Pseudo-Compact Formulations and Branch-and-Cut Approaches for the Capacitated Vehicle Routing Problem with Stochastic Demands

Caio Paziani Tomazella[a,*], Pedro Munari[a], Reinaldo Morabito[a]

[a] *Department of Production Engineering, Federal University of São Carlos, São Carlos-SP, Brazil*

Abstract

In this paper, we address the Capacitated Vehicle Routing Problem with Stochastic Demands (CVRPSD), in which routes are planned a priori and recourse actions are performed to ensure demand fulfillment. These recourse actions are defined through policies and may include replenishment trips or demand backlogging subject to penalties. We develop the first family of pseudo-compact MIP formulations for different recourse policies, which explicitly determine the expected recourse cost of routes. These formulations allow us to write closed models for the CVRPSD and use them straightforwardly within general-purpose MIP solvers. To the best of our knowledge, this is the first exact approach for the CVRPSD that does not rely on tailored cutting-plane or branch-and-price algorithms, making it accessible to a broader range of practitioners and researchers. Extensive computational experiments show that the proposed formulations can be used to solve small to medium-sized instances to optimality using standalone MIP solvers. We also develop simple yet effective branch-and-cut methods upon these formulations, with performance comparable to state-of-the-art methods over different benchmarks, finding proven optimal solutions for previously unsolved asymmetric instances.

Keywords: logistics; vehicle routing; stochastic programming; mixed-integer programming; branch-and-cut.

1. Introduction

Similarly to many combinatorial optimization problems, most studies on the Vehicle Routing Problem (VRP) assume that all input parameters are deterministic and known at the time of route planning. While this assumption may be reasonable in certain situations, real-world applications often involve uncertainty in parameters such as customer demands or travel times. When such uncertainty is ignored, planned routes may become infeasible during execution, leading to costly corrective actions or even service failures (Gendreau et al., 1995; Toth and Vigo, 2014).

Stochastic programming with recourse is one of the most widely used frameworks for modeling and solving stochastic variants of the VRP. In this approach, uncertainty is represented through probability distributions, defined either by finite support vectors or scenarios derived from historical data or expert judgment. To handle the effects of uncertainty, recourse policies incorporate realistic corrective actions that are applied when planned routes become infeasible due to demand realizations (Birge and Louveaux, 2011).

*Corresponding author: caio.tomazella@gmail.com

A major challenge in the context of the Capacitated VRP with Stochastic Demands (CVRPSD) lies in computing and incorporating the Expected Cost of Recourse (ECR) actions into the routing decisions. Unlike the deterministic capacitated VRP, where the objective function consists solely of routing costs and feasibility is enforced through capacity constraints, the CVRPSD requires the evaluation of the ECR for each individual route and each possible demand realization; therefore, the computation of the objective function is not as straightforward.

In this paper, we address the CVRPSD under the *a priori* paradigm (Dror et al., 1989; Bertsimas, 1992) and the underlying setting of *late information*, in which customer demands are revealed only upon arrival at each customer. This paradigm follows a two-phase approach, where routes are planned in advance using demand distributions, and recourse actions are applied during execution once demands are revealed, without altering the planned customer sequence. We consider the CVRPSD with customer demands modeled by discrete distributions with finite support.

The main contribution of this paper is the development of the first pseudo-compact MIP formulations for the CVRPSD, which explicitly compute the ECR of a solution internally in the model. These formulations are pseudo-compact in the sense that their size depends not only on the instance size (i.e., number of nodes), but also on the numerical values of instance parameters such as vehicle capacity. By explicitly modeling the ECR through novel variables and constraints, we obtain closed-form formulations of the CVRPSD that can be directly used within general-purpose MIP solvers, significantly lowering the barrier to implementation.

Within this proposal, we address the main recourse policies proposed in the literature, including policies based on out-and-back replenishment trips to the depot (Laporte et al., 2002; Jabali et al., 2014), optimized restocking trips (Yang et al., 2000; Louveaux and Salazar-González, 2018), and preventive restocking trips based on set rules (Salavati-Khoshghalb et al., 2019c). We also introduce new policies that take direct measures to ensure demand fulfillment, such as deliveries through single-customer dedicated routes and backlogging unmet demand. While several policies have been defined in the literature through recursive structures, we present a unified modeling framework that covers both existing and newly proposed policies. This framework allows all policies to be modeled consistently into the pseudo-compact MIP formulations. We highlight that these formulations are applicable to CVRPSD instances with both Euclidean and asymmetric travel cost structures.

Building upon these formulations, we develop solution methods that are both effective and accessible in terms of implementation. As discussed in Section 2, the literature on exact methods for the CVRPSD is primarily focused on the integer L-shaped (Laporte et al., 2002; Jabali et al., 2014; Hoogendoorn and Spliet, 2023) and branch-price-and-cut (BPC) methods (Christiansen and Lysgaard, 2007; Gauvin et al., 2014; Florio et al., 2020). Although state-of-the-art, these methods are inherently complex, relying on sophisticated algorithmic components such as lower bounding schemes and tailored pricing algorithms. In contrast, the approaches proposed in this paper utilize concepts that are well established in the deterministic VRP literature and can be implemented using commercial general-purpose MIP solvers.

We show that by using the proposed formulations within standalone MIP solvers, one can solve small- to medium-sized instances to optimality without any additional algorithmic en-

hancements. For larger instances, we introduce simple yet effective branch-and-cut (BC) methods derived from the proposed formulations. Extensive computational experiments on different benchmark instances show that these methods achieve a performance that is competitive with, and in some cases superior to, the state-of-the-art on both Euclidean and asymmetric CVRPSD instances. The asymmetric case, in particular, has received limited attention in the literature, with a few notable exceptions (Louveaux and Salazar-González, 2018; Florio et al., 2020).

In summary, the main contributions of this paper are:

- A family of novel pseudo-compact MIP formulations for the CVRPSD that, for the first time, explicitly incorporates the calculation of the ECR under different types of recourse policies when demands follow a discrete distribution defined by finite support vectors;
- Simpler yet effective BC methods developed from the proposed formulations, which significantly improve scalability and allow the solution of both Euclidean and asymmetric CVRPSD instances;
- A unified modeling framework for the main recourse policies studied in the CVRPSD literature (classical, optimal, and rule-based), and new policies based on backlogging and single-customer routes, all expressed through recursive equations or optimization problems;
- Computational experiments that demonstrate that the proposed methods are competitive with state-of-the-art exact approaches. In particular, we show that the proposed BC outperforms the current state-of-the-art for the asymmetric CVRPSD, finding proven optimal solutions for instances that had not been solved to optimality previously.

The remainder of this paper is organized as follows. Section 2 reviews the literature on exact methods for the CVRPSD. Section 3 formally defines the CVRPSD and the recourse policies. Section 4 presents the novel MIP formulations for these policies. Section 5 describes the exact solution methods, and Section 6 reports the computational results. Finally, Section 7 concludes the paper and provides guidelines for future research.

2. Related work

The first study to address the VRP under uncertainty was presented by Tillman (1969), who incorporated stochastic demands in a multi-terminal setting and proposed a heuristic solution method. The first mathematical formulations were introduced by Stewart and Golden (1983), who proposed both chance-constrained models and two-stage stochastic programming with recourse, in which unmet demand is handled through recourse penalties. This latter approach provided the basis for subsequent work on recourse policies in stochastic vehicle routing.

Within this framework, Dror et al. (1989) introduced a recourse policy known today as the *classical* policy, in which vehicles return to a central depot for replenishment when their load is depleted. A more advanced policy was later proposed by Yang et al. (2000) in the form of the *restocking* recourse policy, where the decision of when to perform a replenishment trip between customers is optimized a priori along the route. Because it involves an optimization step for

making a decision, this policy is better known as the *optimal* policy. Both classical and optimal policies have since become standard modeling choices in the CVRPSD literature.

Major contributions to the CVRPSD literature were made by Gendreau et al. (1995) and Laporte et al. (2002), who proposed the first exact methods based on the integer L-shaped method for the problem. These contributions are noteworthy since, until then, this problem had been solved mostly through the use of heuristics. For comprehensive reviews on stochastic variants of the VRP under uncertainty, we refer to Gendreau et al. (2014, 2016), Oyola et al. (2018, 2017) and Florio et al. (2023).

The literature on exact approaches for the CVRPSD can be divided into two classes, based on different modeling approaches: the integer L-shaped and branch-price-and-cut (BPC) methods. The integer L-shaped method is a BC-like method introduced by Laporte and Louveaux (1993) and first applied to the CVRPSD by Gendreau et al. (1995). The ECR is incorporated as a variable in the objective function and its value is determined gradually through cutting planes and lower bounding functionals, which are generated by tailored separation algorithms during the BC process. These methods are generally effective for instances with relatively loose vehicle capacity, but tend to have a weaker performance on instances with a lower customer-to-route ratio (i.e., solutions with shorter routes).

Within this framework, Laporte et al. (2002) proposed lower bounds and lower bound functionals on the expected recourse cost under the classical recourse policy, later strengthened by Jabali et al. (2014). Louveaux and Salazar-González (2018) extended the integer L-shaped method to the CVRPSD with optimal recourse, requiring demands to be represented by a finite discrete support. Their approach, based on a directed arc-flow formulation, enabled the solution of instances with asymmetric travel costs. Subsequent contributions were made by Salavati-Khoshghalb et al. (2019a), who proposed novel lower bounds for routes under the optimal policy, and by Salavati-Khoshghalb et al. (2019b) and Salavati-Khoshghalb et al. (2019c) who introduced hybrid and rule-based recourse policies, respectively, along with a tailored lower bounding procedure for them. These two policies provide rules that trigger preventive restocking trips between customers to improve service consistency and avoid failure during service (e.g., a preventive trip must be made when the vehicle residual load falls below a fraction of its capacity).

Hoogendoorn and Spliet (2023) proposed several enhancements to the integer L-shaped method, including stronger valid inequalities, improved lower bound functionals, and corrections to existing lower bounds for the optimal recourse policy. Recently, Parada et al. (2024) developed new lower bounds and optimality cuts for the CVRPSD with classical recourse under a monotonicity assumption on the ECR.

The BPC method, on the other hand, is based on extended, route-based set-partitioning formulations for the CVRPSD and relies on pricing algorithms in which the ECR is embedded in the reduced cost computation during column generation. While this approach benefits from the stronger dual bounds of the formulation and performs well for instances with low customer-to-route ratios, it often struggles with scalability on larger instances due to the combinatorial nature of the pricing algorithms.

Christiansen and Lysgaard (2007) proposed the first BPC method for the CVRPSD under the a priori paradigm, using a classical recourse policy. Their method was built upon a set-covering

formulation and relied on a dynamic programming algorithm for the pricing. Gauvin et al. (2014) improved this BPC method, applying it to a set-partitioning formulation and introducing several algorithmic enhancements: an aggregate dominance rule and bidirectional label extensions for the labeling algorithm, and a Tabu Search for the pricing subproblem. The method, however, struggled with scalability in larger instances, since it was not possible to store the problem data in memory due to the massive state-space graph. A BPC method for the CVRPSD with optimal recourse was proposed by Florio et al. (2020), with a backward labeling algorithm for the pricing subproblem, along with both exact and heuristic dominance rules. Their method was further enhanced by Florio et al. (2023) through the use of an elementary pricing strategy and stronger bounds for the pricing problem.

Two modeling assumptions are a common thread in the CVRPSD literature: capacity constraints that enforce that the mean expected demand of the customers in a route must not exceed vehicle capacity; and a fixed number of routes, equaling the number of available vehicles. While both assumptions reflect practical constraints and result in balanced plans that are more attractive to decision-makers, their enforcement has implications for solution quality and computational performance. This topic was addressed by Florio et al. (2020), via a BPC method, and Hoogendoorn and Spliet (2025), via an integer L-shaped method, with both authors noting that, by relaxing these constraints, it was possible to find cheaper solutions in certain instances, although at a significant loss of performance on their methods.

Despite the existence of these two classes of exact methods for the addressed CVRPSD, both are based on formulations that cannot be directly used in general purpose MIP solvers, and require the implementation of advanced tailored techniques. This paper fills this gap between effective exact methods and accessible approaches by proposing closed pseudo-compact formulations for the CVRPSD. In addition to being more straightforward to implement, our proposal is applicable to instances under several recourse policies and also to instances with asymmetric distances.

3. The Capacitated VRP with Stochastic Demands

Consider the set of customers $\mathcal{C} = \{1, \ldots, n\}$ and a homogeneous fleet of vehicles represented by the set $\mathcal{M} = \{1, \ldots, m\}$, with each vehicle having capacity Q. The demand of each customer $i \in \mathcal{C}$ is uncertain and modeled as a random variable ξ_i. We assume that these random variables follow a known discrete probability distribution, with the realizations being integer values no larger than Q. These random variables are independent and may follow different distributions. All vehicles are available at a single depot that is represented by indices 0 and $n+1$ for convenience. The CVRPSD is defined using a directed graph $G = (\mathcal{N}, \mathcal{A})$ with the set of nodes $\mathcal{N} = \mathcal{C} \cup \{0, n+1\}$ and the set of arcs $\mathcal{A} = \{(i,j) : i,j \in \mathcal{N}, i \neq j, i \neq n+1, j \neq 0\}$. Given a node $i \in \mathcal{N}$, let $\delta_i^- = \{(j,i) \in \mathcal{A}\}$ denote the set of its incoming arcs, and $\delta_i^+ = \{(i,j) \in \mathcal{A}\}$ the set of its outgoing arcs. For any subset of nodes $\mathcal{S} \subset \mathcal{N}$, let $\mathcal{A}(\mathcal{S}) = \{(i,j) \in \mathcal{A} \mid i,j \in \mathcal{S}\}$. A travel cost c_{ij} is associated with each arc $(i,j) \in \mathcal{A}$ and we assume that the cost matrix satisfies the triangle inequality. The problem consists of designing up to m routes that depart from node 0, visit a subset of customers and return to $n+1$, such that all customers are visited exactly once and the total expected cost of routes is minimized. Variable x_{ij} is a binary variable that

has a value of 1 if arc $(i,j) \in \mathcal{A}$ is traversed, 0 otherwise.

The computation of the total cost of a solution is a key component of the CVRPSD. This total cost consists of the usual deterministic travel costs of the planned routes plus the expected costs associated with the recourse actions. Routes are planned in such a way that the expected demand of customers on the route satisfies the vehicle capacity: given a route denoted as the sequence $(v_0 = 0, v_1, \ldots, v_t, \ldots, v_{T+1} = n+1)$ that visits $T > 0$ customers, we assume that

$$\sum_{t=1}^{T} \mathbb{E}[\xi_{v_t}] \leq Q.$$

This does not mean, of course, that this route will remain feasible after the realization of demand values. Hence, if the actual demand of the customers on a route violates the vehicle capacity, we resort to recourse actions that specify a way of dealing with such an infeasibility. These actions follow a well-defined recourse policy and incur additional costs that are added to the planned route costs, weighted by their probability of occurrence. Therefore, the route design must take into account the corresponding ECR, denoted hereafter as $\mathcal{R}(x)$.

3.1. MIP formulation

Using the defined notation, the CVRPSD can be formulated as follows:

$$\min \quad \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} + \mathcal{R}(x) \tag{1}$$

$$\text{s.t.} \quad \sum_{(i,j)\in\delta_i^+} x_{ij} = 1, \qquad i \in \mathcal{C}, \tag{2}$$

$$\sum_{(j,i)\in\delta_i^-} x_{ji} = \sum_{(i,j)\in\delta_i^+} x_{ij}, \qquad i \in \mathcal{C}, \tag{3}$$

$$\sum_{(0,j)\in\delta_0^+} x_{0j} \leq m, \tag{4}$$

$$\sum_{(i,j)\in A(\mathcal{S})} x_{ij} \leq |\mathcal{S}| - \left\lceil \frac{1}{Q} \sum_{i\in\mathcal{S}} \mathbb{E}[\xi_i] \right\rceil, \quad \mathcal{S} \subset \mathcal{N}, 2 \leq |\mathcal{S}| \leq n, \tag{5}$$

$$x_{ij} \in \{0,1\}, \qquad (i,j) \in \mathcal{A}. \tag{6}$$

The objective function (1) consists of minimizing the total expected costs of the routes. The vehicle flow constraints (2) and (3) guarantee that the planned routes visit each customer exactly once and that a vehicle departs from each visited customer. Constraint (4) imposes that the number of routes cannot exceed the total number of vehicles. Constraints (5) are the so-called Expected Capacity Inequalities (ECI), which enforce the condition that the expected demand of a route does not exceed the vehicle capacity, and also eliminate subtours. The number of ECI is exponential in terms of the number of customers, and it is well-known that they can be replaced by sets of constraints that are polynomial in n, such as commodity-flow constraints or the following MTZ-based constraints (Miller et al., 1960):

$$y_j \leq y_i - \mathbb{E}[\xi_i]x_{ij} + Q(1 - x_{ij}), \quad (i,j) \in \mathcal{A} : i \in \mathcal{C}, \tag{7}$$

$$0 \leq y_i \leq Q, \qquad i \in \mathcal{N}. \tag{8}$$

Even with these polynomial-sized constraints, the formulation does not become compact or pseudo-compact, as there is no closed-form for $\mathcal{R}(x)$ that is polynomial in terms of the number of nodes. Finally, constraints (6) impose the domain of the decision variables. The routes in a feasible solution of model (1)-(6) are called *first-stage* routes and determine the sequence of customer visits, which cannot be modified by recourse actions under the a priori paradigm.

3.2. Recourse policies

The definition of $\mathcal{R}(x)$ depends on the recourse policy adopted in the execution of the first-stage routes. Since we build upon the a priori paradigm, the sequence of visited customers in the first-stage routes cannot be altered, thus the recourse policies involve corrective actions such as replenishment trips and outsourcing to ensure demand fulfillment.

In the remainder of this section, we will define these recourse policies, their corrective actions and present equations to calculate their respective ECR for a first-stage route. We introduce recourse policies based on outsourcing and backlogging of unfulfilled demand, and also present the policies from the literature, providing a comprehensive framework on recourse policies for the CVRPSD. We assume that customer demands follow discrete probability distributions with finite support, which allows the ECR to be computed exactly. A finite support vector $\mathcal{K}$ consists of K values for each customer i, denoted by $\xi_i^1 < \xi_i^2 < \cdots < \xi_i^K$ with respective probabilities $p_i^1, \ldots, p_i^K$.

Given a first-stage route represented as the sequence $(v_0 = 0, v_1, \ldots, v_t, \ldots, v_{T+1} = n+1)$ defined before, let $f(t, q)$ be the ECR after the vehicle finishes servicing the t-th node of this route with a residual load q, for $t \in \{0, \ldots, T\},\ q \in \{0, \ldots, Q\}$. The ECR of the entire route is given by $f(0, Q)$, as the vehicle begins the route at the depot with a full load, whereas $f(T, q) = 0,\ \forall\ q \in \{0, \ldots, Q\}$ serves as a boundary condition, as no recourse actions are taken after the last customer is serviced.

3.2.1. P1: Classical policy

We start with the *classical* recourse policy, introduced by Dror et al. (1989), which involves out-and-back trips to the depot in case of a *failure*, a term coined for the situation in which the vehicle arrives at a customer with a load strictly smaller than demand realization. When a failure occurs, the vehicle delivers its full residual load, returns to the depot for replenishment, and then resumes the route on the same customer to deliver the remaining quantity. Because we are assuming discrete demand realizations, we can find a case in which the customer's actual demand is exactly equal to the vehicle residual load, thus becoming empty after servicing it. For this particular case, a replenishment trip to the depot is made instead of following the route directly to the next customer. The ECR of the classical policy, which we also refer to as P1, is

calculated via the following recursion, for all $t \in \{0, \ldots, T-1\}$ and $q \in \{0, \ldots, Q\}$.

$$f(t,q) = \begin{cases} \sum\limits_{k \in \mathcal{K}} p^k_{v_{t+1}} f(t+1, Q - \xi^k_{v_{t+1}}) + c_{v_t(n+1)} + c_{0v_{t+1}} - c_{v_t v_{t+1}}, & q = 0, \\ \sum\limits_{\substack{k \in \mathcal{K} \\ \xi^k_{v_{t+1}} > q}} p^k_{v_{t+1}} \Big(f(t+1, Q - \xi^k_{v_{t+1}} + q) + c_{v_{t+1}(n+1)} + c_{0v_{t+1}} \Big) & \\ \qquad + \sum\limits_{\substack{k \in \mathcal{K} \\ \xi^k_{v_{t+1}} \leq q}} p^k_{v_{t+1}} f(t+1, q - \xi^k_{v_{t+1}}), & q > 0. \end{cases} \tag{9}$$

3.2.2. P2: Rule-based policy

The second policy (P2) establishes rules to trigger replenishment trips between customers whenever the residual load of the vehicle falls below a certain level. More precisely, we define a threshold parameter θ_{v_t} associated with customer v_t. If, after finishing servicing customer v_{t-1}, the vehicle has a load of $q < \theta_{v_{t-1}}$, the vehicle performs a replenishment trip instead of proceeding directly to v_t. The values of θ_{v_t} are predefined and can be determined by a number of rules, such as a fixed fraction of the vehicle capacity Q or the expected cumulative demand of upcoming customers on the route (Salavati-Khoshghalb et al., 2019c). The recursion for the ECR of a route under the rule-based policy P2 is

$$f(t,q) = \begin{cases} \sum\limits_{k \in \mathcal{K}} p^k_{v_{t+1}} f(t+1, Q - \xi^k_{v_{t+1}}) + c_{v_t(n+1)} + c_{0v_{t+1}} - c_{v_t v_{t+1}}, & q < \theta_{v_{t+1}}, \\ \sum\limits_{\substack{k \in \mathcal{K} \\ \xi^k_{v_{t+1}} > q}} p^k_{v_{t+1}} \Big(f(t+1, Q - \xi^k_{v_{t+1}} + q) + c_{v_{t+1}(n+1)} + c_{0v_{t+1}} \Big) & \\ \qquad + \sum\limits_{\substack{k \in \mathcal{K} \\ \xi^k_{v_{t+1}} \leq q}} p^k_{v_{t+1}} f(t+1, q - \xi^k_{v_{t+1}}), & q \geq \theta_{v_{t+1}}, \end{cases} \tag{10}$$

for all $t \in \{0, \ldots, T-1\}$ and $q \in \{0, \ldots, Q\}$. Notice that (10) is a generalization of (9) in the sense that, by setting $\theta_j = 1$ for all $j \in \mathcal{C}$, we obtain the classical recourse policy.

3.2.3. P3: Optimal restocking policy

A recourse policy based on determining when it is optimal to make a preventive restocking trip was proposed by Yang et al. (2000) and revisited by Louveaux and Salazar-González (2018), Salavati-Khoshghalb et al. (2019a) and Hoogendoorn and Spliet (2023). Referred to in the literature as *optimal restocking* or *preventive*, we hereafter denote it as P3. In this policy, a vehicle may perform a restocking trip to the depot after servicing any customer, resuming the route with full capacity and minimizing the risk of failure. These preventive trips bring more flexibility to the recourse policy, as the vehicle can return to the depot at a more appropriate point of the route (e.g., from a customer closer to the depot). On the other hand, it increases the complexity by introducing a decision in the recourse. Indeed, after servicing a given customer, one needs to decide whether the vehicle should return to the depot or continue directly to the next customer. This optimal decision is made through a dynamic programming problem, represented

by the following recursion, for all $t \in \{0, \ldots, T-1\}$ and $q \in \{0, \ldots, Q\}$.

$$f(t,q) = \min \begin{cases} \sum_{k \in \mathcal{K}} p^k_{v_{t+1}} f(t+1, Q - \xi^k_{v_{t+1}}) + c_{v_t(n+1)} + c_{0v_{t+1}} - c_{v_t v_{t+1}}, \\ \sum_{\substack{k \in \mathcal{K} \\ \xi^k_{v_{t+1}} > q}} p^k_{v_{t+1}} \Big(f(t+1, Q - \xi^k_{v_{t+1}} + q) + c_{v_{t+1}(n+1)} + c_{0v_{t+1}} \Big) \\ \qquad + \sum_{\substack{k \in \mathcal{K} \\ \xi^k_{v_{t+1}} \leq q}} p^k_{v_{t+1}} f(t+1, q - \xi^k_{v_{t+1}}). \end{cases} \tag{11}$$

The first term corresponds to the choice of a preventive trip between v_t and v_{t+1}, while the second corresponds to the choice of resuming the a priori route, with the possibility of an out-and-back trip to the depot in case of failure at v_{t+1}. Notice that solving the dynamic programming problem (11) will provide the optimal load thresholds (θ_{v_t}) for triggering restocking trips in planned routes.

3.2.4. P4 and P5: Penalty-based policies

We now introduce two new policies (P4 and P5) that, instead of relying on restocking trips, penalizes the unfulfilled demand of a route via the ECR, based on the idea discussed by Stewart and Golden (1983). Under these policies, a vehicle follows the route delivering the actual demand of each customer until it becomes empty, when it returns to the depot definitively. The residual demand of this last visited customer, plus the demand of the remaining customers on the route is then penalized in the recourse cost function. There are several choices of penalties (e.g., outsourcing, backlogging demand to the following planning horizon, single-customer trips with dedicated vehicles), which influence how their respective costs are incorporated into the ECR calculation.

In policy P4, the penalties are a function of how many units are not delivered by a route, reflecting a case with outsourcing, backlog penalties or revenue loss. We apply a penalty π_{v_t} (measured in the same cost units as the travel costs c_{ij}) to each non-delivered unit of customer v_t. The ECR of a route under P4 is computed using:

$$f(t,q) = \begin{cases} f(t+1,0) + \pi_{v_{t+1}} \sum_{k \in \mathcal{K}} p^k_{v_{t+1}} \xi^k_{v_{t+1}} - c_{v_{t+1}v_{t+2}}, & q = 0, \\ \sum_{\substack{k \in \mathcal{K} \\ \xi^k_{v_{t+1}} \geq q}} p^k_{v_{t+1}} \Big(f(t+1,0) + \pi_{v_{t+1}}(\xi^k_{v_{t+1}} - q) + c_{v_{t+1}(n+1)} - c_{v_{t+1}v_{t+2}} \Big) \\ \qquad + \sum_{\substack{k \in \mathcal{K} \\ \xi^k_{v_{t+1}} < q}} p^k_{v_{t+1}} f(t+1, q - \xi^k_{v_{t+1}}), & q > 0, \end{cases} \tag{12}$$

for all $t \in \{0, \ldots, T-1\}$ and $q \in \{0, \ldots, Q\}$. The recursion in (12) is divided into three cases that can occur after servicing v_t: (I) the vehicle has already returned to the depot after being emptied ($q = 0$), thus the demand of v_{t+1} is entirely penalized; (II) failure occurs at v_{t+1} or the vehicle becomes empty after servicing it ($\xi^k_{v_{t+1}} > q$), so it returns to the depot and any residual demand is penalized; (III) no failure occurs. In the first two cases the non-traversed arcs are subtracted from the cost function.

In P5, penalties are a function of the customers whose demands were not fully satisfied, representing a case in which additional routes are performed to fulfill their residual demand, regardless of its value. These routes are assumed to contain a single customer and to be performed with a dedicated vehicle, incurring larger costs than the first-stage routes. Thus, a single-customer route to v_t has a cost of $\hat{\pi}_{v_t}(c_{0v_t} + c_{v_t(n+1)})$, which is the traveled distance multiplied by the penalty factor $\hat{\pi}_{v_t}$ (unlike π, $\hat{\pi}$ has no unit as it is a multiplier to the travel costs). The ECR is calculated using a recursion similar to (12), with the corresponding penalty terms being substituted by $\hat{\pi}_{v_t}(c_{0v_t} + c_{v_t(n+1)})$, except for when $\xi^k_{v_{t+1}} = q$, and the demand of v_{t+1} is completely satisfied, so a dedicated route is not needed. Thus, the recursion for computing the ECR of a route under P5 is:

$$f(t,q) = \begin{cases} f(t+1,0) + \hat{\pi}_{v_{t+1}}(c_{0v_{t+1}} + c_{v_{t+1}(n+1)}) - c_{v_{t+1}v_{t+2}}, & q = 0, \\ \displaystyle\sum_{\substack{k\in\mathcal{K} \\ \xi^k_{v_{t+1}} > q}} p^k_{v_{t+1}} \Big(f(t+1,0) + \hat{\pi}_{v_{t+1}}(c_{0v_{t+1}} + c_{v_{t+1}(n+1)}) + c_{v_{t+1}(n+1)} - c_{v_{t+1}v_{t+2}} \Big) \\ \quad + \displaystyle\sum_{\substack{k\in\mathcal{K} \\ \xi^k_{v_{t+1}} = q}} p^k_{v_{t+1}} \Big(f(t+1,0) + c_{v_{t+1}(n+1)} - c_{v_{t+1}v_{t+2}} \Big) \\ \quad + \displaystyle\sum_{\substack{k\in\mathcal{K} \\ \xi^k_{v_{t+1}} < q}} p^k_{v_{t+1}} f(t+1, q - \xi^k_{v_{t+1}}), & q > 0, \end{cases} \tag{13}$$

for all $t \in \{0,\dots,T-1\}$ and $q \in \{0,\dots,Q\}$.

It is worth mentioning that although very simple, these recourse policies promote a robust behavior in the first-stage routes, in the sense that routes will be designed to absorb the variations of the actual demands with respect to the expected values, leaving a safety load in the vehicles. This is a typical way of dealing with uncertainty in practice, empirically used by companies to protect their routes against uncertainties, and is also relevant in cases in which the use of restocking trips is not viable due to timing constraints.

4. Pseudo-compact MIP formulations for the CVRPSD

In this section, we propose pseudo-compact formulations for the CVRPSD that explicitly model the recourse actions presented in Section 3. To obtain them, we convert the recursive equations defined for these policies into linear constraints, resulting in the hereafter named Explicit ECR (EECR) inequalities, which allow us to embed the ECR calculation in MIP formulations of the CVRPSD.

Consider the same notation and decision variables defined for the MIP model (1)-(6). In addition to the arc-based variables x_{ij}, we introduce two other types of variables as follows. Let Θ_j be a continuous variable that represents the ECR of a route that starts by visiting customer j first; and f_{iq} be the continuous variable that represents the ECR of a route after the vehicle departs from customer $i \in \mathcal{C}$ with a residual load q. For all purposes, we set $q \in \{0,1,\dots,Q\}$.

4.1. P1: Classical policy

For the classical recourse policy P1, we can use the following EECR inequalities as linear counterparts of Equation (9):

$$f_{i0} \geq \sum_{k\in\mathcal{K}} p_j^k \left(f_{j(Q-\xi_j^k)} + c_{i(n+1)} + c_{0j} - c_{ij}\right) - M(1-x_{ij}), \quad (i,j) \in \mathcal{A}(\mathcal{C}), \tag{14}$$

$$f_{iq} \geq \sum_{\substack{k\in\mathcal{K} \\ \xi_j^k > q}} p_j^k \left(f_{j(Q-\xi_j^k+q)} + c_{j(n+1)} + c_{0j}\right) + \sum_{\substack{k\in\mathcal{K} \\ \xi_j^k \leq q}} p_j^k f_{j(q-\xi_j^k)} - M(1-x_{ij}), \quad (i,j) \in \mathcal{A}(\mathcal{C}),\ q>0, \tag{15}$$

$$\Theta_j \geq \sum_{k\in\mathcal{K}} p_j^k f_{j(Q-\xi_j^k)} - M(1-x_{0j}), \quad j \in \mathcal{C}. \tag{16}$$

Constraints (14) and (15) guarantee that f_{iq} is determined from the subsequent customer (j, when $x_{ij} = 1$), and represent, respectively, the cases $q = 0$ and $q > 0$ from Equation (9). Constraints (16) define the total expected cost of a route, considering only nodes that are the first on a route.

4.2. P2: Rule-based policy

Constraints (14) and (15) can be easily modified to model recourse policy P2, yielding constraints (17) and (18).

$$f_{iq} \geq \sum_{k\in\mathcal{K}} p_j^k \left(f_{j(Q-\xi_j^k)} + c_{i(n+1)} + c_{0j} - c_{ij}\right) - M(1-x_{ij}), \quad (i,j) \in \mathcal{A}(\mathcal{C}), q < \theta_j, \tag{17}$$

$$f_{iq} \geq \sum_{\substack{k\in\mathcal{K} \\ \xi_j^k > q}} p_j^k \left(f_{j(Q-\xi_j^k+q)} + c_{j(n+1)} + c_{0j}\right) + \sum_{\substack{k\in\mathcal{K} \\ \xi_j^k \leq q}} p_j^k f_{j(q-\xi_j^k)} - M(1-x_{ij}), \quad (i,j) \in \mathcal{A}(\mathcal{C}),\ q \geq \theta_j. \tag{18}$$

This formulation is applicable to cases in which the definition of θ_j does not depend on the entire route, such as *Rule 1* from Salavati-Khoshghalb et al. (2019c) (based on a fraction of the vehicle's capacity).

4.3. P3: Optimal restocking policy

To obtain linear constraints that are counterparts of the dynamic programming problem (11), we introduce a variable to denote the decision of performing a preventive restocking trip. Let z_{iq} be a binary variable that assumes the value of 1 if and only if a preventive trip is performed when the vehicle departs from customer i with a residual load q. Then, the value of f_{iq} must satisfy the following linear constraints:

$$f_{iq} \geq \sum_{k\in\mathcal{K}} p_j^k (f_{j(Q-\xi_j^k)}) + c_{i(n+1)} + c_{0j} - c_{ij} - M(2-x_{ij}-z_{iq}), \quad (i,j) \in \mathcal{A}(\mathcal{C}),\ q, \tag{19}$$

$$f_{iq} \geq \sum_{\substack{k\in\mathcal{K}:\\ \xi_j^k>q}} p_j^k (f_{j(Q-\xi_j^k+q)} + c_{j(n+1)} + c_{0j}) + \sum_{\substack{k\in\mathcal{K}:\\ \xi_j^k\leq q}} p_j^k f_{j(q-\xi_j^k)} - M(1 - x_{ij} + z_{iq}), \quad (i,j)\in\mathcal{A}(\mathcal{C}),\ q, \tag{20}$$

$$z_{iq} \in \{0,1\}, \quad (i,j)\in\mathcal{A}(\mathcal{C}),\ q. \tag{21}$$

Constraints (19) define f_{iq} when a preventive trip occurs right after i, while constraints (20) define it when the vehicle proceeds directly to the next customers (j, when $x_{ij}=1$). Constraints (21) define the domain for z_{iq}.

4.4. P4 and P5: Penalty-based policies

Policy P4, in which the vehicle returns to the depot after becoming empty, is modeled using Constraints (22) and (23). Constraints (22) model the case when $q = 0$: the recourse cost associated with customer j is the expected penalty of not fulfilling its entire demand, minus the cost of traversing the arc between j and its following node on the route, l. Constraints (23) account for the other cases when $q > 0$: the first summation represents the case in which the vehicle returns to the depot incurring penalties for unfulfilled demand and the difference of traversed arcs, while the second summation represents the case in which no failure occurs.

$$f_{i0} \geq f_{j0} + \pi_j \sum_{k\in\mathcal{K}} p_j^k \xi_j^k - \sum_{(j,l)\in\delta_j^+} c_{jl}x_{jl} - M(1 - x_{ij}), \quad (i,j)\in\mathcal{A}(\mathcal{C}), \tag{22}$$

$$f_{iq} \geq \sum_{\substack{k\in\mathcal{K}:\\ \xi_j^k\geq q}} p_j^k \Big(f_{j0} + \pi_j(\xi_j^k - q) + c_{j(n+1)} - \sum_{(j,l)\in\delta_j^+} c_{jl}x_{jl}\Big) + \sum_{\substack{k\in\mathcal{K}:\\ \xi_j^k< q}} p_j^k f_{j(q-\xi_j^k)} - M(1 - x_{ij}), \quad (i,j)\in\mathcal{A}(\mathcal{C}),\ q>0. \tag{23}$$

Alternatively, we can model the recourse costs using inequalities that represent a pair of subsequently traveled arcs, (i,j) and (j,l), explicitly defining which arcs are subtracted and eliminating the summation term over variable x_{jl} in constraints (22) and (23). While the use of this formulation is preferable in some of the proposed approaches in Section 5, it can quickly become intractable due to its size being proportional to $|\mathcal{A}|^2$.

$$f_{i0} \geq f_{j0} + \pi_j \sum_{k\in\mathcal{K}} p_j^k \xi_j^k - c_{jl} - M(2 - x_{ij} - x_{jl}), \quad (i,j)\in\mathcal{A}(\mathcal{C}),\ (j,l)\in\delta_j^+, \tag{24}$$

$$f_{iq} \geq \sum_{\substack{k\in\mathcal{K}:\\ \xi_j^k\geq q}} p_j^k \Big(f_{j0} + \pi_j(\xi_j^k - q) + c_{j(n+1)} - c_{jl}\Big) + \sum_{\substack{k\in\mathcal{K}:\\ \xi_j^k< q}} p_j^k f_{j(q-\xi_j^k)} - M(2 - x_{ij} - x_{jl}), \quad (i,j)\in\mathcal{A}(\mathcal{C}),\ (j,l)\in\delta_j^+,\ q>0. \tag{25}$$

For P5, in which unfulfilled demand is served by single-customer routes, the penalty terms in constraints (22) and (23) are replaced by the costs of these routes, which are constant for any

quantity of unfulfilled demand greater than zero. In the second set of constraints, when $\xi_j^k = q$ the demand of j is completely fulfilled, and no single-customer route is required. Therefore, P5 is defined by constraints (22) with the updated penalty term and

$$\begin{aligned} f_{iq} \geq & \sum_{\substack{k\in\mathcal{K}:\\ \xi_j^k>q}} p_j^k \Big(f_{j0} + \hat{\pi}_j(c_{0j} + c_{j(n+1)}) + c_{j(n+1)} - \sum_{(j,l)\in\delta_j^+} c_{jl}x_{jl} \Big) \\ & + \sum_{\substack{k\in\mathcal{K}:\\ \xi_j^k=q}} p_j^k \Big(f_{j0} + c_{j(n+1)} - \sum_{(j,l)\in\delta_j^+} c_{jl}x_{jl} \Big) \\ & + \sum_{\substack{k\in\mathcal{K}:\\ \xi_j^k<q}} p_j^k f_{j(q-\xi_j^k)} - M(1 - x_{ij}), \qquad (i,j) \in \mathcal{A}(\mathcal{C}),\ q > 0. \end{aligned} \tag{26}$$

5. Solution approaches for the CVRPSD

By using the EECR inequalities proposed in the previous section, we can develop MIP formulations for the CVRPSD that endogenously determine the ECR of a solution, and thus obtain solutions for the CVRPSD directly with general-purpose MIP solvers. The resulting formulation for the CVRPSD has the following structure, in which the Θ_j variables replace term $\mathcal{R}(x)$ as the ECR in objective function (27), and the EECR inequalities are included according to the chosen policies.

$$\min \quad \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} + \sum_{j\in\mathcal{C}} \Theta_j \tag{27}$$

$$\text{s.t.} \quad (2)-(4), (6)-(8), (16)$$

$$\text{EECR inequalities}$$

$$f_{jq} \geq 0, \qquad j \in \mathcal{C},\ q, \tag{28}$$

$$\Theta_j \geq 0, \qquad j \in \mathcal{C}. \tag{29}$$

Table 1 provides an overview of the proposed formulations and their corresponding EECR inequalities. Constraints (16), used to determine the value of Θ_j, and (28) and (29), used to define the domain of the EECR-related variables, are common to all formulations.

Due to the pseudo-compact size of the EECR inequalities sets, e.g. (15), the models can become intractable in cases with large vehicle capacity (Q). This fact, in addition to the weak lower bounds provided by the MTZ-based constraints, requires us to use more effective methods for handling larger instances. To address this issue, we make use of different approaches based on the BC method. For the remainder of this section, we refer to the MIP formulation for the

Table 1: Overview of the CVRPSD formulations using the novel EECR.

Policy	Description	EECR inequalities	Observation
$P1$	Classical	(14), (15)	
$P2$	Rule-based	(17), (18)	
$P3$	Optimal restocking	(19) to (21)	
$P4$	Backlogging	(22), (23)	Alternatively, (24), (25)
$P5$	Single-route	(22), (26)	Modified penalty in (22)

CVRPSD with classical recourse, $P1$. Unless explicitly stated, these approaches can be adapted to other policies with minimal modifications.

5.1. Approach 1: General-purpose MIP solver

The first approach consists of directly solving instances using the proposed pseudo-compact formulation within a general-purpose MIP solver. This approach offers value in its simplicity and ease of implementation. To the best of our knowledge, this is the first approach that one can use to solve instances of the addressed CVRPSD without requiring advanced algorithms based on callback functions.

5.2. Approach 2: BC method with capacity cuts

The second approach is a BC method that improves upon Approach 1 by replacing the MTZ-based constraints (7) and (8) with the ECI in (5), and makes use of a classical and well-established separation algorithm to gain efficiency. The BC method begins with set (5) relaxed, and violated ECI are added through callbacks. To identify violated ECI in both integer and fractional solutions, we use the separation algorithm from the CVRPSEP package (Lysgaard, 2003). Although this approach requires the use of callbacks, it makes use of well-established algorithms (such as the aforementioned package) for the deterministic capacitated VRP, which are available for a direct integration with the user's framework. The single additional requirement for the user is the inclusion of the EECR inequalities corresponding to the chosen recourse policy.

5.3. Approach 3: BC method with EECR inequalities

Enumerating all the EECR inequalities can place a significant computational burden on the solver, making Approaches 1 and 2 inefficient or even impractical for large-scale instances (i.e., instances with large n and/or Q). Hence, the third approach aims at improving the performance of the BC method by relaxing constraints (14) and (15) and applying a tailored separation algorithm that identifies and adds only the violated constraints through callback functions. This algorithm is called when the solver finds a solution that does not violate the integrality of x_{ij} (6) nor the ECI (5).

Using a similar definition to the one in Section 3, let $v = \{v_0 = 0, v_1, \ldots, v_t, \ldots, v_{T+1} = n+1\}$ be a first-stage route found in the BC tree. The algorithm starts by using the recursive equations to compute the ECR of the routes. Let $\overline{\Theta}_{v_1}$ represent the ECR of v. If v has already been discovered, the value of Θ_{v_1} will be equal to $\overline{\Theta}_{v_1}$. Otherwise, we will have $\overline{\Theta}_{v_1} > \Theta_{v_1}$, and the ECR constraints for arcs (i, j) traversed in v must be added to the model.

Since demands are discrete and limited to a finite number of realizations (K), the possible load values a vehicle can carry when traversing arc (i, j) on a given route are restricted. Therefore, it is unnecessary to add ECR constraints for all $q = 0, \ldots, Q$ values for that particular arc. Starting from the first customer (v_1), we compute the K possible load values a vehicle can carry when departing from it, denoted as $Q_{v_1} = \{Q - \xi^1_{v_1}, \ldots, Q - \xi^K_{v_1}\}$. EECR inequalities (14) and (15) are then added for arc (v_1, v_2) and for each $q \in Q_{v_1}$. Next, we obtain Q_{v_2} based on the elements of Q_{v_1} and the demand support vector of v_2. This procedure, detailed in Algorithm 1, is repeated until cuts for arc (v_{T-1}, v_T) are added.

Algorithm 1: Separation of EECR inequalities for a route under policy P1.

```
Create list Q_{v_1} = ∅;
for k ← 1 to K do
    Q_{v_1} ← Q_{v_1} ∪ {Q − ξ^k_{v_1}} ;
end
for i ← 1 to T − 1 do
    for q' ∈ Q_{v_i} do
        if q' = 0 then
            Add cut (14) for (i, j) = (v_i, v_{i+1}) and q = q' to the MIP model;
        else
            Add cut (15) for (i, j) = (v_i, v_{i+1}) and q = q' to the MIP model;
        end
    end
    Create list Q_{v_{i+1}} = ∅;
    for q' ∈ Q_{v_i} do
        for k ← 1 to K do
            if q' = 0 then
                Q_{v_{i+1}} ← Q_{v_{i+1}} ∪ {Q − ξ^k_{v_{i+1}}};
            else
                if ξ^k_{v_{i+1}} > q' then
                    Q_{v_{i+1}} ← Q_{v_{i+1}} ∪ {Q − ξ^k_{v_{i+1}} + q'};
                else
                    Q_{v_{i+1}} ← Q_{v_{i+1}} ∪ {q' − ξ^k_{v_{i+1}}};
                end
            end
        end
    end
end
```

We also use optimality cuts that exclude solutions proven to be non-optimal. These cuts are added after we find a new feasible solution and compute its ECR. If the expected costs exceed the incumbent's cost, $Z(\overline{x})$, Algorithm 1 is skipped and instead, cut (30) is added. This procedure helps to reduce the size of the model by avoiding the inclusion of ECR constraints for non-optimal solutions. Let $A(\overline{x})$ be the set of arcs that represent solution $\overline{x}$. The combinatorial (no-good) cut used to prune this solution is

$$\sum_{(i,j)\in A(\overline{x})} x_{ij} \leq |A(\overline{x})| - 1. \tag{30}$$

5.4. Approach 4: BC method with optimality cuts

This last approach is a BC algorithm that uses an optimality cut to pass the value of a route to its respective Θ_j variable. This algorithm starts with the relaxed model (2)-(6), (27)-(29). Similarly to Approaches 2 and 3, the ECI in (5) are added through the CVRPSEP separation algorithm. Whenever a feasible solution is found, we call the same algorithm used for Approach 3 to identify the routes and calculate their respective ECRs. Then, instead of adding the EECR inequalities, we add a single optimality cut, called hereafter Theta cut.

Let $A(\overline{x})$ be the set of arcs that define one route solution $\overline{x}$, $\overline{\Theta}(\overline{x})$ the calculated ECR of this route, and j its customer with the lowest index. Then, its respective Theta cut (31) guarantees that variable Θ_j has the value of $\overline{\Theta}(\overline{x})$ when this route is part of a feasible solution.

$$\Theta_j \geq \overline{\Theta}(\overline{x})\left(\sum_{(i,j)\in A(\overline{x})} x_{ij} - |A(\overline{x})| + 1\right). \tag{31}$$

Unlike the previous approaches, this BC method does not rely on explicitly modeling the

ECR through the use of inequalities. However, it still requires the implementation of routines to compute the ECR for each route using the calculations presented in Section 3 and 4. Similar cuts are applied in the form of lower bounding functionals (LBFs) by Hoogendoorn and Spliet (2023), under the name of *Route-Split Inequalities*.

One major advantage of this approach over the previous is that, since it does not require the EECR inequalities, it can be used with a smaller CVRPSD formulation to solve instances with Euclidean distances ($c_{ij} = c_{ji}$). In such case, this formulation assumes that $G = (\mathcal{N}, \mathcal{A})$ is a non-directed graph, and variables x_{ij} indicate whether the vehicle traverses the arc between nodes i and j in any direction, with $i < j$. This model is used in most of the CVRPSD approaches, and the reader is referred to Laporte et al. (2002) and Jabali et al. (2014) for a complete description.

6. Computational experiments

In this section, we present the results of computational experiments performed to validate and evaluate the performance of the proposed formulations and BC methods. We first discuss the results on small to medium-sized instances (20 to 40 nodes) to assess the performance of Approaches 1 and 2. Following that, we analyze the scalability of Approaches 2 to 4 on larger instances (up to 60 nodes), and compare the results of the best performing methods with the state-of-the-art.

All proposed models and methods were implemented in C++, and solved using Gurobi version 11.0.3, with a time limit of 3600 seconds. The experiments were performed on a cluster containing 2 Intel Xeon E5-2680v2 processors each, 10 cores, 2.8GHz and 128GB DDR3 1866MHz RAM.

For the main experimentation, we use the instance dataset from Salavati-Khoshghalb et al. (2019a). These instances were proposed for the CVRPSD with optimal restocking recourse (P3) and later used in Salavati-Khoshghalb et al. (2019c) for the CVRPSD with rule-based recourse (P2). This dataset contains instances with $n \in \{20, 30\}$ nodes and $m = 2$ vehicles, and $n \in \{40, 50, 60\}$ nodes and $m \in \{2, 3, 4\}$ vehicles, and four different fill rate values ($fr \in \{0.90, 0.92, 0.94, 0.96\}$). The fill rates denote the capacity usage of the vehicle in relation to the average expected demand of a route. Thus, the higher their value, the tighter the vehicle capacity. In all instances, there are $K = 5$ possible demand realizations for each customer, and the distances between nodes are symmetric. It is also worth noting that the n nodes include the depot, thus there are $n - 1$ customers in each instance.

Under P2, we use $\theta_j = \max_{k \in \mathcal{K}}\{\xi_j^k\}$ for all $j \in \mathcal{C}$ (the maximum value the demand of j can assume). This strategy, although conservative, ensures that no failure occurs when visiting a customer. Under P4, the backlogging penalty is fixed at $\pi_j = 50$ per unit of unfulfilled demand, while under P5, the distance of a single-customer route is weighted by a factor of $\hat{\pi}_j = 1$.

We set different values for the parameter M used in the EECR inequalities, based on the recourse policy. Under P1, P2 and P3, we calculate the number of full vehicles needed to fulfill the maximum demand ($m' = \lceil \sum_j \max_{k \in \mathcal{K}} \xi_j^k / Q \rceil$) and set M equal to the sum of the m' most expensive out-and-back trips to the depot. Under P4, we set M as the total maximum demand times the penalty factor, and under P5 as the cost of single routes for all the customers.

Table 2 summarizes the methods proposed in this paper and their main features. MIP corresponds to solving instances with the standalone MIP solver. Preliminary tests showed that

the general-purpose solver (Gurobi) can require a considerable amount of time to find feasible solutions for certain instances. Based on this behavior, we introduce MIP^H, in which we apply the HGS metaheuristic (Vidal et al., 2012; Vidal, 2022) using average customer demands and a time limit of 2 seconds to generate a first-stage solution that is used as a MIP warm start. In BC_{CAP}, we use the CVRPSEP separation algorithm to add the ECI inequalities (5). BC^L_{CAP} is a variant of this method in which we implement the EECR inequalities as *lazy constraints*. In Gurobi, this implementation consists of modifying the *Lazy* attribute of a constraint to 3, so that it is kept inactive and is automatically added by the solver to the model when violated. We remind that Gurobi, as well as other solvers, also use the term *Lazy* for constraints added via callbacks, which is not the case in this particular method. Finally, BC_{EECR} and BC_Θ correspond to the BC methods described in Approaches 3 and 4, respectively. Preliminary tests showed that warm starts did not improve these methods, and were therefore not used.

Table 2: Overview of the proposed exact methods.

Approach	Method	Warm start	CVRPSEP	ECR computation
1	MIP			EECR inequalities
1	MIP^H	✓		EECR inequalities
2	BC_{CAP}	✓	✓	EECR inequalities
2	BC^L_{CAP}	✓	✓	EECR inequalities as *LazyConstraints*
3	BC_{EECR}		✓	EECR inequalities via separation algorithm; no-good cuts
4	BC_Θ		✓	Optimality cuts

6.1. Results for small instances

We begin by evaluating the overall performance of MIP, MIP^H, BC_{CAP} and BC^L_{CAP} on instances with up to 40 nodes. The charts in Figure 1 summarize the results obtained with policies P1 to P5. Figures 1a and 1b show, respectively, the number of instances solved to optimality and the average computation time, in seconds, for each class of instances. Figure 1c shows the average optimality gap, as reported by Gurobi, while Figure 1d indicates the average relative gap with respect to the Best Known Solution (BKS), which we also refer to as Gap_{BKS}. The BKS corresponds to the best upper bound found across all tested methods, including BC_{EECR} and BC_Θ, reported in the next subsection. In Salavati-Khoshghalb et al. (2019a), these instances were solved under P3 only, and the results for individual instances were not reported, thus we use only our solutions as the BKS for this part of the experimentation. Table 3 complements the graphs by providing the average Gap_{BKS} of these four methods separated by number of nodes and vehicles. The detailed results are found in Tables A1 and A2 of the supplementary material.

Table 3: Average Gap_{BKS} of MIP, MIP^H, BC_{CAP} and BC^L_{CAP} for instances with up to 40 nodes.

n	m	MIP	MIP^H	BC_{CAP}	BC^L_{CAP}
20	2	0.34%	0.19%	0.01%	0.03%
30	2	6.57%	1.27%	0.19%	0.27%
40	2	18.64%	1.45%	0.57%	0.40%
	3	19.10%	3.04%	0.84%	0.94%
	4	16.53%	5.09%	1.61%	2.72%

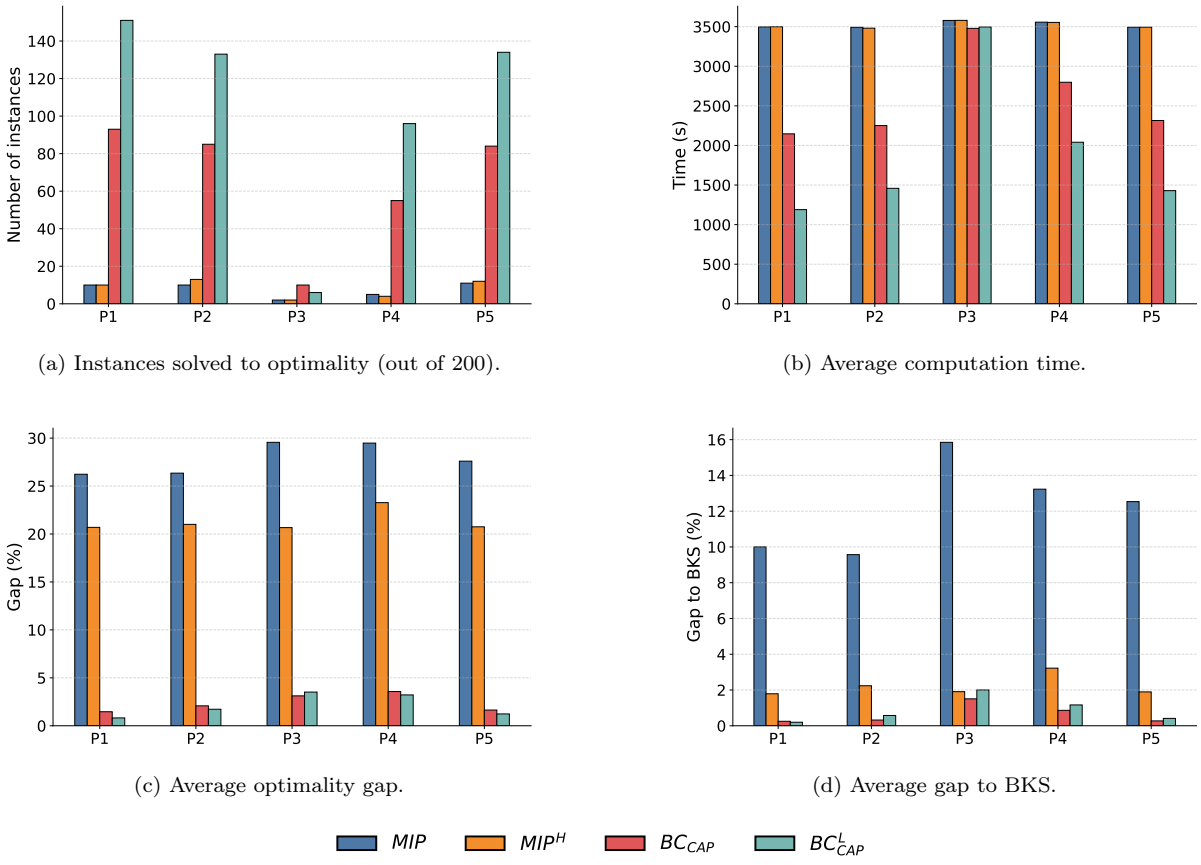
The results indicate that by using the pseudo-compact formulations alone (MIP), it is possible to solve small instances with up to 20 nodes to optimality, and obtain feasible solutions relatively close to the optimum for instances with up to 30 nodes. However, we observe large optimality gaps, mainly due to weak dual bounds caused by the MTZ-based constraints. In

terms of solution quality, the Gap_{BKS} increases with instance size, indicating a limitation for the solver to improve the incumbent solution. Providing a MIP start with HGS improves solution quality, as reflected in smaller Gap_{BKS} values. As the number of optimal solutions remains almost unchanged, the optimality gaps decrease only slightly because of improved upper bounds. For 40-node instances, the incumbent solution is often the initial solution provided by the MIP start, with the correct computation of the ECR value.

The BC_{CAP} approach delivers a significant computational performance boost, as a result of replacing the MTZ-based constraints by the ECI, which improves the dual bound of the formulation. As a consequence, nearly all 20-node instances were solved to optimality, and most under P1 and P2 were solved in under one minute. Some 40-node instances, most of which with higher customer-to-route ratios, were also solved to optimality. Also, solution quality improves significantly, with Gap_{BKS} values consistently below 0.5%.

Furthermore, defining the EECR inequalities as lazy constraints in BC^{L}_{CAP} further enhances the performance of the BC method by reducing model size, enabling the solver to handle the problem more effectively. In general, more optimal solutions are found and less computation time was required. On the other hand, BC_{CAP} is slightly superior in improving the incumbent solutions in harder instances, as indicated by the Gap_{BKS} metric.

Results under P3, which uses optimal preventive trips, deviate from the patterns observed in other policies. Here, modeling the ECR requires a pseudo-polynomial number of binary variables, increasing model size and weakening dual bounds. As a result, proving optimality



(a) Instances solved to optimality (out of 200).

(b) Average computation time.

(c) Average optimality gap.

(d) Average gap to BKS.

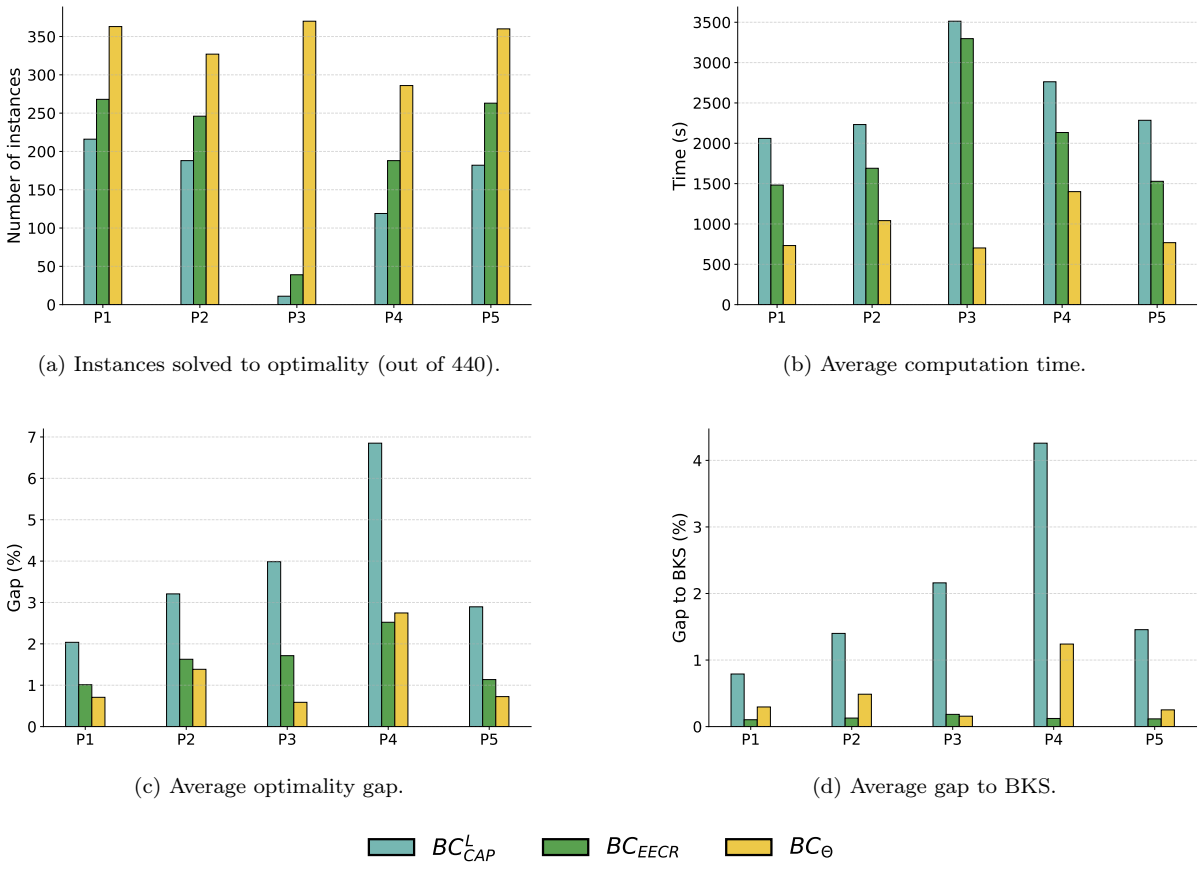
Figure 1: Summarized results of MIP, MIP^{H}, BC_{CAP} and BC^{L}_{CAP} for instances with up to 40 nodes.

becomes harder. Even so, the incumbent solutions remain of high quality, as indicated by the low Gap_{BKS} values, especially when using BC^L_{CAP}.

Overall, the results show that models incorporating EECR inequalities can consistently produce good, and sometimes optimal, solutions for the CVRPSD. The two enhancements (warm starts and implementing EECR inequalities as lazy constraints) significantly improve performance and the number of instances solved to optimality. The main limitations of these methods are that proving optimality remains a challenge for instances under the optimal policy, and that the performance decreases for instances with a lower customer-to-route ratio ($m = 4$).

6.2. Results for larger instances

Figure 2 summarizes the performance of BC^L_{CAP}, BC_{EECR} and BC_Θ on instances with up to 60 nodes, and Table 4 shows the average Gap_{BKS} of these methods. The detailed results are found in Table A3 of the supplementary material.



(a) Instances solved to optimality (out of 440).

(b) Average computation time.

(c) Average optimality gap.

(d) Average gap to BKS.

Figure 2: Summarized results of BC^L_{CAP}, BC_{EECR} and BC_Θ for all instances.

BC^L_{CAP} remains effective for instances with high customer-to-route ratios, where it finds strong incumbent solutions. However, a more detailed look at the results indicates that its performance does not scale well with instance sizes, particularly in cases with a low customer-to-route ratio. Another weakness of this method is evident under P4, where BC^L_{CAP} shows high optimality gaps and a lower quality of solutions indicated by Gap_{BKS}. These results suggest that the formulation is highly affected by the large big-M coefficients required by the EECR inequalities, which weaken the linear relaxation and limit the effectiveness of the method.

Table 4: Average Gap_{BKS} of BC^{L}_{CAP}, BC_{EECR} and BC_{Θ} for all instances.

n	m	BC^{L}_{CAP}	BC_{EECR}	BC_{Θ}
20	2	0.03%	0.00%	0.00%
30	2	0.27%	0.00%	0.01%
40	2	0.40%	0.00%	0.00%
	3	0.94%	0.04%	0.05%
	4	2.72%	0.46%	0.33%
50	2	1.10%	0.00%	0.00%
	3	2.81%	0.05%	0.31%
	4	4.51%	0.46%	1.19%
60	2	1.48%	0.01%	0.01%
	3	4.28%	0.09%	0.90%
	4	3.62%	0.32%	2.56%

In contrast, BC_{EECR} delivers a consistent improvement across all larger instances. The tailored separation algorithm strengthens the formulation, resulting in most two-vehicle instances with $n \geq 40$ solved to optimality and significant reductions in computation times. The performance gains are particularly pronounced under policies P4 and P5, where the EECR inequalities (24) and (25) are employed. These formulations were not tractable for BC^{L}_{CAP} due to prohibitive memory requirements, which already caused model construction to fail for instances with $n = 30$. Moreover, BC_{EECR} consistently achieves the smallest Gap_{BKS} values, demonstrating its superiority in generating high-quality solutions even when optimality cannot be proven. We highlight the fact that, even though BC_{EECR} found few optimal solutions under P3, the incumbents were competitive with BC_{Θ}, and superior in some cases (e.g., 60-node instances).

The use of optimality cuts in BC_{Θ} provides the best results in terms of computational effectiveness and scalability. Nearly all 20- and 30-node instances are solved to optimality, typically within a few seconds. This includes the instances under P3, since this method uses stronger cuts that do not require the z_{iq} variables. More importantly, BC_{Θ} outperforms the other methods as the number of vehicles increases, proving optimality in a substantially larger number of medium and large instances and consistently delivering stronger dual bounds. Across the entire test set, BC_{Θ} emerges as the most reliable and scalable approach.

A direct comparison between BC_{EECR} and BC_{Θ} highlights a complementary trade-off. While BC_{Θ} proves optimality more frequently, especially for instances with four vehicles, BC_{EECR} often achieves lower optimality gaps and Gap_{BKS} values in the largest instances. This indicates that EECR-based formulations are particularly effective at identifying high-quality incumbent solutions, whereas optimality cuts are more effective at strengthening the dual bound and closing the remaining gap.

In summary, these results show that the model size is a major factor behind the scalability of the methods. While BC^{L}_{CAP} struggles to deal with larger instances, BC_{EECR} and BC_{Θ} successfully address this limitation via tailored separation algorithms that reduce the number of inequalities needed to incorporate the ECR to the model. Among them, BC_{Θ} provides the best overall balance between effectiveness and scalability, while BC_{EECR} remains highly competitive when solution quality is the primary objective. These findings highlight the effectiveness of the proposed methods and establish both BC_{EECR} and BC_{Θ} as valuable approaches for large-scale instances of the problem.

6.3. Comparison with the state-of-the-art

In this subsection, we evaluate how the proposed methods compare with state-of-the-art approaches. We start by comparing the results of the two best performing methods, BC_{EECR} and BC_{Θ}, to the integer L-shaped method of Salavati-Khoshghalb et al. (2019a), for the CVRPSD under P3. Table 5 reports the results, including the number of solutions solved to optimality, the average computation time in seconds and the average optimality gap. In their paper, the authors do not provide results for individual instances, and computation times are reported as averages for the instances solved to optimality (using a time limit of 36000 seconds per instance). Those values are still included in the table for reference in column $Time^*$, but should be interpreted with caution due to differences in the hardware used for the experimentation.

BC_{Θ} outperforms the integer L-shaped method in terms of both scalability and computational efficiency. In particular, it solved a larger number of three- and four-vehicle instances to optimality and, for two-vehicle instances, it managed to solve them in significantly shorter average computation times, regardless of the number of nodes. BC_{EECR}, due to its inherent difficulty in dealing with instances under P3, had a less effective performance than the Integer L-shaped method, resulting in fewer proven optimal solutions and higher computation times.

We further evaluate the performance of BC_{EECR} and BC_{Θ} on the benchmark set from Louveaux and Salazar-González (2018), which consists of seven instances of varying sizes (four Euclidean and three asymmetric) each with eight variants generated by altering the number of vehicles, capacities, and demand realizations. This dataset was originally proposed for experimentation on the CVRPSD under the optimal restocking policy P3, and the state-of-the-art are the integer L-shaped method of Hoogendoorn and Spliet (2023) for the Euclidean instances and the integer L-shaped of Louveaux and Salazar-González (2018) and the BPC methods of Florio et al. (2020) for the asymmetric ones.

Table 6 compares the performance of BC_{EECR} and BC_{Θ} against the integer L-shaped method of Hoogendoorn and Spliet (2023), specifically its RS variant, which is the most time-efficient. The results show that BC_{Θ} matches the integer L-shaped in solution quality, proving optimality for the same number of instances and finding solutions with the same cost for the only two instances that were not solved to optimality. Moreover, BC_{Θ} is significantly faster on several instances and achieves shorter computation times in most of them. BC_{EECR}, on the other hand, found optimal solutions for 7 of the 32 instances. It, however, showed consistency in finding good solutions, including the optimal (as proven by other methods) for more than half of the dataset,

Table 5: Results of BC_{EECR}, BC_{Θ} and the integer L-shaped method of Salavati-Khoshghalb et al. (2019a) under the optimal restocking policy P3.

		BC_{EECR}			BC_{Θ}			Integer L-shaped		
n	m	Opt.	Time	Gap	Opt.	Time	Gap	Opt.	Time*	Gap
20	2	12	2621	1.09%	40	< 1	0.00%	40	22	0.00%
30	2	5	3175	0.91%	40	1	0.00%	37	886	0.06%
40	2	5	3155	0.37%	40	1	0.00%	39	185	0.00%
	3	1	3519	1.44%	40	159	0.00%	19	6750	0.67%
	4	0	3600	3.55%	29	1430	0.77%	0	0	3.37%
50	2	10	2728	0.45%	40	5	0.00%	36	460	0.09%
	3	0	3600	1.20%	37	401	0.30%	12	3827	0.97%
	4	0	3600	3.44%	19	2120	1.66%	3	2812	3.26%
60	2	6	3071	0.35%	40	5	0.00%	34	782	0.05%
	3	0	3600	1.88%	30	1096	0.99%	6	7185	1.71%
	4	0	3600	4.19%	15	2509	2.74%	1	9785	3.26%

Table 6: Results for the Euclidean instances from Louveaux and Salazar-González (2018) under the optimal restocking policy P3, and comparison with the RS integer L-shaped from Hoogendoorn and Spliet (2023).

			BC_{EECR}			BC_{Θ}			RS L-shaped		
m	Q	K	UB	Time	Gap	UB	Time	Gap	UB	Time	Gap
Instance class E031-09h, $n = 31$											
2	84	3	332.75	832.63	0.00%	332.75	0.11	0%	332.75	0.10	0%
2	79	3	335.30	3600.00	0.39%	335.30	0.16	0%	335.30	0.13	0%
2	84	9	337.67	3600.00	1.68%	337.67	0.25	0%	337.67	0.66	0%
2	79	9	344.53	3600.00	3.05%	344.53	1.18	0%	344.53	3.19	0%
3	59	3	358.95	61.21	0.00%	358.95	0.38	0%	358.95	0.23	0%
3	56	3	364.10	3600.00	0.85%	364.07	1.02	0%	364.07	0.99	0%
3	59	9	367.25	3600.00	2.52%	367.16	1.79	0%	367.16	3.71	0%
3	56	9	375.05	3600.00	4.55%	372.78	12.32	0%	372.78	73.90	0%
Instance class E051-05e, $n = 51$											
2	139	3	441.00	31.29	0.00%	441.00	0.83	0%	441.00	0.45	0%
2	132	3	441.31	3600.00	0.07%	441.31	1.20	0%	441.31	1.51	0%
2	139	9	443.37	3600.00	0.53%	443.01	5.70	0%	443.01	3.72	0%
2	132	9	448.53	3600.00	1.68%	448.08	75.77	0%	448.08	196.03	0%
3	99	3	459.00	24.30	0.00%	459.00	0.53	0%	459.00	0.52	0%
3	93	3	459.05	30.20	0.00%	459.05	1.01	0%	459.05	1.12	0%
3	99	9	460.55	3600.00	0.34%	460.55	1.06	0%	460.55	1.85	0%
3	93	9	465.90	3600.00	1.48%	465.63	18.80	0%	465.63	51.72	0%
Instance class E076-07s, $n = 76$											
2	209	3	549.01	19.33	0.00%	549.01	0.75	0%	549.01	0.93	0%
2	198	3	550.16	3600.00	0.21%	550.16	10.06	0%	550.16	4.70	0%
2	209	9	550.92	3600.00	0.35%	550.82	5.77	0%	550.82	7.83	0%
2	198	9	556.45	3600.00	1.34%	554.80	159.59	0%	554.80	428.31	0%
3	148	3	567.13	3600.00	0.20%	567.13	4.50	0%	567.13	12.26	0%
3	139	3	569.27	3600.00	0.22%	569.27	82.85	0%	569.27	41.65	0%
3	148	9	570.48	3600.00	0.78%	569.95	22.62	0%	569.95	119.74	0%
3	139	9	576.02	3600.00	1.91%	573.25	2212.23	0%	573.25	1190.24	0%
Instance class E101-08e, $n = 101$											
2	278	3	640.00	37.72	0.00%	640.00	1.57	0%	640.00	3.17	0%
2	264	3	641.73	3600.00	0.27%	641.73	29.41	0%	641.73	51.44	0%
2	278	9	641.30	3600.00	0.20%	641.30	14.82	0%	641.30	91.66	0%
2	264	9	646.12	3600.00	0.95%	646.12	3600.00	0.63%	646.12	3600.00	0.37%
3	197	3	655.35	3600.00	0.05%	655.35	6.50	0%	655.35	7.50	0%
3	186	3	658.30	3600.00	0.20%	658.30	24.87	0%	658.30	106.79	0%
3	197	9	659.02	3600.00	0.61%	658.98	96.25	0%	658.98	223.07	0%
3	186	9	667.40	3600.00	1.80%	666.56	3600.00	0.96%	666.56	3600.00	1.12%

with most optimality gaps being lower than 1%.

Table 7 summarizes the results of BC_{EECR} and BC_{Θ} for the asymmetric instances from Louveaux and Salazar-González (2018) under P3, comparing them to those from the integer L-shaped method of Louveaux and Salazar-González (2018) and the BPC of Florio et al. (2020). For the latter, the authors do not report the optimality gaps nor solve instances with 71 nodes. BC_{Θ} showed an overall more effective performance than the two state-of-the-art methods, solving to optimality all instances with 34 and 48 nodes, which was not achieved by the other two methods, and with consistently better computation times. Furthermore, it solved to optimality two previously unsolved instances (A034-02f-m2-Q87-K9 and A071-03f-m3-Q130-K3), and found a new BKS for instance A071-03f-m3-Q130-K9. BC_{EECR} found proven optimal solutions for two of the 24 instances, and for 12 of them it terminated with a feasible solution that, although not proven by the method, is the optimal solution reported in the literature. These results are in line with the previous results of this paper, indicating a difficulty of this method in tightening the dual bound of the models.

In summary, our results show that BC_{Θ} combines computational effectiveness with relatively accessible implementation requirements, which makes it a valuable method for both researchers and practitioners. It quickly proves optimality for 2-vehicle instances regardless of the recourse

Table 7: Results for the asymmetric instances from Louveaux and Salazar-González (2018) under the optimal restocking policy P3, and comparisons with the integer L-Shaped of Louveaux and Salazar-González (2018) and the BPC of Florio et al. (2020).

			BC_{EECR}			BC_{Θ}			Integer L-Shaped			BPC	
m	Q	K	UB	Time	Gap	UB	Time	Gap	UB	Time	Gap	UB	Time
Instance class A034-02f, $n = 34$													
2	92	3	1404.63	3600	1.18%	1404.64	< 1	0.00%	1404.64	< 1	0.00%	1404.64	6480
2	87	3	1418.72	3600	0.90%	1418.72	1	0.00%	1418.72	1	0.00%	1418.72	2736
2	92	9	1441.28	3600	3.70%	1436.02	6	0.00%	1436.02	14	0.00%	1436.02	4140
2	87	9	1484.13	3600	5.40%	1484.13	1582	0.00%	1484.13	18000	0.75%	1484.13	18000
3	65	3	1557.81	499	0.00%	1557.84	1	0.00%	1557.84	33	0.00%	1557.84	108
3	62	3	1605.28	3600	3.26%	1595.84	6	0.00%	1595.84	266	0.00%	1595.84	144
3	65	9	1644.64	3600	5.57%	1613.76	331	0.00%	1613.76	18000	1.19%	1613.76	360
3	62	9	1656.80	3600	6.27%	1648.70	2829	0.00%	1679.79	18000	6.37%	1648.70	252
Instance class A048-03f, $n = 48$													
2	131	3	1812.06	1	0.00%	1812.06	< 1	0.00%	1812.02	< 1	0.00%	-	18000
2	124	3	1818.16	3600	0.34%	1818.16	< 1	0.00%	1818.16	1	0.00%	-	18000
2	131	9	1827.96	3600	0.87%	1827.96	1	0.00%	1827.96	8	0.00%	-	18000
2	124	9	1864.33	3600	2.81%	1864.33	84	0.00%	1864.33	1628	0.00%	-	18000
3	93	3	1953.15	3600	1.13%	1953.15	9	0.00%	1953.15	123	0.00%	1953.15	18000
3	88	3	1960.16	3600	0.37%	1960.17	8	0.00%	1960.17	219	0.00%	1960.17	5544
3	93	9	1982.48	3600	2.58%	1968.48	86	0.00%	1968.48	2723	0.00%	1968.48	15588
3	88	9	1989.60	3600	1.84%	1989.60	307	0.00%	1989.60	18000	1.33%	1989.60	6264
Instance class A071-03f, $n = 71$													
2	195	3	1979.42	3600	0.02%	1979.43	2	0.00%	1979.43	2	0.00%		
2	185	3	1985.90	3600	0.35%	1985.93	5	0.00%	1985.93	13	0.00%		
2	195	9	1997.85	3600	0.94%	1993.74	83	0.00%	1993.74	8036	0.00%		
2	185	9	2028.60	3600	2.44%	2027.83	3600	1.57%	2027.83	18000	2.02%		
3	138	3	2098.32	3600	1.83%	2093.40	811	0.00%	2093.40	17491	0.00%		
3	130	3	2105.27	3600	0.63%	2105.27	428	0.00%	2105.27	18000	0.29%		
3	138	9	2114.99	3600	2.60%	2113.13	3600	1.25%	2113.13	18000	1.79%		
3	130	9	2153.36	3660	3.76%	2141.85	3600	1.94%	2146.38	18000	3.22%		

policy, and finds high-quality solutions for larger instances, being applicable to settings with both Euclidean and asymmetric distance matrices. Its performance is comparable, and in some cases surpasses the state-of-the-art across different datasets.

7. Conclusion

In this paper, we addressed the Capacitated Vehicle Routing Problem with Stochastic Demands (CVRPSD) with the aim of developing exact methods that are both effective, accessible, and generalizable across different recourse policies. The motivation of this study comes from the fact that state-of-the-art approaches rely on complex and often problem-specific algorithms. In contrast, the proposed methods combine computational efficiency with ease of implementation, providing a simple yet powerful alternative.

We introduced a novel family of pseudo-compact MIP formulations that explicitly compute the Expected Cost of Recourse (ECR) within the model, enabling for the first time the exact solution of instances of the addressed CVRPSD using a general-purpose MIP solver. We considered different recourse policies in the form of recursive equations, which were converted to linear constraints and used to define the pseudo-compact MIP formulations. These constraints were named Explicit ECR inequalities, and their implementation in a MIP solver removes the need of problem-specific procedures within the method to compute the ECR of a priori routes. From these formulations, we then developed tailored branch-and-cut (BC) methods that differ in their level of implementation complexity and practical accessibility.

Computational experiments with benchmark instances reveal that, via the proposed pseudo-

compact MIP formulations, a general-purpose MIP solver is able to prove optimality for small instances, more specifically those with 20 nodes and two vehicles, without relying on tailored techniques. By incorporating a separation algorithm for the Expected Capacity Inequalities, we manage to improve the performance of the BC while maintaining the use of the pseudo-compact formulations unchanged. We highlight the value of this approach due to its effectiveness in solving small- to medium-sized instances of the CVRPSD, and its low implementation complexity: the separation algorithm is the same as that used for deterministic variants, with several well-established frameworks, such as CVRPSEP (Lysgaard, 2003), readily available for integration.

For larger instances, a more effective BC method, incorporating a tailored separation algorithm for the Explicit ECR inequalities, solves instances with up to 60 nodes. In addition, we introduce a BC method using route-based optimality cuts (BC_Θ), which achieves computational performance comparable to state-of-the-art methods. More specifically, for several Euclidean instances under the optimal restocking recourse policy, this method approach matches, and in some cases outperforms, the state-of-the-art integer L-shaped method (Hoogendoorn and Spliet, 2023). For asymmetric instances, we show that BC_Θ is superior to other existing methods (Louveaux and Salazar-González, 2018; Florio et al., 2020) applicable to this setting, solving to optimality instances that had not previously been solved. Due to its relatively simple implementation when compared to the integer L-shaped and branch-price-and-cut methods, BC_Θ is an attractive alternative for practitioners and researchers seeking to address the CVRPSD under different classes of recourse policies.

As for future research, we suggest the investigation of recourse policies with dynamic decisions, such as changing customer order along the way or choosing how many demand units are fulfilled. Such policies present a challenge in the sense of how they can be effectively modeled via a closed form MIP formulation. We also suggest addressing the CVRPSD where demands are modeled via scenarios. Under this approach, we would be able to solve instances in which there is a correlation between demands of different customers. Although some recent work has been published in this field (Florio et al., 2020; Fukasawa and Gunter, 2023; Ota and Fukasawa, 2025), several gaps remain in the literature regarding exact methods and formulations for this class of problems. We strongly believe that the approaches proposed in this paper offer a promising direction for bridging these gaps.

Acknowledgements

This study was financed, in part, by the São Paulo Research Foundation (FAPESP), Brazil (grant numbers 2013/07375-0, 2022/05803-3, 2024/06580-3 and 2025/06567-0); the National Council for Scientific and Technological Development (CNPq), Brazil (grant numbers 405702/2021-3, 314079/2023-8 and 304618/2023-3); and was carried out using the computational resources of the Center for Mathematical Sciences Applied to Industry (CeMEAI) funded by FAPESP under grant number 2013/07375-0.

Fundação de Amparo à Pesquisa do Estado de São Paulo

References

Bertsimas, D.J., 1992. A vehicle routing problem with stochastic demand. Operations Research 40, 574–585. doi:10.1287/opre.40.3.574.

Birge, J.R., Louveaux, F., 2011. Introduction to stochastic programming. Springer series in operations research and financial engineering. second edition. ed., Springer, New York NY. doi:10.1007/978-1-4614-0237-4.

Christiansen, C.H., Lysgaard, J., 2007. A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. Operations Research Letters 35, 773–781. doi:10.1016/j.orl.2006.12.009.

Dror, M., Laporte, G., Trudeau, P., 1989. Vehicle routing with stochastic demands: Properties and solution frameworks. Transportation Science 23, 166–176. doi:10.1287/trsc.23.3.166.

Florio, A.M., Gendreau, M., Hartl, R.F., Minner, S., Vidal, T., 2023. Recent advances in vehicle routing with stochastic demands: Bayesian learning for correlated demands and elementary branch-price-and-cut. European Journal of Operational Research 306, 1081–1093. doi:10.1016/j.ejor.2022.10.045.

Florio, A.M., Hartl, R.F., Minner, S., 2020. New exact algorithm for the vehicle routing problem with stochastic demands. Transportation Science 54, 1073–1090. doi:10.1287/trsc.2020.0976.

Fukasawa, R., Gunter, J., 2023. The complexity of branch-and-price algorithms for the capacitated vehicle routing problem with stochastic demands. Operations Research Letters 51, 11–16. doi:10.1016/j.orl.2022.11.005.

Gauvin, C., Desaulniers, G., Gendreau, M., 2014. A branch-cut-and-price algorithm for the vehicle routing problem with stochastic demands. Computers & Operations Research 50, 141–153. doi:10.1016/j.cor.2014.03.028.

Gendreau, M., Jabali, O., Rei, W., 2014. Stochastic vehicle routing problems, in: Toth, P., Vigo, D. (Eds.), Vehicle routing: problems, methods, and applications, 2nd edn. Society for Industrial and Applied Mathematics, p. 213–239. doi:10.1137/1.9781611973594.ch8.

Gendreau, M., Jabali, O., Rei, W., 2016. 50th anniversary invited article—future research directions in stochastic vehicle routing. Transportation Science 50, 1163–1173. doi:10.1287/trsc.2016.0709.

Gendreau, M., Laporte, G., Séguin, R., 1995. An exact algorithm for the vehicle routing problem with stochastic demands and customers. Transportation Science 29, 143–155. doi:10.1287/trsc.29.2.143.

Hoogendoorn, Y.N., Spliet, R., 2023. An improved integer l-shaped method for the vehicle routing problem with stochastic demands. INFORMS Journal on Computing 35, 423–439. doi:10.1287/ijoc.2023.1271.

Hoogendoorn, Y.N., Spliet, R., 2025. An evaluation of common modeling choices for the vehicle routing problem with stochastic demands. European Journal of Operational Research 321, 107–122. doi:10.1016/j.ejor.2024.09.007.

Jabali, O., Rei, W., Gendreau, M., Laporte, G., 2014. Partial-route inequalities for the multi-vehicle routing problem with stochastic demands. Discrete Applied Mathematics 177, 121–136. doi:10.1016/j.dam.2014.05.040.

Laporte, G., Louveaux, F., van Hamme, L., 2002. An integer l-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. Operations Research 50, 415–423. doi:10.1287/opre.50.3.415.7751.

Laporte, G., Louveaux, F.V., 1993. The integer L-shaped method for stochastic integer programs with complete recourse. Operations Research Letters 13, 133–142. doi:10.1016/0167-6377(93)90002-X.

Louveaux, F.V., Salazar-González, J.J., 2018. Exact approach for the vehicle routing problem with stochastic demands and preventive returns. Transportation Science 52, 1463–1478. doi:10.1287/trsc.2017.0780.

Lysgaard, J., 2003. CVRPSEP: A package of separation routines for the Capacitated Vehicle Routing Problem. Technical Report. Aarthus University.

Miller, C.E., Tucker, A.W., Zemlin, R.A., 1960. Integer programming formulation of traveling salesman problems. Journal of the ACM (JACM) 7, 326–329. doi:10.1145/321043.321046.

Ota, M.J., Fukasawa, R., 2025. Hardness of pricing routes for two-stage stochastic vehicle routing problems with scenarios. Operations Research 73, 2177–2187. doi:10.1287/opre.2023.0569.

Oyola, J., Arntzen, H., Woodruff, D.L., 2017. The stochastic vehicle routing problem, a literature review, part ii: solution methods. EURO Journal on Transportation and Logistics 6, 349–388. doi:10.1007/s13676-016-0099-7.

Oyola, J., Arntzen, H., Woodruff, D.L., 2018. The stochastic vehicle routing problem, a literature review, part i: models. EURO Journal on Transportation and Logistics 7, 193–221. doi:10.1007/s13676-016-0100-5.

Parada, L., Legault, R., Côté, J.F., Gendreau, M., 2024. A disaggregated integer l-shaped method for stochastic vehicle routing problems with monotonic recourse. European Journal of Operational Research 318, 520–533. doi:10.1016/j.ejor.2024.05.012.

Salavati-Khoshghalb, M., Gendreau, M., Jabali, O., Rei, W., 2019a. An exact algorithm to solve the vehicle routing problem with stochastic demands under an optimal restocking policy. European Journal of Operational Research 273, 175–189. doi:10.1016/j.ejor.2018.07.039.

Salavati-Khoshghalb, M., Gendreau, M., Jabali, O., Rei, W., 2019b. A hybrid recourse policy for the vehicle routing problem with stochastic demands. EURO Journal on Transportation and Logistics 8, 269–298. doi:10.1007/s13676-018-0126-y.

Salavati-Khoshghalb, M., Gendreau, M., Jabali, O., Rei, W., 2019c. A rule-based recourse for the vehicle routing problem with stochastic demands. Transportation Science 53, 1334–1353. doi:10.1287/trsc.2018.0876.

Stewart, W.R., Golden, B.L., 1983. Stochastic vehicle routing: A comprehensive approach. European Journal of Operational Research 14, 371–385. doi:10.1016/0377-2217(83)90237-0.

Tillman, F.A., 1969. The multiple terminal delivery problem with probabilistic demands. Transportation Science 3, 192–204. doi:10.1287/trsc.3.3.192.

Toth, P., Vigo, D., 2014. Vehicle routing : problems, methods, and applications. MOS-SIAM series on optimization. second edition. ed., Society for Industrial and Applied Mathematics, Philadelphia, PA. doi:10.1137/1.9781611973594.

Vidal, T., 2022. Hybrid genetic search for the CVRP: Open-source implementation and SWAP* neighborhood. Computers & Operations Research 140, 105643. doi:10.1016/j.cor.2021.105643.

Vidal, T., Crainic, T.G., Gendreau, M., Lahrichi, N., Rei, W., 2012. A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. Operations Research 60, 611–624. doi:10.1287/opre.1120.1048.

Yang, W.H., Mathur, K., Ballou, R.H., 2000. Stochastic vehicle routing problem with restocking. Transportation Science 34, 99–112. doi:10.1287/trsc.34.1.99.12278.

Pseudo-Compact Formulations and Branch-and-Cut Approaches for the Capacitated Vehicle Routing Problem with Stochastic Demands - Supplementary Material

1. Detailed results of Section 6

Table A1: Detailed results of MIP and MIP^H for instances with up to 40 nodes.

		MIP				MIP^H			
n	m	Opt.	Time	Gap	Gap_{BKS}	Opt.	Time	Gap	Gap_{BKS}
Policy P1 - Classical									
20	2	10	3085	10.29%	0.31%	10	3097	10.17%	0.20%
30	2	-	3600	18.65%	5.10%	-	3600	15.29%	1.04%
40	2	-	3600	25.96%	15.96%	-	3600	15.12%	0.90%
	3	-	3600	35.10%	15.15%	-	3600	26.78%	2.45%
	4	-	3600	41.16%	13.49%	-	3600	35.95%	4.33%
Policy P2 - Rule, $\theta_j = \max \xi_j$									
20	2	10	3058	10.53%	0.18%	13	3005	9.60%	0.15%
30	2	-	3600	18.38%	4.36%	-	3600	15.38%	1.08%
40	2	-	3600	25.76%	15.07%	-	3600	15.16%	1.22%
	3	-	3600	35.34%	15.47%	-	3600	27.47%	3.09%
	4	-	3600	41.72%	12.77%	-	3600	37.40%	5.65%
Policy P3 - Optimal									
20	2	2	3493	11.48%	0.89%	2	3497	10.47%	0.35%
30	2	-	3600	23.67%	11.57%	-	3600	15.66%	1.26%
40	2	-	3600	29.14%	22.18%	-	3600	14.57%	0.98%
	3	-	3600	38.82%	23.27%	-	3600	26.55%	2.37%
	4	-	3600	44.68%	21.33%	-	3600	36.07%	4.57%
Policy P4 - Backlog, $\pi_j = 50$									
20	2	5	3384	12.38%	0.17%	4	3369	12.41%	0.11%
30	2	-	3600	21.46%	6.32%	-	3600	17.88%	1.77%
40	2	-	3600	28.45%	19.03%	-	3600	17.06%	2.97%
	3	-	3600	39.08%	21.16%	-	3600	29.67%	4.78%
	4	-	3600	46.02%	19.47%	-	3600	39.33%	6.47%
Policy P5 - Single Route, $\hat{\pi}_j = 1$									
20	2	11	3063	10.07%	0.15%	12	3066	9.76%	0.12%
30	2	-	3600	19.12%	5.51%	-	3600	15.65%	1.18%
40	2	-	3600	28.83%	20.97%	-	3600	15.24%	1.19%
	3	-	3600	37.67%	20.45%	-	3600	26.94%	2.53%
	4	-	3600	42.28%	15.59%	-	3600	36.15%	4.45%

Table A2: Detailed results of BC_{CAP} and BC^{L}_{CAP} for instances with up to 40 nodes.

		BC_{CAP}				BC^{L}_{CAP}			
n	m	Opt.	Time	Gap	Gap_{BKS}	Opt.	Time	Gap	Gap_{BKS}
Policy P1 - Classical									
20	2	40	45	-	0.00%	40	5	-	0.00%
30	2	32	1083	0.35%	0.00%	38	307	0.05%	0.00%
40	2	14	2718	0.80%	0.18%	39	442	0.02%	0.00%
	3	7	3300	1.99%	0.46%	28	1845	0.61%	0.10%
	4	-	3600	4.16%	0.61%	6	3348	3.42%	0.89%
Policy P2 - Rule, $\theta_j = \max \xi_j$									
20	2	40	53	-	0.00%	40	13	-	0.00%
30	2	30	1315	0.59%	0.02%	37	502	0.14%	0.00%
40	2	12	2839	1.04%	0.25%	37	647	0.05%	0.00%
	3	3	3448	2.64%	0.35%	19	2530	1.62%	0.47%
	4	-	3600	6.11%	0.98%	-	3600	6.80%	2.40%
Policy P3 - Optimal									
20	2	4	3333	1.41%	0.03%	3	3348	1.54%	0.14%
30	2	3	3436	1.81%	0.76%	1	3510	2.27%	1.30%
40	2	3	3425	1.45%	0.78%	2	3422	1.64%	1.07%
	3	-	3600	3.80%	2.14%	-	3600	4.11%	2.55%
	4	-	3600	7.11%	3.82%	-	3600	8.01%	4.95%
Policy P4 - Backlog, $\pi_j = 50$									
20	2	36	875	0.26%	0.00%	39	241	0.01%	0.00%
30	2	13	2773	2.44%	0.14%	28	1332	1.09%	0.05%
40	2	5	3230	2.99%	1.46%	21	1946	1.93%	0.92%
	3	1	3513	4.18%	0.84%	8	3083	4.00%	1.29%
	4	-	3600	7.98%	1.87%	-	3600	9.05%	3.56%
Policy P5 - Single Route, $\hat{\pi}_j = 1$									
20	2	40	50	-	0.00%	40	7	-	0.00%
30	2	30	1741	0.59%	0.02%	38	392	0.11%	0.00%
40	2	12	2738	0.95%	0.19%	37	722	0.06%	0.00%
	3	2	3447	2.18%	0.39%	17	2547	1.23%	0.26%
	4	-	3600	4.48%	0.76%	2	3478	4.77%	1.80%

Table A3: Detailed results of BC^{L}_{CAP}, BC_{EECR} and BC_{Θ} for instances with up to 60 nodes.

		BC^{L}_{CAP}				BC_{EECR}				BC_{Θ}			
n	m	Opt.	Time	Gap	Gap_{BKS}	Opt.	Time	Gap	Gap_{BKS}	Opt.	Time	Gap	Gap_{BKS}
Policy P1 - Classical													
20	2	40	5	-	0.00%	40	1	-	0.00%	40	<1	-	0.00%
30	2	38	307	0.05%	0.00%	39	139	0.01%	0.00%	40	1	-	0.00%
40	2	39	442	0.02%	0.00%	40	6	-	0.00%	40	1	-	0.00%
	3	28	1845	0.61%	0.10%	26	1349	0.41%	0.01%	39	155	0.00%	0.00%
	4	6	3348	3.42%	0.89%	1	3518	2.74%	0.41%	27	1487	0.87%	0.14%
50	2	27	1519	0.56%	0.16%	39	129	0.04%	0.00%	40	4	-	0.00%
	3	11	2787	2.11%	0.92%	26	1599	0.50%	0.03%	37	412	0.21%	0.11%
	4	4	3351	5.21%	2.21%	4	3339	2.86%	0.44%	18	2221	2.00%	0.71%
60	2	20	2019	0.71%	0.35%	39	139	0.01%	0.00%	40	4	-	0.00%
	3	3	3441	4.11%	2.19%	11	2733	1.14%	0.06%	29	1137	0.77%	0.28%
	4	0	3600	5.60%	1.88%	3	3354	3.41%	0.19%	13	2635	3.93%	2.01%
Policy P2 - Rule, $\theta_j = \max \xi_j$													
20	2	40	13	-	0.00%	40	8	-	0.00%	40	<1	-	0.00%
30	2	37	502	0.14%	0.00%	38	312	0.13%	0.00%	40	34	-	0.00%
40	2	37	647	0.05%	0.00%	40	13	-	0.00%	40	1	-	0.00%
	3	19	2530	1.62%	0.47%	20	2078	1.01%	0.05%	35	567	0.36%	0.06%
	4	0	3600	6.80%	2.40%	0	3600	4.82%	0.59%	15	2617	2.91%	0.61%
50	2	28	1375	0.92%	0.48%	37	334	0.06%	0.00%	40	11	-	0.00%
	3	6	3117	3.12%	1.40%	19	2119	0.96%	0.05%	33	744	0.48%	0.11%
	4	2	3456	7.72%	3.39%	3	3395	4.42%	0.43%	10	2888	4.02%	1.22%
60	2	17	2216	1.16%	0.66%	39	234	0.03%	0.00%	40	15	-	0.00%
	3	2	3496	5.67%	3.50%	10	2900	1.61%	0.13%	27	1539	1.28%	0.53%
	4	0	3600	8.08%	3.10%	0	3600	4.87%	0.17%	7	3039	6.18%	2.83%
Policy P3 - Optimal													
20	2	3	3348	0.02	0.14%	12	2621	0.01	0.01%	40	<1	-	0.00%
30	2	1	3510	2.27%	1.30%	5	3175	0.91%	0.01%	40	1	-	0.00%
40	2	2	3422	1.64%	1.07%	5	3155	0.37%	0.00%	40	1	-	0.00%
	3	0	3600	4.11%	2.55%	1	3519	1.44%	0.13%	40	159	-	0.00%
	4	0	3600	8.01%	4.95%	0	3600	3.55%	0.54%	29	1430	0.77%	0.08%
50	2	3	3333	1.48%	0.85%	10	2728	0.45%	0.01%	40	5	-	0.00%
	3	0	3600	4.10%	2.41%	0	3600	1.20%	0.08%	37	401	0.30%	0.20%
	4	0	3600	7.57%	4.48%	0	3600	3.44%	0.51%	19	2120	1.66%	0.35%
60	2	2	3431	1.27%	0.76%	6	3071	0.35%	0.02%	40	5	-	0.00%
	3	0	3600	4.83%	2.69%	0	3600	1.88%	0.17%	30	1096	0.99%	0.47%
	4	0	3600	7.04%	2.57%	0	3600	4.19%	0.54%	15	2509	2.74%	0.62%
Policy P4 - Backlog, $\pi_j = 50$													
20	2	39	241	0.00	0.00%	38	317	0.00	0.00%	40	2	-	0.00%
30	2	28	1332	1.09%	0.05%	32	773	0.65%	0.00%	38	244	0.23%	0.02%
40	2	21	1946	1.93%	0.92%	31	855	0.38%	0.01%	40	188	-	0.01%
	3	8	3083	4.00%	1.29%	13	2495	2.04%	0.03%	27	1379	1.14%	0.17%
	4	0	3600	9.05%	3.56%	0	3600	5.69%	0.43%	11	2774	4.17%	0.67%
50	2	11	2798	4.55%	3.52%	28	1174	0.47%	0.00%	38	315	0.06%	0.01%
	3	2	3467	9.51%	7.09%	12	2712	2.15%	0.06%	26	1428	2.22%	1.04%
	4	1	3513	13.11%	8.46%	1	3510	5.92%	0.43%	9	3050	7.06%	3.07%
60	2	9	3206	5.76%	4.91%	27	1246	0.49%	0.01%	36	442	0.14%	0.02%
	3	0	3600	12.07%	9.41%	6	3187	2.75%	0.02%	17	2296	4.42%	2.75%
	4	0	3600	14.27%	7.63%	0	3600	7.10%	0.36%	4	3291	10.78%	5.86%
Policy P5 - Single Route, $\hat{\pi}_j = 1$													
20	2	40	7	-	0.00%	40	1	-	0.00%	40	<1	-	0.00%
30	2	38	392	0.11%	0.00%	38	199	0.07%	0.00%	40	2	-	0.00%
40	2	37	722	0.06%	0.00%	40	6	-	0.00%	40	1	-	0.00%
	3	17	2547	1.23%	0.26%	26	1425	0.47%	0.00%	38	254	0.05%	0.02%
	4	2	3478	4.77%	1.80%	0	3600	3.02%	0.35%	26	1573	1.05%	0.15%
50	2	21	2004	1.15%	0.50%	39	175	0.04%	0.00%	40	7	-	0.01%
	3	5	3308	3.84%	2.24%	25	1674	0.64%	0.05%	36	459	0.22%	0.05%
	4	2	3439	7.02%	4.01%	3	3355	3.12%	0.47%	17	2336	2.00%	0.58%
60	2	18	2194	1.18%	0.70%	39	189	0.01%	0.00%	40	4	-	0.01%
	3	2	3444	5.61%	3.59%	10	2794	1.29%	0.07%	30	1162	0.98%	0.48%
	4	0	3600	6.86%	2.92%	3	3398	3.83%	0.33%	13	2652	3.67%	1.46%