22	9
C207	684.80	684.80		1117.00	684.80	684.80		2005.12	1558.98	1517.53	107482	71	5092	2497	8800	32	9
C208	684.80	684.80		365.00	684.80	684.80		362.16	192.29	194.50	10608	33	2573	453	3160	17	9
R201	843.00	843.00		80.00	843.00	843.00		29.43		11.60	1664	25	1777	59	0	0	8
R202	750.80	**782.70**	**4.25%**	–	**751.18**	824.10	8.85%	–	574.54	52.91	39333	21	24134	6746	6376	15	10
R203	697.78	736.70	5.58%	–	**703.23**	**733.70**	**4.15%**	–	644.56	639.01	28428	30	24688	2073	1400	5	9
R204	682.51	691.90	1.38%	–	**691.90**	691.90	⋆	3503.48	231.49	843.78	49424	62	17886	1238	1840	6	8
R205	758.80	758.80		3286.00	758.80	758.80		768.25	756.06	675.93	22662	37	18436	1209	1071	9	8
R206	719.59	728.10	1.18%	–	**728.10**	728.10	⋆	2276.24	1330.56	1862.29	75161	52	23125	2654	4442	16	8
R207	691.47	712.00	2.97%	–	**698.03**	**711.30**	**1.87%**	–	2192.78	686.73	51551	65	28580	1776	2212	8	8
R208	673.82	**691.90**	**2.68%**	–	**675.85**	704.10	4.01%	–	2891.16	818.71	20026	43	22672	2116	6756	21	8
R209	702.84	**725.40**	3.21%	–	**706.90**	728.90	**3.02%**	–	360.66	361.56	74702	37	29773	2415	2386	5	8
R210	**722.45**	**745.70**	**3.22%**	–	713.88	779.70	8.44%	–	243.58	7.35	23443	33	24661	4787	3536	10	9
R211	674.39	**715.80**	**6.14%**	–	**674.60**	726.40	7.13%	–	95.13	7.28	17985	51	23085	2292	3888	12	8
RC201	966.20	966.20		2.00	966.20	966.20		114.44	84.66	112.33	8018	67	193	1086	3631	16	10
RC202	946.50	946.50		64.00	946.50	946.50		291.84	283.49	286.90	18903	73	381	3499	9232	42	10
RC203	926.40	926.40		22.00	926.40	926.40		298.87	128.67	212.85	60213	45	27	3563	1251	7	10
RC204	915.90	915.90		1.00	915.90	915.90		113.87	89.26	109.12	5778	46	560	2276	2653	11	10
RC205	946.70	946.70		4.00	946.70	946.70		97.11	61.85	86.08	8434	80	24	818	2505	19	10
RC206	940.80	940.80		1.00	940.80	940.80		27.59	21.57	6.77	511	22	58	15	445	2	10
RC207	924.10	924.10		8.00	924.10	924.10		164.20	5.69	36.19	13641	42	21	1663	2977	14	10
RC208	911.90	911.90		1.00	911.90	911.90		152.69	100.83	112.99	5603	52	160	816	3257	17	10

⋆ Instance solved to proven optimality for the first time

Table EC.13 SDVRP-UF instances solved to proven optimality for the first time (time limit: 24 hours).

Instance	n	Q	$\overline{m}$	*Opt value*	T_{total}	T_{best}^{MIP}	T_{best}^{CPL}	*Nodes*	*Check*	*Cap*	*Conn*	*Reg*	*F/O*	*nR*
eil22	21	6000	4	375.28	2.67	2.47	2.61	0	29	111	17	146	7	4
eil23	22	4500	3	568.56	1.86	1.60	1.77	0	37	34	43	44	4	3
eil30	29	4500	3	505.01	3.88	3.80	3.83	0	49	76	311	290	2	4
eil33	32	8000	4	837.06	102.26	87.38	98.97	14797	115	1212	738	2627	40	4
eil51	50	160	5	524.61	115.62	40.90	94.92	6718	97	2950	461	253	6	5
S51D1	50	160	3	459.50	14.46	5.37	7.71	118	28	255	190	100	1	3
S51D2	50	160	9	708.42	2459.86	1323.06	1371.25	84492	46	15815	526	2545	17	9
S51D3*	50	160	15	947.97	23358.40	845.91	3585.44	466425	94	20612	542	1909	14	15
S51D4*	50	160	27	1560.88	6779.78	5927.21	6589.23	177407	132	23361	116	4613	31	27
S51D5*	50	160	23	1333.67	9418.48	5321.93	6042.65	282543	137	13168	65	2931	18	23
S51D6*	50	160	41	2169.10	13973.59	8.65	3302.64	473614	113	15138	1	203	1	41
S76D1	75	160	4	598.94	1111.36	397.45	916.88	11865	47	17361	10520	1873	6	4
p01_1030	50	160	11	756.71	5446.83	5270.76	5356.94	110712	160	19509	707	2923	23	11
p01_1050*	50	160	16	1005.75	74676.79	12570.51	57283.02	1179093	101	51929	1733	4686	30	16
p01_1090*	50	160	26	1487.18	7820.99	222.27	2013.81	193142	99	17485	36	1764	11	26
p01_110	50	160	3	459.50	12.24		3.90	54	17	349	119	0	0	3
p01_3070*	50	160	26	1481.71	27833.26	6916.15	11192.56	875244	127	33578	158	1637	16	26
SD3	16	100	12	430.58	0.67		0.65	0	68	48	0	0	0	12
SD4	24	100	18	631.05	1.98	1.16	1.92	0	152	101	0	50	1	18
SD5	32	100	24	1390.57	72.23	8.02	57.10	4102	161	777	29	476	9	24
SD6	32	100	24	831.24	7.85	3.29	7.04	0	134	258	0	194	2	24
SD7	40	100	30	3640.00	33.21	5.65	16.78	871	32	451	9	240	2	30
SD9	48	100	36	2044.20	322.30	10.91	177.03	12639	75	3191	37	388	4	36
SD10*	64	100	48	2684.88	4066.66	430.87	2579.66	69810	102	9915	61	641	5	48
SD11	80	100	60	13280.00	2211.45	2142.96	1556.85	8494	21	5277	113	2343	7	60

$*$ Instance solved to proven optimality for the first time for the SDVRP-LF as well.

Table EC.14 SDVRPTW instances solved to proven optimality for the first time (time limit: 24 hours).

Instance	*Q*	*Opt value*	T_{total}	T_{best}^{MIP}	T_{best}^{CPL}	*Nodes*	*Check*	*Cap*	*Conn*	*Reg*	*F/O*	*nR*
R103	30	1564.20	4070.72	2441.04	3448.20	141447	88	12100	218	12820	49	25
R104		1548.60	14880.17	9152.55	9285.13	366481	64	13996	496	9800	22	25
R107		1548.60	3016.97	2419.74	2563.38	64970	42	12234	259	7622	25	25
R108		1548.60	13976.65	7414.29	8354.98	363688	94	17338	371	7324	29	25
R110		1557.90	65645.07	63862.55	59224.51	590249	53	24561	307	6701	27	25
R111		1552.80	2098.55	1722.85	2050.62	41531	31	17862	369	7503	18	25
C202		1778.30	179.51	90.31	12.35	5484	14	809	68	2497	6	29
C203		1778.30	316.95	230.29	284.63	6393	19	1118	154	4274	12	29
C204		1778.30	530.70	329.76	145.89	53031	15	1793	324	2469	8	29
C206		1778.30	118.02	60.53	8.77	8210	14	603	35	1344	11	29
C207		1778.30	199.35	88.58	61.21	6496	16	1297	59	2635	10	29
C208		1778.30	168.14	72.68	106.60	5657	18	517	35	3161	13	29
R203		1548.60	3204.27	1012.77	2494.98	73595	47	12138	337	4936	19	25
R204		1548.60	16829.17	6560.85	6622.81	356032	63	16065	582	7919	21	25
R205		1565.40	4823.69	4708.19	4730.06	139500	64	15725	102	6236	28	25
R206		1556.90	1556.59	1189.73	1313.70	30246	53	10431	290	5467	27	25
R207		1548.60	1130.34	846.88	982.36	35846	49	8401	176	3751	22	25
R208		1548.60	10337.63	5710.67	2500.96	283606	79	19659	572	8417	48	25
R210		1552.80	2882.48	2875.20	2831.69	63495	37	15476	233	6331	22	25
R211		1548.60	14793.09	3954.19	4316.67	529185	78	13647	272	8556	50	25
R102	50	1114.20	14205.13	2973.33	14096.08	247435	113	29639	1134	9867	28	15
R103		1081.30	8310.75	6671.07	8220.11	165767	137	24331	865	14854	96	15
R104		1054.70	3023.23	1940.84	2043.92	68025	28	13339	306	4034	15	15
R107		1063.90	16280.85	6842.18	15036.56	135554	54	28383	1382	5349	23	15
R108		1053.50	11641.64	4736.12	5140.85	136639	36	17425	323	4353	16	15
R109		1081.80	694.97	67.73	338.16	23376	39	10107	45	196	2	15
C202		1156.90	915.94	259.94	734.63	18051	481	3625	210	42050	461	18
C203		1156.90	1173.36	676.27	16.13	74424	23	4622	412	4442	16	18
C204		1156.90	3302.65	1793.17	1943.53	177182	43	6929	939	10983	34	18
C205		1156.90	209.37	70.58	124.77	8322	29	1265	85	2675	8	18
C206		1156.90	362.57	231.67	258.47	23503	28	1449	178	4918	15	18
C208		1156.90	798.24	493.42	699.89	61555	35	3649	325	5931	17	18
R203		1059.20	4314.16	2359.94	2535.76	35554	41	22859	796	5675	14	15
R204		1053.50	20781.90	3449.29	2897.92	234994	45	17197	499	4539	15	15
R205		1085.90	15636.66	12728.64	8082.26	216134	76	48817	844	5892	28	15
R206		1071.50	32670.49	31700.78	32393.74	263321	97	47236	1765	5980	25	15
R207		1059.20	1110.80	226.79	436.76	28948	23	11652	306	2138	6	15
R209		1054.50	1230.45	788.63	748.39	36640	36	10986	263	1582	11	15
R210		1072.90	4937.76	3893.42	4049.02	111666	54	21394	402	4729	19	15
R211		1053.50	59873.27	2189.67	2827.55	705865	43	17281	627	3536	11	15
R204	100	691.90	3503.48	231.49	843.78	49424	62	17886	1238	1840	6	8
R206		728.10	2276.24	1330.56	1862.29	75161	52	23125	2654	4442	16	8