Column Enumeration via Strategic Row Aggregation for Split Delivery Vehicle Routing with Time Windows

Ying Yang[a], Xiaodeng Hao[b], Silong Zhang[b], Shuaian Wang[b], Zhou Xu[b,*]
[a]Singapore-MIT Alliance for Research and Technology, Singapore;
[b]Department of Logistics and Maritime Studies, Faculty of Business, The Hong Kong Polytechnic University, Hong Kong
[*]zhou.xu@polyu.edu.hk

Abstract. This paper presents a new exact method based on column enumeration for the split delivery vehicle routing problem with time windows (SDVRPTW), where each customer's demand may be split across multiple vehicles subject to time-window constraints. Although column-enumeration methods have been successful for many other vehicle routing variants, they remain largely unexplored for the SDVRPTW. We develop a new column-enumeration-via-strategic-row-aggregation (CESRA) method for the SDVRPTW. It enumerates routes rather than patterns to preserve column integrality. A key challenge is that this method introduces an exponential number of column-dependent constraints, making reduced-cost evaluation difficult. To overcome this challenge, we propose a strategic row-aggregation scheme that replaces these constraints with a polynomial number of aggregated arc-based constraints, thereby enabling efficient route enumeration. We further strengthen CESRA with several problem-specific enhancements. Computational results show that CESRA is the first column-enumeration method for the SDVRPTW that achieves performance comparable to, and sometimes better than, the state-of-the-art exact method based on branch-price-and-cut. Compared with the only existing enumeration-based exact method for the SDVRPTW, CESRA solves significantly more instances to optimality while enumerating three orders of magnitude fewer columns. It opens a new direction for solving integer programming models with extreme columns through strategic row aggregation.

Key words: Exact algorithm, split delivery vehicle routing problem with time windows, column enumeration, row aggregation

1. Introduction

The split delivery vehicle routing problem (SDVRP) is a variant of the classical vehicle routing problem (VRP), in which a customer's demand may be jointly served by multiple vehicles rather than a single one (Dror and Trudeau 1989, 1990). This relaxation significantly increases the flexibility of delivery operations in many practical applications and has been shown to result in substantial savings of up to 50% (Archetti et al. 2006). Despite the pronounced practical benefits of the SDVRP, it involves both discrete routing and continuous delivery quantity decisions, resulting in a computationally challenging combinatorial optimization problem (Dror and Trudeau 1990). Over the past decades, a variety of exact approaches have been developed to tackle the SDVRP and its time-window variant (SDVRPTW), including branch-and-cut (B&C) algorithms (Belenguer

et al. 2000, Archetti et al. 2014, Bianchessi and Irnich 2019, Bianchessi et al. 2019, Munari and Savelsbergh 2022, Gamst et al. 2024) and branch-and-price (B&P) or B&P augmented with cutting planes (BPC) algorithms (Desaulniers 2010, Archetti et al. 2011, Luo et al. 2017, Balster et al. 2023). Given that the SDVRP is a special case of the SDVRPTW and time windows are common in practical applications, the remainder of this paper focuses on the SDVRPTW.

Even with decades of methodological progress along these lines, the SDVRPTW remains computationally challenging. The most recent state-of-the-art methods still struggle to solve instances exceeding 50 customers within a reasonable time limit (Balster et al. 2023, Gamst et al. 2024), in stark contrast to the standard capacitated VRP, where benchmark instances with up to 200 customers are now within reach of state-of-the-art exact methods (Pessoa et al. 2020, You et al. 2026). This persistent scalability gap suggests that the SDVRPTW poses distinctive computational challenges, and that the methodological landscape would benefit from exploring solution paradigms beyond the established B&C, B&P, and BPC frameworks.

One such paradigm is the column enumeration framework first introduced by Baldacci et al. (2008). Column enumeration has been successfully applied to VRPs, such as the capacitated VRP (Baldacci et al. 2008), pickup and delivery problem (Baldacci et al. 2011a), and multi-trip VRP (Yang 2023, 2025), with the goal of exhaustively identifying all columns (i.e., integer variables associated with feasible routes) that may appear in optimal solutions. The main principle, inherently involving an application of variable fixing (Dantzig et al. 1954), is that only columns with reduced costs smaller than or equal to the integrality gap can enhance the best-known upper bound, allowing columns exceeding this gap to be excluded without compromising optimality. Once these columns are enumerated, an optimal solution can be obtained by directly solving the resulting formulation as an integer program (IP) restricted to a subset of variables. Compared to the classical B&P and BPC framework, this approach is particularly effective when the number of promising columns is moderate and can be enumerated efficiently. It is therefore well-suited to instances with a small root-node duality gap and a small average number of customers per route (Toth and Vigo 2014). Besides, it demonstrates strong performance for problems where the pricing subproblem is highly complex or when branching strategies introduce significant overhead (Baldacci et al. 2008). The SDVRPTW exhibits each of these characteristics. Specifically, in most benchmark instances in Solomon (1987), the duality gap at the root node is relatively small, and the number of customers in a route in the optimal solutions is relatively small (Desaulniers 2010). Moreover, the pricing

subproblem of SDVRPTW is highly complex due to the presence of quantity-related decisions, and the standard B&P approach often requires extensive branching.

Despite these favorable characteristics, the application of column enumeration to the SDVRPTW remains largely unexplored. In the SDVRPTW, a column can represent either a *route*, which specifies only the sequence of customer visits, or a *pattern*, which specifies both the visit sequence and the corresponding delivery quantities. To the best of our knowledge, Casazza et al. (2018) contribute the only existing work that applies column enumeration to the SDVRP and the split pickup and delivery vehicle routing problem (SPDVRP). Their approach directly enumerates all patterns whose reduced cost is less than or equal to the duality gap. However, since a single route can map to numerous distinct patterns, the total number of patterns to enumerate is enormous, which leads to poor computational performance. For instance, within a one-hour time limit, their enumeration approach can only solve 17-customer instances where all delivery demands are multiples of 10. Beyond this single attempt, broader progress on column enumeration for the SDVRPTW has been hindered by two fundamental obstacles when enumerating either patterns or routes. First, as shown by Casazza et al. (2018), enumerating columns that correspond to patterns is computationally prohibitive because the number of patterns is vast. Second, enumerating columns that correspond to routes is difficult, as computing their reduced costs is non-trivial: the computation requires dual variables associated with column-dependent constraints in the master problem. Motivated by these challenges, this paper seeks to address the following key research question:

"*How can column enumeration be applied to solve the SDVRPTW effectively?*"

In response to the above research question, our paper makes the following contributions:

• *We propose a novel exact column-enumeration-via-strategic-row-aggregation (CESRA) method that can effectively solve the SDVRPTW.* Unlike existing B&C, B&P, and BPC frameworks, this is the application of an enumeration-based approach to the SDVRPTW. Specifically, built upon the extreme-pattern-based formulation, we propose a strategic row-aggregation technique. This technique transforms the original exponentially many column-dependent constraints that link routes to patterns into a polynomial number of aggregated arc-based constraints, thereby enabling affordable route enumeration while guaranteeing global optimality.

• *Beyond standard acceleration techniques, we develop the following tailored enhancements to improve the performance of the CESRA method for the SDVRPTW*.

(i) We develop a tailored rollback dominance rule for pattern generation by deriving a lower bound on the delivery quantity through each label extension, which significantly accelerates the pattern pricing process.

(ii) We propose new rollback and generic dominance rules for route enumeration based on the duality gap, which accelerate the route enumeration stage.

(iii) We deploy instance-adaptive IP formulations in the optimal solution computation stage in CESRA. For instances with large vehicle capacities, we solve a route-based formulation (Balster et al. 2023) coupled with strong capacity inequalities (Baldacci et al. 2008); for instances with small vehicle capacities, we propose a variable fixing rule tailored to patterns and employ a pattern-based formulation for IP solving.

• *Extensive computational experiments demonstrate that CESRA is the first column-enumeration method for the SDVRPTW to achieve performance comparable to, and in some cases surpassing, that of the state-of-the-art BPC algorithm proposed by Balster et al. (2023). CESRA also substantially outperforms the only existing column-enumeration method for the SDVRPTW (Casazza et al. 2018).*

(i) For the SDVRPTW benchmark instances with 50 customers, our CESRA method outperforms the state-of-the-art algorithm, reducing the average solution time by 20% to 70% across all three instance classes and closing four previously open instances to proven optimality. For the SDVRPTW benchmark instances with 100 customers, CESRA maintains competitive performance, matching the state-of-the-art algorithm in proven optimality while obtaining feasible solutions on additional instances. Beyond this size, both the state-of-the-art method and CESRA become prohibitively time- or memory-intensive.

(ii) Compared with the only existing enumeration-based method, CESRA closes all tested instances to proven optimality with an average solution time of 228 seconds, whereas the baseline method closes only two instances within a one-hour time limit and enumerates three orders of magnitude more columns.

(iii) The ablation experiments further isolate the contribution of each proposed enhancement for CESRA, with the tailored rollback and generic dominance rules yielding 65% to 90% reductions in computational time and memory usage on the most challenging R2 subclass of the 50-customer benchmark. The pattern-based IP formulation, combined with the tailored variable fixing rule for patterns, reduces the computational time for small-capacity R-class instances by more than 70%.

• *The newly proposed CESRA framework establishes a highly adaptable exact solution paradigm, opening a new direction for solving integer programming models with extreme columns through strategic row aggregation.*

The rest of this paper is organized as follows. Section 2 reviews the relevant literature of SDVRPTW and column enumeration, highlighting the research gaps. Section 3 formally defines

the SDVRPTW and introduces the pattern-based formulation used in this study. Section 4 presents the CESRA, starting with an outline of the framework and followed by a detailed description of each step. The computational results are detailed in Section 5. Section 6 concludes the paper with discussions on future applications of our CESRA method. All proofs are provided in Appendix A.

2. Literature Review

This section first reviews the exact solution approaches for SDVRPTW. Then, we give an overview of the column enumeration algorithm framework and its limitations in addressing SDVRPTW.

2.1. Exact Methods for the SDVRP and SDVRPTW

Heuristic algorithms for the SDVRPTW have been studied extensively for a long time; see Archetti and Speranza (2012) and He and Hao (2023) for comprehensive surveys. More recently, however, growing attention has been directed toward the development of exact solution algorithms, particularly B&C, B&P, and BPC algorithms, which we review below.

2.1.1. B&C Methods. B&C algorithms applied to arc-based formulations are one of the predominant techniques for SDVRPTW in the existing literature (Gamst et al. 2024). Their performance largely depends on tightness of the relaxation model and effectiveness of feasibility verification and valid inequalities.

Belenguer et al. (2000) develop a relaxed two-index vehicle-flow formulation for the SDVRP, study its polyhedral structure, and derive a cutting-plane procedure for lower-bound computation. They also show that checking whether a solution to this relaxation is feasible for the original SDVRP is NP-hard. Building on this work, Archetti et al. (2014) propose two B&C algorithms, based respectively on the relaxation of Belenguer et al. (2000) and a single-commodity flow formulation. They further introduce polynomial-time heuristic feasibility checks derived from Archetti et al. (2008), together with capacity and connectivity inequalities. Lusby et al. (2020) instead develop a branch-and-Benders-cut algorithm for the SDVRP. Their master problem optimizes a linear arc-flow formulation, while the subproblem verifies flow feasibility and generates Benders cuts, with integrality enforced through a standard branch-and-bound (B&B) framework.

Bianchessi et al. (2019) extend the problem scope to SDVRPTW and incorporate customer-inconvenience constraints, such as limiting the number of visits by a vehicle and synchronizing deliveries, and develop a corresponding B&C algorithm. Bianchessi and Irnich (2019) further strengthen this line of research through a relaxed two-commodity flow formulation, where the two

commodities represent vehicle flow and service-time flow. Their method combines strengthened feasibility cuts with cluster-based decomposition and solves 277 of 504 SDVRPTW benchmark instances with 25, 50, and 100 customers to optimality. Munari and Savelsbergh (2022) introduce new binary variables indicating whether two arcs are traversed consecutively and derive three compact formulations based on Miller–Tucker–Zemlin or commodity-flow constraints (Miller et al. 1960). Their B&C algorithm starts from a relaxed two-index formulation and dynamically adds constraints from the compact models when incumbent solutions violate feasibility. Tested on 392 benchmark instances, comprising 224 SDVRP instances with up to about 100 customers and 168 SDVRPTW instances with 50 customers, the algorithm demonstrates substantially improved computational efficiency, solving 95 instances to optimality for the first time and significantly tightening bounds for many others.

Gouveia et al. (2023) study the multi-depot SDVRP and propose a B&C algorithm based on an IP formulation with single-customer route variables, capacity inequalities, depot-consistency constraints, and additional cuts for eliminating infeasible solutions.

Most recently, Gamst et al. (2024) establish the current state of the art for the SDVRP. Their B&C algorithm combines the two-index relaxation of Belenguer et al. (2000) and Archetti et al. (2014), in-out stabilization for cut separation (Ben-Ameur and Neto 2007), a reduced-graph adaptation of the two-edge commodity-flow formulation of Munari and Savelsbergh (2022) for feasibility verification, and an adaptive large-neighborhood search heuristic for upper-bound generation. On 352 benchmark instances under a two-hour limit, the method attains an average gap of 1.82‰, solves 102 instances to optimality, and improves 70 lower bounds and 83 upper bounds. Nevertheless, most instances with 75 or more customers remain unsolved within the time limit, underscoring the persistent challenge of scaling exact methods for split-delivery routing.

2.1.2. B&P and BPC Methods. Several studies have investigated B&P and BPC algorithms for the SDVRPTW by decomposing the problem into a master problem and pricing subproblem. There are generally two decomposition methods for the SDVRPTW: *route-based* formulations that optimize delivery quantities in the master problem (Gendreau et al. 2006), and *pattern-based* formulations that determine the delivery quantities in the pricing subproblem (Desaulniers 2010, Archetti et al. 2011, Luo et al. 2017, Balster et al. 2023). In route-based formulations, delivery quantities are handled in the master problem through exponentially many column-dependent constraints, which complicate reduced-cost computation and often require costly simultaneous column-and-row generation (Feillet et al. 2010, Muter et al. 2013). State-of-the-art B&P methods therefore

typically adopt pattern-based formulations that shift delivery decisions to the subproblem. While this choice increases the complexity of solving the subproblem, it generally leads to better overall computational performance.

Desaulniers (2010) is the first work to solve the SDVRPTW using B&P with the pattern-based formulation. He proposes the concept of an *extreme pattern*, which consists of customers whose demands are unsatisfied, partially satisfied, or fully satisfied, with at most one partially satisfied customer per pattern. Additionally, he introduces a novel labeling framework tailored for the SDVRPTW, where the reduced cost is defined as a linear function of the delivery quantities. He also proposes a branching strategy based on adjacent arcs. Archetti et al. (2011) propose a B&P algorithm for the SDVRPTW, where a tabu search is embedded in the column generation (CG) procedure. They extend the framework by adopting several classes of valid inequalities, including strong minimum number of vehicles inequalities, subset row inequalities (Jepsen et al. 2008), and k-path inequalities (Kohl et al. 1999). Following Desaulniers (2010), Luo et al. (2017) focus on optimizing the pricing algorithm by proposing an aggregated label definition and designing a stronger set dominance rule based on the lower convex envelope of a piecewise linear function. Their approach achieves high efficiency at the root node of the B&B tree.

To the best of our knowledge, Balster et al. (2023) propose the current state-of-the-art BPC approach based on a family of route-based formulations. By showing that an optimal solution exists in which each delivery quantity is a multiple of the greatest common divisor of the customer demands and vehicle capacity, they substantially reduce the number of admissible delivery quantities. Under a one-hour time limit, their proposed method solves 102 of the 352 SDVRP benchmark instances to optimality, including 14 previously open instances, and improves the best-known lower bounds for 116 instances. On the 504 Solomon SDVRPTW benchmark instances, it solves most 50-customer instances to optimality and proves optimality for 84 instances for the first time. Nevertheless, for instances with 75 or 100 customers, only those in the C class are consistently solved, while only a few instances in the R and RC classes can be solved to optimality.

2.2. Column Enumeration Technique and Its Limitations on SDVRPTW

Column enumeration is first introduced by Baldacci et al. (2008) and has been successfully applied to various combinatorial optimization problems, including the location-routing problem (Contardo et al. 2014), arc-routing problem (Bartolini et al. 2013), and VRPs (Baldacci et al. 2011a,b,c,

Andelmin and Bartolini 2017, Paradiso et al. 2020, Yang 2023). The efficiency of column enumeration depends heavily on the size of the enumerated set. In light of this, three main acceleration directions for column enumeration have been proposed: (i) reducing the duality gap to shrink the enumerated column set, which can be achieved by constructing stronger lower bounds via valid cuts or generating better upper bounds to narrow the gap (Costa et al. 2019); (ii) utilizing multiple dual solutions to compute reduced costs across the dual space, which allows for more effective variable fixing (Yang 2023, 2025); and (iii) embedding column enumeration into a B&B framework, which divides the problem into manageable subproblems so that enumeration can be selectively applied to nodes with smaller gaps (Pecin et al. 2017).

While column enumeration has proven effective in various applications, its application to the SDVRPTW remains an open challenge. Casazza et al. (2018) contribute the only existing literature that develops an enumeration-based method for the SDVRP and SPDVRP. Their algorithm primarily targets the multiple vehicle balancing problem in bike-sharing systems and, as a generalization, can also handle the SDVRP. However, its computational scalability is rather limited: on their SDVRP instances, only 3 out of 9 tested instances with up to 25 customers are solved to proven optimality within a six-hour time limit. The challenge primarily stems from two critical requirements: (i) the necessity for variables to be integer and (ii) the need for fully computable reduced costs of these variables. These conditions are difficult to satisfy simultaneously in existing formulations for the SDVRPTW. Specifically, when delivery decisions are modeled in the master problem through a route-based formulation, column-dependent rows hinder efficient pricing. When they are shifted to the subproblem, the resulting formulation either uses continuous pattern variables and requires additional variables to enforce integrality, as in an extreme pattern-based formulation, or generates prohibitively many columns for enumeration, as in the general pattern-based formulation used in Casazza et al. (2018).

Our work addresses this challenge by introducing a strategic row-aggregation technique into the column enumeration method, proposing a CESRA framework for solving the SDVRPTW. Building on the extreme pattern-based formulation, the row-aggregation technique transforms the exponential number of column-dependent constraints linking routes to patterns into a polynomial number of aggregated arc-based constraints. These aggregated rows simplify the reduced cost computation for routes, thereby enabling tractable route enumeration. On the SDVRPTW benchmark instances, CESRA achieves performance comparable to, and in some cases better than, that of the state-of-the-art BPC algorithm proposed by Balster et al. (2023), reducing the average solution time by 20%

to 70% and solving four additional instances to proven optimality on the 50-customer benchmark while remaining competitive on the 100-customer benchmark. Besides, CESRA outperforms the only existing enumeration-based method proposed by Casazza et al. (2018).

3. Problem Description and Mathematical Formulation

This section presents the problem definition alongside its extreme pattern-based formulation.

3.1. Problem Description

The SDVRPTW is defined on a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V} = \{0, 1, ..., n, n+1\}$, indexed by i, is the vertex set, and $\mathcal{A} = \{(i, j) | i, j \in \mathcal{V}, i \neq j, i \neq n+1, j \neq 0\}$, indexed by a, is the arc set. Vertices 0 and $n+1$ are the source and sink representations of the depot, respectively, and the set of remaining vertices $\mathcal{V}_c = \{1, ..., n\}$ denotes the set of n customers. Each vertex i has a demand quantity d_i, and its service start time s_i must lie within the time window $[e_i, l_i]$. For each arc $(i, j) \in \mathcal{A}$, c_{ij} is the travel cost from vertex i to j, and t_{ij} denotes the travel time from i to j that includes, if any, the time to execute service at i. In particular, $d_0 = d_{n+1} = 0, e_0 = e_{n+1} = 0$, and $l_0 = l_{n+1} \geq \max\{l_i : i \in \mathcal{V}_c\}$. Let $\mathcal{V}^+(i)$ and $\mathcal{V}^-(i)$ be the set of successor and predecessor vertices of vertex i.

Assume that there are an unlimited number of homogeneous vehicles, each with a capacity Q. A route r is a path from vertex 0 to vertex $n+1$. Let $\mathcal{V}_c(r)$ and $\mathcal{A}(r)$ denote the set of customer vertices and arcs in a route r, respectively, and let $c_r = \sum_{(i,j)\in\mathcal{A}(r)} c_{ij}$ denote its total travel cost. A route is feasible if the service start time s_i at every vertex is within its time window, and the total quantity delivered at the visited customers does not exceed Q. A feasible solution of the problem consists of a set of feasible routes in which the total delivered quantity to every customer $i \in \mathcal{V}_c$ is equal to d_i. The objective of the SDVRPTW is to find a feasible solution that minimizes the total cost of the routes.

We make use of the following property of optimal SDVRPTW solutions, which is established by Balster et al. (2023). A direct consequence of Property 1 is that, in such an optimal solution, every visit delivers at least $\bar{q}$ units, and hence each route visits at most $Q/\bar{q}$ customers. This bound on route length is exploited throughout our algorithm to prune the search space during both pricing and enumeration.

PROPERTY 1. (Balster et al., 2023) Let $\bar{q} = \gcd(Q, d_1, d_2, \ldots, d_n)$ denote the greatest common divisor of the vehicle capacity and all customer demands. There exists an optimal solution to the SDVRPTW in which the delivery quantity at every visit is an integer multiple of $\bar{q}$.

3.2. Extreme Pattern-Based Formulation

The formulation relies on the following additional notation:

- $\mathcal{R}$: the set of all routes that admit a feasible schedule with respect to the time windows, indexed by r.
- $\mathcal{W}_r$: the set of all extreme delivery patterns having the same visiting sequence as route r and respecting the vehicle capacity, indexed by w.
- δ_{iw}: the quantity delivered to customer $i \in \mathcal{V}_c$ in extreme pattern w.
- x_{rw}: the nonnegative continuous variable indicating the number of vehicles assigned to pattern w compatible with route r.
- x_r: the nonnegative integer variable indicating the number of vehicles assigned to route r.

Here, the *extreme pattern* refers to a pattern that is composed of zero delivery ($\delta_{iw} = 0$), full delivery ($\delta_{iw} = d_i$), and at most one split delivery ($0 < \delta_{iw} < d_i$). Extreme delivery patterns cannot be expressed as a convex combination of other feasible patterns.

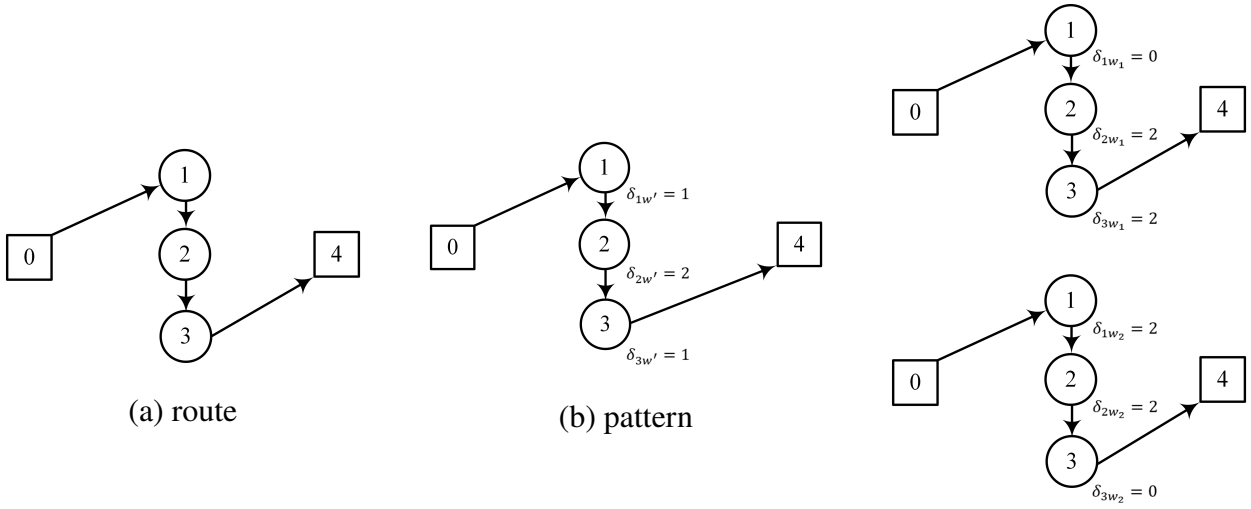

Figure 1 Illustration of route, pattern, and extreme pattern

EXAMPLE 1. Consider an instance with three customers $1, 2, 3$ having demands $d_1 = d_3 = 5$ and $d_2 = 2$, served by vehicles with capacity $Q = 4$. Assume that the time windows at all vertices are sufficiently wide for the illustrated route to be feasible with respect to the time window constraints. Figure 1 illustrates the concepts of a route, a pattern, and extreme patterns.

Figure 1(a) shows a route that starts from depot 0, visits customers $1, 2, 3$ in sequence, and returns to depot 4. Figure 1(b) presents a pattern w' compatible with this route, delivering 1 unit to customer

1, 2 units to customer 2, and 1 unit to customer 3. This pattern is a non-extreme pattern because it involves split deliveries at both customer 1 and customer 3, whereas an extreme pattern admits at most one split delivery. Figure 1(c) displays two extreme delivery patterns associated with the same route. Extreme pattern w_1 involves zero delivery at customer 1 ($\delta_{1w_1} = 0$), full delivery at customer 2 ($\delta_{2w_1} = d_2 = 2$), and a split delivery at customer 3 ($0 < \delta_{3w_1} < d_3$). Extreme pattern w_2 involves zero delivery at customer 3 ($\delta_{3w_2} = 0$), full delivery at customer 2 ($\delta_{2w_2} = d_2 = 2$), and a split delivery at customer 1 ($0 < \delta_{1w_2} < d_1$).

The extreme pattern-based formulation is presented below (Desaulniers 2010).

$$\min \sum_{r\in\mathcal{R}} \sum_{w\in\mathcal{W}_r} c_r x_{rw} \tag{1a}$$

$$\text{s.t.} \sum_{r\in\mathcal{R}} \sum_{w\in\mathcal{W}_r} \delta_{iw} x_{rw} \geqslant d_i \qquad \forall i \in \mathcal{V}_c \tag{1b}$$

$$x_r = \sum_{w\in\mathcal{W}_r} x_{rw} \qquad \forall r \in \mathcal{R} \tag{1c}$$

$$x_{rw} \in \mathbb{R}^+ \qquad \forall r \in \mathcal{R}, w \in \mathcal{W}_r \tag{1d}$$

$$x_r \in \mathbb{Z}^+ \qquad \forall r \in \mathcal{R}. \tag{1e}$$

Objective (1a) minimizes the total costs. Constraints (1b) ensure that the demand of each customer is fulfilled. Constraints (1c) specify the relationship between x_r and x_{rw} subject to the integrality requirements. Constraints (1d) and (1e) define the variable domains.

CG can be applied to solve the linear relaxation of model (2). In each iteration, the master problem selects combinations of extreme delivery patterns from the current pool to cover all customer demands, while the pricing subproblem generates new extreme delivery patterns with negative reduced costs and adds them to the pool. Note that constraints (1c) and (1e), although required in the integer formulation (1), are redundant in the linear relaxation (2) and are therefore omitted (Desaulniers 2010).

$$\min \sum_{r\in\mathcal{R}} \sum_{w\in\mathcal{W}_r} c_r x_{rw} \tag{2a}$$

$$\text{s.t.} \sum_{r\in\mathcal{R}} \sum_{w\in\mathcal{W}_r} \delta_{iw} x_{rw} \geqslant d_i \qquad \forall i \in \mathcal{V}_c \tag{2b}$$

$$x_{rw} \in \mathbb{R}^+ \qquad \forall r \in \mathcal{R}, w \in \mathcal{W}_r. \tag{2c}$$

The pricing subproblem aims to generate a master problem pattern that has the minimum reduced cost under a given dual solution to model (2). We denote by $\boldsymbol{\pi} = (\pi_i)_{i\in\mathcal{V}_c} \geq \mathbf{0}$ the vector of dual variables associated with constraints (2b). The subproblem is a combination of the elementary shortest path problem with resource constraints (ESPPRC) and the linear relaxation of a bounded knapsack problem (LK-BKP). Let x_{ij} denote a binary variable equal to 1 if arc (i, j) is traversed and 0 otherwise, s_i denote the service start time at vertex i, and δ_i denote the quantity delivered to customer i. The pricing subproblem is formulated as follows (Desaulniers 2010):

$$\min \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} - \sum_{i\in\mathcal{V}_c} \pi_i\delta_i \tag{3a}$$

$$\text{s.t.} \sum_{i\in\mathcal{V}_c} x_{0i} = 1 \tag{3b}$$

$$\sum_{j\in\mathcal{V}^+(i)} x_{ij} - \sum_{j\in\mathcal{V}^-(i)} x_{ji} = 0 \qquad \forall i \in \mathcal{V}_c \tag{3c}$$

$$x_{ij}(s_i + t_{ij} - s_j) \leq 0 \qquad \forall (i,j) \in \mathcal{A} \tag{3d}$$

$$e_i \leq s_i \leq l_i \qquad \forall i \in \mathcal{V} \tag{3e}$$

$$\sum_{i\in\mathcal{V}_c} \delta_i \leq Q \tag{3f}$$

$$0 \leq \delta_i \leq \min\{d_i, Q\} \sum_{j\in\mathcal{V}^+(i)} x_{ij} \qquad \forall i \in \mathcal{V}_c \tag{3g}$$

$$x_{ij} \in \{0,1\} \qquad \forall (i,j) \in \mathcal{A}. \tag{3h}$$

Objective (3a) minimizes the reduced cost to generate a feasible vehicle route and its corresponding delivery quantities (i.e., an extreme pattern). Constraint (3b) enforces departure from the depot, while Constraints (3c) ensure flow conservation. Constraints (3d) and (3e) maintain the service time relations and time window constraints. Constraint (3f) ensures that the total vehicle load respects capacity limits. Constraints (3g) impose that the delivered quantity at customer i is bounded above by the smaller of d_i and Q, and is zero when a vehicle does not visit i. Constraints (3h) define the binary requirement.

4. Column Enumeration via Strategic Row Aggregation

Our CESRA framework is summarized in Algorithm 1. It begins by computing a lower bound (LB) from model (2) with cuts (4) and an upper bound (UB) from model (B.2). If LB = UB, optimality is established; otherwise, CESRA proceeds to route enumeration. To enable this, Step 2 reformulates

the extreme pattern-based model through strategic row aggregation, replacing exponentially many column-dependent constraints with a polynomial number of arc-based constraints and thereby making route reduced costs directly computable. Steps 3 and 4 then obtain a nontrivial dual solution and multiple dual solutions for dual picking, respectively, to support effective route enumeration. Besides, to accelerate both the pricing and enumeration stages, we develop tailored label pruning rules for pattern pricing and route enumeration, including a novel rollback dominance rule (Proposition 2) by deriving a lower bound on the delivery quantity for the pattern pricing process, new rollback and generic dominance rules (Proposition 4 and Proposition 5) in the enumeration stage. In Step 5, we employ instance-adaptive IP formulations and, for small-capacity instances, an extreme-pattern variable-fixing rule (Proposition 6) to efficiently generate patterns from the enumerated routes and compute the optimal solution.

Algorithm 1: Column Enumeration via Strategic Row Aggregation

Input: SDVRPTW instance
Output: Optimal solution $\boldsymbol{x}^*$
/ Step 1. Lower and upper bound computation (Section 4.1) */*
$(LB, \hat{\mathcal{W}}) \leftarrow$ CCG on model (2) with cuts (4);
$\hat{\mathcal{R}} \leftarrow \{r(w) : w \in \hat{\mathcal{W}}\}$;
$UB \leftarrow$ solve model (B.2) over $\hat{\mathcal{R}}$;
if $LB = UB$ **then return** $\boldsymbol{x}^*$;
/ Step 2. Model reformulation via strategic row aggregation (Section 4.2) */*
Build aggregated model (5) with route variables $\{x_r\}$ and arc-based constraints (5c);
/ Step 3. Non-trivial dual computation for the aggregated model (Section 4.3) */*
$\bar{\phi} = (\bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\lambda}}, \bar{\omega}) \leftarrow$ solve model (7) subject to (7d) via cut generation;
/ Step 4. Column enumeration (Section 4.4) */*
$\mathcal{R}^* \leftarrow \{r : \bar{c}_r(\bar{\phi}) \le UB - LB(\bar{\phi})\}$;
$\{\bar{\mathcal{R}}_k\} \leftarrow k$-means clustering on $\mathcal{R}^*$;
foreach $\bar{\mathcal{R}}_k$ **do**
  Solve model (B.3) on $\bar{\mathcal{R}}_k$ and prune $\mathcal{R}^*$ via Proposition 3;
end
/ Step 5. Optimal solution computation via instance-adaptive IP models (Section 4.5) */*
if $Q \ge$ *a threshold* **then**
  $\boldsymbol{x}^* \leftarrow$ solve model (8) over $\mathcal{R}^*$ with callbacks;
else
  $\mathcal{W}^* \leftarrow$ apply Algorithm B.1 on $\mathcal{R}^*$;
  $\boldsymbol{x}^* \leftarrow$ solve model (1) over $\mathcal{W}^*$;
end
return $\boldsymbol{x}^*$;

4.1. Step 1: Lower and Upper Bound Computation

We first apply a column-and-cut generation (CCG) procedure to solve model (2), dynamically adding the rounded capacity cuts (4). Let $\mathcal{U}$ denote a subset of customers, $k_{\mathcal{U}} = \lceil \sum_{i\in\mathcal{U}} d_i/Q \rceil$ denote the minimum number of paths required to serve the total demand from $\mathcal{U}$, and $\mathcal{A}^-(\mathcal{U})$ denote the set of outgoing arcs from $\mathcal{U}$. The binary parameter β_{ar} equals 1 if arc a is included in route r, and 0 otherwise. We generate rounded capacity cuts (Baldacci et al. 2008) of the form

$$\sum_{r\in\mathcal{R}} \sum_{w\in\mathcal{W}_r} \sum_{a\in\mathcal{A}^-(\mathcal{U})} \beta_{ar} x_{rw} \geqslant k_{\mathcal{U}} \tag{4}$$

to strengthen the lower bound. We further denote by $\boldsymbol{\lambda} = (\lambda_{\mathcal{U}})_{\mathcal{U}\subseteq\mathcal{V}_c}$ the dual values associated with the rounded capacity cuts, and by $\bar{c}_{ij}$ the modified cost of arc $(i,j) \in \mathcal{A}$, which incorporates the dual contribution induced by the rounded capacity cuts.

To identify new patterns with negative reduced cost, the pricing subproblem is solved using a labeling algorithm (Luo et al. 2017). Let $\mathcal{L} = (i, \tau, V_c, \hat{Q}, G(p,\hat{Q}))$ be a label vector for a pattern ending at vertex $i \in \mathcal{V}$. We denote by $i(\mathcal{L}), \tau(\mathcal{L}), V_c(\mathcal{L})$, $\hat{Q}(\mathcal{L})$ and $G(p(\mathcal{L}), \hat{Q}(\mathcal{L}))$ the components of label $\mathcal{L}$ associated with path $p(\mathcal{L})$, with each component defined as follows:

- i: the last vertex on path p.
- τ: the service starting time at vertex i, which must lie within $[e_i, l_i]$.
- V_c: the set of all visited customers.
- $\hat{Q}$: the set of feasible delivery quantities $\hat{Q} = \{q : 0 \leq q \leq \min\{\sum_{j\in V} d_j, Q\}\}$.
- $G(p,\hat{Q})$: the reduced cost interval of path p with delivery quantity set $\hat{Q}$. $G(p,\hat{Q}) = \{C^{\text{fix}}(p) + C^{\text{var}}(p,q) | \forall q \in \hat{Q}\}$. Here, we denote $\mathcal{A}(p)$ as the set of visited arcs of a partial path p and $C^{\text{fix}}(p) = \sum_{(j,k)\in\mathcal{A}(p)} \bar{c}_{jk}$ as the fixed cost of the path p induced by its visited arcs. We denote $C^{\text{var}}(p,q)$ as the variable cost of the path p determined by the delivered quantity to each vertex on the route for a given total delivered quantity $q \in \hat{Q}$. $C^{\text{var}}(p,q)$ is computed using an efficient exact procedure based on sorting the unit dual profits (Luo et al. 2017). $C^{\text{var}}(p,q)$ is a convex and continuous piecewise linear function, where the slopes, denoted by sl_k, for $k = 1, \cdots, |V_c|$, are $-\pi_j$, for $j \in V_c$.

The detailed label initialization and extension procedures follow that of Luo et al. (2017) and are given in Appendix B.1. The following dominance rule is employed in the labeling algorithm.

PROPOSITION 1. *(Luo et al. 2017) Given two labels $\mathcal{L}_1$ and $\mathcal{L}_2$ ending at the same vertex i. $\mathcal{L}_1$ dominates $\mathcal{L}_2$ if the following conditions are satisfied: (i) $\tau(\mathcal{L}_1) \leq \tau(\mathcal{L}_2)$; (ii) $V_c(\mathcal{L}_1) \subseteq V_c(\mathcal{L}_2)$;*

(iii) for every $0 \le \hat{q} \le Q$, $C^{\text{fix}}(p(\mathcal{L}_1)) + C^{\text{var}}(p(\mathcal{L}_1), \hat{q}) \le C^{\text{fix}}(p(\mathcal{L}_2)) + C^{\text{var}}(p(\mathcal{L}_2), \hat{q})$*; (iv) at least one inequality/subset in (i), (ii), and (iii) is strict.*

The dominance rule in Proposition 1 provides a valid criterion for discarding sub-optimal paths, but its practical implementation is memory- and computation-intensive. It requires *inter-label* comparisons over a large label pool, and condition (iii) demands comparing the two piecewise-linear convex functions at the union of their breakpoints across the entire capacity domain $[0, Q]$.

To overcome this bottleneck, we introduce a lightweight pruning mechanism, the rollback dominance rule in Proposition 2, with the proof given in Appendix A.1. Instead of comparing two distinct labels globally, the rollback rule performs an *intra-label* check that yields a computationally inexpensive sufficient condition for functional dominance based on the local topological detour. It relies only on scalar arithmetic and therefore operates with an almost-zero-memory footprint, requiring neither the storage of historical labels nor pairwise functional comparisons.

PROPOSITION 2 **(Rollback Dominance Rule in Pattern Generation)**. *Consider a feasible label* $\mathcal{L}$ *corresponding to path* $p(\mathcal{L}) = (0, \cdots, m, i, j, k)$ *and a feasible label* $\mathcal{L}'$ *corresponding to path* $p(\mathcal{L}') = (0, \cdots, m, i, k)$*. Both labels follow the same path from vertex* 0 *to* i*. The label* $\mathcal{L}$ *is dominated by the label* $\mathcal{L}'$*, if* $\bar{c}_{ij} + \bar{c}_{jk} - \bar{c}_{ik} \ge \delta_j \pi_j$.

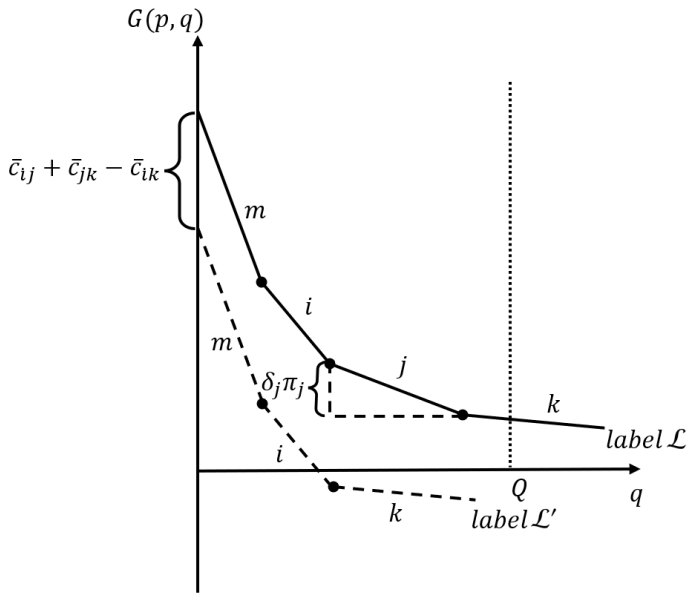

Figure 2 Illustration of rollback dominance rule in pattern generation

Figure 2 illustrates the rollback dominance rule, where label $\mathcal{L}$ ($m \to i \to j \to k$, solid) is compared against $\mathcal{L}'$ ($m \to i \to k$, dashed) that bypasses vertex j. Unlike Proposition 1, which

necessitates generating, storing, and actively querying the alternative label $\mathcal{L}'$ from a massive label pool for functional comparison, our approach performs an intra-label evaluation by checking if the local detour cost ($\bar{c}_{ij} + \bar{c}_{jk} - \bar{c}_{ik}$) outweighs the vertex's dual profit ($\delta_j \pi_j$). Because this condition relies solely on scalar evaluation, we prune path $\mathcal{L}$ instantly without ever needing to construct $\mathcal{L}'$ or evaluate complex piecewise linear functions, thereby reducing the computational time of label dominance.

During the labeling extension process, we first apply Proposition 2 to verify if bypassing the immediate predecessor j is beneficial. If the condition is met, the new label is fathomed. Otherwise, we iteratively apply this rollback check backward along the historical path. For instance, we evaluate a multi-step bypass skipping both preceding vertices i and j, which fathoms the label if $\bar{c}_{mi} + \bar{c}_{ij} + \bar{c}_{jk} - \bar{c}_{mk} \geq \delta_i \pi_i + \delta_j \pi_j$. This process is conducted recursively until the label $\mathcal{L}$ is either successfully dominated or the rollback reaches the depot vertex 0.

After the CCG terminated, we can compute a valid upper bound for the SDVRPTW, with the generated patterns, which, however, is quite loose according to our preliminary experiment. Therefore, we first project the pattern set $\hat{\mathcal{W}}$ obtained onto the corresponding set of routes $\hat{\mathcal{R}} = \{r(w) : w \in \hat{\mathcal{W}}\}$. Then, we adopt an assignment model as shown in model (B.2) of Appendix B.2, which is widely used to obtain an upper bound for the SDVRPTW (Archetti et al. 2008).

4.2. Step 2: Model Reformulation via Strategic Row Aggregation

In the extreme pattern-based formulation (1), the pattern variables are continuous, which prevents direct enumeration. A natural workaround is to incorporate the redundant constraints (1c) into the linear relaxation of the master problem, which allows us to enumerate feasible routes instead of individual patterns, but introduces column-dependent constraints (1c). To address this issue, we propose a strategic row aggregation technique that transfers route-based constraints (1c) to arc-based constraints (5c). Instead of equating flows between individual routes and their corresponding patterns, this formulation aggregates the equations to match the flows on each specific arc. Through this formulation, we reduce the exponentially large set of column-dependent constraints to exactly $|\mathcal{A}|$ arc-based constraints. Let $\mathcal{P}(\mathcal{V}_c)$ denote the set of subsets of customers corresponding to generated rounded capacity cuts. The resulting aggregated model is presented as follows:

[Aggregated model]

$$\min \sum_{r \in \mathcal{R}} \sum_{w \in \mathcal{W}_r} c_r x_{rw}$$

$$\text{s.t.} \sum_{r\in\mathcal{R}} \sum_{w\in\mathcal{W}_r} \delta_{iw} x_{rw} \geqslant d_i \qquad \forall i \in \mathcal{V}_c \tag{5a}$$

$$\sum_{r\in\mathcal{R}} \sum_{w\in\mathcal{W}_r} \sum_{a\in\mathcal{A}^-(\mathcal{U})} \beta_{ar} x_{rw} \geqslant k_{\mathcal{U}} \qquad \forall \mathcal{U} \in \mathcal{P}(\mathcal{V}_c) \tag{5b}$$

$$\sum_{r\in\mathcal{R}} \beta_{ar} x_r - \sum_{r\in\mathcal{R}} \sum_{w\in\mathcal{W}_r} \beta_{ar} x_{rw} = 0 \qquad \forall a \in \mathcal{A} \tag{5c}$$

$$x_{rw} \in \mathbb{R}^+ \qquad \forall r \in \mathcal{R}, w \in \mathcal{W}_r \tag{5d}$$

$$x_r \in \mathbb{Z}^+ \qquad \forall r \in \mathcal{R}. \tag{5e}$$

Intuitively, the constraints (1c) link each route variable with the combinations of patterns, while the constraints (5c) relax constraints (1c) and link the flow on each arc of all routes with that of all patterns. We note that this relaxation does not compromise the optimality or feasibility of the model (2) because the constraints (1c) are redundant. The new constraints (5c) play a crucial role in two respects: (i) they enable the explicit enumeration of integer-valued route variables; and (ii) they facilitate efficient pricing of new routes by avoiding column-dependent rows, as shown below.

Considering the linear relaxation of aggregated model (5), we denote $\boldsymbol{\omega} = (\omega_a)_{a\in\mathcal{A}}$ as the vector of dual variables of constraints (5c). Let $\phi = (\boldsymbol{\pi}, \boldsymbol{\lambda}, \boldsymbol{\omega})$ be a feasible dual solution to linear relaxation of model (5) and Φ be the set of all feasible dual solutions. Let $LB(\phi)$ denote the objective value of the linear relaxation of the aggregated model (5) associated with the dual solution ϕ, which provides a lower bound on the optimal objective value of the aggregated model. Let $\bar{c}_r(\phi) = -\sum_{a\in\mathcal{A}} \beta_{ar}\omega_a$ and $\bar{c}_{rw}(\phi) = c_r - \sum_{i\in\mathcal{V}_c} \delta_{iw}\pi_i - \sum_{\mathcal{U}\in\mathcal{P}(\mathcal{V}_c)} \sum_{a\in\mathcal{A}^-(\mathcal{U})} \beta_{ar}\lambda_{\mathcal{U}} + \sum_{a\in\mathcal{A}} \beta_{ar}\omega_a$ denote the associated reduced cost of route r and pattern rw with respect to dual solution ϕ, respectively. Let UB denote the upper bound of model (1). Then, applying the variable fixing rule (Baldacci et al. 2008), we derive the following proposition concerning the properties of route variables in optimal solutions. The proof is given in Appendix A.2.

PROPOSITION 3 **(Route Variable Fixing)**. *For any $r \in \mathcal{R}$ and $\phi \in \Phi$, if $\bar{c}_r(\phi) > UB - LB(\phi)$, it follows that the variable $x_r = 0$ in any optimal solution to the original problem.*

4.3. Step 3: Non-Trivial Dual Computation for the Aggregated Model

With the aggregated model (5) established, we can enumerate routes with easily computed reduced costs. However, since constraints (5c) are redundant, the associated dual variables $\boldsymbol{\omega}$ may take

trivial values with all ω_a equal to zero. In such case, the reduced cost of any route $r \in \mathcal{R}$ evaluates to

$$\bar{c}_r(\phi) = -\sum_{a \in \mathcal{A}} \beta_{ar}\omega_a = 0, \tag{6}$$

which implies that no route can be pruned by Proposition 3. To address this issue, we exploit the observation that the optimal dual solution to the aggregated model (5) is generally not unique and therefore seek a non-trivial optimal dual solution that enables more route eliminations during the enumeration process.

To obtain a non-trivial dual solution, we formulate model (7) to minimize the sum of the dual variables associated with constraints (5c), while ensuring that the objective value remains equal to the optimal dual objective, z^*. This approach can be viewed as finding an alternative optimal dual solution.

[Perturbed dual problem]

$$\max -\sum_{a \in \mathcal{A}} \omega_a \tag{7a}$$

$$\text{s.t.} \sum_{i \in \mathcal{V}_c} \delta_{iw}\pi_i + \sum_{\mathcal{U} \in \mathcal{P}(\mathcal{V}_c)} \sum_{a \in \mathcal{A}^-(\mathcal{U})} \beta_{ar}\lambda_{\mathcal{U}} - \sum_{a \in \mathcal{A}} \beta_{ar}\omega_a \leqslant c_r \qquad \forall r \in \mathcal{R}, w \in \mathcal{W}_r \tag{7b}$$

$$\sum_{a \in \mathcal{A}} \beta_{ar}\omega_a \leqslant 0 \qquad \forall r \in \mathcal{R} \tag{7c}$$

$$\sum_{i \in \mathcal{V}_c} d_i\pi_i + \sum_{\mathcal{U} \in \mathcal{P}(\mathcal{V})} k_{\mathcal{U}}\lambda_{\mathcal{U}} = z^* \tag{7d}$$

$$\pi_i \in \mathbb{R}^+ \qquad \forall i \in \mathcal{V}_c \tag{7e}$$

$$\lambda_{\mathcal{U}} \in \mathbb{R}^+ \qquad \forall \mathcal{U} \in \mathcal{P}(\mathcal{V}_c) \tag{7f}$$

$$\omega_a \in \mathbb{R} \qquad \forall a \in \mathcal{A}. \tag{7g}$$

Objective (7a) minimizes the sum of the arc-based dual variables. Constraints (7b) and (7c) are associated with the duals of pattern and route variables, respectively. Constraint (7d) ensures that the new solution yields the same objective value as the optimal dual objective.

While in Step 1 we have performed CCG on model (2) and obtained its optimal solution, the perturbed dual problem corresponds to a modified primal formulation. Consequently, to obtain an optimal dual solution to the perturbed dual problem, we employ a cut-generation paradigm. The separation problem of the cuts entails applying a labeling algorithm to generate new columns (both

patterns and routes) with negative reduced costs. Specifically, the labeling algorithm described in Section 4.1 is applied to generate new patterns with negative reduced costs. Additionally, another labeling algorithm described below is applied to generate new routes with negative reduced costs. Let $\overline{\mathcal{L}} = (i, \tau, V_c, \bar{c})$ denote a label associated with a route path p ending at vertex $i \in \mathcal{V}$, where

- i: the last vertex on path p,
- τ: the service starting time at vertex i, which must lie within $[e_i, l_i]$,
- V_c: the set of all visited customers,
- $\bar{c}$: the reduced cost of the path p.

We denote by $i(\overline{\mathcal{L}}), \tau(\overline{\mathcal{L}}), V_c(\overline{\mathcal{L}})$ and $\bar{c}(\overline{\mathcal{L}})$ the components of label $\overline{\mathcal{L}}$. The labeling algorithm is identical to that used for solving the ESPPRC, except that the capacity constraint is excluded.

4.4. Step 4: Column Enumeration

Through Step 3, we obtain an optimal solution $\bar{\phi} = (\bar{\boldsymbol{\pi}}, \bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{\omega}})$ to the perturbed dual problem. Based on the non-trivial dual vector $\bar{\boldsymbol{\omega}}$, we can now enumerate a set of routes $\mathcal{R}^*$ with reduced costs no greater than the duality gap $UB - LB(\bar{\phi})$, which may be included in the optimal solution according to Proposition 3. We apply the labeling algorithm described in Section 4.3 to enumerate routes with the rollback and generic dominance rules modified for the enumeration process. Specifically, we employ the rollback dominance rule established in Proposition 4 below, which eliminates labels from which no complete route with a reduced cost smaller than $UB - LB(\bar{\phi})$ can be extended. The proof is provided in Appendix A.3.

PROPOSITION 4 **(Rollback Dominance Rule in Route Enumeration)**. *Consider a feasible label $\overline{\mathcal{L}}$ corresponding to path $p(\overline{\mathcal{L}}) = (0, \cdots, i, j, k)$ and a feasible label $\overline{\mathcal{L}'}$ corresponding to path $p(\overline{\mathcal{L}'}) = (0, \cdots, i, k)$, where both labels follow the same path from vertex 0 to i. The label $\overline{\mathcal{L}}$ is dominated by the label $\overline{\mathcal{L}'}$, if $\bar{\omega}_{ik} - \bar{\omega}_{ij} - \bar{\omega}_{jk} > UB - LB(\bar{\phi})$.*

In the same spirit, we further develop a new generic dominance rule based on the duality gap as specified in Proposition 5 below and proved in Appendix A.4.

PROPOSITION 5 **(Generic Dominance Rule in Route Enumeration)**. *Given two labels $\overline{\mathcal{L}}_1$ and $\overline{\mathcal{L}}_2$ ending at the same vertex i. $\overline{\mathcal{L}}_1$ dominates $\overline{\mathcal{L}}_2$ if the following conditions are satisfied: (i) $\tau(\overline{\mathcal{L}}_1) \le \tau(\overline{\mathcal{L}}_2)$; (ii) $V_c(\overline{\mathcal{L}}_1) \subseteq V_c(\overline{\mathcal{L}}_2)$; (iii) $\bar{c}(\overline{\mathcal{L}}_1) + UB - LB(\bar{\phi}) < \bar{c}(\overline{\mathcal{L}}_2)$.*

Moreover, considering that the size of set $\mathcal{R}^*$ may still be very large, we discover multiple dual solutions to conduct further variable fixing by adapting the dual picking technique proposed by Yang (2025). The implementation of the dual picking technique is detailed in Appendix B.3.

4.5. Step 5: Optimal Solution Computation via Instance-Adaptive IP Models

Using the set of routes enumerated in Step 4, we can directly solve the IP formulation using an IP solver to obtain the optimal solution. We develop two solution approaches tailored to instances with large and small vehicle capacities, defined as capacities at or above and below a prescribed threshold, respectively.

For instances with large vehicle capacities, we solve the route-based IP model proposed by Balster et al. (2023). Let $h_{r\mathcal{U}}$ denote a binary variable that equals 1 if route $r \in \mathcal{R}^*$ leaves the subset $\mathcal{U} \in \mathcal{P}(\mathcal{V}_c)$ and 0 otherwise. The IP formulation is written as follows:

$$\min \sum_{r\in\mathcal{R}^*} c_r x_r \tag{8a}$$

$$\text{s.t.} \sum_{r\in\mathcal{R}^*} h_{r\mathcal{U}} x_r \geqslant \lceil \sum_{i\in\mathcal{U}} d_i/Q \rceil \qquad \forall \mathcal{U} \in \mathcal{P}(\mathcal{V}_c) \tag{8b}$$

$$x_r \in \mathbb{Z}^+ \qquad \forall r \in \mathcal{R}^*. \tag{8c}$$

Constraints (8b) are the strong capacity cuts (Baldacci et al. 2008). Leveraging Theorem 1 in Balster et al. (2023), we can obtain the optimal solution to the SDVRPTW by solving formulation (8) via an IP solver implementing callbacks.

For instances with small vehicle capacities, we adopt the pattern-based IP formulation, because preliminary computational experiments show that the number of constraints (8b) can become prohibitively large under the route-based formulation. We generate the set of extreme delivery patterns from the enumerated routes following the procedure detailed in Appendix B.4. To further reduce the size of the candidate pattern set, we introduce an extreme pattern-based variable fixing rule. Specifically, let $\mathcal{W}_r^*$ denote the set of extreme delivery patterns for a route $r \in \mathcal{R}^*$. Each extreme pattern $w \in \mathcal{W}_r^*$ uniquely partitions the customer set $\mathcal{V}_c(r)$ into three disjoint subsets based on the delivery quantities δ_{iw}: (i) full delivery set: $\mathcal{F}_w = \{i \in \mathcal{V}_c(r) \mid \delta_{iw} = d_i\}$; (ii) zero delivery set: $\mathcal{Z}_w = \{i \in \mathcal{V}_c(r) \mid \delta_{iw} = 0\}$; (iii) split delivery set: $\mathcal{S}_w = \{i \in \mathcal{V}_c(r) \mid 0 < \delta_{iw} < d_i\}$, where $|\mathcal{S}_w| \leq 1$. We define adjacent extreme delivery patterns in Definition 4.1 and present an illustrative example in Appendix C.

DEFINITION 4.1 **(Adjacency of Extreme Delivery Patterns)**. *Two distinct extreme delivery patterns $w, w' \in \mathcal{W}_r^*$ $(w \neq w')$ are defined as adjacent, denoted as $w' \in \mathcal{N}(w)$, if and only if they share exactly $|\mathcal{V}_c(r)| - 2$ identical boundary constraints on the continuous knapsack polytope. This condition is expressed by the intersection of their respective boundary delivery sets:*

$$|\mathcal{F}_w \cap \mathcal{F}_{w'}| + |\mathcal{Z}_w \cap \mathcal{Z}_{w'}| = |\mathcal{V}_c(r)| - 2. \tag{9}$$

Based on the definition of adjacent extreme delivery patterns, the pattern elimination strategy is formally outlined in Proposition 6 and proved in Appendix A.5

PROPOSITION 6 **(Extreme Pattern Variable Fixing)**. *For any route $r \in \mathcal{R}$, extreme delivery pattern $w \in \mathcal{W}_r^*$, and dual solution $\phi \in \Phi$, suppose that the reduced cost of w and the reduced costs of all its adjacent extreme delivery patterns $w' \in \mathcal{N}(w)$ strictly exceed the optimality gap, mathematically expressed as:*

$$\min_{\bar{w} \in \{w\} \cup \mathcal{N}(w)} \bar{c}_{r\bar{w}}(\phi) > UB - LB(\phi). \tag{10}$$

Then, there exists an optimal solution to formulation (1) *in which $x_{rw} = 0$.*

After applying the extreme pattern variable fixing rule outlined in Proposition 6, we solve the extreme pattern-based model (1) to optimality using an IP solver.

5. Numerical Experiments

We conduct extensive experiments to demonstrate that (i) CESRA achieves performance comparable to, and in some cases better than, that of the state-of-the-art BPC algorithm proposed by Balster et al. (2023) (Section 5.1); (ii) CESRA outperforms the best-known column enumeration method for SDVRPTW (Section 5.2); and (iii) our newly developed acceleration techniques for CESRA contribute to these improvements (Section 5.3).

The experiments are carried out on the standard SDVRPTW benchmark set derived from Solomon's classical VRPTW instances, as described in detail in Appendix D. All experiments were implemented in C++ and conducted on a desktop running Windows 10, equipped with an AMD Ryzen 9950X CPU @ 4.3 GHz and 128 GB of RAM. IPs are solved using IBM ILOG CPLEX 22.1.1. To ensure a fair comparison, we adjust the reported computational times of the BPC by normalizing for CPU frequency (Yang 2025). Specifically, as Balster et al. (2023) used a 2.6 GHz processor, we divide their computational time by a factor of 1.65, corresponding to the ratio of 4.3/2.6.

5.1. Comparison with the State-of-the-Art Method

Table 1 presents the overall computational comparison between CESRA and the BPC algorithm of Balster et al. (2023) on the 50-customer Solomon instances with detailed results provided in Table E.7 in Appendix E. Table 1 details the number of customers (n), the group size (#Instance), and the vehicle capacity (Q). Performance is evaluated based on the average optimality gap in

which a feasible solution is found (Gap (%)), the average solution time in seconds (Time (s)), the number of instances solved to proven optimality (#Opt), and the number of instances with a feasible solution found (#Fea). The final two columns report the relative change in average solution time (Time (%)) and the absolute difference in the number of instances solved to optimality (Δ#Opt). We note that Time (%) is computed only over the instances that are solved to proven optimality by both algorithms. Bold rows provide the average results for each instance class across the three capacity settings.

Table 1 Comparison of CESRA and BPC (Balster et al. 2023) on 50-customer instances

Class	n	#Instance	Q	BPC[a]				CESRA				Improvement	
				Gap (%)	Time (s)	#Opt	#Fea	Gap (%)	Time (s)	#Opt	#Fea	Time (%)	Δ#Opt
C1	50	9	30	0.00	49	9	9	0.00	18	9	9	−63	0
			50	0.00	49	9	9	0.00	9	9	9	−82	0
			100	0.00	51	9	9	0.00	58	9	9	+14	0
C2	50	8	30	0.00	50	8	8	0.00	18	8	8	−64	0
			50	0.00	50	8	8	0.00	11	8	8	−78	0
			100	0.00	54	8	8	0.00	129	8	8	+139	0
C	**50**	**51**	**–**	**0.00**	**50**	**51**	**51**	**0.00**	**40**	**51**	**51**	**−20**	**0**
R1	50	12	30	0.23	1104	10	12	0.03	652	11	12	−61	+1
			50	0.00	395	12	12	0.00	100	12	12	−75	0
			100	0.00	514	12	12	0.00	396	12	12	−23	0
R2	50	11	30	0.00	1467	8	11	0.00	986	11	11	−42	+3
			50	0.00	803	11	11	0.00	339	11	11	−58	0
			100	0.20	1075	8	11	0.08	1790	8	11	−34	0
R	**50**	**69**	**–**	**0.07**	**884**	**61**	**69**	**0.02**	**696**	**65**	**69**	**−48**	**+4**
RC1	50	8	30	0.00	50	8	8	0.00	18	8	8	−64	0
			50	0.00	50	8	8	0.00	19	8	8	−62	0
			100	0.00	50	8	8	0.00	5	8	8	−90	0
RC2	50	8	30	0.00	50	8	8	0.00	21	8	8	−58	0
			50	0.00	49	8	8	0.00	17	8	8	−65	0
			100	0.00	50	8	8	0.00	10	8	8	−80	0
RC	**50**	**48**	**–**	**0.00**	**50**	**48**	**48**	**0.00**	**15**	**48**	**48**	**−70**	**0**

[a] Computational times of Balster et al. (2023) are normalized for CPU frequency by dividing by 1.65 (the ratio of the processors).

For the 50-customer C class, both algorithms attain optimality on all 51 instances with a zero optimality gap. CESRA reduces the average solution time by 20% at the class level, with consistent reductions across the $Q = 30$ and $Q = 50$ settings of C1 and C2. On the C instances with $Q = 100$, however, CESRA is slower than BPC, indicating that the route enumeration phase may become less competitive when vehicle capacities are large. The 50-customer RC class shows a more uniformly favorable result for CESRA. Both algorithms solve all 48 instances to optimality with a zero gap, while CESRA reduces the average solution time by 70% at the class level. The reductions are

consistent across all six subgroups, ranging from 58% on the RC2 instances with $Q = 30$ to 90% on the RC1 instances with $Q = 100$. These results suggest that CESRA matches BPC in solution quality, while requiring relatively less computational effort.

The 50-customer R class is more challenging for both methods. BPC solves 61 of the 69 instances to optimality with an average gap of 0.07%, whereas CESRA solves 65 instances to optimality with an average gap of 0.02% and reduces the average solution time by 48% among commonly solved instances. The improvement is most pronounced on the tightly constrained settings with $Q = 30$, where CESRA closes three additional R2 instances and one additional R1 instance compared to BPC. On the R2 instances with $Q = 100$, both algorithms solve the same eight instances to optimality.

Table 2 Comparison of CESRA and BPC (Balster et al. 2023) on 100-customer instances

Class	n	#Instance	Q	BPC[a]				CESRA				Improvement	
				Gap (%)	Time (s)	#Opt	#Fea	Gap (%)	Time (s)	#Opt	#Fea	Time (%)	Δ#Fea
C1	100	9	30	0.00	215	9	9	0.00	193	9	9	−10	0
			50	0.00	269	9	9	0.00	230	9	9	−14	0
			100	0.00	211	9	9	0.00	80	9	9	−62	0
C2	100	8	30	0.00	202	8	8	0.00	342	8	8	+69	0
			50	0.00	212	8	8	0.00	225	8	8	+6	0
			100	0.00	226	8	8	0.00	360	8	8	+59	0
C	**100**	**51**	**–**	**0.00**	**223**	**51**	**51**	**0.00**	**234**	**51**	**51**	**+5**	**0**
R1	100	12	30	2.53	2182	0	12	1.76	3600	0	12	–[b]	0
			50	2.50	2182	0	12	2.00	3600	0	12	–	0
			100	1.97	1839	3	12	2.79	2870	2	12	−98	0
R2	100	11	30	2.24	2182	0	11	2.13	3600	0	11	–	0
			50	2.29	2182	0	11	1.77	3600	0	11	–	0
			100	2.23	2182	0	8	3.01	3600	0	11	–	+3
R	**100**	**69**	**–**	**2.30**	**2123**	**3**	**66**	**2.24**	**3473**	**2**	**69**	**−98**	**+3**
RC1	100	8	30	0.54	2182	0	8	0.41	3600	0	8	–	0
			50	0.73	2182	0	8	0.77	3600	0	8	–	0
			100	1.67	2182	0	7	1.63	3600	0	8	–	+1
RC2	100	8	30	0.50	2182	0	8	0.33	3600	0	8	–	0
			50	0.80	2182	0	8	0.69	3600	0	8	–	0
			100	1.98	2182	0	6	1.53	3600	0	8	–	+2
RC	**100**	**48**	**–**	**0.98**	**2182**	**0**	**45**	**0.89**	**3600**	**0**	**48**	**–**	**+3**

[a] Computational times of Balster et al. (2023) are normalized for CPU frequency by dividing by 1.65 (the ratio of the processors).

[b] "–" indicates that no instance is solved to optimality by both methods.

Table 2 reports the overall comparison on the 100-customer Solomon instances with detailed results provided in Table E.8 in Appendix E. The column structure mirrors that of Table 1, with the last column (Δ#Fea) reporting the absolute change in the number of instances for which a feasible

solution is obtained within the time limit. This is due to the increased difficulty of the 100-customer instances, where neither algorithm is able to close the optimality gap for most R and RC classes. We note that, because both algorithms frequently reach the one-hour time limit, after applying the CPU frequency calibration, the calibrated time for BPC on instances that exhausted the time limit is reduced from 3600 to 2182 seconds, while the corresponding CESRA time remains at 3600 seconds. The comparison on this benchmark is therefore focused more on solution quality (optimality gap and the number of optimal or feasible solutions), rather than on solution time.

For the 100-customer C class, both algorithms still solve all 51 instances to optimality with a zero gap. CESRA outperforms BPC on the C1 instances, reducing the average solution time by 10%, 14%, and 62% under $Q = 30$, $Q = 50$, and $Q = 100$, respectively. On the C2 instances, however, CESRA is slower than BPC after calibration. The 100-customer R class proves substantially harder for both algorithms at this scale. BPC solves three instances to optimality with an average gap of 2.30%, all of them in the R1 group with $Q = 100$, whereas CESRA solves two instances to optimality with an average gap of 2.24%. The two algorithms therefore deliver comparable performance in terms of the residual optimality gap, with CESRA producing a slightly lower average and BPC closing one additional instance. A more visible difference appears in the number of feasible solutions, where CESRA returns a feasible solution for all R instances within the time limit, including three R2 instances with $Q = 100$ for which BPC does not obtain a feasible solution. For the 100-customer RC class, neither algorithm solves any of the 48 instances to optimality within the time limit. The average gaps are 0.98% for BPC and 0.89% for CESRA, and CESRA returns a feasible solution for three additional instances.

5.2. Comparison with the Best-Known Column Enumeration Method

This section compares CESRA with the pattern-based enumeration method of Casazza et al. (2018), the only existing enumeration-based exact approach for SDVRPs to our knowledge. We adapt their method to the SDVRPTW by enumerating routes, generating the associated patterns, and solving their pattern-based IP formulation (P of Casazza et al. (2018)). The comparison is performed on 100-customer C-class instances with $Q = 50$, the largest benchmark subset satisfying the common-divisor requirement for the method in Casazza et al. (2018). Without this property, the number of feasible delivery quantities grows exponentially, making their pattern enumeration more computationally challenging even for small instances. Table 3 reports the detailed comparison. The columns “LB” and “UB” report the best lower bound and best upper bound, respectively. The

column "Size" reports the total number of enumerated columns, which corresponds to patterns for Casazza et al. (2018) and to routes for CESRA.

Table 3 Comparison of the enumeration method of Casazza et al. (2018) and CESRA on C-class 100-customer instances

Instance	Casazza et al. (2018)					CESRA				
	LB	UB	Gap (%)	Time (s)	Size	LB	UB	Gap (%)	Time (s)	Size
C101	2398.93	2457.20	2.37	3600	389,911	2457.10	2457.10	0.00	212	71
C102	2388.52	2454.60	2.69	3600	3,769,411	2454.00	2454.00	0.00	271	2,909
C103	2385.92	2453.60	2.76	3600	7,290,815	2453.20	2453.20	0.00	352	7,720
C104	2384.43	2450.60	2.70	3600	5,993,185	2450.10	2450.10	0.00	285	4,723
C105	2392.32	2458.10	2.68	3600	1,307,852	2457.10	2457.10	0.00	218	1,320
C106	2396.10	2456.00	2.44	3600	906,524	2456.00	2456.00	0.00	183	64
C107	2392.11	2457.70	2.67	3600	1,799,902	2457.10	2457.10	0.00	204	848
C108	2385.97	2452.50	2.71	3600	1,534,229	2452.20	2452.20	0.00	203	64
C109	2386.26	2450.90	2.64	3600	2,109,012	2450.50	2450.50	0.00	144	1,043
C1	**2390.06**	**2454.58**	**2.63**	**3600**	**2,788,982**	**2454.14**	**2454.14**	**0.00**	**230**	**2,085**
C201	2512.15	2548.80	1.44	3600	5,237,261	2547.40	2547.40	0.00	191	420
C202	2502.64	2545.20	1.67	3600	8,851,545	2543.50	2543.50	0.00	203	5,248
C203	2506.81	2541.50	1.36	3600	5,679,704	2541.10	2541.10	0.00	267	988
C204	2537.20	2537.20	0.00	266	0	2537.20	2537.20	0.00	266	0
C205	2503.82	2543.60	1.56	3600	3,773,611	2542.20	2542.20	0.00	267	1,068
C206	2531.58	2539.00	0.29	3600	285,417	2538.70	2538.70	0.00	226	979
C207	2537.60	2537.60	0.00	157	0	2537.60	2537.60	0.00	157	0
C208	2504.02	2540.20	1.42	3600	4,582,300	2537.60	2537.60	0.00	222	3,782
C2	**2516.98**	**2541.64**	**0.97**	**2753**	**3,551,230**	**2540.66**	**2540.66**	**0.00**	**225**	**1,561**

The results demonstrate that CESRA substantially outperforms the adapted pattern-based enumeration approach across all instances. On the C1 subclass, CESRA closes all nine instances to proven optimality with an average solution time of 230 seconds, while the pattern-based method fails to close any of them within the one-hour time limit and terminates with an average optimality gap of 2.63%. On the C2 subclass, CESRA again solves all eight instances to optimality with an average solution time of 225 seconds, whereas the pattern-based method solves only 2 instances to optimality and leaves the remaining six instances with a residual gap between 0.29% and 1.67%.

The reduction in enumeration size is even more pronounced. On the C1 subclass, CESRA enumerates on average 2,085 routes per instance, compared with roughly 2.8 million patterns enumerated by the pattern-based method, a reduction of three orders of magnitude. A similar gap is observed on the C2 subclass, where CESRA enumerates 1,561 routes on average against roughly 3.55 million patterns. These results indicate that enumerating routes rather than patterns yields a substantially smaller candidate set and a correspondingly lighter downstream integer programming solve, which together account for the observed improvements in both solution time and optimality gap. In instances C204 and C207, both methods reach optimality directly at the root node of the CG procedure, so no columns need to be enumerated by either method.

5.3. Effectiveness of the Tailored Enhancements

In this section, we evaluate the effectiveness of the tailored enhancements proposed for CESRA through three ablation studies that isolate their impact on the pricing, enumeration, and IP solving stages, respectively. We conduct all experiments in this section on the R-class 50-customer instances with $Q = 30$, which we identify as the most challenging configuration that remains tractable. The trends reported below extend to the other classes and capacity settings.

5.3.1. Tailored Rollback Dominance Rule for Pattern Generation. Table 4 reports the impact of the proposed rollback dominance rule (Proposition 2) on pattern generation. For each instance, we record the number of CG iterations (#Iters), the cumulative pricing time in seconds (Time (s)), and the total number of labels generated (#Labels). Instances that exceed the available memory are marked as OOM. The last two columns report the relative reductions in pricing time (Time (%)) and label count (#Labels (%)) obtained by incorporating the rollback dominance rule. Specifically, the two metrics are computed as Time (%) = $(T_{\text{with}} - T_{\text{without}})/T_{\text{without}} \times 100$ and #Labels (%) = $(L_{\text{with}} - L_{\text{without}})/L_{\text{without}} \times 100$, where T_{with} and L_{with} denote the pricing time and label count with the rollback dominance rule, respectively, and T_{without} and L_{without} denote the corresponding quantities without it. The bold rows summarize the results for each subclass, with averages computed over the instances that are not out-of-memory.

On the R1 instances, the rollback dominance rule reduces the average pricing time by 93% and the average number of labels by 49%. The per-instance reductions are consistent across the entire subclass. The number of CG iterations decreases only marginally, indicating that the improvement stems primarily from a reduction in the per-iteration labeling effort. The R2 instances exhibit a more pronounced effect. On the four instances solvable by pricing without rollback, the rollback dominance rule reduces the average pricing time from 24 seconds to less than one second and the average label count from 29.2 million to around 2.7 million, both corresponding to a reduction of more than 90%. More importantly, the configuration without rollback runs out of memory on 7 of the 11 R2 instances, whereas the configuration with rollback solves all 11 instances within the available time and memory budget. This improvement arises because the wide time windows in R2 permit long routes and generate many partial paths, conditions under which the rollback rule offers substantial advantages.

5.3.2. Tailored Rollback and Generic Dominance Rules for Route Enumeration. Table 5 reports the impact of the proposed rollback and generic dominance rules (Proposition 4 and Proposition 5) on route enumeration performance.

Table 4 Impact of the rollback dominance rule on pattern generation on R-class 50-customer instances

Instance	Without rollback			With rollback			Improvement	
	#Iters	Time (s)	#Labels	#Iters	Time (s)	#Labels	Time (%)	#Labels (%)
R101	29	0	757,678	26	0	529,252	/	−30
R102	28	0	2,365,199	32	0	1,458,976	/	−38
R103	27	1	4,815,240	31	0	2,524,418	−100	−48
R104	35	9	21,141,658	33	1	7,376,811	−89	−65
R105	32	0	1,336,474	30	0	777,720	/	−42
R106	30	0	3,079,911	33	0	1,754,112	/	−43
R107	33	2	6,327,741	31	0	2,897,432	−100	−54
R108	35	9	20,488,246	31	1	7,087,949	−89	−65
R109	31	0	2,328,272	29	0	1,299,298	/	−44
R110	28	1	5,164,334	27	0	2,558,388	−100	−50
R111	33	1	6,258,075	31	0	2,942,618	−100	−53
R112	31	4	11,840,180	30	1	5,642,100	−75	−52
R1	**31**	**2**	**7,158,584**	**30**	**0**	**3,070,756**	**−93**	**−49**
R201	29	1	2,582,815	28	0	1,214,741	−100	−53
R202	33	7	13,587,081	32	0	3,018,097	−100	−78
R203	OOM	OOM	OOM	32	2	7,344,376	/	/
R204	OOM	OOM	OOM	31	20	46,285,274	/	/
R205	34	2	6,703,355	30	0	2,419,614	−100	−64
R206	33	84	93,912,081	30	1	4,067,067	−99	−96
R207	OOM	OOM	OOM	36	3	11,240,109	/	/
R208	OOM	OOM	OOM	34	28	68,208,335	/	/
R209	OOM	OOM	OOM	35	1	6,975,184	/	/
R210	OOM	OOM	OOM	30	1	4,359,552	/	/
R211	OOM	OOM	OOM	30	10	26,918,076	/	/
R2[a]	**32**	**24**	**29,196,333**	**30**	**0**	**2,679,880**	**−100**	**−73**

[a] **R2** row is computed over the four R2 instances solved by both configurations.

On the R1 instances, the proposed enhancements reduce the average enumeration time by 22% and the average label count by 4%. The per-instance reductions are consistent across the subclass, with time reductions reaching 50% on R108 and R112, and label reductions reaching 17% on R108 and 9% on R104. Mirroring the trend observed for pricing in Table 4, the benefits of the proposed enhancements become considerably more pronounced on the R2 instances, where the wide time windows induce long routes that produce a large label population during enumeration. The average enumeration time drops from 165 to 1 seconds, and the average label count drops from about 313 million to about 8 million. The average label reduction percentages are 83% and 60% as reported in the summary row. More importantly, the configuration without the enhancements runs out of memory on 4 of the 11 R2 instances, whereas the configuration with the enhancements completes enumeration on all 11 instances within the available time and memory budget.

5.3.3. Instance-Adaptive Model for IP Solving. Table 6 compares the route-based and pattern-based IP formulations. #Cuts denotes the number of valid inequalities generated during

Table 5 Impact of the rollback and generic dominance rules on route enumeration on R-class 50-customer instances

Instance	Without rollback or dominance rule		With rollback and dominance rule		Improvement	
	Time (s)	#Labels	Time (s)	#Labels	Time (%)	#Labels (%)
R101	0	216,188	0	215,576	/	0
R102	0	1,891,487	0	1,870,016	/	−1
R103	1	7,072,067	1	6,955,328	0	−2
R104	18	86,026,238	12	77,957,375	−33	−9
R105	0	419,321	0	414,629	/	−1
R106	0	2,706,620	0	2,684,690	/	−1
R107	0	2,223,395	0	2,107,625	/	−5
R108	8	31,370,609	4	25,952,471	−50	−17
R109	0	1,841,915	0	1,828,094	/	−1
R110	1	9,283,325	1	9,253,133	0	0
R111	1	11,336,789	1	11,261,564	0	−1
R112	2	10,390,637	1	9,907,463	−50	−5
R1	**3**	**13,731,549**	**2**	**12,533,997**	**−22**	**−4**
R201	1	10,096,571	1	10,067,960	0	0
R202	2	5,186,189	0	2,707,079	−100	−48
R203	625	1,090,419,881	2	8,885,627	−100	−99
R204	OOM	OOM	112	412,731,728	/	/
R205	13	4,497,842	1	4,254,317	−92	−5
R206	15	108,741,791	2	10,504,061	−87	−90
R207	OOM	OOM	5	21,762,362	/	/
R208	OOM	OOM	270	1,009,466,102	/	/
R209	494	966,016,193	4	19,300,439	−99	−98
R210	5	10,271,195	0	2,257,871	−100	−78
R211	OOM	OOM	851	3,059,735,972	/	/
R2[a]	**165**	**313,604,237**	**1**	**8,282,479**	**−83**	**−60**

[a] **R2** row is computed over the seven R2 instances solved by both configurations.

the route-based IP solving process. To ensure a fair comparison, the reported solution time for the pattern-based formulation includes all additional steps beyond IP solving, namely, converting the enumerated routes into compatible patterns, performing pattern variable fixing, and finally solving the resulting IP.

The pattern-based formulation produces tighter lower bounds. The average lower bound improves from 1551.21 (route-based) to 1564.10 (pattern-based) on R1 instances, and from 1547.19 to 1555.84 on R2 instances. This improvement stems from two sources. First, the pattern-based formulation is structurally tighter: by encoding the delivery quantities explicitly within the variables, it eliminates a portion of the relaxation slack that arises when the route-based formulation handles delivery quantities through continuous auxiliary decisions. Second, the pattern-based formulation closes the gap more efficiently, requiring far fewer cuts than the route-based formulation, which generates an average of 2,733 cuts per R1 instance and 1,655 cuts per R2 instance, with a peak of 14,037 cuts on R101.

Table 6 Comparison of route-based and pattern-based IP formulations on R-class 50-customer instances

Instance	Route-based					Pattern-based				Improvement	
	LB	UB	Gap (%)	Time (s)	#Cuts	LB	UB	Gap (%)	Time (s)	Gap (%)	Time (%)
R101	1600.28	1619.10	1.18	3600	14,037	1618.90	1618.90	0.00	79	−100	−98
R102	1565.43	1587.20	1.39	3600	4,747	1581.30	1581.30	0.00	152	−100	−96
R103	1550.91	1564.20	0.86	3600	1,614	1564.20	1564.20	0.00	1187	−100	−67
R104	1536.44	1560.40	1.56	3600	823	1548.60	1548.60	0.00	496	−100	−86
R105	1571.11	1580.30	0.58	3600	2,156	1580.10	1580.10	0.00	106	−100	−97
R106	1554.63	1559.60	0.32	3600	1,743	1559.60	1559.60	0.00	104	−100	−97
R107	1543.05	1548.60	0.36	3600	1,255	1548.60	1548.60	0.00	107	−100	−97
R108	1530.96	1562.00	2.03	3600	1,414	1548.60	1548.60	0.00	541	−100	−85
R109	1551.54	1562.80	0.73	3600	1,464	1562.80	1562.80	0.00	156	−100	−96
R110	1533.90	1558.00	1.57	3600	1,373	1555.15	1558.00	0.25	3600	−84	0
R111	1545.25	1552.80	0.49	3600	1,692	1552.80	1552.80	0.00	511	−100	−86
R112	1530.98	1548.60	1.15	3600	474	1548.60	1548.60	0.00	781	−100	−78
R1	**1551.21**	**1566.97**	**1.02**	**3600**	**2,733**	**1564.10**	**1564.34**	**0.02**	**652**	**−99**	**−82**
R201	1566.61	1578.60	0.77	3600	4,946	1578.60	1578.60	0.00	607	−100	−83
R202	1555.28	1559.60	0.28	3600	3,134	1559.60	1559.60	0.00	276	−100	−92
R203	1548.45	1548.60	0.01	3600	1,621	1548.60	1548.60	0.00	232	−100	−94
R204	1537.73	1548.60	0.71	3600	1,736	1548.60	1548.60	0.00	1328	−100	−63
R205	1551.18	1566.20	0.97	3600	1,031	1565.40	1565.40	0.00	405	−100	−89
R206	1547.56	1556.90	0.60	3600	786	1556.90	1556.90	0.00	694	−100	−81
R207	1546.50	1548.60	0.14	3600	1,164	1548.60	1548.60	0.00	529	−100	−85
R208	1530.17	1548.70	1.21	3600	207	1548.60	1548.60	0.00	1864	−100	−48
R209	1539.45	1557.90	1.20	3600	1,055	1557.90	1557.90	0.00	3600	−100	0
R210	1552.80	1552.80	0.00	1387	670	1552.80	1552.80	0.00	222	0	−84
R211	1543.31	1548.60	0.34	3600	1,850	1548.60	1548.60	0.00	1095	−100	−70
R2	**1547.19**	**1555.92**	**0.57**	**3399**	**1,655**	**1555.84**	**1555.84**	**0.00**	**987**	**−91**	**−72**

In terms of solution quality and efficiency, the pattern-based formulation closes 22 of the 23 instances to proven optimality, with the only exception being R110, which terminates at a 0.25% gap. In contrast, the route-based formulation closes only one instance. Correspondingly, the average solution time drops from 3600 to 652 seconds on R1 (an 82% reduction) and from 3399 to 987 seconds on R2 (a 72% reduction). It is worth noting that the pattern-based runtime includes the additional overheads of converting enumerated routes into compatible patterns and applying pattern variable fixing, which route-based model does not need. Even with these additional steps, the pattern-based formulation still delivers substantial time reductions, which further underscores the advantage of our instance-adaptive IP solving procedure.

6. Conclusions

In this paper, we study the SDVRPTW that permits each customer's demand to be served jointly by multiple vehicles within time windows. Existing exact methods for the SDVRPTW have primarily relied on B&C, B&P, and BPC frameworks. Alongside these established frameworks, column enumeration emerges as a complementary exact paradigm and has been successfully applied to several

other VRP variants; however, it remains largely unexplored for the SDVRPTW. Its application is hindered because pattern variables are continuous, while route variables introduce column-dependent rows that complicate reduced-cost computation. We address these challenges with a new CESRA method that enumerates routes and uses strategic row aggregation to replace exponentially many column-dependent constraints with a polynomial number of arc-based constraints. We further accelerate different stages of the algorithm through several problem-specific enhancements, including a rollback dominance rule based on a lower bound of the delivery quantity for pattern generation, rollback and generic dominance rules tailored to the route enumeration stage, and instance-adaptive IP formulations for computing the optimal solution.

Computational results show that CESRA is the first column-enumeration method for the SDVRPTW to match and, in some cases, outperform the state-of-the-art BPC algorithm of Balster et al. (2023). On the 50-customer benchmark, it reduces average solution time by 20%–70% and proves optimality for four additional instances, while remaining competitive on the 100-customer benchmark. Compared with the only existing enumeration-based method, CESRA solves all tested 100-customer C-class instances to optimality in 228 seconds on average, whereas the existing method solves only two within one hour and enumerates three orders of magnitude more columns. Ablation studies confirm the effectiveness of the proposed dominance rules and instance-adaptive IP formulations.

The proposed CESRA framework establishes a highly adaptable exact solution paradigm, opening several promising avenues for future research. First, the variable fixing power in CESRA can be further amplified by integrating advanced dual-space perturbation strategies (e.g., pre-enumeration dual picking) and richer families of valid inequalities (e.g., subset-row cuts) accompanied by dynamic cut management. Second, the CESRA framework remains theoretically applicable to the SDVRP (without time windows), but the absence of time windows significantly prolongs feasible routes, demanding more sophisticated pricing algorithms and stronger bounding techniques to manage the expanded state space during enumeration. Third, CESRA opens a new direction for solving set covering models with extreme columns by strategically aggregating the column-dependent rows induced by convexity constraints. This idea may apply to other complex problems, such as the multi-item capacitated lot-sizing problem, where production schedules are represented as convex combinations of extreme plans (Degraeve and Jans 2007).

References

Andelmin J, Bartolini E (2017) An exact algorithm for the green vehicle routing problem. *Transportation Science* 51(4):1288–1303.

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

Archetti C, Bouchard M, Desaulniers G (2011) Enhanced branch and price and cut for vehicle routing with split deliveries and time windows. *Transportation Science* 45(3):285–298.

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

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, Savelsbergh MWP (2008) An optimization-based heuristic for the split delivery vehicle routing problem. *Transportation Science* 42(1):22–31.

Baldacci R, Bartolini E, Mingozzi A (2011a) An exact algorithm for the pickup and delivery problem with time windows. *Operations Research* 59(2):414–426.

Baldacci R, Bartolini E, Mingozzi A, Valletta A (2011b) An exact algorithm for the period routing problem. *Operations Research* 59(1):228–241.

Baldacci R, Christofides N, Mingozzi A (2008) An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming* 115(2):351–385.

Baldacci R, Mingozzi A, Wolfler Calvo R (2011c) An exact method for the capacitated location-routing problem. *Operations Research* 59(5):1284–1296.

Balster I, Bulhões T, Munari P, Pessoa AA, Sadykov R (2023) A new family of route formulations for split delivery vehicle routing problems. *Transportation Science* 57(5):1359–1378.

Bartolini E, Cordeau JF, Laporte G (2013) Improved lower bounds and exact algorithm for the capacitated arc routing problem. *Mathematical Programming* 137(1):409–452.

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

Ben-Ameur W, Neto J (2007) Acceleration of cutting-plane and column generation algorithms: Applications to network design. *Networks* 49(1):3–17.

Bianchessi N, Drexl M, Irnich S (2019) The split delivery vehicle routing problem with time windows and customer inconvenience constraints. *Transportation Science* 53(4):1067–1084.

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

738 Casazza M, Ceselli A, Chemla D, Meunier F, Wolfler Calvo R (2018) The multiple vehicle balancing problem. *Networks*
739 72(3):337–357.

740 Contardo C, Cordeau JF, Gendron B (2014) An exact algorithm based on cut-and-column generation for the capacitated
741 location-routing problem. *INFORMS Journal on Computing* 26(1):88–102.

742 Costa L, Contardo C, Desaulniers G (2019) Exact branch-price-and-cut algorithms for vehicle routing. *Transportation*
743 *Science* 53(4):946–985.

744 Dantzig G, Fulkerson R, Johnson S (1954) Solution of a large-scale traveling-salesman problem. *Journal of the*
745 *Operations Research Society of America* 2(4):393–410.

746 Degraeve Z, Jans R (2007) A new Dantzig-Wolfe reformulation and branch-and-price algorithm for the capacitated
747 lot-sizing problem with setup times. *Operations Research* 55(5):909–920.

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

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

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

752 Feillet D, Gendreau M, Medaglia AL, Walteros JL (2010) A note on branch-and-cut-and-price. *Operations Research*
753 *Letters* 38(5):346–353.

754 Gamst M, Lusby RM, Ropke S (2024) Exact and heuristic methods for the split delivery vehicle routing problem.
755 *Transportation Science* 58(4):741–760.

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

758 Gouveia L, Leitner M, Ruthmair M (2023) Multi-depot routing with split deliveries: Models and a branch-and-cut
759 algorithm. *Transportation Science* 57(2):512–530.

760 He P, Hao JK (2023) General edge assembly crossover-driven memetic search for split delivery vehicle routing.
761 *Transportation Science* 57(2):482–511.

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

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

766 Luo Z, Qin H, Zhu W, Lim A (2017) Branch and price and cut for the split-delivery vehicle routing problem with time
767 windows and linear weight-related cost. *Transportation Science* 51(2):668–687.

768 Lusby RM, Gamst M, Ropke S, Spoorendonk S (2020) Simultaneously exploiting two formulations: An exact Benders
769 decomposition approach. *Computers & Operations Research* 123:105041.

770 Miller CE, Tucker AW, Zemlin RA (1960) Integer programming formulation of traveling salesman problems. *Journal*
771 *of the Association for Computing Machinery* 7(4):326–329.

Munari P, Savelsbergh M (2022) Compact formulations for split delivery routing problems. *Transportation Science* 56(4):1022–1043.

Muter I, Birbil Şİ, Bülbül K (2013) Simultaneous column-and-row generation for large-scale linear programs with column-dependent-rows. *Mathematical Programming* 142(1):47–82.

Paradiso R, Roberti R, Laganá D, Dullaert W (2020) An exact solution framework for multitrip vehicle-routing problems with time windows. *Operations Research* 68(1):180–198.

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

Pessoa A, Sadykov R, Uchoa E, Vanderbeck F (2020) A generic exact solver for vehicle routing and related problems. *Mathematical Programming* 183(1):483–523.

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

Toth P, Vigo D (2014) *Vehicle Routing: Problems, Methods, and Applications*. MOS-SIAM Series on Optimization (Philadelphia, PA: Society for Industrial and Applied Mathematics).

Yang Y (2023) An exact price-cut-and-enumerate method for the capacitated multitrip vehicle routing problem with time windows. *Transportation Science* 57(1):230–251.

Yang Y (2025) DeLuxing: Deep Lagrangian underestimate fixing for column-generation-based exact methods. *Operations Research* 73(3):1184–1207.

You Z, Yang Y, Wang X, Yin W (2026) Two-stage learning to branch in branch-price-and-cut algorithms for solving vehicle routing problems exactly. *Operations Research* 74(4):1811–1833.

Ziegler GM (1995) *Lectures on Polytopes*. Graduate Texts in Mathematics (New York: Springer).

Electronic Companions

Appendix A: Proofs of Propositions

A.1. Proof of Proposition 2

Let label $\hat{\mathcal{L}}$ correspond to path $p(\hat{\mathcal{L}}) = (0, \ldots, i)$, which shares the exact routing sequence and delivery quantities from vertex 0 to i with labels $\mathcal{L}$ and $\mathcal{L}'$. For the service start time, we have

$$\begin{aligned}\tau(\mathcal{L}) &= \max\{e_k, \max\{e_j, \tau(\hat{\mathcal{L}}) + t_{ij}\} + t_{jk}\} \\ &\geq \max\{e_k, \tau(\hat{\mathcal{L}}) + t_{ij} + t_{jk}\} \\ &\geq \max\{e_k, \tau(\hat{\mathcal{L}}) + t_{ik}\} = \tau(\mathcal{L}'),\end{aligned}$$

which establishes condition (i) in Proposition 1.

Furthermore, because $V_c(\mathcal{L}) = V_c(\mathcal{L}') \cup \{j\}$, it holds that $V_c(\mathcal{L}') \subsetneq V_c(\mathcal{L})$, satisfying conditions (ii) and (iv) in Proposition 1.

For the fixed part of the reduced cost, we can show that

$$C^{\text{fix}}(p(\mathcal{L})) - C^{\text{fix}}(p(\mathcal{L}')) = C^{\text{fix}}(p(\hat{\mathcal{L}})) + \bar{c}_{ij} + \bar{c}_{jk} - (C^{\text{fix}}(p(\hat{\mathcal{L}})) + \bar{c}_{ik}) = \bar{c}_{ij} + \bar{c}_{jk} - \bar{c}_{ik}.$$

For the variable component of the reduced cost, let q_1 be the total demand of vertices in $\mathcal{L}$ whose unit dual profit π_v strictly exceeds π_j, i.e., $q_1 = \sum_{v \in V_c(\mathcal{L}): \pi_v > \pi_j} d_v$.

- For $0 \leq q \leq q_1$: The vertex j receives zero allocation ($\delta_j = 0$). Then, $C^{\text{var}}(p(\mathcal{L}), q)$ and $C^{\text{var}}(p(\mathcal{L}'), q)$ are identical.
- For $q_1 < q \leq q_1 + d_j$: Capacity is allocated to vertex j in $\mathcal{L}$, yielding an allocation amount $\delta_j(q) = q - q_1$ and adding $\pi_j \delta_j(q)$ to the profit. In $\mathcal{L}'$, because j is bypassed, this identical capacity is allocated to subsequent vertices with unit profits $\pi_v \leq \pi_j$. Since $\pi_v \delta_j(q) \leq \pi_j \delta_j(q)$, we have

$$\begin{aligned}C^{\text{var}}(p(\mathcal{L}), q) =& C^{\text{var}}(p(\mathcal{L}'), q) - (\pi_j - \pi_v)\delta_j(q) \\ &\geq C^{\text{var}}(p(\mathcal{L}'), q) - \pi_j \delta_j(q)\end{aligned}$$

- For $q > q_1 + d_j$: Vertex j is fully satisfied in $\mathcal{L}$, i.e., $\delta_j(q) = d_j$. Any further capacity is allocated to vertices with $\pi_v \leq \pi_j$ in both labels. The variable cost reduction that $\mathcal{L}$ can maintain over $\mathcal{L}'$ is strictly bounded by the total profit provided by j, yielding $C^{\text{var}}(p(\mathcal{L}), q) \geq C^{\text{var}}(p(\mathcal{L}'), q) - \pi_j \delta_j(q)$.

Combining the fixed and variable components, the total reduced cost function $G(p(\mathcal{L}), \{q\})$ satisfies:

$$\begin{aligned} G(p(\mathcal{L}), \{q\}) &= C^{\text{fix}}(p(\mathcal{L})) + C^{\text{var}}(p(\mathcal{L}), q) \\ &\geq \left(C^{\text{fix}}(p(\mathcal{L}')) + \bar{c}_{ij} + \bar{c}_{jk} - \bar{c}_{ik}\right) + \left(C^{\text{var}}(p(\mathcal{L}'), q) - \pi_j \delta_j(q)\right) \\ &= G(p(\mathcal{L}'), \{q\}) + \underbrace{(\bar{c}_{ij} + \bar{c}_{jk} - \bar{c}_{ik} - \pi_j \delta_j(q))}_{\geq 0} \\ &\geq G(p(\mathcal{L}'), \{q\}), \end{aligned}$$

for every $0 \leq q \leq Q$. This satisfies condition (iii) of Proposition 1. Consequently, label $\mathcal{L}$ is dominated by label $\mathcal{L}'$.

A.2. Proof of Proposition 3

Denote by $\boldsymbol{\gamma} = (\gamma_r)_{r \in \mathcal{R}}$ the dual variables of constraints (1c). We can write out the dual problem of the linear relaxation of the master problem (1), strengthened by generated constraints (4), as follows:

$$\max \sum_{i \in \mathcal{V}_c} d_i \pi_i + \sum_{\mathcal{U} \in \mathcal{P}(\mathcal{V}_c)} k_{\mathcal{U}} \lambda_{\mathcal{U}} \tag{A.1a}$$

$$\text{s.t.} \sum_{i \in \mathcal{V}_c} \delta_{iw} \pi_i + \sum_{\mathcal{U} \in \mathcal{P}(\mathcal{V}_c)} \sum_{a \in \mathcal{A}^-(\mathcal{U})} \beta_{ar} \lambda_{\mathcal{U}} - \gamma_r \leqslant c_r \qquad \forall r \in \mathcal{R}, w \in \mathcal{W}_r \tag{A.1b}$$

$$\pi_i \in \mathbb{R}^+ \qquad \forall i \in \mathcal{V}_c \tag{A.1c}$$

$$\lambda_{\mathcal{U}} \in \mathbb{R}^+ \qquad \forall \mathcal{U} \in \mathcal{P}(\mathcal{V}_c) \tag{A.1d}$$

$$\gamma_r \in \mathbb{R} \qquad \forall r \in \mathcal{R}. \tag{A.1e}$$

We can also write out the dual problem of the linear relaxation of the aggregated model (5) as follows:

$$\max \sum_{i \in \mathcal{V}_c} d_i \pi_i + \sum_{\mathcal{U} \in \mathcal{P}(\mathcal{V}_c)} k_{\mathcal{U}} \lambda_{\mathcal{U}} \tag{A.2a}$$

$$\text{s.t.} \sum_{i \in \mathcal{V}_c} \delta_{iw} \pi_i + \sum_{\mathcal{U} \in \mathcal{P}(\mathcal{V}_c)} \sum_{a \in \mathcal{A}^-(\mathcal{U})} \beta_{ar} \lambda_{\mathcal{U}} - \sum_{a \in \mathcal{A}} \beta_{ar} \omega_a \leqslant c_r \qquad \forall r \in \mathcal{R}, w \in \mathcal{W}_r \tag{A.2b}$$

$$\sum_{a \in \mathcal{A}} \beta_{ar} \omega_a \leqslant 0 \qquad \forall r \in \mathcal{R} \tag{A.2c}$$

$$\pi_i \in \mathbb{R}^+ \qquad \forall i \in \mathcal{V}_c \tag{A.2d}$$

$$\lambda_{\mathcal{U}} \in \mathbb{R}^+ \qquad \forall \mathcal{U} \in \mathcal{P}(\mathcal{V}_c) \tag{A.2e}$$

$$\omega_a \in \mathbb{R} \qquad \forall a \in \mathcal{A}. \tag{A.2f}$$

For any feasible dual solution $\phi = (\boldsymbol{\pi}, \boldsymbol{\lambda}, \boldsymbol{\omega})$ of the linear relaxation of the aggregated model (5), we can construct a feasible dual solution $\psi = (\boldsymbol{\pi}, \boldsymbol{\lambda}, \boldsymbol{\gamma})$ for original model (1) by setting $\gamma_r = \sum_{a \in \mathcal{A}} \beta_{ar} \omega_a, \forall r \in \mathcal{R}$ and keeping the value of $\boldsymbol{\pi}$ and $\boldsymbol{\lambda}$.

According to Baldacci et al. (2008), given a feasible dual solution $\psi = (\boldsymbol{\pi}, \boldsymbol{\lambda}, \boldsymbol{\gamma})$ of the original model (1) with objective value $LB(\psi)$, any route variable x_r must be 0 in the optimal integer solution if its reduced cost strictly exceeds the optimality gap. Based on constraints (A.1), this fixing condition is expressed as:

$$-\gamma_r > UB - LB(\psi) \implies x_r = 0.$$

By our construction, the feasible dual solution $\phi = (\boldsymbol{\pi}, \boldsymbol{\lambda}, \boldsymbol{\omega})$ of the aggregated model (5) maps to ψ such that $\gamma_r = \sum_{a \in \mathcal{A}} \beta_{ar} \omega_a$. Consequently, the reduced cost evaluated under ϕ is $\bar{c}_r(\phi) = -\gamma_r$. Since the dual objective functions are identical, the bounds are strictly equal, i.e., $LB(\phi) = LB(\psi)$. Substituting these into the fixing condition directly yields:

$$\bar{c}_r(\phi) > UB - LB(\phi) \implies x_r = 0,$$

which completes the proof.

A.3. Proof of Proposition 4

Consider any feasible extension p_{ext} that extends $\overline{\mathcal{L}}$ from vertex k to the depot, yielding the complete route $p_1 = p(\overline{\mathcal{L}}) \oplus p_{\text{ext}}$.

Since the partial label $\overline{\mathcal{L}'}$ bypasses vertex j, it consumes fewer or equal resources (e.g., time and cumulative load) compared to $\overline{\mathcal{L}}$ upon reaching vertex k. Therefore, any extension p_{ext} that is feasible for $\overline{\mathcal{L}}$ is guaranteed to be a physically feasible extension for $\overline{\mathcal{L}'}$. Let $p_2 = p(\overline{\mathcal{L}'}) \oplus p_{\text{ext}}$ be this valid complete route.

The reduced cost of the complete route p_1 can be expressed as:

$$\begin{aligned}
\bar{c}(p_1) &= \bar{c}(\overline{\mathcal{L}}) + \bar{c}(p_{\text{ext}}) \\
&= \left(\bar{c}(\overline{\mathcal{L}'}) + \omega_{ik} - \omega_{ij} - \omega_{jk}\right) + \bar{c}(p_{\text{ext}}) \\
&= \left(\bar{c}(\overline{\mathcal{L}'}) + \bar{c}(p_{\text{ext}})\right) + \omega_{ik} - \omega_{ij} - \omega_{jk} \\
&= \bar{c}(p_2) + \omega_{ik} - \omega_{ij} - \omega_{jk}.
\end{aligned}$$

Because p_2 is a feasible complete route generated under the feasible dual solution, the weak duality condition guarantees that its reduced cost is non-negative:

$$\bar{c}(p_2) \geq 0.$$

Substituting this into the decomposition of $\bar{c}(p_1)$, we establish a strict lower bound for any complete route extended from $\overline{\mathcal{L}}$:

$$\bar{c}(p_1) \geq 0 - \omega_{ij} - \omega_{jk} + \omega_{ik}.$$

Given the premise that $\omega_{ik} - \omega_{ij} - \omega_{jk} > UB - LB(\bar{\phi})$, we have $\bar{c}(p_1) > UB - LB(\bar{\phi})$. Because this holds for every possible feasible extension p_{ext}, no complete route originating from the partial label $\overline{\mathcal{L}}$ can have reduced cost less than or equal to $UB - LB(\bar{\phi})$. Thus, according to Proposition 3, the partial label $\overline{\mathcal{L}}$ can be eliminated.

A.4. Proof of Proposition 5

Consider any feasible extension p_{ext} that extends $\overline{\mathcal{L}}_2$ from vertex i to the depot, yielding the complete route $p_2 = p(\overline{\mathcal{L}}_2) \oplus p_{\text{ext}}$.

By condition (i), $\overline{\mathcal{L}}_1$ consumes no more resources (e.g., time) at vertex i than $\overline{\mathcal{L}}_2$, so p_{ext} remains resource-feasible when appended to $\overline{\mathcal{L}}_1$. By condition (ii), $V_c(\overline{\mathcal{L}}_1) \subseteq V_c(\overline{\mathcal{L}}_2)$, so any vertex in p_{ext} that does not appear in $V_c(\overline{\mathcal{L}}_2)$ does not appear in $V_c(\overline{\mathcal{L}}_1)$ either, preserving elementarity. Therefore, p_{ext} is also a feasible extension for $\overline{\mathcal{L}}_1$, and $p_1 = p(\overline{\mathcal{L}}_1) \oplus p_{\text{ext}}$ defines a feasible complete route.

The reduced costs of the two complete routes share the same extension contribution, and their difference can be expressed as:

$$\bar{c}(p_2) - \bar{c}(p_1) = \left(\bar{c}(\overline{\mathcal{L}}_2) + \bar{c}(p_{\text{ext}})\right) - \left(\bar{c}(\overline{\mathcal{L}}_1) + \bar{c}(p_{\text{ext}})\right) = \bar{c}(\overline{\mathcal{L}}_2) - \bar{c}(\overline{\mathcal{L}}_1).$$

Because p_1 is a feasible complete route generated under the feasible dual solution, the weak duality condition guarantees that its reduced cost is non-negative, i.e., $\bar{c}(p_1) \geq 0$. Substituting this lower bound and condition (iii) into the relation above, we obtain:

$$\bar{c}(p_2) = \bar{c}(p_1) + \left(\bar{c}(\overline{\mathcal{L}}_2) - \bar{c}(\overline{\mathcal{L}}_1)\right) > 0 + UB - LB(\bar{\phi}) = UB - LB(\bar{\phi}).$$

Because this bound holds for every feasible extension p_{ext}, every complete route originating from the partial label $\overline{\mathcal{L}}_2$ has reduced cost larger than $UB - LB(\bar{\phi})$. Thus, according to Proposition 3, the partial label $\overline{\mathcal{L}}_2$ can be eliminated.

A.5. Proof of Proposition 6

Let $\boldsymbol{x}^*$ be an arbitrary optimal solution to formulation (1), with objective value $z^* \leq UB$. If $x^*_{rw} = 0$, the result follows immediately. Otherwise, we have $x^*_{rw} \geq 1$ and $x^*_r \geq 1$. Let $\widetilde{w} = \sum_{u \in \mathcal{W}^*_r} \theta_u u$ denote a general pattern realized by route r, where $\theta_u = x^*_{ru}/x^*_r$. In this case, the original representation of

$\widetilde{w}$ uses extreme pattern w as shown in the illustrative Figure A.1. Let $\mathcal{H}_r = \mathrm{conv}(\mathcal{W}_r^*)$ denote the continuous delivery-pattern polytope associated with route r. By the variable-fixing principle of Baldacci et al. (2008), the reduced-cost contribution of route r satisfies

$$x_r^* \bar{c}_{r\widetilde{w}}(\phi) = \sum_{u \in \mathcal{W}_r^*} x_{ru}^* \bar{c}_{ru}(\phi) \le z^* - LB(\phi) \le UB - LB(\phi).$$

Since $x_r^* \ge 1$, we have $\bar{c}_{r\widetilde{w}}(\phi) \le UB - LB(\phi)$, i.e., $\widetilde{w} \in \overline{\mathcal{H}}_r := \{u \in \mathcal{H}_r : \bar{c}_{ru}(\phi) \le UB - LB(\phi)\}$.

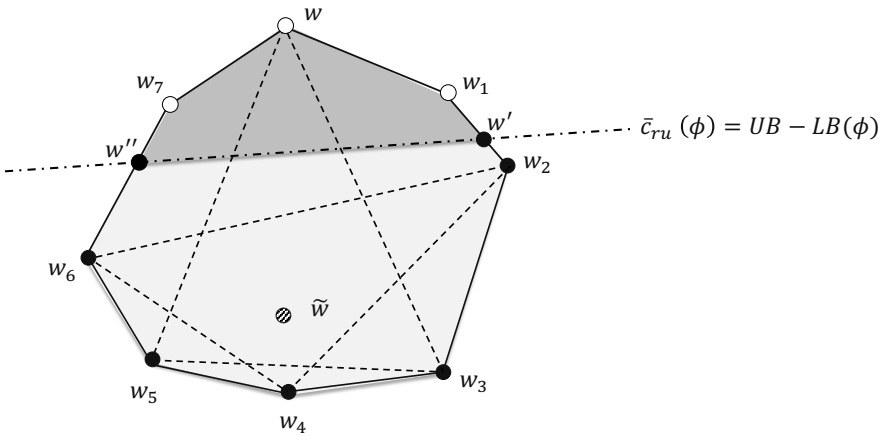

Figure A.1 An illustrative example of the case where $x_{rw}^* \ge 1$

Notes. $\mathcal{H}_r = \mathrm{conv}(\mathcal{W}_r^*) = \mathrm{conv}(\{w, w_1, w_2, w_3, w_4, w_5, w_6, w_7\})$ denotes the pattern polytope associated with route r; $\overline{\mathcal{H}}_r = \{u \in \mathcal{H}_r : \bar{c}_{ru}(\phi) \le UB - LB(\phi)\} = \mathrm{conv}(\{w_2, w_3, w_4, w_5, w_6, w', w''\})$ denotes the retained subpolytope; $\bar{c}_{ru}(\phi) = UB - LB(\phi)\}$ denotes the bounding hyperplane; w is the target extreme pattern; $\mathcal{N}(w) = \{w_1, w_7\}$ denotes the set of extreme patterns adjacent to w; and $\widetilde{w}$ denotes the general pattern, whose original representation is given by the extreme patterns $\{w, w_3, w_5\}$, whereas its equivalent representation is given by $\{w_2, w_4, w_6\}$.

By the premise of the proposition, w and all its adjacent extreme patterns have reduced costs strictly greater than $UB - LB(\phi)$. Hence, w and all vertices adjacent to it lie outside $\overline{\mathcal{H}}_r$, as illustrated in Figure A.1.

By the standard halfspace-truncation property of polytopes, every vertex of $\overline{\mathcal{H}}_r$ is either an original vertex of $\mathcal{H}_r$ satisfying the reduced-cost bound or the intersection of the bounding hyperplane $\bar{c}_{ru}(\phi) = UB - LB(\phi)$ with an edge of $\mathcal{H}_r$ (Ziegler 1995, Chapter 1). Since no edge incident to w intersects $\overline{\mathcal{H}}_r$, none of these vertices requires w in its representation. Moreover, by the Minkowski–Weyl theorem, every point in $\overline{\mathcal{H}}_r$, including $\widetilde{w}$, can be expressed as a convex combination of the vertices of $\overline{\mathcal{H}}_r$ (Ziegler 1995, Theorem 1.1). Consequently, there exists an equivalent representation

$$\widetilde{w} = \sum_{u \in \mathcal{W}_r^* \setminus \{w\}} \widehat{\theta}_u u, \qquad \sum_{u \in \mathcal{W}_r^* \setminus \{w\}} \widehat{\theta}_u = 1, \qquad \widehat{\theta}_u \ge 0,$$

that assigns zero weight to w. Replacing the original representation by this equivalent one preserves the realized delivery pattern and the objective value while setting $x^*_{rw} = 0$. Therefore, an optimal solution exists in which $x_{rw} = 0$.

Appendix B: Supplementary Implementation Details

B.1. Label Initialization and Extension for Pattern Generation

Let $\mathcal{L}_0 = (0, 0, \emptyset, \{0\}, \{0\})$ be the initial label at vertex 0. Extending a label $\mathcal{L}$ to vertex $j \in \mathcal{V} \setminus (V_c(\mathcal{L}) \cup \{0\})$ yields a new label $\mathcal{L}'$, computed according to the following relations:

$$\tau(\mathcal{L}') = \max\{e_j, \tau(\mathcal{L}) + t_{ij}\} \tag{B.1a}$$

$$V_c(\mathcal{L}') = \begin{cases} V_c(\mathcal{L}) \cup \{j\}, & j \neq n+1, \\ V_c(\mathcal{L}), & j = n+1. \end{cases} \tag{B.1b}$$

$$C^{\text{fix}}(p(\mathcal{L}')) = C^{\text{fix}}(p(\mathcal{L})) + \bar{c}_{ij}. \tag{B.1c}$$

If $j \neq n+1$, the function $C^{\text{var}}(p(\mathcal{L}'), q)$ is updated by the following procedure: (i) identifying the index k such that $sl_{k-1} \leq -\pi_j \leq sl_k$, (ii) shifting the portion of $C^{\text{var}}(p(\mathcal{L}'), q)$ corresponding to $k' \geq k$ to the right by d_j units and downward by $\pi_j d_j$ units, and (iii) truncating the function at $q = Q$. The generated label $\mathcal{L}'$ is feasible if the time window constraint is satisfied, i.e., $\tau_j \in [e_j, l_j]$.

B.2. Assignment Model Used to Compute the Upper Bound

We first introduce the following notation and decision variables:

- $\hat{\mathcal{R}}$: a subset of routes projected from the patterns generated in model (2).
- μ_{ir}: nonnegative continuous variable indicating the quantity delivered at customer $i \in \mathcal{V}_c$ in route $r \in \hat{\mathcal{R}}$.
- x_r: nonnegative integer variable indicating the number of vehicles assigned to route r.

Then, the assignment model used to compute the upper bound can be presented as follows:

$$\min \sum_{r \in \hat{\mathcal{R}}} c_r x_r \tag{B.2a}$$

$$\text{s.t.} \sum_{r \in \hat{\mathcal{R}}} \mu_{ir} \geq d_i \qquad \forall i \in \mathcal{V}_c \tag{B.2b}$$

$$\sum_{i \in \mathcal{V}_c(r)} \mu_{ir} \leq Q x_r \qquad \forall r \in \hat{\mathcal{R}} \tag{B.2c}$$

$$\mu_{ir} = 0 \qquad \forall r \in \hat{\mathcal{R}}, i \notin \mathcal{V}_c(r) \tag{B.2d}$$

$$x_r \in \mathbb{Z}^+ \qquad \forall r \in \hat{\mathcal{R}} \tag{B.2e}$$

$$\mu_{ir} \in \mathbb{R}^+ \qquad \forall r \in \hat{\mathcal{R}}, i \in \mathcal{V}_c. \tag{B.2f}$$

Objective (B.2a) minimizes the total costs. Constraints (B.2b) ensure that each customer is fully served. Constraints (B.2c)–(B.2d) enforce the load capacity of the vehicles. Constraints (B.2e)–(B.2f) define the domains of the decision variables.

B.3. Dual-Picking Technique

We first cluster the routes based on their structural similarities by k-means algorithm, where the distance between two routes $r_1, r_2 \in \mathcal{R}^*$ is defined as the cardinality of the symmetric difference between $\mathcal{A}(r_1)$ and $\mathcal{A}(r_2)$, i.e., $|(\mathcal{A}(r_1)\backslash\mathcal{A}(r_2)) \cup (\mathcal{A}(r_2)\backslash\mathcal{A}(r_1))|$. For each cluster $\bar{\mathcal{R}} \subseteq \mathcal{R}^*$, we construct and solve a linear program designed to find a targeted dual solution that minimizes the elimination of those specific redundant variables, which is given as follows:

$$\begin{aligned} &\max - \sum_{r\in\bar{\mathcal{R}}} \sum_{a\in\mathcal{A}} \beta_{ar}\omega_a \\ &\text{s.t. (7b)} - \text{(7c)}, \text{(7e)} - \text{(7g)}. \end{aligned} \tag{B.3}$$

After obtaining an optimal dual to the above formulation, we can apply Proposition 3 to further eliminate unpromising routes.

B.4. Pseudocode for Extreme Delivery Patterns Generation Based on Routes

The pseudocode for generating extreme delivery patterns based on routes is presented in Algorithm B.1.

Appendix C: An Illustrative Example of Adjacent Delivery Patterns

To illustrate Definition 4.1, consider a route r visiting three customers, i.e., $\mathcal{V}_c(r) = \{1, 2, 3\}$, and three extreme delivery patterns $w_1, w_2, w_3 \in \mathcal{W}_r^*$ configured as follows:

- w_1: Customer 1 receives a full delivery, customer 2 receives zero, and customer 3 receives a split delivery. Thus, $\mathcal{F}_{w_1} = \{1\}$ and $\mathcal{Z}_{w_1} = \{2\}$.
- w_2: Customer 1 receives a full delivery, customer 3 receives zero, and customer 2 receives a split delivery. Thus, $\mathcal{F}_{w_2} = \{1\}$ and $\mathcal{Z}_{w_2} = \{3\}$.
- w_3: Customer 3 receives a full delivery, customer 2 receives zero, and customer 1 receives a split delivery. Thus, $\mathcal{F}_{w_3} = \{3\}$ and $\mathcal{Z}_{w_3} = \{2\}$.

Algorithm B.1: Generation of Extreme Delivery Patterns

Input: A route r, capacity Q, demands $d_i, i \in \mathcal{V}_c$.
Output: A set of all extreme delivery patterns $\mathcal{P}_r$.

Let $\mathcal{V}_c(r) = \{i_1, i_2, \ldots, i_{|\mathcal{V}_c(r)|}\}$ be the set of customers in r;
if $\sum_{i \in \mathcal{V}_c(r)} d_i \leq Q$ **then**
  Initialize pattern $p \leftarrow (0, \ldots, 0)$;
  for $i \in \mathcal{V}_c(r)$ **do**
    $p_i \leftarrow d_i$
  end
  return $\{ p \}$;
else
  $\mathcal{P}_r \leftarrow \emptyset$;
  for *each subset* $\bar{\mathcal{V}}_c(r) \subset \mathcal{V}_c(r)$ *such that* $\sum_{i \in \bar{\mathcal{V}}_c(r)} d_i \leq Q$ **do**
    $q_{\text{full}} \leftarrow \sum_{i \in \bar{\mathcal{V}}_c(r)} d_i$;
    for *each* $i \in \mathcal{V}_c(r) \setminus \bar{\mathcal{V}}_c(r)$ **do**
      if $q_{full} + d_i > Q$ **then**
        Initialize pattern $p \leftarrow (0, \ldots, 0)$;
        for $j \in \bar{\mathcal{V}}_c(r)$ **do**
          $p_j \leftarrow d_j$
        end
        $p_i \leftarrow Q - q_{\text{full}}$
        $\mathcal{P}_r \leftarrow \mathcal{P}_r \cup \{p\}$;
      end
    end
  end
  return $\mathcal{P}_r$;
end

According to the adjacency condition (9), pattern w_1 is adjacent to w_2 because they share the identical full delivery set ($\mathcal{F}_{w_1} = \mathcal{F}_{w_2} = \{1\}$). Similarly, w_1 is adjacent to w_3 because their zero delivery sets are identical ($\mathcal{Z}_{w_1} = \mathcal{Z}_{w_3} = \{2\}$). However, w_2 and w_3 are **not** adjacent. Neither their full delivery sets ($\{1\} \neq \{3\}$) nor their zero delivery sets ($\{3\} \neq \{2\}$) match. Geometrically, transitioning directly between w_2 and w_3 requires altering the bounds of all three customers simultaneously, meaning they do not share a 1-dimensional edge on the knapsack polytope.

Appendix D: Description of Benchmark Instances

The standard SDVRPTW benchmark set is derived from Solomon's classical VRPTW instances, comprising 56 instances with 50 customers and 100 customers, respectively, each spanning six classes (C1, C2, R1, R2, RC1, RC2). The C, R, and RC classes correspond to clustered, randomly distributed, and mixed customer locations, respectively. Within each geographic type, the type-1

subclass features narrow time windows and short scheduling horizons (small l_{n+1}) that limit the number of customers per route, while the type-2 subclass features wide time windows and long horizons (large l_{n+1}) that allow longer routes. The C instances are generally the easiest, while the R and RC instances tend to be harder due to their weaker geographic structure and longer feasible routes.

For each instance, we consider three vehicle capacities $Q \in \{30, 50, 100\}$, resulting in 336 test instances in total. As detailed in Section 4.5, we compute the optimal solution using the pattern-based formulation for instances with a small vehicle capacity ($Q = 30$ in the experiments) and using the route-based formulation for instances with large vehicle capacities ($Q = 50$ and $Q = 100$). We follow the preprocessing protocol of Balster et al. (2023), where travel times are first rounded to one decimal place, each arc travel time is then replaced by the shortest path between its endpoints, and the service time of the tail vertex is finally embedded into each outgoing arc travel time. To ensure a fair comparison, a one-hour time limit is imposed per instance, consistent with Balster et al. (2023).

Appendix E: Table E.7 and Table E.8 of Detailed Experiment Results

Table E.7: Detailed results of CESRA on 50-customer instances

Instance	Q	LB	UB	Gap (%)	Time (s)	Instance	Q	LB	UB	Gap (%)	Time (s)
C101	30	1599.50	1599.50	0.00	15	C102	30	1599.50	1599.50	0.00	19
C103	30	1598.30	1598.30	0.00	13	C104	30	1598.30	1598.30	0.00	28
C105	30	1599.50	1599.50	0.00	19	C106	30	1599.50	1599.50	0.00	22
C107	30	1599.50	1599.50	0.00	19	C108	30	1598.30	1598.30	0.00	17
C109	30	1598.30	1598.30	0.00	15	C101	50	1015.80	1015.80	0.00	7
C102	50	1013.00	1013.00	0.00	10	C103	50	1012.30	1012.30	0.00	8
C104	50	1010.20	1010.20	0.00	18	C105	50	1015.80	1015.80	0.00	4
C106	50	1015.80	1015.80	0.00	5	C107	50	1015.80	1015.80	0.00	7
C108	50	1011.80	1011.80	0.00	11	C109	50	1010.10	1010.10	0.00	9
C101	100	587.50	587.50	0.00	5	C102	100	584.60	584.60	0.00	8
C103	100	582.10	582.10	0.00	41	C104	100	578.80	578.80	0.00	431
C105	100	587.50	587.50	0.00	6	C106	100	587.50	587.50	0.00	6
C107	100	587.50	587.50	0.00	9	C108	100	584.00	584.00	0.00	7
C109	100	579.80	579.80	0.00	11	C201	30	1778.30	1778.30	0.00	14
C202	30	1778.30	1778.30	0.00	18	C203	30	1778.30	1778.30	0.00	21
C204	30	1778.30	1778.30	0.00	23	C205	30	1778.30	1778.30	0.00	11
C206	30	1778.30	1778.30	0.00	13	C207	30	1778.30	1778.30	0.00	25
C208	30	1778.30	1778.30	0.00	15	C201	50	1159.40	1159.40	0.00	8
C202	50	1156.90	1156.90	0.00	8	C203	50	1156.90	1156.90	0.00	13
C204	50	1156.90	1156.90	0.00	19	C205	50	1156.90	1156.90	0.00	7
C206	50	1156.90	1156.90	0.00	11	C207	50	1156.90	1156.90	0.00	12
C208	50	1156.90	1156.90	0.00	12	C201	100	693.10	693.10	0.00	28
C202	100	686.20	686.20	0.00	18	C203	100	685.40	685.40	0.00	51
C204	100	684.80	684.80	0.00	735	C205	100	684.80	684.80	0.00	17

Continued on next page

Table E.7 (continued)

Instance	Q	LB	UB	Gap (%)	Time (s)	Instance	Q	LB	UB	Gap (%)	Time (s)
C206	100	684.80	684.80	0.00	37	C207	100	684.80	684.80	0.00	105
C208	100	684.80	684.80	0.00	43						
R101	30	1618.90	1618.90	0.00	79	R102	30	1581.30	1581.30	0.00	152
R103	30	1564.20	1564.20	0.00	1187	R104	30	1548.60	1548.60	0.00	496
R105	30	1580.10	1580.10	0.00	106	R106	30	1559.60	1559.60	0.00	104
R107	30	1548.60	1548.60	0.00	107	R108	30	1548.60	1548.60	0.00	541
R109	30	1562.80	1562.80	0.00	156	R110	30	1555.15	1558.00	0.25	3600
R111	30	1552.80	1552.80	0.00	511	R112	30	1548.60	1548.60	0.00	781
R101	50	1190.70	1190.70	0.00	10	R102	50	1114.20	1114.20	0.00	15
R103	50	1081.30	1081.30	0.00	40	R104	50	1054.70	1054.70	0.00	268
R105	50	1132.30	1132.30	0.00	44	R106	50	1080.20	1080.20	0.00	31
R107	50	1063.90	1063.90	0.00	56	R108	50	1053.50	1053.50	0.00	317
R109	50	1081.80	1081.80	0.00	53	R110	50	1061.80	1061.80	0.00	131
R111	50	1061.50	1061.50	0.00	51	R112	50	1053.50	1053.50	0.00	184
R101	100	1043.80	1043.80	0.00	0	R102	100	913.20	913.20	0.00	0
R103	100	804.70	804.70	0.00	22	R104	100	708.60	708.60	0.00	3458
R105	100	918.10	918.10	0.00	7	R106	100	821.50	821.50	0.00	15
R107	100	753.90	753.90	0.00	34	R108	100	701.30	701.30	0.00	494
R109	100	804.10	804.10	0.00	13	R110	100	744.40	744.40	0.00	25
R111	100	752.70	752.70	0.00	374	R112	100	702.40	702.40	0.00	312
R201	30	1578.60	1578.60	0.00	607	R202	30	1559.60	1559.60	0.00	276
R203	30	1548.60	1548.60	0.00	232	R204	30	1548.60	1548.60	0.00	1328
R205	30	1565.40	1565.40	0.00	405	R206	30	1556.90	1556.90	0.00	694
R207	30	1548.60	1548.60	0.00	528	R208	30	1548.60	1548.60	0.00	1864
R209	30	1557.90	1557.90	0.00	3600	R210	30	1552.80	1552.80	0.00	222
R211	30	1548.60	1548.60	0.00	1095	R201	50	1107.70	1107.70	0.00	31
R202	50	1080.20	1080.20	0.00	78	R203	50	1059.20	1059.20	0.00	110
R204	50	1053.50	1053.50	0.00	657	R205	50	1085.90	1085.90	0.00	97
R206	50	1071.50	1071.50	0.00	165	R207	50	1059.20	1059.20	0.00	341
R208	50	1053.50	1053.50	0.00	841	R209	50	1054.50	1054.50	0.00	43
R210	50	1072.90	1072.90	0.00	804	R211	50	1053.50	1053.50	0.00	558
R201	100	843.00	843.00	0.00	18	R202	100	771.60	771.60	0.00	33
R203	100	717.50	717.50	0.00	872	R204	100	691.38	693.90	0.36	3600
R205	100	758.80	758.80	0.00	12	R206	100	728.10	728.10	0.00	104
R207	100	707.00	707.00	0.00	1007	R208	100	686.92	689.60	0.39	3600
R209	100	720.40	720.40	0.00	208	R210	100	742.20	742.20	0.00	2189
R211	100	691.48	691.90	0.06	3600						
RC101	30	2739.50	2739.50	0.00	10	RC102	30	2739.50	2739.50	0.00	14
RC103	30	2739.50	2739.50	0.00	8	RC104	30	2739.50	2739.50	0.00	23
RC105	30	2739.60	2739.60	0.00	24	RC106	30	2739.50	2739.50	0.00	32
RC107	30	2739.50	2739.50	0.00	19	RC108	30	2739.50	2739.50	0.00	11
RC101	50	1708.30	1708.30	0.00	32	RC102	50	1700.50	1700.50	0.00	12
RC103	50	1696.80	1696.80	0.00	21	RC104	50	1696.70	1696.70	0.00	16
RC105	50	1700.10	1700.10	0.00	7	RC106	50	1699.00	1699.00	0.00	11
RC107	50	1698.60	1698.60	0.00	28	RC108	50	1696.70	1696.70	0.00	22
RC101	100	990.50	990.50	0.00	4	RC102	100	960.20	960.20	0.00	4
RC103	100	936.20	936.20	0.00	8	RC104	100	915.90	915.90	0.00	6
RC105	100	957.40	957.40	0.00	5	RC106	100	936.40	936.40	0.00	5
RC107	100	915.10	915.10	0.00	5	RC108	100	911.90	911.90	0.00	6
RC201	30	2739.50	2739.50	0.00	16	RC202	30	2739.50	2739.50	0.00	16
RC203	30	2739.50	2739.50	0.00	21	RC204	30	2739.50	2739.50	0.00	29
RC205	30	2739.50	2739.50	0.00	25	RC206	30	2739.50	2739.50	0.00	15
RC207	30	2739.50	2739.50	0.00	21	RC208	30	2739.50	2739.50	0.00	26
RC201	50	1708.30	1708.30	0.00	11	RC202	50	1700.50	1700.50	0.00	19

Continued on next page

Table E.7 (continued)

Instance	Q	LB	UB	Gap (%)	Time (s)	Instance	Q	LB	UB	Gap (%)	Time (s)
RC203	50	1696.80	1696.80	0.00	15	RC204	50	1696.70	1696.70	0.00	13
RC205	50	1700.40	1700.40	0.00	11	RC206	50	1699.00	1699.00	0.00	20
RC207	50	1698.60	1698.60	0.00	24	RC208	50	1696.70	1696.70	0.00	22
RC201	100	966.20	966.20	0.00	3	RC202	100	946.50	946.50	0.00	8
RC203	100	926.40	926.40	0.00	16	RC204	100	915.90	915.90	0.00	14
RC205	100	946.70	946.70	0.00	8	RC206	100	940.80	940.80	0.00	11
RC207	100	924.10	924.10	0.00	9	RC208	100	911.90	911.90	0.00	13

Table E.8: Detailed results of CESRA on 100-customer instances

Instance	Q	LB	UB	Gap (%)	Time (s)	Instance	Q	LB	UB	Gap (%)	Time (s)
C101	30	3889.60	3889.60	0.00	89	C102	30	3889.50	3889.50	0.00	146
C103	30	3888.10	3888.10	0.00	213	C104	30	3887.60	3887.60	0.00	172
C105	30	3889.60	3889.60	0.00	161	C106	30	3889.10	3889.10	0.00	149
C107	30	3889.60	3889.60	0.00	195	C108	30	3887.70	3887.70	0.00	247
C109	30	3887.60	3887.60	0.00	362	C101	50	2457.10	2457.10	0.00	212
C102	50	2454.00	2454.00	0.00	271	C103	50	2453.20	2453.20	0.00	352
C104	50	2450.10	2450.10	0.00	285	C105	50	2457.10	2457.10	0.00	218
C106	50	2456.00	2456.00	0.00	183	C107	50	2457.10	2457.10	0.00	204
C108	50	2452.20	2452.20	0.00	203	C109	50	2450.50	2450.50	0.00	144
C101	100	1391.10	1391.10	0.00	120	C102	100	1377.50	1377.50	0.00	101
C103	100	1376.40	1376.40	0.00	114	C104	100	1370.60	1370.60	0.00	135
C105	100	1389.30	1389.30	0.00	65	C106	100	1381.70	1381.70	0.00	56
C107	100	1382.30	1382.30	0.00	31	C108	100	1377.20	1377.20	0.00	32
C109	100	1374.60	1374.60	0.00	63	C201	30	3957.10	3957.10	0.00	163
C202	30	3955.40	3955.40	0.00	349	C203	30	3950.80	3950.80	0.00	419
C204	30	3949.70	3949.70	0.00	376	C205	30	3949.70	3949.70	0.00	391
C206	30	3949.70	3949.70	0.00	337	C207	30	3949.70	3949.70	0.00	357
C208	30	3949.70	3949.70	0.00	341	C201	50	2547.40	2547.40	0.00	191
C202	50	2543.50	2543.50	0.00	203	C203	50	2541.10	2541.10	0.00	267
C204	50	2537.20	2537.20	0.00	266	C205	50	2542.20	2542.20	0.00	267
C206	50	2538.70	2538.70	0.00	226	C207	50	2537.60	2537.60	0.00	157
C208	50	2537.60	2537.60	0.00	222	C201	100	1465.70	1465.70	0.00	315
C202	100	1462.50	1462.50	0.00	721	C203	100	1454.00	1454.00	0.00	309
C204	100	1446.10	1446.10	0.00	776	C205	100	1455.90	1455.90	0.00	145
C206	100	1452.70	1452.70	0.00	215	C207	100	1453.30	1453.30	0.00	194
C208	100	1451.00	1451.00	0.00	207						
R101	30	2878.57	2932.50	1.87	3600	R102	30	2849.33	2874.60	0.89	3600
R103	30	2809.82	2849.60	1.42	3600	R104	30	2797.80	2862.60	2.32	3600
R105	30	2833.61	2930.70	3.43	3600	R106	30	2818.42	2889.20	2.51	3600
R107	30	2800.33	2849.50	1.76	3600	R108	30	2797.55	2834.70	1.33	3600
R109	30	2810.33	2838.90	1.02	3600	R110	30	2798.28	2835.10	1.32	3600
R111	30	2802.85	2854.40	1.84	3600	R112	30	2795.59	2835.20	1.42	3600
R101	50	1996.95	2026.70	1.49	3600	R102	50	1941.50	1977.90	1.87	3600
R103	50	1877.06	1922.20	2.40	3600	R104	50	1853.35	1899.10	2.47	3600
R105	50	1923.76	1952.50	1.49	3600	R106	50	1897.85	1938.80	2.16	3600
R107	50	1859.54	1909.30	2.68	3600	R108	50	1850.76	1880.90	1.63	3600
R109	50	1868.30	1897.40	1.56	3600	R110	50	1852.40	1886.30	1.83	3600
R111	50	1861.00	1901.80	2.19	3600	R112	50	1845.04	1887.00	2.27	3600
R101	100	1638.40	1638.40	0.00	3	R102	100	1478.90	1478.90	0.00	36
R103	100	1264.35	1283.20	1.49	3600	R104	100	1150.55	1204.00	4.65	3600

Continued on next page

Table E.8 (continued)

Instance	Q	LB	UB	Gap (%)	Time (s)	Instance	Q	LB	UB	Gap (%)	Time (s)
R105	100	1384.22	1401.00	1.21	3600	R106	100	1294.58	1323.90	2.26	3600
R107	100	1193.42	1222.50	2.44	3600	R108	100	1145.46	1205.10	5.21	3600
R109	100	1235.44	1289.90	4.41	3600	R110	100	1180.36	1239.00	4.97	3600
R111	100	1181.50	1225.70	3.74	3600	R112	100	1137.00	1172.10	3.09	3600
R201	30	2832.11	2884.70	1.86	3600	R202	30	2822.53	2865.60	1.53	3600
R203	30	2798.40	2848.30	1.78	3600	R204	30	2796.58	2862.20	2.35	3600
R205	30	2811.17	2876.30	2.32	3600	R206	30	2807.26	2863.20	1.99	3600
R207	30	2795.04	2862.70	2.42	3600	R208	30	2788.18	2850.70	2.24	3600
R209	30	2788.19	2859.90	2.57	3600	R210	30	2796.79	2850.00	1.90	3600
R211	30	2795.59	2863.20	2.42	3600	R201	50	1908.14	1931.20	1.21	3600
R202	50	1889.40	1913.50	1.28	3600	R203	50	1856.53	1890.60	1.84	3600
R204	50	1850.10	1883.60	1.81	3600	R205	50	1872.25	1919.30	2.51	3600
R206	50	1864.39	1900.40	1.93	3600	R207	50	1850.24	1880.00	1.61	3600
R208	50	1844.99	1886.40	2.24	3600	R209	50	1852.45	1879.80	1.48	3600
R210	50	1864.56	1884.60	1.07	3600	R211	50	1844.99	1891.80	2.54	3600
R201	100	1304.10	1337.50	2.56	3600	R202	100	1240.22	1259.90	1.59	3600
R203	100	1166.86	1189.00	1.90	3600	R204	100	1137.14	1159.40	1.96	3600
R205	100	1210.31	1269.70	4.91	3600	R206	100	1177.81	1205.40	2.34	3600
R207	100	1153.13	1216.60	5.50	3600	R208	100	1132.81	1157.50	2.18	3600
R209	100	1167.07	1213.00	3.94	3600	R210	100	1177.45	1196.00	1.58	3600
R211	100	1132.00	1184.40	4.63	3600						
RC101	30	4277.66	4295.50	0.42	3600	RC102	30	4265.22	4276.20	0.26	3600
RC103	30	4260.14	4271.00	0.25	3600	RC104	30	4260.10	4271.90	0.28	3600
RC105	30	4266.02	4275.50	0.22	3600	RC106	30	4265.93	4302.40	0.85	3600
RC107	30	4259.78	4280.10	0.48	3600	RC108	30	4258.57	4280.10	0.51	3600
RC101	50	2776.42	2811.20	1.25	3600	RC102	50	2743.73	2768.00	0.88	3600
RC103	50	2726.95	2736.60	0.35	3600	RC104	50	2723.81	2742.70	0.69	3600
RC105	50	2755.33	2780.60	0.92	3600	RC106	50	2741.16	2759.20	0.66	3600
RC107	50	2727.13	2742.90	0.58	3600	RC108	50	2721.58	2743.60	0.81	3600
RC101	100	1756.63	1794.80	2.17	3600	RC102	100	1680.40	1704.60	1.44	3600
RC103	100	1605.16	1624.70	1.22	3600	RC104	100	1565.35	1578.90	0.87	3600
RC105	100	1699.10	1726.20	1.59	3600	RC106	100	1626.75	1661.90	2.16	3600
RC107	100	1584.48	1612.90	1.79	3600	RC108	100	1556.22	1584.20	1.80	3600
RC201	30	4271.97	4288.00	0.38	3600	RC202	30	4262.13	4268.90	0.16	3600
RC203	30	4259.02	4270.90	0.28	3600	RC204	30	4259.09	4273.40	0.34	3600
RC205	30	4267.73	4282.70	0.35	3600	RC206	30	4267.20	4282.00	0.35	3600
RC207	30	4259.17	4276.50	0.41	3600	RC208	30	4258.20	4274.60	0.39	3600
RC201	50	2764.18	2783.70	0.71	3600	RC202	50	2738.24	2754.20	0.58	3600
RC203	50	2723.66	2739.40	0.58	3600	RC204	50	2722.09	2749.80	1.02	3600
RC205	50	2741.80	2756.40	0.53	3600	RC206	50	2741.65	2755.00	0.49	3600
RC207	50	2726.54	2744.70	0.67	3600	RC208	50	2721.57	2737.00	0.57	3600
RC201	100	1686.68	1718.60	1.89	3600	RC202	100	1615.75	1649.60	2.10	3600
RC203	100	1572.68	1589.80	1.09	3600	RC204	100	1557.56	1577.70	1.29	3600
RC205	100	1633.71	1664.70	1.90	3600	RC206	100	1624.96	1643.30	1.13	3600
RC207	100	1592.04	1601.50	0.59	3600	RC208	100	1556.38	1575.30	1.22	3600