An Integer L-shaped algorithm for the vehicle routing problem with time windows and stochastic demands

Jonathan De La Vega[a,b], Michel Gendreau[c], Reinaldo Morabito[a], Pedro Munari[a,*], Fernando Ordóñez[d]

[a]*Production Engineering Department,Federal University of Sao Carlos, Rodovia Washington Luis, Km 235, Sao Carlos, 13565-905, SP, Brazil*
[b]*SimpliRoute, Holanda 100, Santiago, 7510021, Chile*
[c]*Centre Interuniversitaire de Recherche sur les Réseaux d'Entreprise, la Logistique et le Transport (CIRRELT); D'partement de Mathématiques et de Génie Industriel, Polytechnique Montréal, 2500 Chemin de Polytechnique, Montréal, H3C 3A7, Quebec, Canada*
[d]*Departamento de Ingeniería Industrial, Universidad de Chile, Republica 701, Santiago, 8370439, Chile*

Abstract

This paper addresses the vehicle routing problem with time windows and stochastic demands (VRPTWSD). The problem is modeled as a two-stage stochastic program with recourse, in which routes are designed in the first stage and executed in the second. A failure occurs if the load of the vehicle is insufficient to meet the observed demand of a customer, implying recourse actions to recover feasibility. We consider the classical recourse policy where reactive trips to the depot are made in case of failures and a fixed rule-based recourse policy where, in addition, preventive trips are allowed. These recourse actions delay the vehicle and may cause further failures if the arrival times on the remaining customers of the planned route do not satisfy their time windows. An additional recourse action is used to service the customers whose time windows would be violated in the planned routes. We propose an Integer L-shaped algorithm considering the mentioned recourse actions. To the best of our knowledge, this is the first tailored exact approach for the VRPTWSD. Computational experiments using 112 benchmark instances evaluate the performance of this algorithm as well as the quality of the stochastic problem solutions. The results indicate significant savings in the solutions when using the fixed rule-based policy and round-trip recourse actions instead of the classical policy. Additionally, the algorithm performed better with the fixed rule-based policy, solving to optimality all instances with up to 34 customers, and taking less time on instances that were solved to optimality with both policies.

Keywords: routing, time windows, stochastic demand, Integer L-shaped algorithm, fixed rule-based policy

1. Introduction

The vehicle routing problem (VRP) entails the design of least-cost routes to meet the demand of a number of customers spread across a geographic area. These routes must start and end at the depot; each customer must be visited exactly once; and the vehicle load along each route cannot exceed its capacity. An extensively studied variant of this problem is the VRP with time windows (VRPTW), which imposes the additional requirement that the service at each customer starts within its time window (Toth and Vigo,

*Corresponding author.
Email addresses: jonathan.delavega@simplit-solutions.com (Jonathan De La Vega), michel.gendreau@cirrelt.net (Michel Gendreau), morabito@ufscar.br (Reinaldo Morabito), munari@dep.ufscar.br (Pedro Munari), fordon@dii.uchile.cl (Fernando Ordóñez)

2014; Braekers et al., 2016). Most research addressing the VRPTW assumes that the input parameters are deterministic, i.e., customer demands as well as service and travel times are known a priori from the route design. A different setting is observed in most real logistics configurations though, as random factors act directly and indirectly on these parameters, making them stochastic. However, addressing a problem in its stochastic version typically makes it more difficult to solve than its deterministic counterpart, bringing additional challenges for the design of efficient methods (Gendreau et al., 2016).

In this paper, we address the VRP with time windows and stochastic demands (VRPTWSD) and assume that the actual demands are revealed only when the vehicle arrives at the customer location. The problem is modeled as a two-stage stochastic program with recourse, in which routes are designed in the first stage and executed in the second. In the execution stage, failures can occur if the load of a vehicle is insufficient to meet the observed demand of vertices. In such cases, recourse actions are performed to recover the feasibility of the routes designed in the first stage. To govern the recourse actions, we use a fixed rule-based policy introduced by Salavati-Khoshghalb et al. (2019c) that allows implementing preventive recourse actions to avoid possible future failures along the route. We use two recourse actions commonly used in the literature (Dror et al., 1989; Bertsimas, 1992; Laporte et al., 2002; Mendoza and Villegas, 2013), implemented under fixed rule-based policy as follows. In the first rule, if the vehicle load is not sufficient to meet the observed demand of a vertex, the vehicle performs a *back-and-forth* (BF) trip to the depot, where the vehicle is replenished and then returns to the same vertex. In the second, if the vehicle load after meeting the demand of a given vertex of the route falls below a previously fixed threshold value, the vehicle performs a *restocking trip* (RT) to the depot for replenishing and then proceeds to the next vertex along its route.

An immediate consequence of applying these recourse actions is the change in the arrival times at the remaining vertices of the planned route because of the additional time required by the vehicle to return to the depot and resume its deliveries.We assume that a time window failure occurs if the arrival time in a vertex is outside its time windows in the planned route. Concerning this new failure event, we use an additional recourse action given by a special service that generates an extra cost. This special service may represent different types of strategies for the customers involved in a failure, such as performing separate round trips to each of them; leaving these customers to the next working day but with higher priority to their service; or any other strategy within the policy established between the transport company and customers that can be represented by extra costs. Our recourse action is based on the one introduced by Lei et al. (2011), consisting of using separate round trips to serve customers whose time windows would be violated in the planned route, which generates a cost proportional to the length of the corresponding individual round trips from the depot. Hence, a customer is not serviced in the planned route if the vehicle arrives at this customer outside its time window. Instead, we assume that this customer is serviced by a round trip performed with a smaller and faster vehicle that can manage to satisfy the customer's time window (e.g., using a premium transportation at the expense of higher costs). Hence, we consider hard time windows in the design of the first-stage routes as well as in the round trips used in the recourse actions related to time window failures in the planned routes.

To solve the VRPTWSD with the fixed rule-based policy, we propose an Integer L-shaped algorithm (Laporte and Louveaux, 1993). The overall performance of the algorithm is improved by using the k-path inequalities introduced in Kohl et al. (1999), in which the separation for $k = 2$ uses the labeling algorithm to exactly solve the traveling salesman problem with time windows (TSPTW), as in Desaulniers et al. (2008, 2016) and De La Vega et al. (2020). Furthermore, we design a procedure to determine a valid lower bound on

the expected recourse cost for general partial routes, which is then used in the generation of lower bounding functionals (LBF) as in Jabali et al. (2014). These LBF cuts improve the overall efficiency of the algorithm by tightening the linear relaxation of the subproblem and thus stemming the growth of the search tree. To the best of our knowledge, this is the first tailored exact method for the VRPTWSD considering the fixed rule-based policy. Additionally, no other exact approach for the VRPTWSD has resorted to k-path inequalities and LBF cuts.

The VRPTWSD has only been addressed by Chang (2005) and Lei et al. (2011). Chang (2005) modeled the problem as a two-stage stochastic program with recourse and proposed an Integer L-shaped algorithm for its solution in which the optimality cuts correspond to those proposed in Gendreau et al. (1995). In Lei et al. (2011), the problem was also approached as a two-stage stochastic program with recourse but solved using an adaptive large neighborhood search heuristic. Our approach differs from the one proposed by Chang (2005) as we consider hard time windows on vertices and rely on k-path inequalities and LBF cuts in the Integer L-shaped method. It also differs from Lei et al. (2011) for being an exact method and for using a backward recursive expression to compute the costs associated with recourse actions of the fixed rule-based policy.

In summary, the main contributions of this paper are as follows: 1) we propose the first tailored exact method based on the Integer L-shaped algorithm to effectively solve the VRPTWSD, considering both the classical and the fixed rule-based recourse policies; 2) we extend the recursive expression described by Salavati-Khoshghalb et al. (2019c), which computes the total expected recourse cost, to incorporate the cost of the special service used in the recourse action related to time window failures in the planned routes; and 3) we develop a procedure to determine a valid lower bound for the expected recourse cost for all solutions defined in a given partial solution.

The remainder of this paper is organized as follows. Section 2 highlights the characteristics of related studies that have addressed the VRP variants in their stochastic version. Then, Section 3 presents a formulation for the VRPTWSD based on two-stage stochastic programming with recourse, and Section 4 proposes an Integer L-shaped algorithm to solve this formulation. The results of the numerical experiments are then reported in Section 5. Finally, the conclusions and perspectives of future work are presented in Section 6.

2. Related Literature

In this section, we review the main contributions regarding VRP variants with stochastic demands, focusing on exact solution approaches. Two-stage stochastic programming with recourse has been the most used paradigm for modeling stochastic VRP variants (Gendreau et al., 2016). In this approach, routes are designed in the first stage and executed in the second, which is known as the a priori optimization paradigm in the VRP literature (Bertsimas et al., 1990). The actual demand of a customer is known only when the vehicle arrives at this customer, and in the classical recourse policy, a return trip to the depot occurs every time a vehicle load is smaller than or equal to this demand (Dror et al., 1989).

Exact algorithms based on the L-shaped method have been developed for the VRP with stochastic demands (VRPSD) under the classical policy. Laporte et al. (2002) proposed the Integer L-shaped algorithm to exactly solve this variant, using general lower bounds on the expected recourse cost of solutions. In addition, the idea of partial routes introduced by Hjorring and Holt (1999) was extended to the multiple vehicle case, which further improved the computational performance of the proposed solution method. Jabali et al. (2014) enhanced this Integer L-shaped algorithm by generalizing the concept of partial routes originally

defined by Hjorring and Holt (1999) and extended by Laporte et al. (2002). Additionally, they proposed strengthened LBFs based on these generalized partial routes, as well as an exact separation procedure to detect violated LBF inequalities. Approaches based on the branch-and-price (BP) method have also been proposed for the VRPSD with the classical policy. The first BP algorithm in this context was proposed by Christiansen and Lysgaard (2007) and later enhanced by Gauvin et al. (2014), who solved the pricing subproblem (column/route generator subproblem) using a bidirectional labeling algorithm and a new aggregate dominance rule.

Other policies to govern the return trips to the depot, different from the classical policy, have been considered by Yang et al. (2000), Louveaux and Salazar-González (2018), Salavati-Khoshghalb et al. (2019b,a,c), and (Florio et al., 2020a, 2021). Yang et al. (2000) used an optimal restocking policy that allows preventive return trips to the depot in addition to the traditional back-and-forth trips due to failures. In this policy introduced by Yee and Golden (1980), a sequential decision rule is applied that optimally determines whether the vehicle leaving a given customer with a positive residual capacity should either proceed directly to the next vertex or make a preventive trip to the depot. The resulting problem was solved by means of two heuristic strategies, namely, route-first-cluster-next and cluster-first-route-second. Louveaux and Salazar-González (2018) and Salavati-Khoshghalb et al. (2019a) also addressed the VRPSD with an optimal restocking policy and proposed exact methods based on the Integer L-shaped method. In particular, the algorithm developed in Louveaux and Salazar-González (2018) was the first to solve instances of the problem with up to 100 customers. Florio et al. (2020a) and Florio et al. (2021) developed BP algorithms to solve the VRPSD and a variant considering probabilistic duration constraints, respectively, considering the optimal restocking policy in both cases. As opposed to previous approaches, these BP algorithms were effective for solving instances where feasible solutions involve many routes and few vertices per route.

Salavati-Khoshghalb et al. (2019c) introduced the rule-based recourse policy that resorts to a set of prefixed operational rules that specify when preventive trips to the depot should be made. Given a route, these policies are expressed using threshold values associated with each of the customers along the route. Later, Salavati-Khoshghalb et al. (2019b) proposed a hybrid policy based on the combination of fixed rule-based policies taking into account distance and risk.

Stochastic VRP variants with time windows have received considerably less attention in the literature. Chang (2005) addressed the VRP with soft time windows and stochastic demand. They introduced a recursive expression to determine the expected recourse cost, including penalties, when the time windows of some customers are violated due to returns to the depot. They solved the problem using an L-shaped algorithm similar to the one proposed by Gendreau et al. (1995) and presented numerical experiments based on the benchmark instance C101 of Solomon.

Lei et al. (2011) used the classical policy and assumed that the returns to the depot incur additional times that may make it impossible to serve the remaining customers in the planned route within their time windows. Thus, they proposed an additional recourse action to address this new type of failure event related to time windows, consisting of a single separate trip by a different vehicle. To solve the problem, they proposed an adaptive large neighborhood search heuristic.

As mentioned before, our study differs from Chang (2005) in the use of k-path and LBF cuts in the Integer L-shaped method, as well as in the consideration of hard time windows. It is also different from Lei et al. (2011) because we use an exact method to solve the problem and a backward recursive expression to compute the costs associated with recourse actions defined by the fixed rule-based policy. Additionally, none

of these papers include the possibility of preventive restocking trips to the depot in their recourse policies.

All papers mentioned so far in this section involve recourse actions or wait-and-see decisions, and assume that demands are independent random variables. Nevertheless, other approaches have been developed as well without considering such assumptions, based on other paradigms, such as robust optimization (RO) and stochastic programming with chance constraints (SPCC), considering that routes are here-and-now decisions. In the RO paradigm, routes are designed in a way to satisfy vehicle capacity for all demand realizations within an uncertainty set (Gounaris et al., 2013; Munari et al., 2019; De La Vega et al., 2020; Pessoa et al., 2021). In SPCC, violations in the vehicle capacity are allowed but with low probability (Beraldi et al., 2015; Mendoza et al., 2016). Dinh et al. (2018) proposed an approach based on SPCC that does not require the stochastic demands to be independent. Latorre-Biel et al. (2021) propose simheuristics for a VRPSD in which demands are correlated and may present a common trend, such as a joint increase on their mean values. Dynamic models can also be applied to variants of the VRP with stochastic demands (Secomandi and Margot, 2009; Goodson et al., 2015; Bertazzi and Secomandi, 2018). These models are a good option to reduce operational costs but they can become computationally prohibitive because the number of states increases exponentially with the number of vehicles and vertices. For comprehensive reviews on stochastic vehicle routing, see, e.g., Oyola et al. (2017, 2018) and (Gendreau et al., 2014, 2016).

3. Mathematical Formulation

Let $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ be a complete and directed graph, where $\mathcal{V}$ is the set of vertices and $\mathcal{A}$ is the set of arcs. Set $\mathcal{V}$ contains the n demand vertices that represent customers and the two vertices 0 and $n+1$ that represent the depot. $\mathcal{N}$ represents the set of demand vertices only, i.e., $\mathcal{N} = \mathcal{V} \setminus \{0, n+1\}$. The set of arcs is defined as $\mathcal{A} = \{(0, j) \mid j \in \mathcal{N}\} \cup \{(i, j) \mid i, j \in \mathcal{N} \text{ and } i \neq j\} \cup \{(i, n+1) \mid i \in \mathcal{N}\}$. The travel costs ($c_{ij}$) and travel times ($t_{ij}$), both satisfying the triangle inequality, are associated with the arcs, while the service times (s_i), the time windows ($[a_i, b_i]$) and the stochastic demands (ξ_i) are associated with the vertices. In particular, $s_0 = s_{n+1} = 0$, $a_0 = a_{n+1} = 0$ and $\mathbb{E}(\xi_0) = \mathbb{E}(\xi_{n+1}) = 0$. We assume a fleet of m identical vehicles of capacity Q, which depart from the depot at instant 0. Additionally, we assume that if a vehicle arrives at a customer before the opening of its time window, the vehicle must wait until the beginning of the time window before starting service. A route is a sequence of vertices that a vehicle can visit, starting from the initial depot 0 with a full load, serving the expected demand of each vertex, and finishing at the final depot $n+1$. The VRPTWSD consists of defining a set of routes to visit all vertices and meet their actual demands while minimizing the operating costs of the planned routes as well as the expected costs related to the recourse actions due to failure events. The first-stage decision corresponds to the definition of a set of routes $\mathcal{R}$, such that the service on each of the vertices starts within the time windows. The recourse cost is determined under the following assumptions:

A1) The demands are independent random variables represented by probability distributions that are discrete and known beforehand. Thus, for each customer $i \in \mathcal{N}$, the probability distribution of the stochastic demand ξ_i is represented by a finite number of o_i discrete realizations $\xi_i^1 \leq \xi_i^2 \leq \ldots \leq \xi_i^{o_i}$, where their corresponding probabilities of occurrence are expressed as $p_i^1 = \mathbb{P}\{\xi_i = \xi_i^1\}$, $p_i^2 = \mathbb{P}\{\xi_i = \xi_i^2\}, \ldots, p_i^{o_i} = \mathbb{P}\{\xi_i = \xi_i^{o_i}\}$.

A2) Demand realizations on each customer do not exceed the vehicle capacity Q, i.e., $\mathbb{P}\{\xi_i \leq Q\} = 1$ for all $i \in \mathcal{N}$.

A3) The total expected demand of each route does not exceed the vehicle capacity Q. More specifically, each route r involving n_r customers must respect the condition $\sum_{i=1}^{n_r} \mathbb{E}(\xi_{v_i}) \leq Q$, where v_i is the ith vertex in the route.

A4) At most, one return to the depot can be made on any route. This means that the probability of incurring more than one failure is 0, or equivalently, $\mathbb{P}\{\sum_{i=1}^{n_r} \xi_{v_i} \leq 2Q\} = 1$.

A5) Vertices whose time windows would be violated in the planned routes are served by separate single trips, which generate a cost proportional to the distance traveled in a round trip from the depot. These single trips are performed with smaller and faster vehicles that can manage to visit the customers inside their windows.

Assumptions **A1** and **A2** are standard in the literature. To the best of our knowledge, all previous exact approaches based on the Integer L-shaped algorithm for two-stage stochastic programming variants have considered these assumptions (see, e.g., Laporte et al. (2002); Jabali et al. (2014); Louveaux and Salazar-González (2018); Salavati-Khoshghalb et al. (2019a,b) and Salavati-Khoshghalb et al. (2019c)). Assumption **A3** is a common, simplifying assumption employed in other works on the VRPSD (Laporte et al. (2002); Jabali et al. (2014); Louveaux and Salazar-González (2018); Salavati-Khoshghalb et al. (2019a,b) and Salavati-Khoshghalb et al. (2019c)). Although **A3** does not hold in general (e.g., in waste collection, it is common to observe replenishment trips, as presented by Jaunich et al. (2016)), we employ this assumption in this first work on exact algorithms for the VRPTWSD. However, our algorithm works reasonably well even without assuming **A3**, as suggested by the results of computational experiments presented in Section 5.

Assumptions **A4** and **A5** were introduced in Lei et al. (2011), who were the first to study the VRPTWSD (using heuristic algorithms for its solution). Assumption **A4** helps to plan routes with low risk of failures related to violation of vertex time windows. If this assumption is relaxed, this risk increases because the new arrival times at the planned route vertices become larger with each restocking trip to the depot. Finally, in assumption **A5** we consider that the vehicles used to serve vertices with violated time windows are smaller and faster, reaching very large speeds. However, these vehicles have a high chance of attending the vertices within their time windows because they start their operation as soon as the failure occurs on the planned route.

3.1. Problem Formulation

The VRPTWSD can be formulated as a stochastic program with recourse involving the following decision variables: x_{ij} is a binary variable that takes the value 1 if and only if there is a route that visits vertex j immediately after visiting vertex i; and w_i is a continuous variable that denotes the exact time at which the vehicle starts its service at vertex i. Let $Q(\boldsymbol{x})$ be the expected recourse cost. The formulation can be described as follows:

Min

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

s.t.

$$\sum_{i:(i,j)\in\mathcal{A}} x_{ij} = 1, \quad \forall j \in \mathcal{N}, \tag{2}$$

$$\sum_{j:(i,j)\in\mathcal{A}} x_{ij} = 1, \quad \forall i \in \mathcal{N}, \tag{3}$$

$$\sum_{j\in\mathcal{N}} x_{0j} \leq m, \tag{4}$$

$$w_j \geq w_i + (s_i + t_{ij})x_{ij} - \mathcal{M}_{ij}(1 - x_{ij}), \quad \forall (i,j) \in \mathcal{A}, \tag{5}$$

$$a_i \leq w_i \leq b_i, \quad \forall i \in \mathcal{N} \cup \{n+1\}, \tag{6}$$

$$\sum_{(i,j)\in\delta(\mathcal{S})} x_{ij} \geq k(\mathcal{S}), \ \mathcal{S} \subset \mathcal{N}, \quad 2 \leq |\mathcal{S}| \leq |\mathcal{N}|-1, \tag{7}$$

$$x_{ij} \in \{0,1\}, \quad (i,j) \in \mathcal{A}. \tag{8}$$

Model (1)-(8) involves all the first-stage decisions that minimize the travel costs and the expected value of the costs incurred by the second-stage decisions (recourse actions). Constraints (2)-(3) define the vehicle flow and enforce that all demand vertices are visited. Constraint (4) ensures that the number of routes generated does not exceed the total number of vehicles (m) available in the depot. Time constraints are imposed in (5) and (6). In particular, constraints (6) ensure that services start within the time windows. Parameter $\mathcal{M}_{ij}$ is a sufficiently large number in constraints (5), defined in this paper as $\max\{b_i - a_j, 0\}$.

Constraints (7) are known as the k-path inequalities (Kohl et al., 1999; Desaulniers et al., 2008). Given a set $\mathcal{S} \subset \mathcal{N}$ of demand vertices, the left-hand side of this inequality corresponds to the vehicle flow entering in set $\mathcal{S}$, where $\delta(\mathcal{S}) = \left\{(i,j) \in \mathcal{A} : i \in \overline{\mathcal{S}} \text{ and } j \in \mathcal{S}\right\}$, while $k(\mathcal{S})$ is the minimum number of vehicles required to service all vertices in $\mathcal{S}$ satisfying the time windows, meeting the expected demand of vertices and respecting assumption **A3**. The procedure for separating these inequalities is described in Subsection 4.2. Finally, the binary domain of the vehicle flow variables is imposed by constraints (8).

3.2. Calculating the Expected Recourse Cost

In this section, we show how to determine the expected recourse cost $Q(\boldsymbol{x})$ for a given solution $\boldsymbol{x}$. Let $\mathcal{R}$ be the set of routes associated with this solution. We represent each route $r \in \mathcal{R}$ as a sequence of vertices $(v_1 = 0, v_2, \ldots, v_k, v_{k+1} = n+1)$, with associated threshold vector $(\theta_{v_2}, \ldots, \theta_{v_k})$. We use the rule-based policy known as volume-based, in which threshold values can be set to any estimate of customer demands on route r (Salavati-Khoshghalb et al., 2019c). These threshold values are set as the expected demands, thus one can easily identify when it is time to perform a preventive trip to the depot. In addition, it allows calculating a good lower bound on the recourse cost, as presented afterward. Then, each θ_{v_i}, for $i = 2, \ldots, k$, is defined as the expected demand of vertex $v_{(i+1)}$, i.e., $\theta_{v_i} = \mathbb{E}\left[\xi_{v_{(i+1)}}\right]$. Given a vertex v_i, for $i = 2, \ldots, k$, let q be the load of the vehicle upon arrival at this vertex, such that $\theta_{v_{(i-1)}} \leq q \leq Q$.

When the vehicle arrives at vertex v_i with q units, one of the following three mutually exclusive cases can be observed (Salavati-Khoshghalb et al., 2019c):

Case 1) The vehicle load remains greater than or equal to θ_{v_i} after servicing v_i, i.e., $q - \xi_{v_i} \geq \theta_{v_i}$. In this case, there is no failure related to the load of the vehicle, and therefore, the vehicle moves directly to the location of vertex $v_{(i+1)}$.

Case 2) The demand realization does not exceed q, but it causes the load to fall within the range of $0 \leq q - \xi_{v_i} < \theta_{v_i}$. In this case, although it does not correspond to a failure, a restocking trip to the depot is carried out. The cost associated with this recourse action corresponds to the cost of arcs $(v_i, n+1)$ and

$(0, v_{(i+1)})$, minus the cost of arc $(v_i, v_{(i+1)})$ that will no longer be traversed, i.e., $c_{v_i,n+1} + c_{0,v_{(i+1)}} - c_{v_i,v_{(i+1)}}$. Observe that, when $q - \xi_{v_i} > 0$, this preventive restocking trip to the depot can significantly reduce the total expected recourse cost with respect to a BF trip, as it does not incur the additional cost Γ (assuming that distances satisfy the triangle inequality).

Case 3) The realization of demand for v_i is strictly greater than q units, i.e., $q - \xi_{v_i} < 0$. In this case, a failure occurs and, as a recourse action, a BF trip must be made to the depot immediately after the delivery of q units to the vertex. The cost associated with this recourse is the total distance traveled on the BF trip between the vertex location and the depot, i.e., the cost of arcs $(v_i, n+1)$ and $(0, v_i)$. Moreover, an additional cost Γ is attributed to this recourse, as there is an interruption in the service at vertex v_i (Yang et al., 2000; Salavati-Khoshghalb et al., 2019c).

After applying a recourse action at vertex v_i, the planned arrival times at the remaining vertices on the planned route might be outside their time windows, because of the time required by the vehicle to return to the depot and resume its deliveries. Let $\mathcal{B}^1_{v_i}(q, \xi^\omega_{v_i})$ and $\mathcal{B}^2_{v_i}(q, \xi^\omega_{v_i})$ be the sets of vertices for which time windows would be violated after a BF and a restocking trip, respectively, assuming that the vehicle arrives with load q at vertex v_i and the observed demand of the vertex is $\xi^\omega_{v_i}$ (with $\omega = 1, \ldots, o_{v_i}$). We construct these sets based on the ideas proposed by Lei et al. (2011). First, we determine the new arrival times at vertices $(v_i, v_{i+1}, \ldots, v_k, v_{k+1})$ due to a trip to the depot in vertex v_i. Let w_{v_j} be the time at which the vehicle arrives at vertex v_j; s_{v_j} be the service time of vertex v_j; and $\tilde{w}_{v_j}(v_i)$ be the new arrival time at vertex v_j after a recourse action at vertex v_i. Then, $\tilde{w}_{v_j}(v_i)$ can be computed as follows:

$$\tilde{w}_{v_j}(v_i) = \begin{cases} w_{v_j}, & \text{if } j = 2, \ldots, i, \\ \zeta_{v_j}(v_i), & \text{if } j = i+1, \\ \tau_{v_j}(v_i), & \text{if } j = i+2, \ldots, k+1, \end{cases} \tag{9}$$

where $\tau_{v_j}(v_i)$ is computed as:

$$\tau_{v_j}(v_i) = \begin{cases} \max\left\{a_{v_{(j-1)}}, \tilde{w}_{v_{(j-1)}}(v_i)\right\} + s_{v_{(j-1)}} + t_{v_{(j-1)}v_j}, & \text{if } \tilde{w}_{v_{(j-1)}}(v_i) \le b_{v_{(j-1)}}, \\ \tilde{w}_{v_{(j-1)}}(v_i) + t_{v_{(j-1)}v_j}, & \text{otherwise,} \end{cases} \tag{10}$$

and, for a BF trip, $\zeta_{v_j}(v_i)$ is determined as:

$$\zeta_{v_j}(v_i) = \begin{cases} \max\left\{a_{v_{(j-1)}}, \tilde{w}_{v_{(j-1)}}(v_i)\right\} + s_{v_{(j-1)}} + t_{v_{(j-1)}v_j} + f_{v_{(j-1)}}, & \text{if } \tilde{w}_{v_{(j-1)}}(v_i) + f_{v_{(j-1)}} \le b_{v_{(j-1)}}, \\ \tilde{w}_{v_{(j-1)}}(v_i) + t_{v_{(j-1)}v_j} + f_{v_{(j-1)}}, & \text{otherwise,} \end{cases} \tag{11}$$

with $f_{v_{(j-1)}} = t_{v_{(j-1)}(n+1)} + t_{0v_{(j-1)}}$; while for a restocking trip, we use:

$$\zeta_{v_j}(v_i) = \tilde{w}_{v_{(j-1)}}(v_i) + s_{v_{(j-1)}} + t_{v_{(j-1)}(n+1)} + t_{0v_j}. \tag{12}$$

Additionally, we have the initial conditions $w_{v_2} = t_{v_1v_2}$ and $w_{v_i} = \max\{a_{v_{i-1}}, w_{v_{i-1}}\} + s_{v_{i-1}} + t_{v_{i-1}v_i}$ for $i = 3, \ldots, k+1$. Note that, as defined in (9), we have $\tilde{w}_{v_i}(v_i) = w_{v_i}$.

In (9), the arrival times remain the same for vertices v_2 to v_i, and thus, the corresponding time windows are satisfied. However, after applying a recourse action at v_i, the new arrival times at the remaining vertices in the route may change and cause infeasibility if vertex v_i is serviced in the planned route. A BF trip to the depot immediately after arriving at v_i (at exact time w_{v_i}) incurs the additional time f_{v_i}. Then, as defined in

(11), if $w_{v_i} + f_{v_i} \leq b_{v_i}$, v_i is fully serviced (only after its time window opens) and the vehicle continues to the next vertex. Otherwise, the vehicle cannot service v_i because the time window would be violated, and thus it goes immediately to vertex v_{i+1} after arriving in v_i from the depot. In this case, there is no service time in v_i. Additionally, the vehicle cannot skip this back-and-forth trip as we assume it gets fully replenished after a failure at v_i. However, if a restocking trip occurs, we resort to (12), which assumes that the vehicle services vertex v_i, returns to the depot for replenishment and then goes directly to vertex v_{i+1}. Finally, we update the arrival time at vertices v_{i+2} to v_{k+1} using (10), assuming that a vertex is fully serviced only if its time window is not violated.

Having determined the new arrival times, Proposition 3.1 is then sequentially used to construct sets $\mathcal{B}^1_{v_i}(q, \xi^\omega_{v_i})$ and $\mathcal{B}^2_{v_i}(q, \xi^\omega_{v_i})$ for all values of q and $\xi^\omega_{v_i}$.

Proposition 3.1. *For a given load $q \in [\theta_{v_{(i-1)}}, Q]$ and for a demand realization $\xi^\omega_{v_i}$ of vertex v_i, the sets $\mathcal{B}^1_{v_i}(q, \xi^\omega_{v_i})$ and $\mathcal{B}^2_{v_i}(q, \xi^\omega_{v_i})$ containing the vertices for which time windows would be violated due to a BF and restocking trip to the depot, respectively, can be determined as follows:*

$$\mathcal{B}^1_{v_i}(q, \xi^\omega_{v_i}) = \{v_i \mid w_{v_i} + f_{v_i} > b_{v_i}\} \cup \left\{v_j \mid \tilde{w}_{v_j}(v_i) > b_{v_j} \text{ and } q - \xi^\omega_{v_i} < 0,\ j = i+1, \ldots, k+1\right\}, \tag{13}$$

$$\mathcal{B}^2_{v_i}(q, \xi^\omega_{v_i}) = \left\{v_j \mid \tilde{w}_{v_j}(v_i) > b_{v_j} \text{ and } 0 \leq q - \xi^\omega_{v_i} < \theta_{v_i},\ j = i+1, \ldots, k+1\right\}. \tag{14}$$

Proof. For $q - \xi^\omega_{v_i} < 0$, the BF trip is realized and expression (9) can be used to determine the new arrival times $\tilde{w}_{v_j}(v_i)$, for $j = 2, \ldots, k+1$. According to this expression the recourse action only affects vertex v_i and the remaining vertices in the route. Therefore, the set of vertices that would have their time windows violated due to the BF trip can be constructed by the expression (13). The derivation of (14) is similar, but the recourse now only affects the vertices after vertex v_i ($j = i+1, \ldots, k, k+1$), thus completing the proof. □

Observe that set $\mathcal{B}^1_{v_i}(q, \xi^\omega_{v_i})$ can include the vertex v_i where the failure occurred, as the vehicle may return to this vertex after its time window closes, but no service is performed at this vertex using this vehicle. The vertices after v_i are not serviced by the vehicle if the arrival at them is later than the closing times of their time windows. Assumption **A5** in Section 3 states that they are serviced by separate individual trips that satisfy their time windows, generating a cost Φ times the cost corresponding to a round trip from the depot to each of these vertices. More specifically, the recourse action at vertex v_i generates a cost Φd_{0v_j} in which d_{0v_j} is equal to $c_{0v_j} + c_{v_j(n+1)}$ for $j = i, \ldots, k$. Additionally, we set $d_{0v_{k+1}} = 2\Gamma$, which penalizes the late arrival at the depot.

We can now determine the expected recourse cost associated with a BF or a restocking trip at vertex v_i. Given a route r, let $F^r_{v_i}(q)$ be the expected recourse cost at vertex v_i when the vehicle arrives at this vertex with load q. Then, $F^r_{v_i}(q)$ can be determined as follows:

$$
F^r_{v_i}(q) = \begin{cases}
F^r_{v_{(i+1)}}(q), & \text{if } i = 1, \\
\displaystyle\sum_{\substack{\omega=1,\ldots,o_{v_i}: \\ q-\xi^\omega_{v_i}<0}} p^\omega_{v_i} \left[\Gamma + c_{v_i,n+1} + c_{0,v_i} + \Phi \sum_{v_j \in \mathcal{B}^1_{v_i}(q,\xi^\omega_{v_i})} d_{0v_j} + F^r_{v_{(i+1)}}\left(Q + q - \xi^\omega_{v_i}\right) \right] \\
+ \displaystyle\sum_{\substack{\omega=1,\ldots,o_{v_i}: \\ 0\le q-\xi^\omega_{v_i}<\theta_{v_i}}} p^\omega_{v_i} \left[c_{v_i,n+1} + c_{0v_{(i+1)}} - c_{v_i v_{(i+1)}} + \Phi \sum_{v_j \in \mathcal{B}^2_{v_i}(q,\xi^\omega_{v_i})} d_{0v_j} + F^r_{v_{(i+1)}}(Q) \right] \\
+ \displaystyle\sum_{\substack{\omega=1,\ldots,o_{v_i}: \\ q-\xi^\omega_{v_i}\ge\theta_{v_i}}} p^\omega_{v_i} \left[F^r_{v_{(i+1)}}\left(q - \xi^\omega_{v_i}\right) \right], & \text{if } i = 2,\ldots,k, \\
0, & \text{if } i = k+1.
\end{cases} \tag{15}
$$

Expression (15) takes into consideration the three previously discussed cases, i.e., the cases in which there is a BF or restocking trip and in which there is no failure related to the vehicle load. Observe that $F^r_{v_i}(q)$ accumulates the value of the expected recourse cost of vertex $v_{(i+1)}$. The total expected recourse cost of route r is given by $F^r_{v_1}(Q)$, in which the vehicle starts its route from the initial depot with a full capacity (i.e., with load Q), and is calculated recursively. Finally, the total expected recourse cost of the set of routes $\mathcal{R}$ is determined as $Q(\boldsymbol{x}) = \sum_{r=1}^{|\mathcal{R}|} F^r_{v_1}(Q)$. Note that expression (15) extends the recursive equation introduced in Salavati-Khoshghalb et al. (2019c) by including the expected cost of the recourse action performed on vertices that would have their time windows violated in the planned routes.

It is important to note that for $\theta_{v_i} = 1$, for $i = 2,\ldots,k$, the fixed rule-based policy reduces to the classical policy that applies return trips to the depot when the load is insufficient to meet demand or is exactly equal to the demand, as proposed by Lei et al. (2011). To illustrate how the recourse trips to the depot are governed by both policies and how these recourse trips cause time window violations at vertices, we present an illustrative example in Appendix A of the e-companion.

4. Solution Method

We develop an Integer L-shaped algorithm to solve the VRPTWSD. This algorithm is similar to applying Benders decomposition to model (1)-(8) and further relies on specialized valid inequalities to impose effective lower bounds on the expected recourse cost $Q(\boldsymbol{x})$. We detail the proposed Integer L-shaped algorithm in Subsection 4.1 and the procedure for separating the k-path inequalities in Subsection 4.2. Finally, in Subsection 4.3, we present a procedure for separating the LBF cuts.

4.1. Integer L-shaped Algorithm

The Integer L-shaped algorithm is a BC procedure that starts with the linear programming (LP) relaxation of the formulation (1)-(8), in which the explicit evaluation of $Q(\boldsymbol{x})$ is replaced with the continuous variable Θ. The k-path inequalities (7), which act as subtour elimination constraints and impose vehicle capacity as well as assumption A3 of Section 3, are initially dropped from the formulation and then generated dynamically at the end of each node of the search tree. LBF and optimality cuts are also dynamically generated throughout the search tree. Hence, at iteration 0, the problem is as follows:

Min

$$\sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} + \Theta \tag{16}$$

s.t.

$$\sum_{i:(i,j)\in\mathcal{A}} x_{ij} = 1, \quad \forall j \in \mathcal{N}, \tag{17}$$

$$\sum_{j:(i,j)\in\mathcal{A}} x_{ij} = 1, \quad \forall i \in \mathcal{N}, \tag{18}$$

$$\sum_{(0,j)\in\mathcal{A}} x_{0j} \leq m, \tag{19}$$

$$w_j \geq w_i + (s_i + t_{ij})x_{ij} - \mathcal{M}_{ij}(1 - x_{ij}), \quad \forall (i,j) \in \mathcal{A}, \tag{20}$$

$$a_i \leq w_i \leq b_i, \quad \forall i \in \mathcal{N} \cup \{n+1\}, \tag{21}$$

$$\Theta \geq L, \tag{22}$$

$$x_{ij} \in [0,1], \quad (i,j) \in \mathcal{A}, \tag{23}$$

where L in constraint (22) corresponds to a valid general lower bound on $Q(\boldsymbol{x})$, i.e., $L \leq \min_{\boldsymbol{x}} Q(\boldsymbol{x})$. The algorithm proceeds by adding the mentioned cuts during the BC search until optimality is ensured or until some stopping criterion is reached. We separate 1-path and 2-path inequalities at the root node only, while k-path inequalities are separated at every node with an integer solution. The LBF cuts are stated as

$$L + (\hat{\Theta} - L)\tilde{W}(\boldsymbol{x}) \leq \Theta, \tag{24}$$

where $\hat{\Theta}$ is a valid lower bound for $Q(\boldsymbol{x}^v)$ and $\tilde{W}(\boldsymbol{x})$ is the functional in terms of flow variable $\boldsymbol{x}$. Their separation is detailed in Subsection 4.3. Finally, the optimality cut

$$\sum_{\substack{(i,j)\in\mathcal{A} \\ x_{ij}^v=1}} x_{ij} \leq \sum_{(i,j)\in\mathcal{A}} x_{ij}^v - 1, \tag{25}$$

is added whenever an integer feasible solution $\boldsymbol{x}^v$ is found to avoid revisiting it.

We now outline the main steps of the proposed Integer L-shaped algorithm. This algorithm assumes that given the integer feasible solution $\boldsymbol{x}^v$, the value of $Q(\boldsymbol{x}^v)$ can be computed.

Step 0: Set the iteration counter v to 0 and the initial upper bound $\bar{z}$ to $+\infty$. Push the initial current problem (16)-(23) in the list of pending nodes $list_{PN}$.

Step 1: Select a node from list_{PN}. If none exists, stop.

Step 2: Increment v and solve the current problem (CP) to optimality. Let $(\boldsymbol{x}^v, \boldsymbol{\Theta}^v)$ be the optimal solution of CP.

Step 3: If the solution is integer, perform the following substeps:

- ***Step 3.1:*** Check any violation of inequalities (7). If there are any such violated inequalities, generate the associated cuts and add them to CP. Return to *Step 2*.
- ***Step 3.2:*** Check for a new incumbent integer solution. Compute $Q(\boldsymbol{x}^v)$ via expression (15) and set $z^v := c\boldsymbol{x}^v + Q(\boldsymbol{x}^v)$. If $z^v < \bar{z}$, set $\bar{z} := z^v$.
- ***Step 3.3:*** Check the optimality cuts. If $\boldsymbol{\Theta}^v \geq Q(\boldsymbol{x}^v)$, fathom the current node and return to *Step 1*. Otherwise, impose the optimality cut (25) and return to *Step 2*.

Step 4: If the solution is not integer, perform the following substeps:

> ***Step 4.1:*** Check any violation of inequalities (7). If there are any such violated inequalities, generate the associated cuts and add them to CP. (2-path cuts are generated at the root node only.) Return to *Step 2*.
>
> ***Step 4.2:*** Check any violation of inequalities (24). If there are any such violated inequalities, generate the associated cuts and add them to CP. Return to *Step 2*.
>
> ***Step 4.3:*** Generate the *branching* subproblems and append to pending list list_{PN}. Return to *Step 1*.

4.2. Separation of k-path Cuts

The k-path inequalities correspond to an extension of the subtour elimination constraints to VRP variants defined in directed graphs. For a given solution $(\boldsymbol{x}^v, \boldsymbol{w}^v)$ of formulation (16)-(23) let $\mathcal{G}^v = (\mathcal{V}, \mathcal{E})$ be the graph induced by $\boldsymbol{x}^v$, where $\mathcal{V}$ is defined as before and $\mathcal{E} = \left\{(i,j) \in \mathcal{A} : x_{ij}^v > 0\right\}$. Additionally, given a subset $\mathcal{S}$ of $\mathcal{N}$, recall that $\delta(\mathcal{S})$ is the set of arcs in $\mathcal{G}^v$ entering $\mathcal{S}$. We define $x(\mathcal{S})$ as the flow entering $\mathcal{S}$ and $\hat{k}(\mathcal{S})$ as a lower bound for the minimum number of vehicles necessary to service the vertices in $\mathcal{S}$, while respecting time windows and assumptions **A3** and **A4** (i.e., expected demand scenario and at most one failure per route, respectively). We separate 1- and 2-path inequalities only. To separate 1-path inequalities, which coincide with the subtour elimination constraints, we resort to the min-cut problem. The optimal value of this problem is $x(\mathcal{S})$ and if it is strictly less than 1, then the 1-path inequality is violated. In such case, we add the corresponding cut to model (16)-(23).

The procedure for separating 2-path inequalities starts with the generation of all the sets $\mathcal{S}$ such that $x(\mathcal{S}) < 2$, using the greedy heuristic proposed by Kohl et al. (1999). Then, for each set $\mathcal{S}$, the parameter $\hat{k}(\mathcal{S})$ is initially determined as $\max\left\{\left\lceil \sum_{i\in\mathcal{S}} \mathbb{E}[\xi_i]/Q \right\rceil, \left\lceil \sum_{i\in\mathcal{S}} \xi_i^{o_i}/2Q \right\rceil\right\}$. If $\hat{k}(\mathcal{S}) \geq 2$, the 2-path inequality is violated, and therefore, we add the corresponding cut to the formulation. Otherwise, we solve an instance of the traveling salesman problem with time windows (TSPTW) as a feasibility problem. First, a constructive heuristic is used to determine a feasible solution for the TSPTW for vertices in $\mathcal{S}$. This heuristic seeks a solution that visits customers in ascending order of w_i^v, for all $i \in \mathcal{N}$, in the current solution $(\boldsymbol{x}^v, \boldsymbol{w}^v)$. If the resulting solution is feasible, the corresponding 2-path inequality is not violated. Otherwise, we use the labeling algorithm to exactly solve the TSPTW. For this, we represent the TSPTW as a shortest path problem with resource constraints (SPPRC) in which travel costs c_{ij} are conveniently set to -1 for arcs in $\mathcal{A}(\mathcal{S}) = \{(i,j) \in \mathcal{A} : i \in \mathcal{S} \cup \{0\},\ j \in \mathcal{S} \cup \{n+1\}\, i \neq j\}$; otherwise, set to 0. Then, the SPPRC between vertices 0 and $n+1$ is solved. Let $z^\star$ be the optimal value of this SPPRC. If $|z^\star| < |\mathcal{S}|+1$, then the TSPTW is infeasible and clearly more than one vehicle is required, implying that the corresponding 2-path inequality is violated. Thus, we proceed by adding the corresponding cut to the formulation. A similar idea was used by Desaulniers et al. (2008, 2016) and De La Vega et al. (2020) to exactly separate generalized 2-path inequalities. Our implementation of the labeling algorithm is adapted from the framework described by Álvarez and Munari (2017); Munari et al. (2019); De La Vega et al. (2020).

4.3. Separation of LBF Cuts

We now present a three-stage routine to separate the LBF cuts. In the first stage, we detect the partial routes in the graph $\mathcal{G}^v$ induced by the fractional solution $\boldsymbol{x}^v$. In the second, we build the valid functional for all previously detected partial routes and, finally, in the third stage, we calculate a lower bound of $Q(\boldsymbol{x}^v)$.

4.3.1. Detecting partial routes.

A general partial route can be defined as an alternating sequence of sequenced and nonsequenced components. In particular, the sequenced components are called chains. One example of a general partial route is the partial route with topology α defined in Jabali et al. (2014), comprising two chains and one nonsequenced component (see Figure 1). Figure 1 shows that the chains correspond to a group of vertices with a pre-established order, whereas the nonsequenced component is a set of vertices whose visit sequence can be given by any permutation of these vertices in this component.

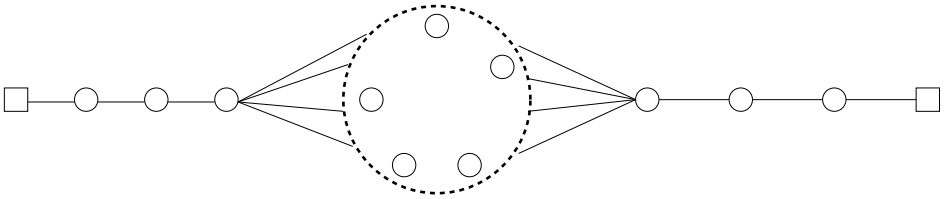

Figure 1: General partial route with topology α.

Let $\mathcal{S}_h = (s_1 = 0, s_2, \ldots, s_{|\mathcal{S}_h|})$ and $\mathcal{T}_h = (t_1, t_2, \ldots, t_{|\mathcal{T}_h|} = n + 1)$ be the sequences of vertices defining the initial and final chains of a given partial route h, respectively, and $\mathcal{U}_h = \{u_1, u_2, \ldots, u_{|\mathcal{U}_h|}\}$ be the set of vertices defining the nonsequenced component of h. In particular, the vertices $s_{|\mathcal{S}_h|}$ and t_1 are called articulation vertices and connect the chains with the nonsequenced component $\mathcal{U}_h$. With the sets $\mathcal{S}_h$, $\mathcal{U}_h$ and $\mathcal{T}_h$, the partial route h can be represented as $\mathcal{S}_h$-$\mathcal{U}_h$-$\mathcal{T}_h$. It is important to highlight that s_1 and $t_{|\mathcal{T}_h|}$ represent the initial and final depots, respectively.

We propose the heuristic procedure presented in Algorithm 1 to detect partial routes with topology α in graph $\mathcal{G}^v$, if they exist. This procedure is based on the one proposed by Laporte et al. (2002) to also detect partial routes following that same topology. Let $\tilde{H}$ be the maximum number of partial routes that can be identified in the graph, determined as the smallest integer greater than or equal to the sum of the flow leaving vertex 0. Additionally, let $\mathcal{P}_h$ be the set of candidate vertices to induce a partial route. For each iteration $h = 1, \ldots, \tilde{H}$, Algorithm 1 creates the set $\mathcal{P}_h$, which is accepted if its vertices induce a partial route that satisfies assumptions **A3** and **A4** (lines 2-12). Set $\mathcal{P}_h$ is initialized using the vertex strongly connected to the depot, and then all the other vertices strongly connected to the vertices in $\mathcal{P}_h$ are iteratively added to this set. Each time a vertex is inserted into $\mathcal{P}_h$, the flow leaving the vertices in $\mathcal{P}_h$ is updated (lines 7-11) and then used as a stopping criterion (line 12) for the construction of this set. Finally, the algorithm creates the chains $\mathcal{S}_h$ and $\mathcal{T}_h$ (lines 15-20 and 21-26, respectively) and the nonsequenced component $\mathcal{U}_h$ (line 27).

This algorithm also detects four different types of partial routes, which are special cases of the partial route with topology α. Figure 2 illustrates these four types. We classify the topology of the partial routes in this figure as α_1, α_2, α_3, and α_4, respectively. The first type is the only one that does not have a nonsequenced component and, thus, it coincides with a full route. The other types of partial routes differ in the number of vertices in the chains. For instance, each chain in the second type has only one vertex, which is the initial depot in the first chain and the final depot in the second.

4.3.2. Building the functional

Let H be the number of partial routes detected by Algorithm 1 in graph $\mathcal{G}^v$ for a given fractional solution $\boldsymbol{x}^v$. In addition, for each partial route h, with $h = 1, \ldots, H$, we define $W_h(x)$ as its functional and $\hat{\Theta}_h$ as its lower bound on the expected recourse cost. The functional $W_h(x)$ is an expression in terms of flow variables

Algorithm 1: Partial route generation algorithm

for $h = 1, \ldots, \tilde{H}$ **do**
  Determine $u = \underset{i \in \mathcal{N}}{\operatorname{argmax}} \left\{ x^v_{0i} + \max_{j \in \mathcal{N}} \left\{ x^v_{ij} \right\} \right\}$;
  Set $\mathcal{P}_h = \{u\}$ and $\mathcal{N} = \mathcal{N} - \{u\}$;
  do
    Determine $u = \underset{j \in \mathcal{N}}{\operatorname{argmax}} \left\{ \sum_{i \in \mathcal{P}_h} \left(x^v_{ij} + x^v_{ji} \right) \right\}$;
    Set $\mathcal{P}_h = \mathcal{P}_h \cup \{u\}$ and $\mathcal{N} = \mathcal{N} - \{u\}$;
    foreach $i \in \mathcal{P}_h$ **do**
      $flow_i = 0$;
      foreach $j \in \mathcal{P}_h$ **do**
        $flow_i = flow_i + x^v_{ij}$;
      $flow_i = flow_i + x^v_{i(n+1)}$;
  while $\sum_{i \in \mathcal{P}_h} flow_i \neq |\mathcal{P}_h|$;
  if $\sum_{j \in \mathcal{P}_h} x^v_{0j} = 1$ *and* $\sum_{i \in \mathcal{P}_h} x^v_{i(n+1)} = 1$ **then**
    if $\sum_{i \in \mathcal{P}_h} \mathbb{E}[\xi_i] \leq Q$ *and* $\sum_{i \in \mathcal{P}_h} \xi_i^{s_i} \leq 2Q$ **then**
      Set $\mathcal{S}_h = \mathcal{S}_h \cup \{0\}$ and $a = 0$;
      do
        Determine $u = \underset{j \in \mathcal{P}_h}{\operatorname{argmax}} \left\{ x^v_{aj} \right\}$ and set $a' = a$;
        if $x^v_{au} = 1$ **then**
          Set $\mathcal{S}_h = \mathcal{S}_h \cup \{u\}$, $\mathcal{P}_h = \mathcal{P}_h - \{u\}$ and $a = u$;
      while $x^v_{a'u} = 1$;
      Set $\mathcal{T}_h = \mathcal{T}_h \cup \{n+1\}$ and $u = n+1$;
      do
        Determine $a = \underset{i \in \mathcal{P}_h}{\operatorname{argmax}} \left\{ x^v_{iu} \right\}$ and set $u' = u$;
        if $x^v_{au} = 1$ **then**
          Set $\mathcal{T}_h = \mathcal{T}_h \cup \{a\}$, $\mathcal{P}_h = \mathcal{P}_h - \{a\}$ and $u = a$;
      while $x^v_{au'} = 1$;
      Set $\mathcal{U}_h = \mathcal{P}_h$;

x_{ij} associated with the arcs of the sequenced components (chains) and arcs that can be in the nonsequenced components of partial route h.

To build the functional $W_h(\boldsymbol{x})$ for any of the addressed topologies, we resort to functions $w^1_h(\boldsymbol{x})$ to $w^5_h(\boldsymbol{x})$ defined below. This functional can also be extended to consider partial routes with two or more nonsequenced components. Sequences $\mathcal{S}_h$ and $\mathcal{T}_h$ and set $\mathcal{U}_h$ are defined as before. To build the functional, we follow the same idea as in Jabali et al. (2014) of multiplying by 3 the flow on arcs (i, j) when $i \neq 0$ and $j \neq n+1$. For simplicity, we write $(s_i, s_j) \in \mathcal{S}_h$ if s_i and s_j are consecutive vertices in $\mathcal{S}_h$, and we use a similar reasoning for $(t_i, t_j) \in \mathcal{T}_h$. In all the addressed topologies, we have at least one vertex in $\mathcal{S}_h$ but $\mathcal{T}_h$ and $\mathcal{U}_h$ can be empty in some of them. For example, for topology α_1 (a full route) all vertices are in the chain $\mathcal{S}_h$, including 0 and $n+1$, and thus $|\mathcal{T}_h| = |\mathcal{U}_h| = 0$. Note that we have $|\mathcal{U}_h| = 0$ if, and only if, $|\mathcal{S}_h| > 1$.

Given a partial route h and its corresponding chain $\mathcal{S}_h$, we define:

$$w^1_h(\boldsymbol{x}) = \begin{cases} 0, & \text{if } |\mathcal{S}_h| = 1, \\ x_{0s_2} + \sum\limits_{\substack{(s_i, s_j) \in \mathcal{S}_h: \\ s_i \neq 0, s_j \neq n+1}} 3x_{s_i s_j} - (1 + 3(|\mathcal{S}_h| - 2)), & \text{otherwise.} \end{cases} \tag{26}$$

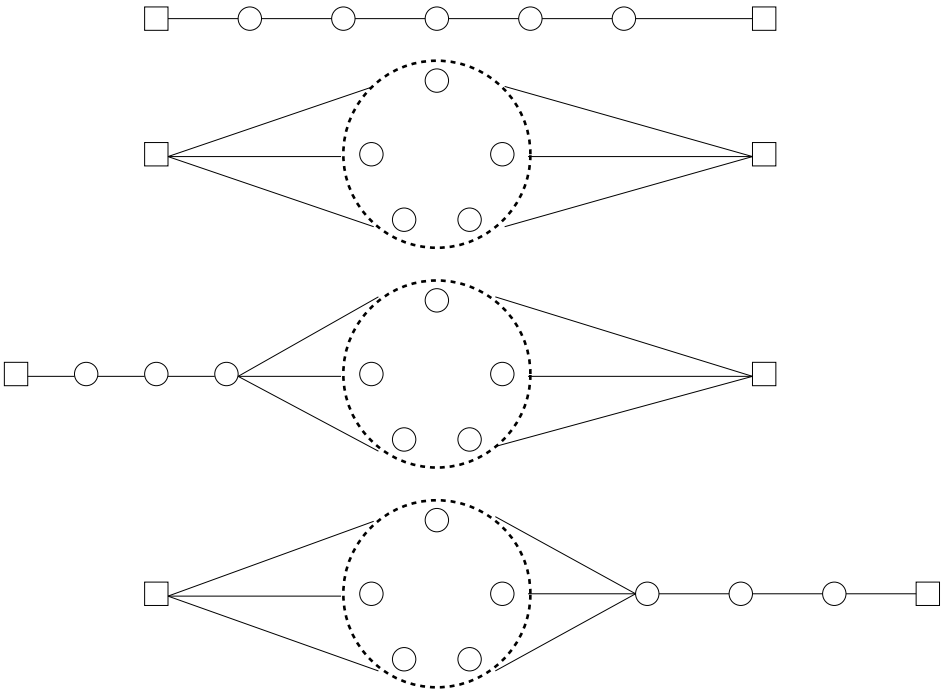

Figure 2: Partial routes with topologies α_1, α_2, α_3 and α_4, respectively.

This definition implies that for solution $\boldsymbol{x}^v$ we have $w_h^1(\boldsymbol{x}^v) = 0$, for any of the addressed topologies. Similarly, for chain $\mathcal{T}_h$, we define:

$$w_h^2(\boldsymbol{x}) = \begin{cases} 0, & \text{if } |\mathcal{T}_h| \leq 1, \\ \sum\limits_{\substack{(t_i,t_j)\in\mathcal{T}_h:\\ t_j\neq n+1}} 3x_{t_it_j} + x_{t_{(|\mathcal{T}_h|-1)}(n+1)} - (3(|\mathcal{T}_h|-2)+1), & \text{otherwise,} \end{cases} \tag{27}$$

and thus we have $w_h^2(\boldsymbol{x}^v) = 0$ for the addressed topologies. For component $\mathcal{U}_h$, we define:

$$w_h^3(\boldsymbol{x}) = \begin{cases} 0, & \text{if } |\mathcal{U}_h| \leq 1, \\ \sum\limits_{\substack{u_i,u_j\in\mathcal{U}_h\\ u_i\neq u_j}} 3x_{u_iu_j} - 3(|\mathcal{U}_h|-1), & \text{otherwise.} \end{cases} \tag{28}$$

Finally, for the articulation vertices between $\mathcal{S}_h$ and $\mathcal{U}_h$ and between $\mathcal{U}_h$ and $\mathcal{T}_h$, we have the following definitions, respectively:

$$w_h^4(\boldsymbol{x}) = \begin{cases} x_{s_{(|\mathcal{S}_h|-1)}n+1} - 1, & \text{if } |\mathcal{U}_h| = 0, \\ \sum\limits_{u_j\in\mathcal{U}_h} x_{0u_j} - 1, & \text{if } |\mathcal{S}_h| = 1, \\ \sum\limits_{u_j\in\mathcal{U}_h} 3x_{s_{|\mathcal{S}_h|}u_j} - 3, & \text{otherwise.} \end{cases} \tag{29}$$

$$w_h^5(\boldsymbol{x}) = \begin{cases} 0, & \text{if } |\mathcal{U}_h| = 0, \\ \sum\limits_{u_j\in\mathcal{U}_h} x_{u_j(n+1)} - 1, & \text{if } |\mathcal{T}_h| = 1, \\ \sum\limits_{u_i\in\mathcal{U}_h} 3x_{u_it_1} - 3, & \text{otherwise.} \end{cases} \tag{30}$$

Having defined functions $w_h^1(\boldsymbol{x})$ to $w_h^5(\boldsymbol{x})$, we build the functional $W_h(\boldsymbol{x})$ as follows:

$$W_h(\boldsymbol{x}) = \sum_{k=1}^{5} w_h^k(\boldsymbol{x}) + 1. \tag{31}$$

Proposition 4.1. *Let $\boldsymbol{x}'$ be a solution satisfying constraints (17)–(21). Constraint (24) with*

$$\hat{\Theta} = \sum_{h=1}^{H} \hat{\Theta}_h \quad \textit{and} \quad \tilde{W}(\boldsymbol{x}') = \sum_{h=1}^{H} W_h(\boldsymbol{x}') - H + 1$$

is a valid inequality for the VRPTWSD.

Proof. Let $\mathcal{X}_h$ be the set of solutions inducing a given partial route h of $\boldsymbol{x}'$. Moreover, let $\boldsymbol{x}'' \notin \mathcal{X}_h$ be a feasible solution regarding (17)-(21) and consider a partial route h'' induced by this solution. Since $\boldsymbol{x}'' \notin \mathcal{X}_h$, there exists at least one arc in h'' that is not in h and hence $W_h(\boldsymbol{x}'') \leq 0$. Therefore, $W_h(\boldsymbol{x}) = 1$ if $\boldsymbol{x} \in \mathcal{X}_h$ and $W_h(\boldsymbol{x}) \leq 0$ otherwise. Then, $W_h(\boldsymbol{x}) \leq 1$ and the expression $\sum_{h=1}^{H} W_h(\boldsymbol{x}) - H + 1$ assumes at most the value 1, which is reached when $W_h(\boldsymbol{x}) = 1$ for all $h = 1, \ldots, H$. In this case, constraint (24) becomes $\Theta \geq \hat{\Theta}$. Otherwise, constraint (24) is redundant. □

4.4. Bounding the Recourse Cost

We now describe a procedure to determine a lower bound on the expected recourse cost for a partial route following topology α. The procedure is also valid for the other topologies of partial routes, including partial routes with two or more nonsequenced components. Let h be a partial route represented as $(s_1 = 0, s_2, \ldots, s_{|\mathcal{S}_h|}, \mathcal{U}_h, t_1, t_2, \ldots, t_{|\mathcal{T}_h|} = n + 1)$, and p be the number of vertices in nonsequenced component $\mathcal{U}_h$ (i.e., $p = |\mathcal{U}_h|$). Thus, another possible representation of partial route h is:

$$h = \left(s_1 = 0, s_2, \ldots, s_{(j-p)}, \{u_{(j-p+1)}, u_{(j-p+2)}, \ldots, u_j\}, t_{(j+1)}, t_{(j+2)}, \ldots, t_{(k+1)} = n + 1\right). \tag{32}$$

where $k + 1 = |\mathcal{S}_h| + |\mathcal{U}_h| + |\mathcal{T}_h| = |\mathcal{P}_h|$ and the articulation vertices $s_{|\mathcal{S}_h|}$ and t_1 are now denoted by $s_{(j-p)}$ and $t_{(j+1)}$, respectively. Since the actual vertices in positions $j - p + 1, j - p + 2, \ldots, j$ of the partial route are unknown, we define an artificial partial route $\tilde{h}$ as follows:

$$\tilde{h} = \left(s_1 = 0, s_2, \ldots, s_{(j-p)}, []_{(j-p+1)}, []_{(j-p+2)}, \ldots, []_j, t_{(j+1)}, t_{(j+2)}, \ldots, t_{(k+1)} = n + 1\right), \tag{33}$$

where $[]_{(j-p+1)}, []_{(j-p+2)}, \ldots, []_j$ represent any permutation of the vertices in $\mathcal{U}_h$ such that the resulting sequence is feasible regarding the time windows. In what follows, we use ℓ to denote the ℓth position of artificial partial route $\tilde{h}$. Given a solution x^v, the procedure for computing a valid lower bound for $Q(x^v)$ starts by estimating the lower bounds on time-related parameters, i.e., arrival times $\hat{w}_\ell$ and service times $\hat{s}_\ell$ as well as estimated time windows $[\hat{a}_\ell, \hat{b}_\ell]$ for each position ℓ of $\tilde{h}$. Moreover, it also determines the lower bounds for the travel times in the arcs traversed in $\tilde{h}$.

4.4.1. Building lower bounds on time-related parameters

The estimated time windows for the positions of the previously sequenced portions of $\tilde{h}$ are the actual time windows of the vertices corresponding to those positions. To determine the new time windows of the positions of the nonsequenced portion of $\tilde{h}$, we first proceed to determine lower bounds for their opening times and then upper bounds for their closing times. Let $(a_{[1]}, a_{[2]}, \ldots, a_{[p]})$ be the opening times of the time

windows of the vertices in $\mathcal{U}_h$ sorted in nondescending order. We attribute the value $a_{[1]}$ to the $(j-p+1)$-th position, $a_{[2]}$ to the $(j-p+2)$-th position and so on, until reaching the last nonsequenced position j. The closing time of the estimated time window is overestimated for these positions as $\hat{b}_{(j-p+1)} = \hat{b}_{(j-p+2)} = \ldots = \hat{b}_j = \max_{u \in \mathcal{U}_h}\{b_u\}$.

Due to the estimated time windows, some vertices in $\mathcal{U}_h$ cannot be allocated in certain positions of the nonsequenced portion. Let $\mathcal{C}_h(\ell)$ be the set containing the vertices that can be allocated to position ℓ of $\tilde{h}$, for $\ell = 1, \ldots, k+1$. These sets, for each position ℓ, can be found as follows:

$$\mathcal{C}_h(\ell) = \begin{cases} \{s_\ell\}, & \text{if } \ell = 1, \ldots, j-p. \\ \{u \in \mathcal{U}_h : b_u \geq \hat{a}_\ell\}, & \text{if } \ell = j-p+1, \ldots, j. \\ \{t_\ell\}, & \text{if } \ell = j+1, \ldots, k+1. \end{cases} \tag{34}$$

Note from expression (34) that, for $\ell = j-p+1, \ldots, j$, vertex $u \in \mathcal{U}_h$ belongs to set $\mathcal{C}_h(\ell)$ if and only if inequality $b_u \geq \hat{a}_\ell$ holds. In the other positions, sets $\mathcal{C}_h(\ell)$ are of unit cardinality, whose only element is associated with the vertex (customer or depot) allocated in the corresponding position. Using sets $\mathcal{C}_h(\ell)$, we can now estimate a lower bound for the travel time in each arc of the artificial partial route $\tilde{h}$ as follows:

$$\hat{t}_{\ell(\ell+1)} = \min_{\substack{u \in \mathcal{C}_h(\ell),\, u' \in \mathcal{C}_h(\ell+1) \\ u \neq u'}} \{t_{uu'}\} \quad \text{for } \ell = 1, \ldots, k. \tag{35}$$

A lower bound for the service time in each of the positions of $\tilde{h}$ is determined as $\hat{s}_\ell = \min_{u \in \mathcal{C}_h(\ell)} \{s_u\}$. Finally, lower bounds for the arrival times are calculated as follows:

$$\hat{w}_\ell = \begin{cases} \hat{a}_\ell, & \text{if } \ell = j-p+1, \ldots, j. \\ \max\{\hat{a}_{(\ell-1)}, \hat{w}_{(\ell-1)}\} + \hat{s}_{(\ell-1)} + \hat{t}_{(\ell-1),\ell}, & \text{otherwise.} \end{cases} \tag{36}$$

for $\ell = 2, \ldots, k+1$, with the initial condition $\hat{w}_1 = 0$.

4.4.2. Building the lower bound on recourse cost

After applying a recourse action in the ℓth position of the artificial partial route $\tilde{h}$, it is necessary to identify the positions $\ell, \ell+1, \ldots, k+1$ in which the estimated time windows are violated. Similar to the definitions introduced in Section 3.2, let $\widehat{\mathcal{B}}^1_\ell(q, \xi^\omega_u)|_{\ell \to u}$ and $\widehat{\mathcal{B}}^2_\ell(q, \xi^\omega_u)|_{\ell \to u}$ be the sets of positions of vertices with estimated time windows that are violated after a BF and restocking trip, respectively. In both sets, we assume that the vehicle arrives with load q at a vertex u in $\mathcal{C}_h(\ell)$ allocated in the ℓth position of $\tilde{h}$. Additionally, note that the components in these sets are positions at the artificial partial route $\tilde{h}$, instead of vertex indices. Propositions 4.2 and 4.3 show how to determine new lower bounds for the arrival times and how to construct these sets.

Proposition 4.2. *Let $\hat{w}_m$ be a lower bound for the exact time at which the vehicle arrives in the vertex in the mth position of artificial partial route $\tilde{h}$, with an estimated time window $[\hat{a}_m, \hat{b}_m]$. Let $\hat{s}_m$ be a lower bound for the service time and $\tilde{w}_m(\ell \to u)$ be a lower bound for the new arrival time in this same vertex after applying the recourse action at vertex $u \in \mathcal{C}_h(\ell)$ in the ℓth position. Then, $\tilde{w}_m(\ell \to u)$ can be computed as follows:*

$$\tilde{w}_m(\ell \to u) = \begin{cases} \hat{w}_m, & \text{if } m = 2, \ldots, \ell. \\ \hat{\zeta}_m(\ell), & \text{if } m = \ell+1. \\ \hat{\tau}_m(\ell), & \text{if } m = \ell+2, \ldots, k+1. \end{cases} \tag{37}$$

with initial conditions $\hat{w}_m$ given by (36). $\hat{\zeta}_m(\ell)$ can be determined as:

$$\hat{\zeta}_m(\ell) = \begin{cases} \max\{\hat{a}_{(m-1)}, \tilde{w}_{(m-1)}(\ell \to u)\} + s_u + \tilde{t}_{um} + f_u, & \text{if } \tilde{w}_{(m-1)}(\ell \to u) + f_u \le b_u, \\ \tilde{w}_{(m-1)}(\ell \to u) + \tilde{t}_{um} + f_u, & \text{otherwise}, \end{cases} \tag{38}$$

for a BF trip, where $\tilde{t}_{um} = \min_{\substack{u' \in \mathscr{C}_h(m) \\ u \neq u'}} \{t_{uu'}\}$ and $f_u = t_{u(n+1)} + t_{0u}$, whereas for a restocking trip, we have:

$$\hat{\zeta}_m(\ell) = \tilde{w}_{(m-1)}(\ell \to u) + s_u + t_{u(n+1)} + \tilde{t}_{0m}, \tag{39}$$

where $\tilde{t}_{0m} = \min_{u' \in \mathscr{C}_h(m)} \{t_{0u'}\}$. Finally,

$$\hat{\tau}_m(\ell) = \begin{cases} \max\{\hat{a}_{(m-1)}, \tilde{w}_{(m-1)}(\ell \to u)\} + \widehat{s}_{(m-1)} + \hat{t}_{(m-1)m}, & \text{if } \tilde{w}_{(m-1)}(\ell \to u) \le \hat{b}_{(m-1)}, \\ \tilde{w}_{(m-1)}(\ell \to u) + \hat{t}_{(m-1)m}, & \text{otherwise}. \end{cases} \tag{40}$$

Proof. See Appendix B of the e-companion.

Proposition 4.3. *For a given load $q \in [0, Q]$, sets $\widehat{\mathscr{B}}^1_\ell(q, \xi^\omega_u)|_{\ell \to u}$ and $\widehat{\mathscr{B}}^2_\ell(q, , \xi^\omega_u)|_{\ell \to u}$ containing positions of vertices with estimated time windows that would be violated after a BF and restocking trip, respectively, can be constructed as follows:*

$$\widehat{\mathscr{B}}^1_\ell(q, \xi^\omega_u)|_{\ell \to u} = \left\{ m \mid \tilde{w}_m(\ell \to u) > \hat{b}_m \text{ and } q - \xi^\omega_u < 0,\ m = \ell, \ell+1, \ldots, k+1 \right\}, \tag{41}$$

$$\hat{\mathscr{B}}^2_\ell(q, \xi^\omega_u)|_{\ell \to u} = \left\{ m \mid \tilde{w}_m(\ell \to u) > \hat{b}_m \text{ and } 0 \le q - \xi^\omega_u < \underline{\theta}_{(\ell+1)},\ m = \ell+1, \ldots, k+1 \right\}, \tag{42}$$

where $\underline{\theta}_{(\ell+1)}$ is the minimum value of the fixed rule-based policy of the vertices in $\mathscr{C}_h(\ell+1)$, i.e., $\underline{\theta}_{(\ell+1)} = \min_{u' \in \mathscr{C}_h(\ell+1)} \{\theta_{u'}\}$.

Proof. See Appendix C of the e-companion.

With the presented definitions, we can now determine $\tilde{F}_\ell(q)$, a lower bound on the expected recourse cost at the ℓth position of $\tilde{h}$ given that the vehicle arrives at this position with load q, such that $\overline{\theta}_{(\ell-1)} \le q \le Q$ and $\overline{\theta}_{(\ell-1)} = \max_{u \in \mathscr{C}_h(\ell-1)} \{\mathbb{E}[\xi_u]\}$. Additionally, $\hat{F}_\ell(q)|_{\ell \to u}$ is a lower bound on the expected recourse cost for the specific vertex u in $\mathscr{C}_h(\ell)$ assigned to the ℓth position of $\tilde{h}$, given that the vehicle arrives at this position with load q. $\hat{F}_\ell(q)|_{\ell \to u}$ is determined in terms of $\tilde{F}_\ell(q)$ as follows:

$$\hat{F}_\ell(q)|_{\ell \to u} = \begin{cases} \tilde{F}_{\ell+1}(q), \quad \text{if } \ell = 1, \\ \displaystyle\sum_{\substack{\omega=1,\ldots,o_u: \\ q - \xi^\omega_u < 0}} p^\omega_u \left[\Gamma + 2c_{0u} + \Phi \sum_{m \in \widehat{\mathscr{B}}^1_\ell(q, \xi^\omega_u)|_{\ell \to u}} \tilde{d}_{0m} + \tilde{F}_{(\ell+1)}(Q + q - \xi^\omega_u) \right] + \\ \displaystyle\sum_{\substack{\omega=1,\ldots,o_u: \\ 0 \le q - \xi^\omega_u < \underline{\theta}_\ell}} p^\omega_u \left[\tilde{c}_u + \Phi \sum_{m \in \hat{\mathscr{B}}^2_\ell(q, \xi^\omega_u)|_{\ell \to u}} \tilde{d}_{0m} + \tilde{F}_{(\ell+1)}(Q) \right] + \\ \displaystyle\sum_{\substack{\omega=1,\ldots,o_u: \\ q - \xi^\omega_u \ge \overline{\theta}_\ell}} p^\omega_{v_i} \left[\tilde{F}_{(\ell+1)}\left(q - \xi^\omega_u\right) \right], \quad \text{if } \ell = 2, \ldots, k, \\ 0, \quad \text{if } \ell = k+1, \end{cases} \tag{43}$$

where

$$\tilde{d}_{0m} = 2\Gamma \text{ if } m = k+1 \text{ and } \tilde{d}_{0m} = \min_{u' \in \mathscr{C}_h(m)} \{c_{0u'} + c_{u'(n+1)}\} \text{ otherwise,}$$

$$\tilde{c}_u = \min_{u' \in \mathscr{C}_h(m+1)} \{c_{0u} + c_{0u'} - c_{uu'}\}, \; \underline{\theta}_\ell = \min_{u' \in \mathscr{C}_h(\ell+1)} \left\{\mathbb{E}\left[\xi_{v_{u'}}\right]\right\}, \quad \text{and} \quad \overline{\theta}_\ell = \max_{u' \in \mathscr{C}_h(\ell+1)} \left\{\mathbb{E}\left[\xi_{v_{u'}}\right]\right\}.$$

From (43), we have that $\hat{F}_\ell(q)|_{\ell \to u} \leq F_u(q)$ for all q, because $\hat{F}_\ell(q)|_{\ell \to u}$ is determined in terms of the lower bound $\tilde{F}_\ell(q)$, while $F_u(q)$ is the exact recourse cost given by (15). For each position of $\tilde{h}$, $\tilde{F}_\ell(q)$ can be determined as follows. In the sequenced portion of $\tilde{h}$ given by the vertices in $\mathscr{T}_h$, $\tilde{F}_\ell(q)$ coincides with the exact recourse cost given by (15), for $\ell = j+1, j+2, \ldots, k+1$. Then, for position j of $\tilde{h}$, we have:

$$\tilde{F}_j(q) = \min_{u \in \mathscr{C}_h(j)} F_j(q)|_{j \to u} \quad \forall q \in \left[\overline{\theta}_{(j-1)}, Q\right], \tag{44}$$

where $F_j(q)|_{j \to u}$ is computed by allocating vertex $u \in \mathscr{C}_h(j)$ at position j and then applying the exact recourse function (15). Finally, we use function $\hat{F}_\ell(q)|_{\ell \to u}$ in (43) inside expression (44) to compute $\tilde{F}_\ell(q)$ for positions $2, \ldots, j-1$. Thus, function $\tilde{F}_\ell(q)$ for $\ell = j-1, \ldots, 2$ can now be stated as follows:

$$\tilde{F}_\ell(q) = \min_{u \in \mathscr{C}_h(\ell)} \hat{F}_\ell(q)|_{\ell \to u} \quad \forall q \in \left[\overline{\theta}_{(\ell-1)}, Q\right]. \tag{45}$$

Expression (45) is first used to determine $\tilde{F}_{(j-1)}(q)$ and then is sequentially used for the remaining positions $\ell = j-2, \ldots, 2$. Observe that (45) is also used to find a lower bound for the sequenced positions referring to the vertices of $\mathcal{S}_h$. This is because for the positions $\ell = 2, \ldots, j-p$, we must compute the cost related to the penalties due to a time window failure at vertex s_ℓ. Given that there are nonsequenced positions with unknown vertices, this cost must be estimated. Finally, $\hat{\Theta}_h = \tilde{F}_1(Q)$ can be used as a valid lower bound on the expected recourse cost of the artificial route $\tilde{h}$. If H partial routes are detected in the current solution, then $\hat{\Theta} = \sum_{h=1}^{H} \hat{\Theta}_h$ corresponds to a lower bound of $Q(\boldsymbol{x}^v)$.

5. Computational Results

In this section, we report the results of computational experiments performed with the following objectives: 1) analyze the computational performance of the proposed Integer L-shaped algorithm and verify the impact of k-path and LBF inequalities; and 2) quantify the savings in the total operating cost brought about by the explicit consideration of stochastic demands and fixed rule-based policy in relation to the expected value strategy and the classical recourse policy. The experiments were performed using the modified instances introduced by Lei et al. (2011), which were based on the instance classes 1 and 2 proposed by Solomon (1987). Class 1 consists of subclasses **R1**, **C1** and **RC1**, while class 2 consists of subclasses **R2**, **C2** and **RC2**. The geographic distribution of customers in Solomon's instances varies according to the instance subclass: a clustered distribution (subclasses **C1** and **C2**), a random distribution (subclasses **R1** and **R2**), and a combination of clustered and random distribution (subclasses **RC1** and **RC2**). The following values were used as vehicle capacities: 80 in subclasses **C1** and **C2**, 50 in subclasses **R1** and **R2**, and 100 in subclasses **RC1** and **RC2**. The numbers of vehicles and customers varied from one instance to another, as shown in Tables 1 and 2. The parameters related to the transport cost c_{ij} and to the travel time t_{ij} for each arc $(i, j) \in \mathcal{A}$ correspond to the Euclidean distance, determined in terms of the coordinates given in the instances.

We assumed that customer demands follow a triangular probability distribution with the same number of realizations S (i.e., $s_1 = s_2 = \ldots = s_n = S$) and expected values equal to the demands defined in Solomon's instances. As in Louveaux and Salazar-González (2018), we used $S = 3$ and $S = 9$, and the value of the sth realization of the random demand ξ_i was determined as $\xi_i^s = \mathbb{E}[\xi_i] - \lfloor S/2 \rfloor + s - 1$, for $s = 1, \ldots, S$.

	C1									R1												RC1							
	01	02	03	04	05	06	07	08	09	01	02	03	04	05	06	07	08	09	10	11	12	01	02	03	04	05	06	07	08
n	19	24	17	22	29	15	11	21	34	24	29	19	14	11	35	15	21	19	31	44	49	21	30	16	19	22	39	9	14
m	8	9	6	9	12	6	5	9	12	9	11	7	5	5	13	5	8	8	11	14	16	8	11	6	7	8	13	4	5
Q	75	75	75	75	75	75	75	75	75	50	50	50	50	50	50	50	50	50	50	50	50	100	100	100	100	100	100	100	100

Table 1: Values of parameters n, m and Q for each of the instances in class 1.

	C2								R2											RC2							
	01	02	03	04	05	06	07	08	01	02	03	04	05	06	07	08	09	10	11	01	02	03	04	05	06	07	08
n	19	21	17	24	30	14	10	39	29	24	16	11	32	15	23	19	31	41	49	16	12	20	24	10	11	34	44
m	8	8	6	9	11	6	5	13	11	9	6	5	12	6	8	7	11	13	17	6	5	8	9	5	5	11	14
Q	75	75	75	75	75	75	75	75	50	50	50	50	50	50	50	50	50	50	50	100	100	100	100	100	100	100	100

Table 2: Values of parameters n, m and Q for each of the instances in class 2.

If $s < \lceil S/2 \rceil$, the probability of occurrence is $p_i^s = \mathbb{P}[\xi_i = \xi_i^s] = s/\lceil S/2 \rceil^2$; otherwise, this probability is $p_i^s = \mathbb{P}[\xi_i = \xi_i^s] = (S - s + 1)/\lceil S/2 \rceil^2$. Additionally, we set $\Phi = 2$ (penalty that incurs in single-customer trips) and $\Gamma = \sum_{i \in \mathcal{N}} c_{0i}/n$ (penalty for interruptions in the service of a customer vertex due to a BF trip). The general lower bound on the expected recourse cost L was set to 0. In total, 112 instances were used in the experiments. For each value of S, we have 9 instances in subclass **C1**, 12 in **R1**, 8 in **RC1**, **C2**, and **RC2**, and 11 in **R2**.

When Algorithm 1 identifies a partial route coinciding with a full route and its number of demand vertices is four or more, the route is transformed into a partial route with topology α, considering $\mathcal{S}_h$ as the $\lfloor 0.25 * (|\mathcal{P}_h|-2) \rfloor + 1$ first vertices of $|\mathcal{P}_h|$, $\mathcal{U}_h$ as the next $\lceil 0.5 * (|\mathcal{P}_h|-2) \rceil$ vertices of $|\mathcal{P}_h|$ and $\mathcal{T}_h$ as the last $|\mathcal{P}_h|-|\mathcal{S}_h|-|\mathcal{U}_h|$ vertices of $\mathcal{P}_h$. This is because preliminary computational experiments have shown a better performance of the algorithm if partial routes with topology α are used instead of full routes.

The Integer L-shaped algorithm was coded in C++ using the Concert library of the IBM CPLEX Optimization Studio v.12.8. For separating k-path inequalities, we used the push-relabel flow algorithm (Goldberg and Tarjan, 1988) available in the Concorde library (Cook, 2015). The experiments were performed on a Linux PC with 16 GB of RAM and an Intel Core i7-4790 3.6 GHz CPU. The stopping criteria for the Integer L-shaped algorithm were reaching a relative optimality gap below 0.01% or a time limit of 18,000 seconds (five hours). An integer feasible solution with a relative optimality gap less than 0.01% is assumed optimal.

5.1. Computational performance and impact of k-path and LBF Cuts

In this subsection, we analyze the performance of the proposed Integer L-shaped method and the impact of incorporating k-path and LBF cuts into the method. The instances were solved by three variants of the method: without k-path and LBF cuts (**LS1**), with k-path cuts only (**LS2**) and with both k-path and LBF cuts (**LS3**).

To compare the quality of the solutions obtained by the three approaches, we used the performance profiles introduced by Dolan and More (2002). Generally, a performance profile of a method can be defined as the cumulative distribution function for a given performance metric. We used the objective function value as metric and hence, for each method, the performance profile shows the percentage of instances for which the objective value of the solution obtained by this method is within 2^τ times ($\tau \geq 0$) the best objective value obtained by the three methods. Figure 3 shows the performance profiles of **LS1** to **LS3**. For a method

$t \in \{\mathbf{LS1}, \mathbf{LS2}, \mathbf{LS3}\}$ and an instance p in the set of instances, let Obj_{pt} be the objective value obtained by method t for instance p. The performance profile of t is then the percentage of instances p in the set such that $Obj_{pt} \leq 2^{\tau} \times \min_{t' \neq t}\{Obj_{pt'}\}$ for a given τ. If a method did not find a feasible solution for an instance within the time limit, it is said that this method failed to solve this instance. Extreme values of the performance profiles for each method are shown in Table 3, where r_M is the extreme value of the abscissa axis in Figure 3 (see Dolan and More (2002) for a detailed explanation of the performance profiles).

From the performance profiles, we observe that **LS3** obtained the best performance in relation to the other algorithms in 90% of the instances ($\tau = 0$) (see also Table 3). Since the curve associated with **LS3** is above the others throughout interval $[0, r_M]$, this method dominates the others, in the sense that it solved more problems within a factor 2^{τ} of the performance of any other algorithm. Similarly, **LS2** clearly dominates **LS1**. Therefore, even though none of the methods found feasible solutions to all instances (their curves did not reach the value 1 for $\tau \in [0, r_M]$), **LS3** is the most likely to succeed, as it obtained the highest performance profile value (96%) at $\tau \to r_M$, in addition to having achieved the best overall performance.

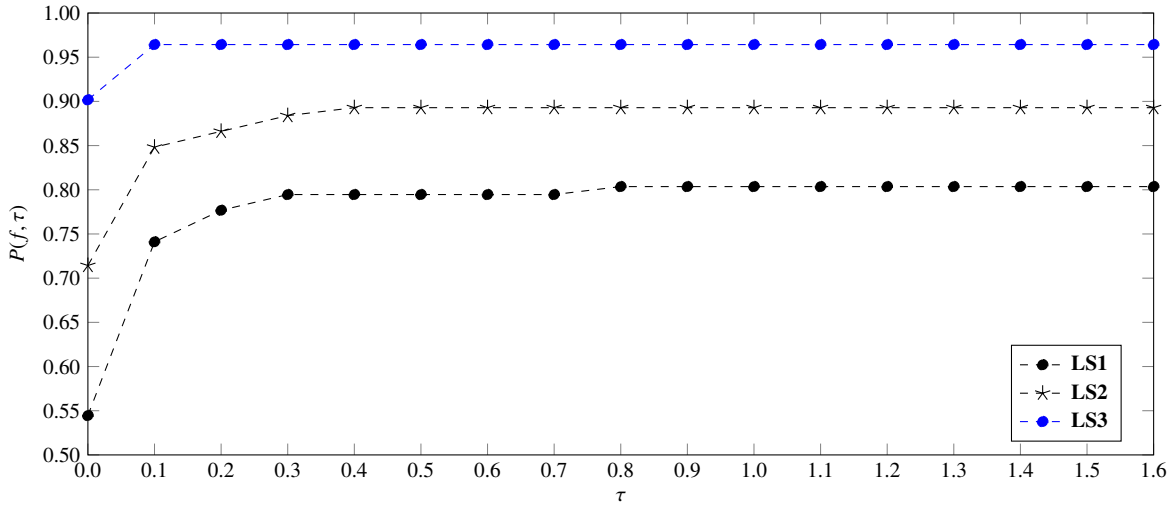

Figure 3: Performance profile on the 112 instances of the three solution algorithms in relation to the objective function value metric.

Algorithm	$\tau = 0$	$\tau \to r_M$
LS1	0.55	0.80
LS2	0.71	0.89
LS3	0.90	0.96

Table 3: Extreme values of the performance profiles for each algorithm based on the objective function value metric.

Figure 4 shows bar charts that summarize the main results of methods **LS1** to **LS3**, regarding the number of instances solved to optimality (Nopt); the average computing time in seconds (Time) considering only instances solved to optimality by the three methods; and the number of instances in which the corresponding method failed to find a feasible solution within the time limit (Nfail). According to these graphs, **LS3** was the method with most instances solved to optimality (74 out of 112), representing approximately 66% of the total number of instances. This algorithm was also the fastest, on average, taking less than 40 seconds to solve to optimality the instances that were solved by the three methods. Compared to the average time of the other methods, **LS3** was 31 times faster than **LS1** and 11 times faster than **LS2**. Moreover, it failed to find feasible solutions in four instances only, two of subclass **R1** and two of subclass R2, while **LS1** and **LS2** failed in 19 and 12, respectively. **LS2** outperformed **LS1**, as it solved to optimality 9 more instances

than **LS1**. Additionally, it took less time on average, being three times faster than **LS1** to prove optimality.

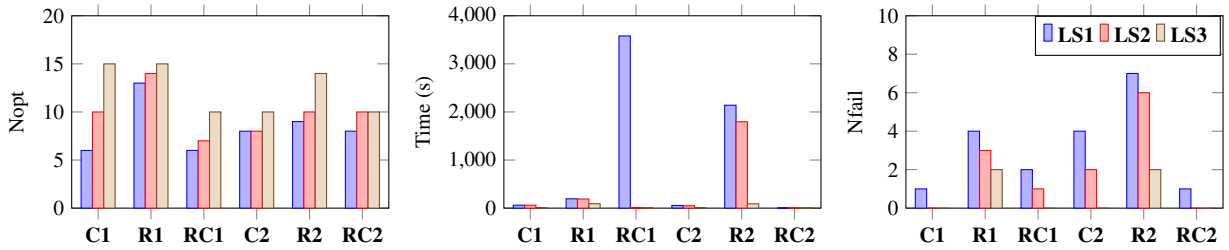

Figure 4: Bar charts summarizing the main results of the three algorithms.

Therefore, the results summarized in Figures 3 and 4 reveal that for the considered instances, **LS3** obtained the best performance regarding both solution quality and computing time, showing the importance of incorporating k-path and LBF cuts in the L-shaped algorithm.

5.2. Comparison of Deterministic and Stochastic Solutions

In this subsection, we compare the solutions of the deterministic and stochastic problems in terms of total expected cost and computing time, considering the fixed rule-based and classical policies. The stochastic problem was solved using **LS3** for both policies, and the expected recourse cost for the classical policy was computed via function (15) with $\theta_i = 1$, for all $i \in \mathcal{N}$. The deterministic problem was solved by a variant of the proposed L-shaped method that does not resort to LBF inequalities (24) and optimality cuts (25), referred to as **LS0**. It is worth mentioning that we also tested a straightforward approach for solving the deterministic problem, which consists of using the general-purpose branch-and-cut (BC) method of CPLEX to solve an equivalent flow formulation defined using Miller-Tucker-Zemlin (MTZ) constraints to impose time windows, vehicle capacity and assumption **A3**. Compared to the CPLEX BC approach, the **LS0** algorithm increased the number of instances solved to optimality from 25 to 45. In addition, for instances solved to optimality by both **LS0** and CPLEX BC, the former was ≈ 2.6 times faster. These results revealed the superiority in both solution quality and computational effort of the **LS0** algorithm over the CPLEX BC algorithm to solve the deterministic problem, and for this reason, we report the results obtained by **LS0** only.

Table 4 presents the average results obtained for the following problems: deterministic, stochastic with fixed rule-based policy, and stochastic with classical policy. These results are grouped by instance subclass and number of demand realizations (S), as specified in the first two columns of the table, and the averages include only instances that were solved to optimality for the three problems. For the deterministic problem, C denotes the cost of the optimal routes obtained by **LS0**. Then, for these routes, we computed the expected recourse cost related to the fixed rule-based and classical policies, denoted in the table as Q_r^d and Q_c^d, respectively. The table also shows the expected total cost for these two policies, denoted as $\mathcal{Z}_r^d$ and $\mathcal{Z}_c^d$, respectively. For the stochastic approaches, C denotes the cost of the first-stage routes, and the table also presents the expected recourse cost and expected total cost of the corresponding recourse policy, using similar headers as in the deterministic approach but with the superscript s. Finally, the last four columns

present comparisons between the approaches based on savings computed as follows:

$$\begin{aligned} \text{Sav}_1 &= (\mathcal{Z}_r^d/\mathcal{Z}_r^s - 1) * 100, \\ \text{Sav}_2 &= (\mathcal{Z}_c^d/\mathcal{Z}_c^s - 1) * 100, \\ \text{Sav}_3 &= (\mathcal{Z}_c^s/\mathcal{Z}_r^s - 1) * 100, \\ \text{Sav}_4 &= (\mathcal{Z}_c^d/\mathcal{Z}_r^d - 1) * 100, \end{aligned}$$

considering only instances that were solved to optimality for the three problems. Hence, these savings represent the relative reductions in the expected total cost given by solving the stochastic problem with fixed rule-based policy instead of applying the fixed rule-based policy in the deterministic solution (Sav_1), solving the stochastic problem with classical policy instead of applying the classical policy in the deterministic solution (Sav_2), solving the stochastic problem with fixed rule-based policy instead of the stochastic problem with the classical policy (Sav_3), and applying the fixed rule-based policy in the deterministic solution instead of applying the classical policy in the deterministic solution (Sav_4). As already mentioned, the results in Table 4 are grouped by instance subclasses and S and include only instances that were solved to optimality with the three problems. For detailed results regarding each instance, see Tables D.15-D.18 in Appendix D of the e-companion.

		Deterministic					Stochastic Fixed rule-based policy			Stochastic Classical policy						
Subclass	S	C	Q_r^d	$\mathcal{Z}_r^d$	Q_c^d	$\mathcal{Z}_c^d$	C	Q_r^s	$\mathcal{Z}_r^s$	C	Q_c^s	$\mathcal{Z}_c^s$	Sav_1 (%)	Sav_2 (%)	Sav_3 (%)	Sav_4 (%)
C1	3	307.9	0.0	307.9	0.0	307.9	307.9	0.0	307.9	307.9	0.0	307.9	0.00	0.00	0.00	0.00
R1	3	435.0	11.4	446.3	45.0	480.0	437.7	3.6	441.3	443.1	1.5	444.6	1.01	7.19	0.77	6.92
RC1	3	324.1	22.0	346.1	131.8	455.9	325.3	11.2	336.5	345.0	0.0	345.0	3.27	34.87	2.82	34.41
C2	3	324.7	0.0	324.7	0.0	324.7	324.7	0.0	324.7	324.7	0.0	324.7	0.00	0.00	0.00	0.00
R2	3	432.6	10.6	443.3	33.8	466.4	433.1	7.8	440.8	442.5	3.0	445.5	4.73	4.41	0.94	4.73
RC2	3	268.7	3.5	272.2	20.4	289.1	269.3	1.5	270.8	277.8	0.0	277.8	6.62	4.71	2.57	6.62
C1	9	247.7	15.4	263.1	25.6	273.3	250.4	9.8	260.2	253.9	11.3	265.2	1.19	3.34	1.55	3.76
R1	9	405.5	21.0	426.5	66.2	471.7	406.6	13.7	420.4	415.7	11.5	427.2	1.02	8.84	1.58	9.43
RC1	9	324.1	26.0	350.1	158.0	482.1	325.5	13.7	339.2	345.9	2.8	348.7	3.63	42.13	2.85	41.12
C2	9	324.7	9.1	333.8	15.1	339.8	325.3	6.9	332.2	326.9	9.2	336.1	0.39	0.87	1.19	1.67
R2	9	322.7	13.4	336.1	38.3	361.1	325.0	8.3	333.3	334.9	5.2	340.2	0.94	7.01	1.88	7.90
RC2	9	268.7	4.2	272.9	24.7	293.5	269.3	1.9	271.2	277.8	0.4	278.2	0.70	6.18	2.54	8.00
Average		**332.2**	**11.4**	**343.6**	**46.6**	**378.8**	**333.3**	**6.5**	**339.9**	**341.3**	**3.7**	**345.1**	**1.12**	**9.96**	**1.56**	**10.38**

Table 4: Summary of the results of the deterministic and stochastic solutions.

The results reported in Table 4 indicate the following:

- The values in columns Sav_1 and Sav_2 suggest that significant savings can be achieved by solving the stochastic problem instead of the deterministic problem. For instances in class 1, the average savings were 2.1% for the fixed rule-based policy and 16.1% for the classical one, while for class 2, the average savings were 1.1% for the fixed rule-based policy and 3.9% for the classical policy. This result suggests that the use of the fixed rule-based policy instead of the classical one brings benefit to govern the recourse actions, particularly when the solution of the deterministic problem is used. We observe larger average savings Sav_1 and Sav_2 in class 1 because the instances in this class have

a shorter planning horizon and shorter routes than instances in class 2; hence, class 1 instances were more sensitive to demand variations.

- The values in columns Sav_3 and Sav_4 indicate that allowing preventive restocking trips to the depot (as defined in the fixed rule-based policy) in addition to reactive trips (as defined in the classical policy) reduces the total expected recourse cost. These results were expected since preventive restocking trips avoid visiting the same vertex twice, avoiding the extra cost Γ, in addition to reducing travel costs. Yet, the average savings reported are significant, on average 1.56% and 10.38% for Sav_3 and Sav_4, respectively, giving evidence of the importance of resorting to preventive restocking trips.

For the three approaches, we show in Table 5 the corresponding relative optimality gap (Gap) in percentage, computed using the best upper and lower bounds found by the approach; the average computing time in seconds (Time) considering only instances solved to optimality by the approach; number of instances solved to optimality (Nopt); and number of instances in which no feasible solution was found within the time limit (Nfail). We also report the average computing time in seconds ($Time^2$) considering only instances that were solved to optimality for both policies. Detailed results regarding each instance are presented in Tables D.15-D.18 in Appendix D of the e-companion.

		Deterministic				Stochastic Fixed rule-based policy					Stochastic Classical policy				
Subclass	S	Gap (%)	Time (sec.)	Nopt	Nfail	Gap (%)	Time (sec.)	$Time^2$ (sec.)	Nopt	Nfail	Gap (%)	Time (sec.)	$Time^2$ (sec.)	Nopt	Nfail
C1	3	0.84	225.5	8	0	0.66	1346.8	1346.8	8	0	0.01	1589.4	1589.4	8	1
R1	3	3.09	106.6	9	0	4.48	85.0	85.0	8	1	5.93	440.1	440.1	8	1
RC1	3	0.07	45.7	7	0	1.66	1840.8	1.5	5	0	3.86	36.1	36.1	3	0
C2	3	6.78	2217.6	6	0	5.20	27.0	27.0	5	0	2.48	40.2	40.2	5	1
R2	3	7.41	1372.2	9	0	2.63	77.9	77.9	7	1	2.15	750.1	750.1	7	2
RC2	3	1.05	0.0	6	0	1.83	2.4	0.1	5	0	5.13	2832.1	2832.1	4	0
C1	9	0.84	225.5	8	0	1.64	3244.4	7.3	7	0	3.09	112.4	112.4	4	1
R1	9	3.09	106.6	9	0	6.58	62.3	18.3	7	1	11.84	33.1	33.1	6	1
RC1	9	0.07	45.7	7	0	2.06	3921.0	3.5	5	0	4.30	230.2	230.2	3	0
C2	9	6.78	2217.6	6	0	7.52	53.1	53.1	5	0	3.71	175.8	175.8	5	2
R2	9	7.41	1372.2	9	0	4.88	1543.0	1.1	7	1	11.28	12.2	12.2	3	1
RC2	9	1.05	0.0	6	0	1.84	4.2	0.1	5	0	6.20	2259.8	2259.8	4	0
Average		**9.62**	**661.2**			**3.42**	**1017.3**	**135.1**			**5.00**	**709.3**	**709.3**		
Total				90	0				**74**	**4**				**60**	**10**

Table 5: Summary of the results of the deterministic and stochastic solutions.

For the deterministic problem, 45 out of 56 instances were solved to optimality, with an average time of less than 670 seconds. As expected, solving the stochastic problem required more time, on average, than the deterministic problem. For the fixed rule-based policy, proven optimal solutions were obtained for 38 instances with $S = 3$, within 564 seconds on average, and for 36 instances with $S = 9$, within 1472 seconds on average. This also reveals that the difficulty of solving the stochastic problem increased as the number of realizations S increased. For instances R112 and R211, with $S = 3$ and $S = 9$, the stochastic problem became so challenging that the method failed to determine a feasible solution within the time limit (5 hours).

The **LS3** algorithm performed better on the VRPTWSD with fixed rule-based policy than with the classical policy. It solved to optimality 74 instances considering the former policy, while for the latter, it

solved 60. Moreover, the method took less time on average on instances that were solved to optimality with both policies (135.1 seconds for the fixed rule-based policy and 709.3 seconds for the classical one). Note that the optimality gaps of the remaining instances were smaller on average for the fixed rule-based policy. Finally, the number of failures of the **LS3** algorithm increased from 4 to 10 when solving the VRPTWSD with classical policy. These results reveal the weaknesses of the classical policy: in addition to generating solutions with larger expected total costs, it adversely affected the computational performance of the proposed Integer L-shaped algorithm.

To provide further information regarding the benefits of using the stochastic approach, we ran additional experiments in which we reduced the vehicle capacity of the instances to 80%, 85%, 90% and 95% of the original capacity and solved these modified instances using **LS0** and a constructive heuristic based on Solomon's heuristic I1 (**HI1**) (Solomon, 1987; Braysy and Gendreau, 2005). This allows us to verify if a simpler approach, based on reducing vehicle capacities, would lead to similar or even superior results than **LS3** (using the original vehicle capacity Q).

The proposed **HI1** selects the first vertex of the route using the farthest vertex rule. Next, the route is extended until no more feasible vertex insertions are possible. Then, a new route is started if there are vertices not visited yet. The heuristic ends when all vertices have been visited. One of the differences to the original version of the heuristic is that we consider Assumption **A4** (at most one failure per route) in the extensions. Thus, we calculate the maximum sum of demand realizations of the route vertices and use a vehicle capacity equal to $2Q$. Unlike the **LS0** method, the solutions obtained by **HI1** are always feasible for the VRPTWSD, as it verifies time window violations due to recourse actions in the route extensions, using the recursive equation (15). Note, however, that the heuristic may fail to find a solution, as we have a limited number of vehicles in the instances. Hence, given an instance, we apply the heuristic several times, using different combinations of parameters α_1, α_2, μ, and λ, and then we choose the feasible solution with the best expected total cost. Let $\mathcal{R}$ be the set of routes in this best solution and let c, Q, and z be the corresponding first-stage cost, recourse cost, and expected total cost, respectively. The application of the heuristic in a given instance is summarized as follows:

Step 0: Set the expected total cost z to $+\infty$;

Step 1: For all $\alpha_1, \mu, \lambda \in \{0.00, 0.01, 0.02, \ldots, 1.00\}$ do

> ***Step 1.1:*** Run the **HI1** for the current capacity value and with parameters $\alpha_1, \alpha_2 = 1 - \alpha_1, \mu$, and λ;
>
> ***Step 1.2:*** Let $\tilde{\mathcal{R}}$ be the set of routes obtained by **HI1** and $\tilde{c}$ the corresponding first-stage cost;
>
> ***Step 1.3:*** Determine the recourse cost $\tilde{Q}$ of the routes in $\tilde{\mathcal{R}}$ using (15) and set $\tilde{z} := \tilde{c} + \tilde{Q}$;
>
> ***Step 1.4:*** If $\tilde{z} < z$, then $\mathcal{R} := \tilde{\mathcal{R}}$, $c := \tilde{c}$, $Q := \tilde{Q}$ and $z := \tilde{z}$.

We apply this heuristic also to the VRPTWSD with classical policy, but in *Step 1.3* we use $\theta_i = 1$, for all $i \in \mathcal{N}$, when applying the recursive equation (15). For further details on **HI1**, see e.g. Solomon (1987); Braysy and Gendreau (2005) and De La Vega et al. (2019).

Table 6 reports the average results of **HI1** and **LS0** grouped by instance subclass and by five different values of vehicle capacity $\tilde{Q} = \lfloor \gamma Q \rfloor$, with $\gamma = 0.8, 0.85, 0.9, 0.95$ and 1. Different from Table 4, this table considers the solutions of all instances, including those in which optimality was not proven. Columns $\mathcal{Z}_r^{\mathbf{HI1}}$ and $\mathcal{Z}_r^{\mathbf{LS0}}$ are the total expected costs of the solutions obtained by **HI1** and **LS0**, respectively, with the expected recourse cost computed under the rule-based policy. Similarly, $\mathcal{Z}_c^{\mathbf{HI1}}$ and $\mathcal{Z}_c^{\mathbf{LS0}}$ are the total

expected costs under the classical policy. Columns Sav_1, Sav_2 and Sav_4 have a similar meaning as those in Table 4, but we use the total expected cost of the solution obtained by the corresponding approach (e.g., for **HI1** we use $\mathcal{Z}_r^d = \mathcal{Z}_r^{\mathbf{HI1}}$, while for **LS0** we use $\mathcal{Z}_r^d = \mathcal{Z}_r^{\mathbf{LS0}}$). Values $\mathcal{Z}_r^s$ and $\mathcal{Z}_c^s$ are the same used in Table 4 and correspond to the total expected cost of the solution obtained by **LS3** (using the original capacity Q). Columns Time, nFail and nOpt report the average computing time in seconds, the number of instances in which **HI1** failed to find a feasible solution and the number of instances solved to optimality by **LS0**, respectively. We also report in this table two additional savings computed as follows:

$$\begin{aligned} Sav_5 &= (\mathcal{Z}_r^{\mathbf{HI1}}/\mathcal{Z}_r^{\mathbf{LS0}} - 1) * 100, \\ Sav_6 &= (\mathcal{Z}_c^{\mathbf{HI1}}/\mathcal{Z}_c^{\mathbf{LS0}} - 1) * 100, \end{aligned}$$

which quantify the savings obtained in the expected total cost of the solution obtained by **HI1** with respect to the solution obtained by **LS0** under rule-based and classical policies, respectively. Negative values of these metrics mean that the **HI1** solution has a lower expected total cost than the solution with the **LS0** method. Such negative values are possible since the **LS0** method solves the deterministic problem.

The results in columns Sav_1 and Sav_2 of Table 6 indicate that we can achieve significant savings by using the stochastic approach (**LS3**) instead of **HI1** and **LS0** with reduced vehicle capacity values. The more we reduce the vehicle capacity, the more advantageous the stochastic approach tends to be with respect to solution cost. Average savings are 17.3% and 12.8% for the rule-based policy and 30.9% and 28.6% for the classical one. Additionally, the results in column Sav_4 show large savings by using the rule-based instead of the classical policy. The average savings of the rule-based policy over the classical is 15.1% and 17.3% for **HI1** and **LS0**, respectively. These results emphasize the importance of resorting to the stochastic approach to find good-quality solutions of the VRPTWSD. Additionally, they confirm the benefits of resorting to restocking trips to the depot, given the superior results of the rule-based policy with respect to the classical policy.

The cost of the solutions obtained by **LS0** was less than the cost of the solutions obtained by **HI1**, in most instances. Only in very few cases, we observed the opposite (given by the negative values in columns Sav_5 and Sav_6). Another issue is that **HI1** failed to find a solution in 11 instances, given the limited number of vehicles. On the other hand, **LS0** required more computing time than **HI1**, as expected. On average, **LS0** took around 750 seconds, while **HI1** ran for 1.1 seconds. Considering all the 280 instances, **LS0** solved 237 (out of 280) instances to optimality, which corresponds to almost 85%.

5.3. Effect of time windows

The results presented in column Sav_3 of Table 4 indicate average savings of 1.56 for the cost of solutions obtained under the rule-based policy with respect to solutions obtained under the classical policy. This is in accordance with results reported for the VRPSD in the literature, suggesting that the inclusion of recourse actions related to time window violations keeps the rule-based policy advantageous. For example, Salavati-Khoshghalb et al. (2019c) report average savings that range from 0.18 to 1.29 when comparing the rule-based and the classical policies for the VRPSD instances considered in their experiments. Regarding the savings we obtained for the VRPTWSD related to the comparison of stochastic and deterministic solutions under both policies, namely Sav_1 and Sav_2, we observe that they follow a similar behavior as reported for the VRPSD in the literature. As mentioned by Florio et al. (2020a), even with an optimal restocking policy, the cost of the optimal stochastic solutions is less than 1% better than the cost of the determinis-

		HI1							LS0								
Subclass	$\tilde{Q}$	$\mathcal{Z}_r^{\mathbf{HI1}}$	$\mathcal{Z}_c^{\mathbf{HI1}}$	Time (sec.)	nFail	Sav_1 %	Sav_2 %	Sav_4 %	$\mathcal{Z}_r^{\mathbf{LS0}}$	$\mathcal{Z}_c^{\mathbf{LS0}}$	Time (sec.)	nOpt	Sav_1 %	Sav_2 %	Sav_4 %	Sav_5 %	Sav_6 %
	60	417.5	566.6	0.7	0	27.5	68.6	35.5	413.0	571.3	153.4	8	26.4	70.6	38.0	1.0	-0.6
	63	399.1	435.8	0.7	0	23.9	32.2	9.3	393.6	432.7	280.9	7	22.4	31.5	10.1	1.3	0.6
C1	67	370.3	371.9	0.7	0	15.1	12.8	0.5	361.9	363.7	1837.8	7	12.7	10.6	0.6	2.2	2.0
	71	343.4	434.4	0.7	0	7.9	33.3	26.6	342.1	447.0	4.7	7	7.3	35.3	29.1	0.6	-0.9
	75	322.3	336.1	0.7	0	1.5	3.4	4.4	320.1	332.6	61.1	7	0.7	2.1	3.8	0.7	1.3
	40	442.8	556.6	0.6	0	18.2	45.6	24.8	432.1	544.0	640.5	6	15.2	41.1	23.8	2.6	3.7
	42	435.1	467.1	0.6	0	16.3	23.2	7.3	424.7	453.2	620.8	6	13.4	19.0	6.3	2.6	3.7
R1	45	414.1	416.6	0.7	0	11.1	10.4	0.6	403.8	405.2	1226.5	6	8.3	7.3	0.4	2.6	2.9
	47	405.0	472.3	0.7	0	10.1	25.4	15.4	385.4	449.4	610.4	6	3.6	17.8	15.1	6.4	6.7
	50	391.3	401.5	0.7	0	6.8	8.1	2.5	372.4	378.7	629.6	6	0.3	0.6	1.6	6.5	7.4
	80	695.0	871.9	1.0	3	26.9	51.0	23.0	674.1	832.3	540.9	7	23.0	43.3	20.2	3.2	6.2
	85	609.6	678.7	0.8	3	21.4	31.3	11.7	594.4	684.1	422.5	9	19.0	32.0	14.0	2.4	1.4
RC1	90	611.0	718.2	1.0	1	16.8	29.8	14.7	569.4	672.9	424.8	10	9.4	23.2	16.4	6.8	5.3
	95	644.9	759.2	1.5	1	12.7	23.3	14.2	623.9	746.5	1005.9	11	8.0	21.8	17.7	4.4	1.5
	100	565.4	642.2	1.0	0	9.2	19.2	12.7	526.2	613.8	677.0	10	1.2	11.8	14.2	7.9	7.0
	60	604.5	681.9	0.7	1	22.9	33.2	12.0	592.7	670.9	294.1	7	19.3	29.8	12.5	3.7	3.3
	63	586.7	633.7	0.7	2	18.3	22.9	8.0	565.3	634.0	913.9	8	14.5	23.2	11.8	3.3	-0.1
C2	67	602.0	668.5	0.9	0	17.0	23.7	10.2	556.4	618.5	802.2	9	8.1	13.8	9.6	8.2	8.9
	71	626.0	697.5	1.3	0	12.4	17.8	10.6	607.3	691.6	1129.6	10	7.0	13.8	12.3	5.3	4.0
	75	597.4	658.6	1.4	0	7.5	12.0	9.9	569.0	633.8	805.8	10	1.4	7.0	11.3	6.0	4.7
	40	697.4	1047.7	1.3	0	36.7	88.8	41.2	669.0	1048.7	963.0	8	34.1	101.2	53.7	2.0	-5.6
	42	652.4	695.4	1.3	0	29.3	33.0	5.7	626.3	677.9	2206.6	8	26.1	32.3	7.8	2.7	0.7
R2	45	591.8	808.8	1.3	0	16.7	52.7	34.4	596.0	981.4	457.0	8	18.3	81.8	57.3	-1.4	-14.9
	47	565.5	596.2	1.3	0	11.8	13.9	4.7	550.5	599.0	901.6	8	10.1	15.1	7.6	1.6	-1.0
	50	557.9	743.9	1.4	0	9.2	39.0	30.7	523.9	768.6	583.3	8	4.9	50.8	47.2	4.3	-5.0
	80	632.8	790.5	1.6	0	34.5	54.7	21.0	603.6	764.7	964.0	8	30.1	53.1	23.5	3.5	1.8
	85	613.9	635.9	1.6	0	31.2	28.6	3.0	585.0	602.6	1381.1	8	26.5	23.8	2.8	3.8	4.0
RC2	90	556.6	698.9	1.7	0	16.8	37.7	23.7	520.1	669.7	473.1	8	10.6	32.2	25.8	5.7	4.1
	95	540.0	563.2	1.8	0	13.6	11.7	3.4	501.0	520.5	492.3	8	6.8	4.7	3.1	6.3	6.7
	100	546.1	716.6	1.8	0	17.0	41.3	26.7	472.4	554.7	900.0	8	1.2	10.2	14.6	15.5	28.6
Average		**545.7**	**638.9**	**1.1**	**11**	**17.3**	**30.9**	**15.1**	**522.6**	**625.1**	**755.6**	**237**	**12.8**	**28.6**	**17.3**	**4.2**	**3.0**

Table 6: Summary of the results obtained by **LS0** and **HI1**.

tic equivalent solution, for the VRPSD instances introduced by Louveaux and Salazar-González (2018), which are characterized by routes with many customers in an optimal solution. For instances with many vehicles and few customers per route, the authors reported higher gains of the stochastic approach over the deterministic. Again, this is confirmed in our results, as the average Sav_1 and Sav_2 values for instances in class 1 (characterized by short routes) were 2.1% and 16.1%, respectively; while for instances in class 2 (characterized by long routes), these values were 1.1% and 3.9%, respectively.

To analyze the impact of the length of time windows on the performance of our best stochastic approach (**LS3**), we ran one additional experiment using instances in subclass **RC2**, but with the start and end times of the time window of each vertex $i \in \mathcal{V}$ modified as in Lei et al. (2011) and given by:

$$\tilde{w}_i^a = \max\left\{w_0^a, (w_i^a + w_i^b)/2 - \chi(w_i^b - w_i^a)/2\right\}, \tag{46}$$

$$\tilde{w}_i^b = \min\left\{w_0^b, (w_i^a + w_i^b)/2 + \chi(w_i^b - w_i^a)/2\right\}, \tag{47}$$

where $[w_i^a, w_i^b]$ is the original time window of vertex i and χ is a parameter to control the length of the new time window $[\tilde{w}_i^a, \tilde{w}_i^b]$. Note that for $\chi = 1$, we obtain the original time windows of the instance. The larger the value of χ, the wider the new time windows become. Thus, for $\chi \to \infty$, the VRPTWSD becomes a VRPSD, meaning that the time windows are disregarded. Table 7 reports the total expected costs and computing times in seconds for $S = 9$ and $\chi = 0.5, 1, 2, 3$ and ∞. The symbol '>' in the column Time indicates that the **LS3** algorithm reached the time limit of 3600 seconds. These results indicate that as the parameter χ increases, the expected total cost decreases but the runtime of **LS3** increases. The behavior observed in the costs when increasing χ is expected, as the wider the time windows, the lower the probability of not satisfying them in the planned routes. Hence, it reduces the recourse cost associated with using the special service. On the other hand, by increasing χ, the space of feasible solutions is likely to increase, thus resulting in a larger search tree in the L-shaped method. This often leads to more calls to the routine that separates LBF cuts, causing the total time of the solution method to increase as well.

Inst.	$\chi = 0.5$		$\chi = 1.0$		$\chi = 2.0$		$\chi = 3.0$		$\chi \to \infty$	
	$\mathcal{Z}_r^s$	Time (sec.)	$\mathcal{Z}_r^s$	Time (sec.)	$\mathcal{Z}_r^s$	Time (sec.)	$\mathcal{Z}_r^s$	Time (sec.)	$\mathcal{Z}_r^s$	Time (sec.)
RC201	339.6	0.0	334.8	0.0	327.7	0.0	325.3	0.5	325.0	1.4
RC202	275.5	0.7	275.5	0.7	272.5	1.2	272.5	3.5	272.5	3.7
RC203	438.7	81.0	432.8	250.8	432.8	1349.6	429.5	1911.6	425.8	2387.4
RC204	520.9	>	514.4	>	516.2	>	514.2	>	515.5	>
RC205	237.0	0.0	235.3	0.0	235.3	0.0	235.3	0.1	235.3	0.1
RC206	242.3	0.0	239.3	0.0	238.0	0.0	237.8	0.1	237.8	0.1
RC207	767.4	>	758.8	>	752.1	>	756.5	>	758.9	>
RC208	977.1	>	956.0	>	949.7	>	951.6	>	946.3	>
Average	474.8	1360.1	468.4	1381.3	465.5	1518.7	465.3	1589.4	464.6	1649.0

Table 7: Expected total costs and computing times for instances in subclass **RC2** with $S = 9$ and considering modified time windows.

5.4. Effect of variability and stochasticity

In our experiments, we assume that ξ_i follows a triangular distribution with finite support S and most likely realization equal to its expected value $\mathbb{E}[\xi_i]$. Each realization s, for $s = 1, \ldots, S$, is determined as $\xi_i^s = \mathbb{E}[\xi_i] - \delta * (\lfloor S/2 \rfloor - s + 1)$ where δ is the step size between each consecutive realization of the support of ξ_i. All results presented so far assume $\delta = 1$, which leads to small variability and stochasticity in the demand realizations. To verify whether the variability of the demand distribution support impacts the performance of **LS3** algorithm, we performed another experiment in which we vary δ from 1 to 5 for instances in subclass **RC2** with $S = 5$. Table 8 shows the results of this experiment. As expected, the total cost increases as the demand distribution support variability increases. High variability means that the values in the realizations can be significantly greater than the expected value, which increases the route failure probability. Moreover, the problem becomes more difficult for **LS3** as the variability increases. Supports with high variability make the estimate of $\underline{\theta}_\ell$ very low and that of $\overline{\theta}_\ell$ very high for each position ℓ of a partial route. The smaller the value of $\underline{\theta}_\ell$ is (and the larger the value of $\overline{\theta}_\ell$ is), the smaller is the lower bound on recourse cost and, therefore, the less effective the LBF cuts are.

	$\delta = 1$		$\delta = 2$		$\delta = 3$		$\delta = 4$		$\delta = 5$	
Inst.	$\mathcal{Z}_r^s$	Time (sec)	$\mathcal{Z}_r^s$	Time (sec)	$\mathcal{Z}_r^s$	Time (sec)	$\mathcal{Z}_r^s$	Time (sec)	$\mathcal{Z}_r^s$	Time (sec)
RC201	334.4	0.0	335.2	0.0	341.8	0.1	348.7	0.9	351.4	1.4
RC202	275.2	0.8	275.5	0.9	282.1	5.2	287.9	31.8	288.4	33.0
RC203	432.0	67.5	433.3	103.8	443.2	>	448.3	>	449.5	>
RC204	516.4	>	518.4	>	529.8	>	536.3	>	539.8	>
RC205	235.0	0.0	235.6	0.0	239.0	0.1	241.9	0.3	243.0	0.5
RC206	239.0	0.0	239.7	0.0	245.9	0.1	252.0	0.1	253.2	0.2
RC207	752.2	>	756.6	>	774.4	>	797.5	>	805.6	>
RC208	944.3	>	947.9	>	976.3	>	1032.2	>	1018.5	>

Table 8: Expected total cost and computing time for instances in subclass **RC2** with $S = 5$ and different values of δ.

5.5. *Effect of* Γ

Recall that our recourse actions apply the additional cost Γ related to customer inconvenience because of partial delivery. In certain situations, however, there is no such inconvenience (e.g., in waste collection), thus we would have $\Gamma = 0$. To verify the performance of the proposed approach in situations like this, Table 9 presents the expected total costs of the deterministic problem solutions evaluated under rule-based and classical policies, considering stochastic problem solutions under the same policies for two different choices of Γ, namely $\Gamma_1 = 0$ and $\Gamma_2 = \sum_{i \in \mathcal{N}} c_{0i}/n$. This table also reports the savings and the ratio of the stochastic problem costs under both policies for each choice of Γ.

The results in Table 9 show that, for both stochastic problems, the expected total cost is higher for Γ_2, which is expected because of the additional cost of partial deliveries. Regardless of the value of Γ, it is cheaper (see the values reported in columns Sav_1 and Sav_2) to solve the stochastic problem rather than to solve the deterministic problem evaluated under both recourse policies. In addition, the values in the columns Sav_3 indicate that it is possible to have greater savings by solving the stochastic problem under the rule-based policy instead of the classical policy. These savings are even greater in applications where partial delivery is undesirable. Finally, the ratio for each value of Γ highlights the fact that the expected total cost of the stochastic problem under the classical policy is no worse than twice the expected total cost of the stochastic problem under the rule-based policy. These empirical results are aligned with the theoretical results in Bertazzi and Secomandi (2020) who conducted a worst-case analysis finding that the expected cost under the classical policy of each feasible route is at most twice its expected cost under the preventive policy.

5.6. *Effect of* Φ *on the expected total cost of solutions*

We have conducted experiments varying parameter Φ from 0 to 10 with step size 2 to assess its impact on the expected total cost of solutions. Table 10 reports the expected total cost and number of routes (NR) obtained for instances of the subclass **R1**. As expected, increasing this parameter also increases the expected total cost. For certain values of Φ, we observe an increase in the number of planned routes in the solutions, which prevent the high costs of special services associated to customers with time window failures.

5.7. *Expected Value of Perfect Information*

We ran numerical experiments using 6 instances with 5 customers and 1 vehicle to calculate the expected value of perfect information (EVPI), defined as the maximum value a decision maker would be willing to

	Deterministic		$\mathcal{Z}_r^s$		$\mathcal{Z}_c^s$		Savings for Γ_1			Savings for Γ_2			Ratio	
Inst.	$\mathcal{Z}_r^d$	$\mathcal{Z}_c^d$	Γ_1	Γ_2	Γ_1	Γ_2	Sav_1 (%)	Sav_2 (%)	Sav_3 (%)	Sav_1 (%)	Sav_2 (%)	Sav_3 (%)	Γ_1	Γ_2
RC201	334.8	334.8	334.6	334.8	334.7	334.8	0.03	0.00	0.01	0.00	0.00	0.01	1.00	1.00
RC202	276.0	315.8	273.2	275.5	298.6	313.6	1.01	0.72	9.29	0.19	0.72	13.84	1.05	1.14
RC203	438.2	517.0	430.3	432.8	455.4	471.7	1.85	9.60	5.85	1.26	9.60	9.00	1.04	1.09
RC204	524.3	653.4	511.3	514.4	540.7	557.4	2.54	17.23	5.74	1.91	17.23	8.35	1.03	1.08
RC205	241.5	283.8	235.2	235.3	235.2	235.4	2.67	20.60	0.01	2.63	20.60	0.01	1.00	1.00
RC206	239.3	239.4	239.2	239.3	239.3	239.4	0.05	0.00	0.01	0.00	0.00	0.01	1.00	1.00
RC207	769.1	912.3	754.7	755.2	780.9	800.5	1.91	13.97	3.47	1.84	13.97	5.99	1.03	1.06
RC208	967.7	1273.5	941.3	957.3	1048.7	1177.4	2.80	8.16	11.41	1.08	8.16	22.98	1.12	1.23

Table 9: Results of the deterministic and stochastic problems under both policies for instances in **RC2** with $S = 5$ and $\delta = 1$, using different choices for Γ.

	$\Phi = 0$		$\Phi = 2$		$\Phi = 4$		$\Phi = 6$		$\Phi = 8$		$\Phi = 10$	
Inst.	$\mathcal{Z}_r^s$	NR	$\mathcal{Z}_r^s$	NR	$\mathcal{Z}_r^s$	NR	$\mathcal{Z}_r^s$	NR	$\mathcal{Z}_r^s$	NR	$\mathcal{Z}_r^s$	NR
R101	634.3	9	649.0	9	657.6	9	662.8	9	668.1	9	673.3	9
R102	647.2	10	671.7	10	681.7	10	692.4	11	697.7	11	699.5	11
R103	433.1	6	446.8	7	450.7	7	455.3	7	459.5	7	463.7	7
R104	336.4	5	336.4	5	336.4	5	336.4	5	336.4	5	336.4	5
R105	292.7	4	292.7	4	292.8	4	292.8	4	292.8	4	292.8	4
R107	360.3	5	360.4	5	360.6	5	360.7	5	360.8	5	361.0	5
R108	448.5	6	448.5	6	448.5	6	448.5	6	448.5	6	448.5	6
R109	432.7	6	441.8	6	446.2	6	446.2	6	446.2	6	446.2	6

Table 10: Results of the stochastic problems for instances in **R1** with $S = 5$ and $\delta = 1$, using different values of Φ.

pay for complete and accurate information about the future. This concept also expresses how much can be saved by having perfect information. The EVPI is determined as follows: (*i*) solve the deterministic problems (wait-and-see problems) associated with each scenario ω and define $WS(\omega)$ to be their corresponding optimal values; (*ii*) determine the expected value of the wait-and-see solutions using WS = $\sum_\omega \pi(\omega) WS(\omega)$ where $\pi(\omega)$ is the probability of occurrence of scenario ω (we assume the scenarios are equiprobable); and (*iii*) determine the EVPI as the difference between the value of the here-and-now (HN) solutions and the value of the wait-and-see (WS) solutions, i.e., EVPI = HN - WS, where the HN value is the optimal value of the stochastic problem. The higher the EVPI value, the greater the impact of uncertainty in the problem.

The HN problem for each instance involves a demand distribution support with $\delta = 3$ and $S = 5$. For each instance, we generate 300 WS problems corresponding to 300 different demand profiles. Each demand profile or scenario has five realizations, one for each customer of the five considered in each instance. The demand realization of each customer in each of the scenarios is randomly generated between the minimum and maximum values of the demand distribution support. The **LS3** algorithm is used to solve both WS and HN problems. It is important to note that the demand distribution support in the WS problems have a single realization only (equal to the realization in the corresponding scenario) with probability of occurrence equal to 1. For these experiments, we set $\Phi = 1$ and $Q = 50$.

Table 11 presents the HN, WS and EVPI values for the selected instances. Additionally, the last column in this table shows the percentage of the EVPI value with respect to the HN value. All the HN and WS problems were solved to optimality by the **LS3** algorithm in less than 1 second. The EVPI results indicate that it would be possible to achieve significant cost savings (22.07%, on average) if perfect information

about the stochastic parameters were known when the problem is solved.

Inst.	WS	HN	EVPI	EVPI (%)
C101	42.8	57.5	14.7	25.6
C201	75.8	90.5	14.8	16.3
R101	126.7	136.2	9.6	7.0
R201	126.9	136.2	9.3	6.8
RC101	102.9	164.9	61.9	37.6
RC201	100.4	164.9	64.5	39.1

Table 11: HN, WS and EVPI values of the solutions obtained for instances with 5 customers and 1 vehicle.

5.8. Load factors smaller and larger than one

Finally, we ran additional computational experiments to study the performance of the **LS3** algorithm progressively relaxing assumption **A3** by multiplying Q by η, where η is the load factor varying in the set $\{0.80, 1.00, 1.20, 1.35, 1.75, +\infty\}$. Note that we do not change the vehicle capacity of the instance; we only change the way assumption **A3** is checked by using $\eta \times Q$ instead of Q when designing the first-stage routes. These experiments were conducted using the RC2 class instances with $S = 5$, $\delta = 2$ and $\Phi = 10$. They also include the scenario in which assumption **A3** is fully relaxed, represented by $\eta = +\infty$. Table 12 reports the results, presenting for each instance and value of η, the first stage cost, the recourse cost, the ratio between the previous costs, and the execution time in seconds. Additionally, the table shows the *risk* associated to the solution, estimated via Monte-Carlo simulation using 10,000 demand profiles generated within the support of each demand vertex. The risk is calculated as the ratio between the number of times the vehicle capacity was violated and 10,000. As in previous tables, the symbol '>' in column Time indicates that the **LS3** algorithm reached the time limit of 3600 seconds.

There are several results related to those found in Florio et al. (2020a) and Florio et al. (2020b) for the optimal solutions of Table 12 that deserve special attention as the load factor increases:

- The total expected cost decreases because the first stage cost becomes less relevant as the number of routes also decreases.
- The risk of failure increases. This behavior is expected since the solutions have a greater number of customers per route; thus, they are more exposed to failures when implemented. This indicates the practical importance of assuming **A3** to generate first stage routes, as the risk significantly increases even for $\eta = 1.2$.
- The ratio between the recourse cost and the first stage cost increases because the recourse cost becomes more relevant as the solutions have a greater number of customers per route.
- The difficulty for solving the problem by **LS3** algorithm increases. For large values of η, the generation of 2-path cuts is significantly reduced, and they are important to tighten the linear programming relaxation. However, the method is still effective to find feasible solutions in scenarios where assumption **A3** is progressively relaxed because for these cases the solutions have more customers per route. Hence, the LBF cuts become more effective as the recourse cost becomes more relevant and its lower bound stronger.

	$\eta = 0.8$							$\eta = 1.00$							$\eta = 1.2$						
Inst.	C	Q_r^s	$\mathcal{Z}_r^s$	*ratio*	*nr*	Time (sec)	*risk*	C	Q_r^s	$\mathcal{Z}_r^s$	*ratio*	*nr*	Time (sec)	*risk*	C	Q_r^s	$\mathcal{Z}_r^s$	*ratio*	*nr*	Time (sec)	*risk*
RC201	426.9	0.0	426.9	0.0	5	0.2	0.2	334.4	0.8	335.2	0.2	4	0.0	0.3	311.9	16.4	328.2	5.3	3	10.2	0.8
RC202	336.4	0.0	336.4	0.0	4	0.1	0.2	268.8	6.7	275.5	2.5	3	1.0	0.4	243.2	0.9	244.1	0.4	3	0.1	0.6
RC203	518.4	0.0	518.4	0.0	6	25.2	0.2	425.4	7.9	433.3	1.9	5	59.9	0.5	404.6	8.7	413.4	2.2	4	>	0.9
RC204	604.4	0.0	604.4	0.0	7	940.9	0.3	511.5	7.9	519.4	1.6	6	>	0.6	469.5	3.6	473.2	0.8	5	2389.5	0.9
RC205	270.8	0.0	270.8	0.0	3	0.8	0.1	235.0	0.6	235.6	0.3	3	0.0	0.2	216.3	1.4	217.7	0.6	2	0.5	0.6
RC206	337.4	0.0	337.4	0.0	4	0.4	0.1	239.0	0.8	239.7	0.3	3	0.0	0.2	239.0	0.8	239.7	0.3	3	0.0	0.2
RC207	890.8	0.0	890.8	0.0	9	>	0.4	748.2	9.2	757.4	1.2	8	>	0.6	736.0	28.6	764.7	3.9	8	>	0.8
RC208	1142.0	0.0	1142.0	0.0	12	>	0.5	914.3	39.4	953.7	4.3	10	>	0.8	937.9	136.5	1074.5	14.6	9	>	1.0
Average	**565.9**	**0.0**	**565.9**	**0.0**	**6.3**	**1020.9**	**0.3**	**459.6**	**9.2**	**468.7**	**1.5**	**5.3**	**1357.4**	**0.5**	**444.8**	**24.6**	**469.4**	**3.5**	**4.6**	**1649.8**	**0.7**

	$\eta = 1.3$							$\eta = 1.7$							$\eta = \infty$						
Inst.	C	Q_r^s	$\mathcal{Z}_r^s$	*ratio*	*nr*	Time (sec)	*risk*	C	Q_r^s	$\mathcal{Z}_r^s$	*ratio*	*nr*	Time (sec)	*risk*	C	Q_r^s	$\mathcal{Z}_r^s$	*ratio*	*nr*	Time (sec)	*risk*
RC201	309.7	12.8	322.4	4.1	3	5.5	0.8	224.7	50.4	275.1	22.4	2	1.5	1.0	224.7	50.4	275.1	22.4	2	6.0	1.0
RC202	243.2	0.9	244.1	0.4	3	0.1	0.6	182.4	2.3	184.7	1.3	2	1.0	0.9	182.4	2.3	184.7	1.3	2	29.9	0.9
RC203	403.1	4.5	407.6	1.1	4	>	0.9	337.8	1.9	339.7	0.6	3	803.2	1.0	278.6	3.5	282.2	1.3	2	>	1.0
RC204	467.8	18.0	485.7	3.8	5	>	0.9	369.4	5.3	374.7	1.4	4	>	1.0	296.7	7.3	304.0	2.5	2	>	1.0
RC205	211.3	3.0	214.3	1.4	2	0.2	0.6	209.0	2.0	211.0	1.0	2	7.8	0.7	209.0	2.0	211.0	1.0	2	14.0	0.7
RC206	239.0	0.8	239.7	0.3	3	1.3	0.2	185.5	39.0	224.6	21.0	2	3.3	0.9	194.7	1.5	196.2	0.8	1	0.9	1.0
RC207	736.2	24.5	760.6	3.3	7	>	0.9	575.1	172.0	747.1	29.9	5	>	1.0	551.8	23.0	574.7	4.2	3	>	1.0
RC208	930.9	110.6	1041.5	11.9	10	>	1.0	692.6	195.8	888.4	28.3	6	>	1.0	544.6	208.6	753.2	38.3	5	>	1.0
Average	**442.6**	**21.9**	**464.5**	**3.3**	**4.6**	**1800.6**	**0.7**	**347.1**	**58.6**	**405.7**	**13.2**	**3.3**	**1451.9**	**0.9**	**310.3**	**37.3**	**347.6**	**9.0**	**2.4**	**1806.2**	**1.0**

Table 12: Results of the stochastic problems for instances in **RC2** with $S = 5$, $\Phi = 10$ and $\delta = 2$, using different values of η.

6. Conclusions

In this paper, we studied the vehicle routing problem with time windows and stochastic demands (VRPTWSD). Two-stage stochastic programming with recourse was used as the approach to address demand uncertainties, where in the first stage the routes are defined and in the second stage they are executed. We considered a fixed rule-based recourse policy that is combined with additional actions to recover from violated time windows. The problem was formulated using a vehicle flow-based model, for which we developed an Integer L-shaped algorithm that was enhanced by a few additional components. Namely, it resorts to LBF cuts based on general partial routes as well as k-path inequalities, which help to improve the lower bound provided by the linear programming relaxation and thus promote a better overall performance. This is the first exact method proposed for the VRPTWSD with the fixed rule-based policy. We additionally showed how to effectively compute the expected recourse cost for the proposed recourse policy as well as how to separate the valid inequalities based on this policy. Computational experiments carried out with benchmark instances from the literature showed that the incorporation of LBF and k-path cuts improved the computational performance of the proposed L-shaped method. These components resulted in more instances solved to optimality and accelerated the convergence of the method. Moreover, the results indicated significant savings in the expected recourse cost for solutions of the stochastic problem with respect to solutions of deterministic approaches based on the expected value. Furthermore, the fixed rule-based policy resulted in an attractive alternative to the classical policy, as it reduces the expected total cost and promotes a better computational performance of the proposed Integer L-shaped algorithm.

There are several interesting perspectives for future research, and some of the most promising are as follows: 1) develop procedures to determine a general lower bound on the expected recourse cost to further

improve the global performance of the proposed Integer L-shaped algorithm; 2) develop exact methods based on branch-price-and-cut for the VRPTWSD; and 3) extend and adapt the solution approach developed here for other VRP variants, such as the variant that also involves deliverymen allocation decisions on each route, as well as the routing and scheduling decisions (Pureza et al., 2012; De La Vega et al., 2019). In addition, extending our methodology to consider the dependence of the stochastic demands or using other methodologies to treat this dependence are interesting topics to be studied in future works. Another interesting topic for future work is to use other policies to govern restocking trips to the depot, such as the optimal restocking policy. Building a procedure to bound the recourse cost is challenging for this policy, especially in the presence of customer time windows. The performance of **LS3** algorithm is likely to be negatively affected if we relax assumptions **A3** and **A4**. By relaxing these assumptions, the 2-path cuts become weaker because they would be added to the model only if one vehicle was not enough to serve all the vertices in $\mathcal{S}$ within their time windows. Additionally, there would be an increase in the number of calls to the separation procedures of the optimality, feasibility and LBF cuts because the solution space of the problem would grow, causing the global time of **LS3** algorithm to increase. For this reason, adjusting our method **LS3** to allow the total expected demand of a route to exceed the vehicle capacity and more than one failure per route is another topic for further research. Finally, another topic for future research is to extend our approach to consider other recourse actions that represent methods to meet customer time windows that cannot be satisfied in the planned routes.

7. Acknowledgments

The authors would like to thank the Editor and anonymous reviewers for their useful comments and suggestions of revision. This work was supported by the Sao Paulo Research Foundation (FAPESP) [grant numbers 15/14582-7, 17/06434-3, 18/01523-0, 19/23596-2, 16/01860-1]; the National Council for Scientific and Technological Development (CNPq) [grant number 313220/2020-4]; and the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) [Finance code 001].

References

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

Beraldi, P., Bruni, M.E., Laganà, D., Musmanno, R., 2015. The mixed capacitated general routing problem under uncertainty. European Journal of Operational Research 240, 382–392.

Bertazzi, L., Secomandi, N., 2018. Faster rollout search for the vehicle routing problem with stochastic demands and restocking. European Journal of Operational Research 270, 487–497.

Bertazzi, L., Secomandi, N., 2020. Technical note—worst-case benefit of restocking for the vehicle routing problem with stochastic demands. Operations Research 68, 671–675.

Bertsimas, D.J., 1992. A vehicle routing problem with stochastic demand. Operations Research 40, 574–585.

Bertsimas, D.J., Jaillet, P., Odoni, A.R., 1990. A priori optimization. Operations Research 38, 1019–1033.

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

Braysy, O., Gendreau, M., 2005. Vehicle routing problem with time windows, Part I: Route construction and local search algorithms. Transportation Science 39, 104–118.

Chang, M., 2005. A vehicle routing problem with time windows and stochastic demands. Journal of the Chinese Institute of Engineers 28, 783–794.

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.

Cook, W., 2015. Concorde tsp solver. https://www.math.uwaterloo.ca/tsp/concorde/index.html.

De La Vega, J., Munari, P., Morabito, R., 2019. Robust optimization for the vehicle routing problem with multiple deliverymen. Central European Journal of Operations Research 27, 905–936.

De La Vega, J., Munari, P., Morabito, R., 2020. Exact approaches to the robust vehicle routing problem with time windows and multiple deliverymen. Computers & Operations Research 124, 105062.

Desaulniers, G., Errico, F., Irnich, S., Schneider, M., 2016. Exact algorithms for electric vehicle-routing problems with time windows. Operations Research 64, 1388–1405.

Desaulniers, G., Lessard, F., Hadjar, A., 2008. Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle routing problem with time windows. Transportation Science 42, 387–404.

Dinh, T., Fukasawa, R., Luedtke, J., 2018. Exact algorithms for the chance-constrained vehicle routing problem. Mathematical Programming 172, 105–138.

Dolan, E.D., More, J.J., 2002. Benchmarking optimization software with performance profiles. Mathematical Programming Series A 91, 201–213.

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

Florio, A., Hartl, R., Minner, S., 2020a. New exact algorithm for the vehicle routing problem with stochastic demands. Transportation Science 54, 855–1152.

Florio, A.M., Hartl, R.F., Minner, S., 2020b. Optimal a priori tour and restocking policy for the single-vehicle routing problem with stochastic demands. European Journal of Operational Research 285, 172–182.

Florio, A.M., Hartl, R.F., Minner, S., Salazar-González, J.J., 2021. A branch-and-price algorithm for the vehicle routing problem with stochastic demands and probabilistic duration constraints. Transportation Science 55, 1–17.

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.

Gendreau, M., Jabali, O., Rei, W., 2014. Stochastic vehicle routing problems, in: Paolo Toth, D.V. (Ed.), Vehicle Routing: Problems, Methods, and Applications. 2 ed.. SIAM, Philadelphia. MOS-SIAM Series on Optimization. chapter 8, pp. 213–239.

Gendreau, M., Jabali, O., Rei, W., 2016. 50th anniversary invited article - Future research directions in stochastic vehicle routing. Transportation Science 50, 1163–1173.

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.

Goldberg, A.V., Tarjan, R.E., 1988. A new approach to the maximum-flow problem. J. ACM 35, 921–940.

Goodson, J.C., Thomas, B.W., Ohlmann, J.W., 2015. Restocking-Based Rollout Policies for the Vehicle Routing Problem with Stochastic Demand and Duration Limits. Transportation Science 50, 591–607.

Gounaris, C.E., Wiesemann, W., Floudas, C.A., 2013. The robust capacitated vehicle routing problem under demand uncertainty. Operations Research 61, 677–693.

Hjorring, C., Holt, J., 1999. New optimality cuts for a single vehicle stochastic routing problem. Annals of Operations Research 86, 569–584.

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.

Jaunich, M.K., Levis, J.W., DeCarolis, J.F., Gaston, E.V., Barlaz, M.A., Bartelt-Hunt, S.L., Jones, E.G., Hauser, L., Jaikumar, R., 2016. Characterization of municipal solid waste collection operations. Resources, Conservation and Recycling 114, 92–102.

Kohl, N., Desrosiers, J., Madsen, O.B.G., Solomon, M.M., Soumis, F., 1999. 2-path cuts for the vehicle routing problem with time windows. Transportation Science 33, 101–116.

Laporte, G., Louveaux, F., Hamme, L., 2002. An integer L-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. Operations Research 50, 415–423.

Laporte, G., Louveaux, F.V., 1993. The integer L-shaped method for stochastic integer programs with complete recourse. Operations Research Letters 13, 133–142.

Latorre-Biel, J.I., Ferone, D., Juan, A.A., Faulin, J., 2021. Combining simheuristics with petri nets for solving the stochastic vehicle routing problem with correlated demands. Expert Systems with Applications 168, 114240.

Lei, H., Laporte, G., Guo, B., 2011. The capacitated vehicle routing problem with stochastic demands and time windows. Computers & Operations Research 38, 1775–1783.

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.

Mendoza, J.E., Rousseau, L.M., Villegas, J.G., 2016. A hybrid metaheuristic for the vehicle routing problem with stochastic demand and duration constraints. Journal of Heuristics 22, 539–566.

Mendoza, J.E., Villegas, J.G., 2013. A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. Optimization Letters 7, 1503–1516.

Munari, P., Moreno, A., De La Vega, J., Alem, D., Gondzio, J., Morabito, R., 2019. The robust vehicle routing problem with time windows: Compact formulation and branch-price-and-cut method. Transportation Science 53, 1–18.

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.

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.

Pessoa, A.A., Poss, M., Sadykov, R., Vanderbeck, F., 2021. Branch-cut-and-price for the robust capacitated vehicle routing problem with knapsack uncertainty. Operations Research .

Pureza, V., Morabito, R., Reimann, M., 2012. Vehicle routing with multiple deliverymen: Modeling and heuristic approaches for the VRPTW. European Journal of Operational Research 218, 636 – 647.

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.

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.

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.

Secomandi, N., Margot, F., 2009. Reoptimization approaches for the vehicle-routing problem with stochastic demands. Operations Research 57, 214–230.

Solomon, M.M., 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research 35, 254–265.

Toth, P., Vigo, D., 2014. Vehicle Routing: Problems, Methods, and Applications, Second Edition. MOS-SIAM Series on Optimization, SIAM.

Yang, W.H., Mathur, K., Ballou, R.H., 2000. Stochastic vehicle routing problem with restocking. Transportation Science 34, 99–112.

Yee, J.R., Golden, B.L., 1980. A note on determining operating strategies for probabilistic vehicle routing. Naval Research Logistics Quarterly 27, 159–163.

Appendix A. An illustrative example

We present an example to illustrate the proposed rule-based policy with additional actions for time window violations. The example is based on instance RC107 (arbitrarily taken from the benchmark instances used in Section 5), which involves 9 demand vertices and 4 identical vehicles available in the depot with capacity 100. Figure A.5 presents two sets of optimal routes for this instance: one for the deterministic problem (on the left) and another for the stochastic problem with fixed rule-based policy and $S = 9$ (on the right). For a given vertex demand profile, we use these routes to illustrate how the depot returns are governed by fixed rule-based and classical policies, and what these returns imply in the violation of the vertex time windows. The demand profile we use is $(0, 18, 34, 13, 38, 24, 17, 22, 14, 20, 0)$, in which the first and last realizations are the demand of vertices 0 and $n + 1$. The first-stage cost is 202.9 for the routes in the solution of the deterministic problem and 203.6 for the routes in the stochastic problem.

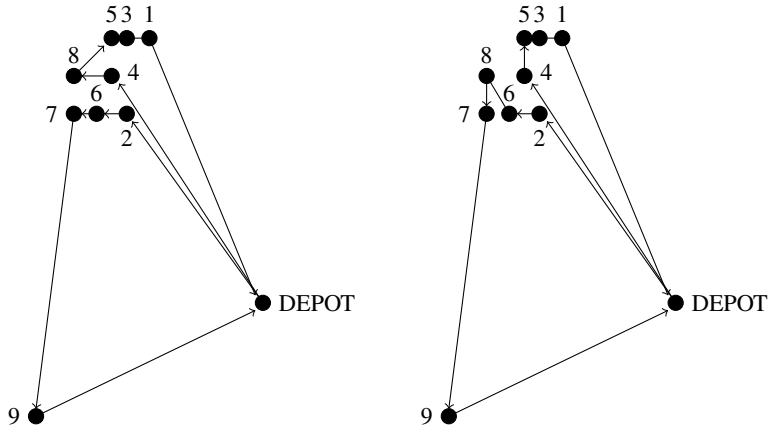

Figure A.5: Optimal routes of instance RC17 for the deterministic problem (on the left) and for the stochastic problem with $S = 9$ (on the right).

Tables A.13 and A.14 summarize the results obtained after applying the recourse actions under classical and fixed rule-based policies in route $(0, 4, 8, 5, 3, 1, n + 1)$ of the deterministic solution and in route $(0, 2, 6, 8, 7, 9, n + 1)$ of the stochastic solution, respectively. The first four columns present the vertex index (Vertex); the expected value of the vertex demand ($\mathbb{E}(\xi)$); the realization of the vertex demand within the considered profile (Profile); and the closing time of the vertex time window (b). Then, for both recourse policies, the tables show the vehicle arrival time at the vertex (t) and the corresponding vehicle load at this time (q). For the fixed rule-based policy, they also report the residual vehicle load ($\tilde{q}$) after serving the vertex as well as the threshold value (θ) used to decide whether a vehicle, after serving a demand vertex, makes a restocking trip to the depot or proceeds directly to the next vertex of the route. We do not apply the recourse policies on the other routes in the solution since the used demand profile does not incur a recourse cost.

For the classical policy, Table A.13 shows that the vehicle arrives at the first vertex (4) in the route with a full load (100 units). After serving this vertex, the vehicle continues its route and visits vertices 8, 5 and 3, arriving at them with 62, 48 and 24 units, respectively. Then, after serving vertex 3, the vehicle continues its route and goes to vertex 1, where it arrives with insufficient load (11) to meet the vertex demand (18). Thus, the vehicle makes a round trip to the depot, returning to vertex 1 at instant 240.2, implying the violation of the vertex time window. In this way, the vehicle does not serve the remaining vertex demand and proceeds directly to vertex $n+1$ where it arrives at instant 242.3, violating the depot time window. The route recourse

				Fixed based-rule policy				Classical policy	
Vertex	$\mathbb{E}(\xi)$	Profile	b	t	q	$\bar{q}$	θ	t	q
0	0	0	240	0.0	100	100	-	0.0	100
4	40	38	193	36.1	100	62	10	36.1	100
8	10	14	155	86.0	62	48	20	86.0	62
5	20	24	113	103.1	48	24	10	103.1	48
3	10	13	146	115.1	24	**11**	20	115.1	24
1	20	18	191	**202.5**	100	100	0	**240.2**	**11**
$n+1$	0	0	**230**	**240.6**	100	-	-	**242.3**	100

Table A.13: Application of recourse policies to route (0,4,8,5,3,1,$n+1$) of the deterministic solution.

				Fixed based-rule policy				Classical policy	
Vertex	$\mathbb{E}(\xi)$	Profile	b	t	q	$\bar{q}$	θ	t	q
0	0	0	240	0.0	100	100	0	0.0	100
2	30	34	97	30.8	100	66	20	30.8	100
6	20	17	164	46.0	66	49	10	46.0	66
8	10	14	155	70.8	49	35	20	70.8	49
7	20	22	118	85.8	35	**13**	20	85.8	35
9	20	20	160	**164.7**	13	100	0	**203.2**	**13**
$n+1$	0	0	**230**	198.3	100	-	-	**236.8**	100

Table A.14: Application of recourse policies to route (0,2,6,8,7,9,$n+1$) of the stochastic solution.

cost for the demand profile used is 371.4 (36.2 for interrupting service at vertex 1, 76.2 for returning to the depot, and 259.0 for violating the time windows of vertices 1 and $n+1$). The total cost is 574.3, of which 202.9 are from the first stage and 371.4 are from the recourse cost or second stage.

Regarding the fixed rule-based policy, the vehicle arrives at vertex 3 with load 24 and leaves with load 11 after serving its demand of 13. As the residual vehicle load at this vertex is lower than the threshold value, the vehicle performs a restocking trip to the depot starting from vertex 3 and returning to vertex 1. The vehicle arrival time at vertex 1 is 202.5, implying the violation of its time window. Hence, the vehicle does not serve this vertex and proceeds directly to final depot $n+1$, where its time window is also violated since the vehicle arrives at 240.6. The recourse cost under the fixed rule-based policy of the route is 336.5 (77.5 for the restocking trip and 259 for violating the time windows of vertices 1 and $n+1$). The total cost is 539.4, 34.9 units smaller than the total cost under the classical policy. Note that on the restocking trip, the cost Γ for interrupting the service at the vertices is saved, which reduces the total cost when compared to the total cost under the classical policy.

When uncertainties are incorporated, the optimal routes of the stochastic problem with fixed rule-based policy change when compared to the optimal routes of the deterministic problem. When applying the recourse actions under classical and fixed rule-based policies to route $(0, 2, 6, 8, 7, 9, n+1)$ of the stochastic problem (see Table A.14), the total cost is smaller when compared to the total cost of the deterministic solution route, changing from 574.3 to 441.1 under the classical policy and from 539.4 to 406.7 under the fixed rule-based policy. This shows the savings that can be achieved if the uncertainties in the vertex demand are incorporated into the problem instead of seeking alternatives such as the expected value to abolish them. The results also show that it is possible to have an additional reduction in the total cost when the fixed rule-based policy is used to govern recourse actions rather than the classical policy, as that policy can reduce the number of failures related to the time window violation. Table A.14, for instance, reports that depot time window $n+1$ is violated under the classical policy but not under the fixed rule-based policy.

Appendix B. Proof of Proposition 4.2

Consider the following 3 cases:

1. Until position ℓ of the artificial route $\tilde{h}$, the arrival times are not affected by the failure in vertex u allocated in position ℓ. Hence, $\tilde{w}_m(\ell \to u)$ coincides with the lower bound of the arrival time $\hat{w}_m$, for $m = 2, \ldots, \ell$;

2. Concerning the position $\ell+1$, if a BF trip is carried out and $\tilde{w}_\ell(\ell \to u) + f_u \leq b_u$, then the vehicle does not violate the time window at vertex u allocated in position ℓ. Therefore, $\tilde{w}_{(\ell+1)}(\ell \to u)$ is equal to $\max\{a_u, \tilde{w}_\ell(\ell \to u)\}$, plus the service time of u allocated in the ℓth position (s_u), the travel time from vertex u to the vertex allocated in position $\ell+1$ of $\tilde{h}$ and the additional time incurred by the return trip from vertex u to the final depot $n+1$ (f_u) . As the actual vertex in position $\ell+1$ can be unknown, we use a lower bound for the travel time at arc $(\ell, \ell+1)$ of $\tilde{h}$, determined as $\tilde{t}_{u(\ell+1)} = \min\limits_{\substack{u' \in \mathcal{C}_h(\ell+1) \\ u \neq u'}} \{t_{uu'}\}$.

   However, if $\tilde{w}_\ell(\ell \to u) + f_u > b_u$ after a BF trip, the vehicle will not attend vertex u allocated in position ℓ; therefore, it proceeds directly to the vertex in the $(\ell+1)$-th position. Thus, $\tilde{w}_{(\ell+1)}(\ell \to u)$ is equal to $\tilde{w}_\ell(\ell \to u)$ plus the travel time between u and the vertex at position $\ell+1$ of $\tilde{h}$ and the additional time incurred by return trip from u_e to the final depot $n+1$ (f_u). As before, the possibility of not knowing the actual vertex in position $\ell+1$ means that we use the lower bound $\tilde{t}_{u(\ell+1)}$ for the travel time between vertices in positions ℓ and $\ell+1$.

   If a restocking trip is carried out, the vehicle returns to the depot after visiting vertex u in position ℓ of $\tilde{h}$ and then proceeds directly to the vertex in position $\ell+1$. Then, $\tilde{w}_{(\ell+1)}(\ell \to u)$ is equal to $\tilde{w}_\ell(\ell \to u)$, plus the service time of u (s_u), the travel time between u and the final depot $n+1$ ($t_{u(n+1)}$) and the travel time from the initial depot 0 to the vertex in position $\ell+1$ of $\tilde{h}$. Again, the actual vertex in position $\ell+1$ may be unknown; therefore, we use a lower bound for the travel time from the initial depot 0 to the vertex in position $\ell+1$, determined as $\tilde{t}_{0(\ell+1)} = \min\limits_{u' \in \mathcal{C}_h(\ell+1)} \{t_{0u'}\}$.

3. Concerning the positions after $\ell+1$, if $\tilde{w}_{(m-1)}(\ell \to u) \leq \hat{b}_{(m-1)}$, for $m = \ell+2, \ldots, k+1$, the vehicle can visit the vertices in these positions inside their estimated time windows. Thus, $\tilde{w}_{(m)}(\ell \to u)$ is equal to $\max\{a_{(m-1)}, \tilde{w}_{(m-1)}(\ell \to u)\}$, plus the service time of the vertex in position $(m-1)$ and the travel time between vertices $v_{(m-1)}$ and v_m, respectively. For this case, the actual vertices at positions $(m-1)$ and m of artificial partial route $\tilde{h}$ may also be unknown. Therefore, we use the previously defined lower bounds for the opening time window at position $m-1$ ($\hat{a}_{(m-1)}$), for the service time at position $m-1$ ($\hat{s}_{(m-1)}$) and fort the travel time of the arc associated with positions $m-1$ and m of $\tilde{h}$ ($\hat{t}_{(m-1)m}$). If $\tilde{w}_{(m-1)}(\ell \to u_e) > \hat{b}_{(m-1)}$, for $m = \ell+2, \ldots, k+1$, the vehicle does not attend the vertex at $(m-1)$, and hence, $\tilde{w}_{(m)}(\ell \to u)$ is equal to $\tilde{w}_{(m-1)}(\ell \to u)$ plus the travel time between vertices $v_{(m-1)}$ and v_m of $\tilde{h}$. Once more, we use the lower bound for the travel time at arc $(m-1, m)$ of $\tilde{h}$ because of possibly not knowing the actual vertex in those positions. Considering this, the proof is completed. □

Appendix C. Proof of Proposition 4.3

For a given value of q such that $q - \xi_u < 0$ (i.e., an RF trip is carried out), Proposition 2 can be applied to determine the new lower bounds for the arrival times $\tilde{w}_m(\ell \to u)$, for $m = 2, \ldots, k+1$. According to the same proposition, the recourse action only affects vertex u in position ℓ and the vertices in the positions

after ℓ. Therefore, the set of positions of the artificial partial route $\tilde{h}$ whose estimated time windows were violated due to the BF trip can be constructed by formula (41). The derivation of (42) is similar, but the recourse now affects the positions after the ℓth position ($m = \ell + 1, \ldots, k + 1$), thus completing the proof. □

Appendix D. Results of the deterministic and stochastic problems

Tables D.15-D.18 report detailed results for all instances and S equal to 3 and 9. Most columns in these tables have the same meaning as in Tables 4 and 5 presented in Section 5.2, except for n, m and Q that define the number of demand vertices, number of vehicles and vehicle capacity, respectively. The symbol "-" in these tables indicates that the corresponding algorithm did not find a feasible solution within the time limit (5 hours). The negative values in the Sav_1 and Sav_2 columns are because the solutions of the deterministic and stochastic problems are not necessarily optimal. Therefore, the total expected cost of a feasible solution of the expected value problem may be less than the expected cost of a feasible solution of the stochastic problem.

Table D.15: Results of the instances in Class 1 with $S = 3$.

				Deterministic							Stochastic with fixed rule-based policy					Stochastic with classical policy								
Inst.	n	m	Q	C	Q_p	$\mathcal{Z}_p$	Q_c	$\mathcal{Z}_c$	Gap (%)	Time (sec.)	C	Q_p	$\mathcal{Z}_p$	Gap (%)	Time (sec.)	C	Q_c	$\mathcal{Z}_c$	Gap (%)	Time (sec.)	Sav_1 (%)	Sav_2 (%)	Sav_3 (%)	Sav_4 (%)
C101	19	8	75	326.0	0.0	326.0	0.0	326.0	0.00	0.2	326.0	0.0	326.0	0.00	0.5	326.0	0.0	326.0	0.00	0.8	0.0	0.00	0.00	0.00
C102	24	9	75	361.3	0.0	361.3	0.0	361.3	0.01	5.5	361.3	0.0	361.3	0.01	16.5	361.3	0.0	361.3	0.01	388.5	0.0	0.00	0.00	0.00
C103	17	6	75	307.5	0.0	307.5	0.0	307.5	0.00	1.7	307.5	0.0	307.5	0.00	1.6	307.5	0.0	307.5	0.01	2.8	0.0	0.00	0.00	0.00
C104	22	9	75	346.2	0.0	346.2	0.0	346.2	0.01	1487.1	346.2	0.0	346.2	0.01	10050.0	346.2	0.0	346.2	0.01	11341.8	0.0	0.00	0.00	0.00
C105	29	12	75	422.2	0.0	422.2	0.0	422.2	0.01	303.2	422.2	0.0	422.2	0.01	688.0	422.2	0.0	422.2	0.01	957.7	0.0	0.00	0.00	0.00
C106	15	6	75	233.7	0.0	233.7	0.0	233.7	0.00	0.2	233.7	0.0	233.7	0.00	0.4	233.7	0.0	233.7	0.00	0.2	0.0	0.00	0.00	0.00
C107	11	5	75	123.5	0.0	123.5	0.0	123.5	0.00	0.0	123.5	0.0	123.5	0.00	0.0	123.5	0.0	123.5	0.00	0.0	0.0	0.00	0.00	0.00
C108	21	9	75	342.9	0.0	342.9	0.0	342.9	0.01	6.1	342.9	0.0	342.9	0.01	17.4	342.9	0.0	342.9	0.01	23.3	0.0	0.00	0.00	0.00
C109	34	12	75	549.1	0	549.1	0	549.1	7.56	18000.2	548.1	0.0	548.1	5.90	13649.5	-	-	-	-	-	0.2	-	-	0.00
R101	24	9	50	617.7	16.8	634.5	61.4	679.2	0.00	1.4	623.3	0.0	623.3	0.00	1.6	623.3	0.0	623.3	0.00	0.5	1.8	8.96	0.00	7.04
R102	29	11	50	625.0	15.0	639.9	65.6	690.6	0.00	45.4	629.5	1.2	630.7	0.01	629.8	629.5	1.9	631.3	0.01	464.9	1.5	9.38	0.10	7.92
R103	19	7	50	418.2	13.7	431.9	63.7	481.9	0.00	0.6	425.0	0.0	425.0	0.01	6.6	425.0	0.0	425.0	0.01	9.5	1.6	13.39	0.00	11.58
R104	14	5	50	332.0	4.0	336.1	14.8	346.8	0.00	0.3	333.5	0.8	334.3	0.01	1.3	333.5	0.8	334.3	0.01	1.5	0.5	3.74	0.00	3.20
R105	11	5	50	291.6	0.0	291.6	0.0	291.6	0.00	0.0	291.6	0.0	291.6	0.00	0.0	291.6	0.0	291.6	0.00	0.0	0.0	0.00	0.00	0.00
R106	35	13	50	746.0	52.0	797.9	186.3	932.2	0.00	910.8	779.2	4.7	783.9	6.41	18000.0	791.5	5.9	797.4	9.78	18000.0	1.8	16.91	1.72	16.83
R107	15	5	50	354.2	0.8	355.0	0.8	355.0	0.00	0.1	354.2	0.8	355.0	0.00	0.2	354.2	0.8	355.0	0.00	0.1	0.0	0.00	0.00	0.00
R108	21	8	50	422.2	16.5	438.7	55.1	477.4	0.00	0.5	423.8	12.2	436.0	0.01	39.3	440.9	8.4	449.3	0.01	3028.9	0.6	6.25	3.05	8.80
R109	19	8	50	418.9	24.1	443.0	98.7	517.7	0.00	0.2	420.5	13.7	434.2	0.01	1.2	447.2	0.0	447.2	0.01	15.2	2.0	15.77	2.98	16.84
R110	31	11	50	657.0	30.5	687.5	48.9	723.9	8.01	18001.2	666.5	3.5	670.0	8.61	18003.5	689.3	5.5	694.8	17.46	18194.8	2.6	4.19	3.70	5.29
R111	44	14	50	949.0	44.9	994.0	114.5	1063.5	11.78	18001.1	1080.2	36.5	1116.7	34.26	18001.2	1085.3	68.3	1153.7	37.99	18000.0	-11.0	-7.82	3.31	6.99
R112	49	16	50	1102.3	87.2	1189.5	98.3	1200.6	17.25	18001.2	-	-	-	-	-	-	-	-	-	-	-	-	-	0.93
RC101	21	8	100	441.8	25.0	466.8	157.4	599.2	0.00	2.4	443.5	17.6	461.1	0.01	3.5	474.2	0.0	474.2	0.01	101.6	1.2	26.34	2.85	28.36
RC102	30	11	100	745.1	40.9	786.0	257.2	1002.3	0.01	309.2	762.1	8.0	770.1	3.99	18001.3	787.0	0.0	787.0	6.55	18000.0	2.1	27.36	2.19	27.52
RC103	16	6	100	327.6	14.4	342.0	90.5	418.1	0.00	0.1	328.7	0.0	328.7	0.00	0.3	328.7	0.0	328.7	0.00	0.1	4.0	27.19	0.00	22.26
RC104	19	7	100	418.2	41.1	459.2	258.6	676.7	0.00	0.1	427.3	7.7	435.0	0.01	7675.6	451.2	0.0	451.2	4.90	18000.0	5.6	49.98	3.74	47.36
RC105	22	8	100	506.9	25.8	532.7	162.5	669.4	0.01	8.1	509.1	15.4	524.5	0.01	1523.9	534.9	0.0	534.9	2.46	18000.0	1.6	25.15	1.98	25.66
RC106	39	13	100	876.5	121.3	997.9	683.9	1560.5	0.56	18000.4	895.4	8.1	903.5	5.85	18000.1	889.2	37.7	926.9	8.86	18000.0	10.4	68.35	2.59	56.38
RC107	9	4	100	202.9	26.8	229.6	147.5	350.4	0.00	0.0	203.6	16.0	219.6	0.01	0.6	231.9	0.0	231.9	0.00	6.4	4.6	51.08	5.61	52.60
RC108	14	5	100	274.0	70.5	344.5	364.0	638.0	0.00	0.1	282.1	22.2	304.3	3.41	18001.6	315.9	0.0	315.9	8.13	18000.0	13.2	101.95	3.81	85.19
Average				**473.8**	**23.1**	**496.9**	**101.0**	**575.4**	**1.56**	**3209.9**	**460.4**	**6.0**	**466.4**	**2.45**	**5082.7**	**466.2**	**4.8**	**471.0**	**3.56**	**5945.9**	**1.58**	**16.60**	**1.39**	**14.85**

Table D.16: Results of the instances in Class 2 with $S = 3$.

				Deterministic							Stochastic with fixed rule-based policy					Stochastic with classical policy								
Inst.	n	m	Q	C	Q_p	$\mathcal{Z}_p$	Q_c	$\mathcal{Z}_c$	Gap (%)	Time (sec.)	C	Q_p	$\mathcal{Z}_p$	Gap (%)	Time (sec.)	C	Q_c	$\mathcal{Z}_c$	Gap (%)	Time (sec.)	Sav_1 (%)	Sav_2 (%)	Sav_3 (%)	Sav_4 (%)
C201	19	8	75	404.4	0.0	404.4	0.0	404.4	0.01	6.6	404.4	0.0	404.4	0.01	10.7	404.4	0.0	404.4	0.01	13.9	0.00	0.00	0.00	0.00
C202	21	8	75	412.6	0.0	412.6	0.0	412.6	0.01	31.0	412.6	0.0	412.6	0.01	104.0	412.6	0.0	412.6	0.01	152.6	0.00	0.00	0.00	0.00
C203	17	6	75	368.2	0.0	368.2	0.0	368.2	0.01	11.8	368.2	0.0	368.2	0.01	20.0	368.2	0.0	368.2	0.01	34.2	0.00	0.00	0.00	0.00
C204	24	9	75	432.9	0.0	432.9	0.0	432.9	0.01	13255.9	433.3	0.0	433.3	1.57	18000.0	433.3	0.0	433.3	4.37	18000.0	-0.08	-0.08	0.00	0.00
C205	30	11	75	543.0	0.0	543.0	0.0	543.0	11.21	18001.0	544.5	0.0	544.5	13.28	18000.1	544.9	0.0	544.9	12.98	18000.0	-0.29	-0.36	0.07	0.00
C206	14	6	75	254.6	0.0	254.6	0.0	254.6	0.00	0.2	254.6	0.0	254.6	0.00	0.5	254.6	0.0	254.6	0.00	0.2	0.00	0.00	0.00	0.00
C207	10	5	75	183.8	0.0	183.8	0.0	183.8	0.00	0.0	183.8	0.0	183.8	0.00	0.0	183.8	0.0	183.8	0.00	0.0	0.00	0.00	0.00	0.00
C208	39	13	75	908.7	0.0	908.7	0.0	908.7	43.01	18001.2	810.2	0.0	810.2	26.75	14823.9	-	-	-	-	-	12.16	-	-	0.00
R201	29	11	50	624.9	18.0	642.9	63.7	688.6	0.01	14.4	625.3	10.0	635.3	0.01	440.4	636.2	2.7	638.9	0.01	681.28	1.20	7.77	0.57	7.10
R202	24	9	50	528.2	15.9	544.1	57.9	586.1	0.01	1.2	528.2	15.9	544.1	0.01	36.0	551.9	0.2	552.1	0.01	241.4	0.00	6.16	1.48	7.73
R203	16	6	50	365.1	4.1	369.2	6.2	371.3	0.01	0.1	365.1	3.6	368.7	0.00	0.2	365.1	5.6	370.8	0.00	0.5	0.14	0.14	0.56	0.56
R204	11	5	50	258.8	8.3	267.1	20.5	279.3	0.00	0.1	259.8	1.3	261.1	0.00	0.1	259.8	1.3	261.1	0.00	0.1	2.29	6.96	0.00	4.56
R205	32	12	50	693.2	18.7	711.9	78.9	772.1	0.01	2633.5	699.1	7.0	706.1	3.06	18000.0	711.1	5.7	716.8	6.68	19619.3	0.83	7.71	1.52	8.45
R206	15	6	50	344.3	2.7	347.0	2.7	347.0	0.00	0.1	344.3	2.7	347.0	0.00	0.1	344.3	2.7	347.0	0.00	0.1	0.00	0.00	0.00	0.00
R207	23	8	50	497.9	14.0	511.8	38.7	536.6	0.01	1.5	497.9	13.3	511.1	0.01	60.5	515.0	8.5	523.5	0.01	3956.8	0.14	2.50	2.42	4.84
R208	19	7	50	409.4	11.5	420.9	46.7	456.0	0.00	0.3	410.9	7.7	418.6	0.01	8.2	425.0	0.0	425.0	0.01	370.5	0.56	7.31	1.53	8.35
R209	31	11	50	641.2	17.8	659.0	75.6	716.8	0.01	9699.2	641.2	17.8	659.0	7.65	18001.2	679.6	0.0	679.6	12.61	18000.0	0.00	5.48	3.12	8.77
R210	41	13	50	1019.0	18.3	1037.3	45.8	1064.8	29.45	18002.1	898.9	26.1	925.0	15.58	18000.8	-	-	-	-	-	12.14	-	-	2.65
R211	49	17	50	1092.9	60.3	1153.2	86.9	1179.8	52.02	18001.2	-	-	-	-	-	-	-	-	-	-	-	-	-	2.31
RC201	16	6	100	334.4	0.0	334.4	0.0	334.4	0.00	0.0	334.4	0.0	334.4	0.00	0.0	334.4	0.0	334.4	0.00	0.1	0.00	0.00	0.00	0.00
RC202	12	5	100	267.7	7.2	274.9	39.5	307.2	0.00	0.0	268.8	6.0	274.7	0.00	0.4	302.9	0.0	302.9	0.01	11328.10	0.05	1.41	10.26	11.75
RC203	20	8	100	424.2	12.4	436.6	78.1	502.3	0.00	0.1	425.4	6.1	431.5	0.01	11.7	457.2	0.0	457.2	5.43	18000.0	1.19	9.87	5.95	15.04
RC204	24	9	100	498.1	23.6	521.8	132.6	630.7	0.00	0.1	505.4	6.9	512.3	0.68	18000.6	537.1	0.0	537.1	6.02	18000.0	1.86	17.44	4.84	20.88
RC205	10	5	100	233.9	6.7	240.5	42.1	275.9	0.00	0.0	235.0	0.0	235.0	0.00	0.0	235.0	0.0	235.0	0.01	0.0	2.36	17.41	0.00	14.71
RC206	11	5	100	239.0	0.0	239.0	0.0	239.0	0.00	0.0	239.0	0.0	239.0	0.00	0.0	239.0	0.0	239.0	0.00	0.0	0.00	0.00	0.00	0.00
RC207	34	11	100	740.4	22.7	763.1	143.0	883.4	4.08	18001.3	742.1	6.6	748.7	5.49	18002.8	773.6	0.0	773.6	9.21	18000.0	1.93	14.19	3.33	15.76
RC208	44	14	100	908.3	48.5	956.8	304.6	1212.9	4.31	18001.7	921.7	21.6	943.2	8.44	18000.2	919.0	128.1	1047.1	20.34	18000.0	1.44	15.83	11.01	26.77
Average				**504.8**	**11.5**	**516.3**	**46.8**	**551.6**	**0.1**	**4950.5**	**475.1**	**5.9**	**481.0**	**3.18**	**6135.5**	**453.7**	**6.5**	**460.1**	**3.24**	**6766.6**	**1.46**	**4.99**	**1.94**	**5.93**

Table D.17: Results of the instances in Class 1 with $S = 9$.

				Deterministic							Stochastic with fixed rule-based policy					Stochastic with classical policy								
Inst.	n	m	Q	C	Q_p	$\mathcal{Z}_p$	Q_c	$\mathcal{Z}_c$	Gap (%)	Time (sec.)	C	Q_p	$\mathcal{Z}_p$	Gap (%)	Time (sec.)	C	Q_c	$\mathcal{Z}_c$	Gap (%)	Time (sec.)	Sav_1 (%)	Sav_2 (%)	Sav_3 (%)	Sav_4 (%)
C101	19	8	75	326.0	28.6	354.6	45.6	371.5	0.00	0.2	328.5	24.8	353.3	0.01	18.7	337.2	29.2	366.4	0.01	394.0	0.36	1.39	3.73	4.79
C102	24	9	75	361.3	23.5	384.8	40.2	401.5	0.01	5.5	366.8	16.0	382.7	0.01	4409.8	368.4	30.0	398.4	6.99	18592.6	0.55	0.78	4.09	4.32
C103	17	6	75	307.5	15.2	322.7	25.2	332.7	0.00	1.7	310.6	6.8	317.4	0.01	9.4	312.4	8.7	321.1	0.01	53.6	1.65	3.61	1.17	3.11
C104	22	9	75	346.2	15.5	361.7	26.6	372.8	0.01	1487.1	346.2	15.2	361.4	2.30	18001.5	346.2	26.5	372.7	10.15	18000.0	0.07	0.02	3.11	3.07
C105	29	12	75	422.2	19.8	442.0	36.3	458.5	0.01	303.2	423.7	18.0	441.7	0.00	18000.0	429.5	26.0	455.6	7.19	18000.0	0.07	0.65	3.14	3.73
C106	15	6	75	233.7	15.5	249.2	25.7	259.4	0.00	0.2	238.2	7.6	245.8	0.00	1.1	241.7	7.4	249.0	0.00	1.8	1.37	4.16	1.30	4.09
C107	11	5	75	123.5	2.5	126.0	6.0	129.5	0.00	0.0	124.3	0.0	124.3	0.00	0.1	124.3	0.0	124.3	0.00	0.0	1.39	4.21	0.00	2.77
C108	21	9	75	342.9	19.0	361.9	33.6	376.4	0.01	6.1	342.9	19.0	361.9	0.01	271.8	349.0	26.1	375.1	0.42	18000.3	0.00	0.36	3.65	4.02
C109	34	12	75	549.1	27.1	576.2	38.6	587.7	7.56	18002.1	553.0	24.6	577.6	12.44	18001.3	-	-	-	-	-				2.00
R101	24	9	50	617.7	60.9	678.7	136.7	754.5	0.00	1.4	623.3	25.6	649.0	0.00	2.5	623.3	33.3	656.7	0.00	1.0	4.57	14.90	1.19	11.17
R102	29	11	50	625.0	42.8	667.8	131.3	756.3	0.01	45.4	625.2	40.0	665.9	1.15	18001.7	654.8	36.1	690.8	7.93	18000.0	0.29	9.48	3.74	13.25
R103	19	7	50	418.2	23.7	441.9	95.1	513.2	0.01	0.6	418.2	23.7	441.9	0.01	100.5	439.0	10.6	449.6	0.01	168.6	0.00	14.15	1.76	16.15
R104	14	5	50	332.0	4.4	336.5	19.2	351.2	0.00	0.3	332.0	4.4	336.4	0.01	3.0	338.1	2.9	341.0	0.01	7.2	0.02	2.98	1.37	4.37
R105	11	5	50	291.6	1.1	292.7	1.6	293.2	0.00	0.0	291.6	1.1	292.7	0.00	0.0	291.6	1.6	293.2	0.01	0.0	0.00	0.00	0.16	0.16
R106	35	13	50	746.0	100.9	846.8	353.0	1098.9	0.01	910.8	777.3	55.1	832.3	13.89	18001.5	815.6	69.8	885.5	22.54	18000.0	1.75	24.11	6.39	29.77
R107	15	5	50	354.2	6.2	360.4	14.1	368.3	0.00	0.1	354.2	6.2	360.4	0.01	0.5	354.2	14.1	368.3	0.00	1.8	0.00	0.00	2.18	2.18
R108	21	8	50	422.2	30.0	452.2	102.0	524.2	0.01	0.5	423.7	24.8	448.5	0.01	326.5	451.9	36.9	488.8	10.90	18000.0	0.83	7.25	8.98	15.91
R109	19	8	50	418.9	29.7	448.7	130.8	549.7	0.00	0.2	420.5	21.3	441.8	0.01	3.4	448.0	6.2	454.2	0.01	20.0	1.56	21.02	2.81	22.52
R110	31	11	50	657.0	71.7	728.7	197.2	854.2	8.01	18002.1	661.4	50.4	711.9	17.03	9795.0	726.8	21.1	747.9	25.41	18000.0	2.36	14.22	5.05	17.22
R111	44	14	50	949.0	110.2	1059.2	330.6	1279.6	11.78	18001.1	1085.5	84.2	1169.7	40.23	18016.2	1132.5	227.9	1360.4	63.44	18000.0	-9.45	-5.94	16.30	20.80
R112	49	16	50	1102.3	131.1	1233.4	213.5	1315.8	17.25	18002.3	-	-	-	-	-	-	-	-	-	-				6.68
RC101	21	8	100	441.8	31.6	473.4	187.1	628.9	0.00	2.4	443.7	22.5	466.2	0.01	9.2	475.9	8.1	483.9	0.01	682.9	1.55	29.96	3.80	32.84
RC102	30	11	100	745.1	50.6	795.8	307.6	1052.7	0.01	309.2	762.1	14.9	777.0	4.90	18001.4	782.2	9.0	791.2	7.20	18000.0	2.41	33.06	1.82	32.29
RC103	16	6	100	327.6	16.0	343.6	107.6	435.2	0.00	0.1	329.3	0.3	329.5	0.01	0.2	329.3	0.3	329.5	0.00	0.2	4.27	32.06	0.00	26.65
RC104	19	7	100	418.2	48.5	466.7	307.0	725.1	0.00	0.1	428.8	8.4	437.2	0.01	12445.3	452.0	0.2	452.2	4.68	18000.0	6.75	60.36	3.43	55.37
RC105	22	8	100	506.9	30.7	537.6	193.9	700.8	0.01	8.1	509.1	19.0	528.0	0.01	7149.1	534.9	3.1	538.0	2.54	18000.0	1.81	30.27	1.88	30.36
RC106	39	13	100	876.5	146.3	1022.8	828.1	1704.7	0.56	18000.4	886.6	28.0	914.6	7.05	18000.0	894.8	59.0	953.8	12.09	18000.0	11.83	78.72	4.29	66.66
RC107	9	4	100	202.9	30.4	233.2	179.3	382.2	0.00	0.0	203.6	18.3	222.0	0.01	1.0	232.5	0.0	232.5	0.00	7.5	5.08	64.36	4.76	63.86
RC108	14	5	100	274.0	77.6	351.6	444.7	718.7	0.00	0.1	282.1	24.6	306.7	4.49	18002.0	315.9	0.4	316.3	7.91	18000.0	14.64	127.23	3.11	104.38
Average				**473.8**	**41.8**	**515.5**	**157.2**	**630.9**	**1.56**	**3210.0**	**460.4**	**21.6**	**482.1**	**3.70**	**7020.5**	**474.1**	**25.7**	**499.9**	**7.02**	**9404.9**	**2.06**	**21.23**	**3.45**	**20.85**

Table D.18: Results of the instances in Class 2 with $S = 9$.

				Deterministic							Stochastic with fixed rule-based policy					Stochastic with classical policy								
Inst.	n	m	Q	C	Q_p	$\mathcal{Z}_p$	Q_c	$\mathcal{Z}_c$	Gap (%)	Time (sec.)	C	Q_p	$\mathcal{Z}_p$	Gap (%)	Time (sec.)	C	Q_c	$\mathcal{Z}_c$	Gap (%)	Time (sec.)	Sav_1 (%)	Sav_2 (%)	Sav_3 (%)	Sav_4 (%)
C201	19	8	75	404.4	16.8	421.2	28.6	433.0	0.01	6.6	405.7	10.4	416.1	0.01	16.6	408.8	12.2	421.0	0.01	36.9	1.23	2.85	1.17	2.79
C202	21	8	75	412.6	13.2	425.7	20.9	433.5	0.01	31.0	412.7	10.3	422.9	0.01	163.8	415.7	12.0	427.7	0.01	566.1	0.67	1.35	1.14	1.82
C203	17	6	75	368.2	8.4	376.6	12.0	380.2	0.01	11.8	369.9	6.3	376.3	0.01	84.2	371.5	8.1	379.5	0.00	275.4	0.08	0.17	0.87	0.96
C204	24	9	75	432.9	18.7	451.6	34.0	466.9	0.01	13255.9	433.3	17.7	451.1	6.19	18001.8	-	-	-	-	-	0.11	-		3.39
C205	30	11	75	543.0	15.2	558.1	24.7	567.7	11.21	18001.0	547.9	18.5	566.4	15.83	18000.9	560.8	16.0	576.8	22.24	18000.1	-1.46	-1.58	1.84	1.71
C206	14	6	75	254.6	5.3	259.9	10.0	264.7	0.00	0.2	254.6	5.3	259.9	0.01	0.6	254.6	10.0	264.7	0.00	0.6	0.00	0.00	1.82	1.82
C207	10	5	75	183.8	2.0	185.8	3.7	187.5	0.00	0.0	183.8	2.0	185.8	0.00	0.0	183.8	3.7	187.5	0.00	0.0	0.00	0.00	0.95	0.95
C208	39	13	75	908.7	27.0	935.8	46.3	955.0	43.01	18001.2	870.1	14.0	884.1	38.13	6114.2	-	-	-	-	-		-		2.05
R201	29	11	50	624.9	55.8	680.7	158.2	783.1	0.01	14.4	633.5	23.0	656.6	0.01	8730.3	653.9	16.4	670.3	4.53	18006.20	3.68	16.83	2.09	15.04
R202	24	9	50	528.2	32.8	561.0	110.9	639.1	0.01	1.2	528.2	32.5	560.7	0.01	367.0	551.9	33.5	585.4	3.49	18000.0	0.04	9.17	4.40	13.92
R203	16	6	50	365.1	20.5	385.6	50.1	415.2	0.01	0.1	365.1	17.0	382.1	0.01	3.0	395.0	0.0	395.0	0.01	34.9	0.90	5.12	3.37	7.69
R204	11	5	50	258.8	12.7	271.5	50.4	309.1	0.00	0.1	265.5	0.9	266.4	0.00	0.1	265.5	1.1	266.7	0.00	0.3	1.91	15.92	0.10	13.86
R205	32	12	50	693.2	41.0	734.2	138.4	831.6	0.01	2633.5	701.7	34.6	736.4	10.38	18001.6	728.0	53.4	781.5	20.39	18000.0	-0.29	6.41	6.13	13.26
R206	15	6	50	344.3	6.9	351.2	14.5	358.8	0.00	0.1	344.3	6.9	351.2	0.00	0.2	344.3	14.5	358.8	0.01	1.3	0.00	0.00	2.16	2.16
R207	23	8	50	497.9	34.9	532.8	101.5	599.3	0.01	1.5	497.9	33.1	531.0	0.01	1651.1	542.5	29.8	572.3	10.83	18000.0	0.34	4.72	7.78	12.49
R208	19	7	50	409.4	19.3	428.6	67.7	477.0	0.00	0.3	410.8	14.9	425.7	0.01	49.5	444.4	4.8	449.1	1.13	18000.0	0.68	6.21	5.49	11.29
R209	31	11	50	641.2	44.5	685.7	145.2	786.5	0.01	9699.2	641.2	45.9	687.1	13.18	18003.3	684.8	50.5	735.3	22.94	18000.0	-0.20	6.95	7.01	14.69
R210	41	13	50	1019.0	55.7	1074.7	78.3	1097.3	29.45	18001.1	925.2	73.5	998.7	25.17	18001.2	1058.2	123.1	1181.4	49.43	18000.0	7.61	-7.12	18.29	2.10
R211	49	17	50	1092.9	122.5	1215.4	175.9	1268.8	52.02	18002.3	-	-	-	-	-	-	-	-	-	-		-		4.39
RC201	16	6	100	334.4	0.4	334.8	0.4	334.8	0.00	0.0	334.4	0.4	334.8	0.00	0.0	334.4	0.4	334.8	0.00	0.0	0.00	0.00	0.01	0.01
RC202	12	5	100	267.7	8.2	276.0	48.1	315.8	0.00	0.0	268.8	6.7	275.5	0.01	0.5	302.9	0.4	303.3	0.01	9039.07	0.19	4.13	10.11	14.44
RC203	20	8	100	424.2	14.0	438.2	92.8	517.0	0.00	0.1	425.4	7.4	432.8	0.01	20.3	457.6	0.7	458.4	5.37	18000.0	1.26	12.79	5.91	17.97
RC204	24	9	100	498.1	27.0	525.2	159.7	657.8	0.00	0.1	505.4	8.8	514.1	0.71	18000.2	540.5	1.4	541.9	6.86	18000.1	2.15	21.39	5.40	25.26
RC205	10	5	100	233.9	7.7	241.5	50.0	283.8	0.00	0.0	235.0	0.3	235.3	0.00	0.0	235.0	0.3	235.4	0.00	0.0	2.63	20.60	0.01	17.52
RC206	11	5	100	239.0	0.4	239.3	0.4	239.4	0.00	0.0	239.0	0.4	239.3	0.00	0.0	239.0	0.4	239.4	0.00	0.0	0.00	0.00	0.01	0.01
RC207	34	11	100	740.4	28.8	769.1	172.0	912.3	4.08	18001.3	742.1	9.4	751.4	5.81	18001.3	773.6	5.2	778.8	9.97	18000.0	2.36	17.14	3.65	18.62
RC208	44	14	100	908.3	58.8	967.1	362.3	1270.7	4.31	18001.7	915.9	25.6	941.6	8.21	18000.1	955.7	150.4	1106.1	27.37	18000.0	2.71	14.88	17.47	31.38
Average				**504.8**	**25.9**	**530.7**	**79.9**	**584.7**	**5.34**	**4950.5**	**479.1**	**16.4**	**495.5**	**4.76**	**6200.5**	**487.6**	**22.9**	**510.5**	**7.69**	**9415.0**	**1.06**	**6.58**	**4.47**	**9.32**