New Valid Inequalities for the Multi-Vehicle Routing Problem with Stochastic Demands

Ola Jabali
Walter Rei
Michel Gendreau
Gilbert Laporte

October 2012

CIRRELT-2012-58

New Valid Inequalities for the Multi-Vehicle Routing Problem with Stochastic Demands

Ola Jabali[1,2,*], Walter Rei[1,3], Michel Gendreau[1,4], Gilbert Laporte[1,5]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Logistics and Operations Research Management, HEC Montréal, 3000 Côte-Sainte-Catherine, Montréal, Canada H3T 2A7

[3] Department of Management and Technology, Université du Québec à Montréal, P.O. Box 8888, Station Centre-Ville, Montréal, Canada H3C 3P8

[4] Department of Mathematics and Industrial Engineering, École Polytechnique de Montréal, P.O. Box 6079, Station Centre-ville, Montréal, Canada H3C 3A7

[5] Department of Management Sciences, HEC Montréal, 3000 Côte-Sainte-Catherine, Montréal, Canada H3T 2A7

Abstract. This paper describes an exact algorithm for a variant of the vehicle routing problem in which customer demands are stochastic. Demands are revealed upon arrival at customer locations. As a result, a vehicle may reach a customer and not have sufficient capacity to collect the realized demand. Such a situation is referred to as a failure. In this paper the following recourse action is then applied when failure occurs: the vehicle returns to the depot to unload and resumes its planned route at the point of failure. The capacitated vehicle routing problem with stochastic demands (VRPSD) consists of minimizing the sum of the planned routes cost and of the expected recourse cost. The VRPSD is formulated as a two-stage stochastic programming model and solved by means of the integer L-shaped algorithm. This paper introduces three lower bounding functional based on the generation of general partial routes, as well as an exact separation procedure to identify violated cuts. Extensive computational results confirm the effectiveness of the proposed algorithm, as measured by a substantial reduction in the number of feasible solutions that have to be explicitly eliminated. This translates into a higher proportion of instances solved to optimality, reduced optimality gaps, and lower computing times.

Keywords: Stochastic vehicle routing problem, stochastic demand, stochastic programming, integer L-shaped algorithm, general partial routes, lower bounding functionals.

Acknowledgements. Thanks are due to Serge Bisaillon for his outstanding assistance with the implementation of the algorithm and Marco Veneroni for his valuable comments. This research was partly supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grants 338816-10, 39682-10, and 227837-04. This support is gratefully acknowledged.

---

* Corresponding author: Ola.Jabali@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2012

1. Introduction

The aim of the vehicle routing problem (VRP) is to construct vehicle routes through a set of customers under side constraints. In the classical VRP, a travel cost matrix between customers is provided, several identical capacitated vehicles are available, and customers have demands to be collected. The routes start and end at the depot, each customer is visited exactly once by a single vehicle, and the demand collected on a route cannot exceed the vehicle capacity. The objective is to minimize the total travel cost.

The VRP has been extensively studied (see, e.g., Laporte [18]). A number of variants of the VRP have been proposed (see, e.g., Toth and Vigo [28] and Golden et al. [12]) to represent more realistic settings, and several efficient solution procedures have been developed for the VRP and its variants. Most of this research deals with deterministic settings, thus implicitly implying that all information concerning the instance parameters is known when the problem is solved. This assumption applies to situations where the estimated variability in the problem parameters is relatively low. However, in practice, data are often stochastic. In such contexts solving a deterministic problem in which stochastic parameters are replaced by their expected values can yield poor solutions (Louveaux [22]). As a result, several stochastic versions of the VRP have been studied in recent years (Cordeau et al. [9]).

In this paper we solve the VRP with stochastic demands (VRPSD) in which demand is only revealed when the vehicle reaches the customer's location. Such problems occur in a number of applications, for example in the delivery and collection of money to and from banks (Bertsimas [4] and Lambert et al. [17]), in home oil delivery (Chepuri and Homem de Mello [7]), beer distribution and garbage collection (Yang et al. [29]).

Because demand is not known beforehand, a vehicle may reach a customer location with insufficient residual capacity to collect the observed demand. This causes a *route failure*, in which case a *recourse* action can be implemented. One of the most common solution frameworks for this class of problems is *a priori optimization*, a concept initially put forward by Bertsimas [6], Jaillet [15] and Bertsimas et al. [5]. It consists of modeling the problem in two stages. In the first stage, a planned, or *a priori* solution is designed, before customer demands are known. It consists of the set of vehicle routes. In the second stage, these routes are performed as demands are gradually revealed. Whenever a failure occurs, a predetermined recourse policy is implemented, which entails an extra recourse cost. The objective of the problem is to minimize the cost of the first-stage solution plus the expected cost of recourse.

Several recourse policies have been proposed for the VRPSD. A *classical* policy is to return to the depot upon failure, offload, and resume collections by following the planned route starting at the point of failure (Christiansen and Lysgaard [8], Gendreau et al. [10], [11], Goodson et al. [13], Hjorring and Holt [14], Laporte et al. [20], Lei et al. [21] and Rei et al. [24]). More involved recourse policies have also been considered, such as restocking rules (Yang et al. [29]), route reoptimization (Secomandi and Margot [25]), pairing strategies (Ak and Erera [1]), and the use of safety stocks (Juan et al. [16]) . Note that the algorithms described in Ak and Erera [1], Hjorring and Holt [14], and Secomandi and Margot [25] have been implemented for the single-vehicle case only. An important managerial advantage of the classical policy is that it yields stable routes which require minimal alterations in the event of failure. Solving the VRPSD under the classical recourse policy also provides a benchmark against which alternative policies can be assessed. Whereas most authors in the field of stochastic vehicle routing cast the problem in the context of stochastic programming, one recent study by Sungur et al. [27] defines it in the context of robust optimization which yields routes that minimize transportation costs while satisfying all demands in a given bounded uncertainty set.

Compared with the classical VRP, the VRPSD is considerably more difficult to solve. For example, state-of-the-art algorithms for the VRP can handle instances involving up to 200 customers and 17 vehicles (Baldacci et al. [2]). In contrast, the best available algorithms for the VRPSD (with multiple vehicles) can only handle instances with 50 customers and three vehicles under a normal demand distribution (Laporte et al. [20]).

As in Laporte et al. [20], we formulate the VRPSD as a two-stage stochastic programming model under the classical recourse policy. We solve it optimally by means of the integer L-shaped method proposed by Laporte and Louveaux [19], an extension of the L-shaped method of Van Slyke and Wets [26] for continuous stochastic programs, itself an application of Benders decomposition [3] to stochastic programming. The integer L-shaped method follows a branch-and-cut framework in which lower optimality cuts are generated to eliminate feasible solutions, and lower bounding functionals (LBFs) are commonly used to improve the efficiency of the algorithm. Their role is to tighten the linear relaxation of the current subproblem, thus stemming the growth of the search tree. When applied to the VRPSD, LBFs are used to strengthen the lower bounds on the recourse cost associated to partial routes encountered throughout the solution process. As was numerically illustrated by Hjorring and Holt [14] and Laporte et al. [20], the use of LBFs is instrumental in optimally solving the VRPSD.

This paper makes three main scientific contributions. It first generalizes the concept of partial routes defined by Hjorring and Holt [14] for the single vehicle case, and by Laporte et al. [20] for the multi-vehicle case. It then proposes strengthened LBFs based on these generalized partial routes. Finally, it describes an exact separation algorithm for these LBFs. Extensive computational experiments on benchmark instances demonstrate that the combination of these improvements yields shorter computing times, thus enabling the exact solution of larger instances than what was previously possible. Moreover, it yields smaller optimality gaps on the unsolved instances.

The remainder of this paper is organized as follows. In Section 2 we present our modeling and solution framework for the VRPSD. Section 3 introduces the generalized definition of partial routes. These are used to generate stronger LBFs in Section 4. The exact separation procedure for the LBFs is presented in Section 5. This is followed by computational results in Section 6 and by conclusions in Section 7.

2. Model and algorithmic framework

We recall in Section 2.1 the two-stage stochastic programming formulation of VRPSD initially proposed by Laporte et al. [20], which constitutes the backbone of our solution framework. We then describe in Section 2.2 the integer L-shaped algorithm for which we introduce improvements in Sections 3, 4, and 5.

2.1 The VRPSD Model

The VRPSD is defined on a complete undirected graph $G = (V, E)$, where $V = \{v_1, \dots, v_n\}$ is the vertex set and $E = \{(v_i, v_j) : v_i, v_j \in V,\ i < j\}$ is the edge set. Vertex v_1 is the depot at which m identical vehicles of capacity D are based, whereas the remaining vertices represent customers. Each customer has a non-negative stochastic demand ξ_i to be collected. We further assume that these demands are identically and independently distributed with expected values μ_i. A travel cost c_{ij} is associated with each edge $(v_i, v_j) \in E$. The aim of the first-stage problem is to design m vehicle routes of least expected cost 1) starting and ending at the depot, 2) such that each customer is visited exactly once by exactly one vehicle, and 3) such that the expected demand of each route does not exceed the vehicle capacity. In this definition the objective function is the sum of the planned routing cost and of the expected second-stage cost of recourse. The third constraint was originally imposed by Laporte et al. [20] in order to avoid the creation of routes that would systematically fail while some vehicles would be underutilized.

The VRPSD can be formulated as a two-index stochastic program as follows. Let x_{ij} $(i < j)$ be an integer decision variable equal to the number of times edge (v_i, v_j) appears in the first-stage solution. In what follows x_{ij} must be interpreted as x_{ji} whenever $i > j$. If $i, j > 1$, then x_{ij} can only take the values 0 or 1; if $i = 1$, then x_{ij} can also be equal to 2 whenever a vehicle makes a return trip between the depot and customer v_j. Furthermore, let $\mathcal{Q}(x)$ denote the expected cost of recourse in solution x. The model is then

$$\text{(VRPSD)} \qquad \text{Minimize} \sum_{i<j} c_{ij} x_{ij} + \mathcal{Q}(x) \tag{1}$$

subject to

$$\sum_{j=2}^{n} x_{1j} = 2m, \tag{2}$$

$$\sum_{i<k} x_{ik} + \sum_{j>k} x_{kj} = 2, \qquad (k = 2, \ldots, n), \tag{3}$$

$$\sum_{v_i, v_j \in S} x_{ij} \leq |S| - \left\lceil \sum_{v_i \in S} \mu_i / D \right\rceil, \qquad (S \subset V \setminus \{v_1\}, 2 \leq |S| \leq n-2) \tag{4}$$

$$0 \leq x_{ij} \leq 1 \qquad (2 \leq i < j \leq n), \tag{5}$$

$$0 \leq x_{1j} \leq 2 \qquad (j = 2, \ldots, n), \tag{6}$$

$$x = (x_{ij}) \text{ integer} \qquad (1 \leq i < j \leq n). \tag{7}$$

In this model constraints (2) and (3) specify the degree of each vertex, whereas constraints (4) eliminate subtours and ensure that the expected demand of any route does not exceed the vehicle capacity.

Given a first-stage solution x, the computation of $\mathcal{Q}(x)$ is separable in the routes. It is well known that the expected cost of route k depends on its orientation:

$$\mathcal{Q}(x) = \sum_{k=1}^{m} \min\{\mathcal{Q}^{k,1}, \mathcal{Q}^{k,2}\}, \tag{8}$$

where $\mathcal{Q}^{k,\delta}$ denotes the expected cost of the recourse of route k for orientation δ. Given an undirected first stage solution, the expected cost of recourse is computed for each direction, and the cheapest orientation is selected. As in Laporte et al. [20], the computation of $\mathcal{Q}^{k,1}$

for route k defined by $(v_{i_1} = v_1, v_{i_2}, \dots, v_{i_{t+1}} = v_1)$ is given by

$$\mathcal{Q}^{k,1} \;=\; 2\sum_{j=2}^{t}\sum_{l=1}^{j-1} P\Big(\sum_{s=2}^{j-1}\xi_{i_s} \leq lD < \sum_{s=2}^{j}\xi_{i_s}\Big)c_{1i_j}. \tag{9}$$

The first factor in the double summation is the probability of incurring the l^{th} failure at customer v_{i_j}. The value of $\mathcal{Q}^{k,2}$ is computed likewise by reversing the orientation of route k.

2.2 The integer L-shaped algorithm

The integer L-shaped algorithm applies branch-and-cut to a relaxation of (VRPSD) in which the recourse term $\mathcal{Q}(x)$ is bounded below by a variable Θ. In addition, the subtour elimination constraints and the integrality requirements are relaxed. Initially Θ is set greater than or equal to a lower bound L on the expected cost of recourse, and its value Θ^ν is computed for the solution of the subproblem solved at iteration ν. We adopt the general lower bound L on $\mathcal{Q}(x)$ described in Proposition 1 of Laporte et al. [20]. This bound is based on the computation of the probability of failure on each route taken separately. The recourse cost is bounded below by considering the m customers closest to the depot, and the total demand is then partitioned among the m vehicles so as to minimize the total cost. As is standard in branch-and-cut, subtour elimination constraints are generated dynamically as they are found to be violated, and integrality is gradually recovered by branching on fractional variables. Optimality cuts are generated at feasible integer solutions. In most applications these cuts are local, and therefore relying solely on them may yield substantial enumeration in the search tree. This is why it often pays to also impose lower bounds on the recourse function $\mathcal{Q}(x)$ in the form of linear functionals computed on the basis of infeasible intermediate solutions. In the following summary of the algorithm, CP denotes the current problem.

Step 0 Compute L and set the iteration counter ν equal to 0. Define the CP as a relaxation of VRPSD in which constraints (4) and (7) are removed, the $\mathcal{Q}(x)$ term of the objective function is replaced with Θ, and the constraint $\Theta \geq L$ is imposed. Set the value of the best known solution to $\bar{z} := \infty$. At this stage the only pendent node is the initial CP.

Step 1 Select a pendent node from the list. If none exists stop.

Step 2 Set $\nu := \nu + 1$ and solve CP. Let (x^ν, Θ) be its optimal solution.

Step 3 Check for any violated subtour elimination constraints and generate them accordingly. At this stage, valid inequalities or LBFs may also be generated. If a violated constraint is found, add it to the CP and return to Step 2. Otherwise, if $cx^\nu + \Theta \geq \bar{z}$, fathom the current node and return to Step 1.

Step 4 If the solution is not integer, then branch on a fractional variable. Append the corresponding subproblems to the list of pendent nodes and return to Step 1.

Step 5 Compute $Q(x^\nu)$ and set $z^\nu := cx^\nu + Q(x^\nu)$. If $z^\nu < \bar{z}$, then $\bar{z} = z^\nu$.

Step 6 If $\Theta \geq Q(x^\nu)$, then fathom the current node and return to Step 1. Otherwise add an optimality cut defined as

$$\sum_{\substack{1<i<j \\ x^\nu_{ij}=1}} x_{ij} \leq \sum_{1<i<j} x^\nu_{ij} - 1 \tag{10}$$

and go to Step 2.

3. General partial routes

The generation of LBFs is based on the identification of partial routes in a solution. Such partial routes yielding LBFs for the single VRPSD were first proposed by Hjorring and Holt [14] and extended to the multi-vehicle case by Laporte et al. [20]. In this section we introduce a generalized definition of partial routes. It is broader than the one proposed by Hjorring and Holt [14] in that it better exploits the structural information provided by partial routes, thus enabling the construction of stronger LBFs. It was indeed shown by Laporte et al. [20] that a rather large number of the Hjorring and Holt [14] LBFs have to be added to the master problem in order to solve the VRPSD.

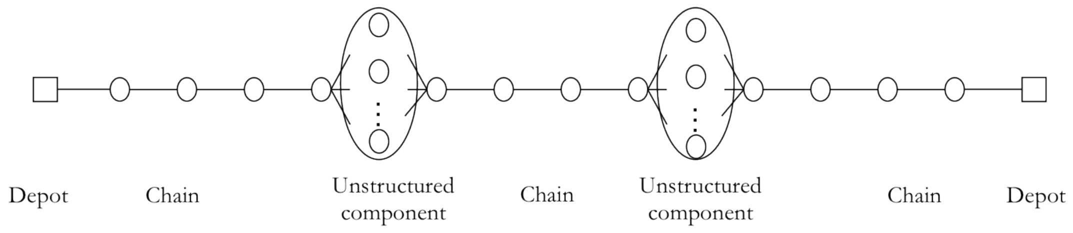

Figure 1: Example of a general partial route

General partial routes h can in principle be defined on any subgraph of G. However, as is common in most branch-and-cut implementations, we identify them on a graph $\bar{G}$ induced by the positive variables of a non-integer and connected solution of the CP solved in Step 2 of the integer L-shaped algorithm. It is convenient to represent such partial routes as in Figure 1 by duplicating the depot. When doing so it is necessary to ensure that each copy of the depot is connected to at least one vertex. A solution can be decomposed into several components anchored at an *articulation vertex*, i.e., a vertex whose removal disconnects the graph into connected components disjoint from each other. These are either *unstructured components* of $\bar{G}$ (ellipses in the figures) whose vertex sets are called *unstructured vertex sets* (UVSs), or *chains* whose vertex sets are called *chain vertex sets* (CVSs). The vertices of a chain are linked together by edges (v_i, v_j) for which $x_{ij} = 1$ in $\bar{G}$.

Let b denote the number of chains, and let $b-1$ denote the number of unstructured components in a partial route h. This partial route then consists of $2b-1$ components. Let S_h^r denote the r^{th} ordered CVS, possibly consisting of a single vertex, defined as $S_h^r = \{v_{1^{rh}}, \ldots, v_{l_{rh}^{rh}}\}$, where $v_{k^{rh}}$ is the k^{th} vertex in S_h^r and l_{rh} is the number of vertices in S_h^r. For simplicity, we write $(v_i, v_j) \in S_h^r$ if v_i and v_j are consecutive in S_h^r, and we write l^{rh} instead of l_{rh}^{rh}. Thus,

$$\sum_{(v_i,v_j)\in S_h^r} x_{ij} = |S_h^r| - 1.$$

Let U_h^r denote the r^{th} UVS in h. Then

$$\sum_{v_i,v_j\in U_h^r} x_{ij} = |U_h^r| - 1.$$

For each $r \leq b-1$,

$$\sum_{v_j\in U_h^r} x_{l^{rh},j} = 1,$$

Similarly, for each $r \geq 2$,

$$\sum_{v_j\in U_h^{r-1}} x_{1^{rh},j} = 1.$$

Given that a chain is a structured special case of a UVS and a single vertex is a degenerate chain, a general solution such as the one depicted in Figure 1 can be viewed differently depending on how chains and vertices are interpreted. For example, in Figure 2a only the first and last chains are viewed as such, whereas the intermediate part of the solution is viewed as a UVS. In Figure 2b, the original alternation of chains and UVSs is maintained. In Figure 2c, each chain is viewed as a UVS and the articulation vertices are considered as degenerate chains. The partial routes corresponding to these three constructions are called α-routes, β-routes and γ-routes, respectively. In Section 4 we will develop LBFs based on these structures.

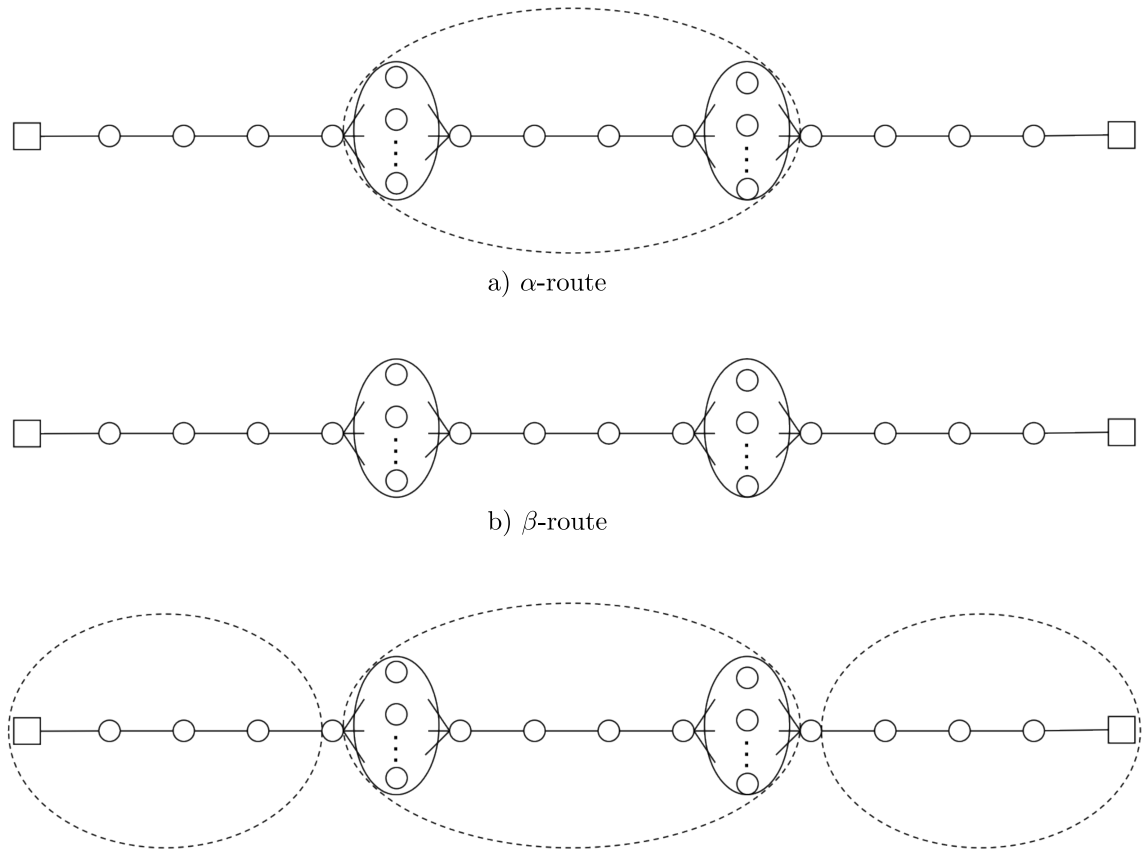

Figure 2: Examples of general partial routes

4. Lower bounding functionals

We present in Section 4.1 a lower bound on the cost of recourse. This bound is applied in Section 4.2 to the derivation of a lower bounding functional based on general partial routes. Three particular cases of the lower bounding functional associated with α-, β- and γ-routes are then presented in Section 4.3.

4.1 Lower bound on the cost of recourse

A lower bound P_h on the cost of recourse associated with general partial route h is constructed by aggregating the demand of each UVS, while the distance to the depot is bounded by the smallest distance between the depot and the vertices within a UVS. For each UVS, $r = 1, \dots, b-1$, we create an artificial customer v^r with demand

$$\xi_r = \sum_{v_i \in U_h^r} \xi_i \quad \text{and} \quad c_{1r} = \min_{v_j \in U_h^r}\{c_{1j}\}. \tag{11}$$

The partial route is then constructed as $(v_0 = v_{1^{1h}}, \dots, v_{l^{rh}}, v^1, v_{1^{2h}}, \dots, v_{l^{2h}}, v^2, \dots, v^{b-1},$ $v_{1^{bh}}, \dots, v_{l^{bh}})$ and the value of $\mathcal{Q}^{k,\delta}$ for this route k is computed as in Equation (9). Then

$$P_h \;=\; \min\{\mathcal{Q}^{k,1}, \mathcal{Q}^{k,2}\}. \tag{12}$$

To compute a lower bound P on the total cost of recourse, first define

$$R_h = (\cup_{r=1}^{b} S_h^r) \cup (\cup_{r=1}^{b-1} U_h^r).$$

Assuming $f \leq m$ partial routes, let P_{r+1} be a lower bound on the cost of recourse for $m-f$ routes involving the customer set $V \setminus \cup_{h=1}^{f} R_h$, with $P_{m+1} = 0$. Then the lower bound is

$$P \;=\; \sum_{h=1}^{f+1} P_h. \tag{13}$$

4.2 Computation of the lower bounding functional

In this section we construct a lower bounding functional that will act as a valid inequality in the integer L-shaped algorithm to eliminate some infeasible solutions. Given a solution x we first construct a lower bounding functional $W_h(x)$ for each partial route h in x. The sum of the functionals $W_h(x)$ of all partial routes in x determines when a cut is active in the solution space. The functional operates on all vertices included in partial route h. It uses variables associated with the edges that are part of chains and with all possible edges between the vertices contained in each UVS. Furthermore, the functional includes all variables associated with edges between each articulation vertex and all vertices of its corresponding UVS.

The remainder of this section is organized as follows. In Section 4.2.1 we formally define $W_h(x)$ and explain each of its terms separately. In Section 4.2.2 we project $W_h(x)$ onto the solution space. We then characterize solutions for which $W_h(x)$ is equal to 1 and those for which it is less than or equal to 0. Using this characterization we construct a valid inequality in Section 4.2.3.

4.2.1 Description of $W_h(x)$

We first define the following functional for a partial route h:

$$\begin{aligned} W_h(x) &= \sum_{r=1}^{b} \sum_{\substack{(v_i,v_j)\in S_h^r \\ v_i\neq v_1}} 3x_{ij} + \sum_{(v_1,v_j)\in S_h^1} x_{1j} + \sum_{(v_1,v_j)\in S_h^b} x_{1j} + \sum_{r=1}^{b-1} \sum_{v_i,v_j\in U_h^r} 3x_{ij} \\ &+ \sum_{r=1}^{b-1} \sum_{\substack{v_j\in U_h^r \\ v_{l^{rh}}\neq v_1}} 3x_{l^{rh},j} + \sum_{r=2}^{b} \sum_{\substack{v_j\in U_h^{r-1} \\ v_{1^{rh}}\neq v_1}} 3x_{1^{rh},j} + \sum_{\substack{v_j\in U_h^1 \\ v_{l^{1h}}=v_1}} x_{l^{1h},j} + \sum_{\substack{v_j\in U_h^{b-1} \\ v_{1^{bh}}=v_1 \\ v_{l^{b-1,h}}\neq v_1}} x_{1^{bh},j} \\ &- (3|R_h|-5). \end{aligned} \tag{14}$$

In Equation (14), the coefficients of all edges containing the depot are equal to 1 while all other edges have a coefficient of 3. The functional ends by subtracting a constant which will be shown to be equal to the sum of all edge variables multiplied by their coefficients, minus 1. In what follows we explain each component of $W_h(x)$, and we show that contrary to what was presented in Proposition 2 of Laporte et al. [20], our definition of $W_h(x)$ is always valid.

The sum of all edge variables that are part of chains and not connected to the depot is expressed by

$$\sum_{r=1}^{b} \sum_{\substack{(v_i,v_j)\in S_h^r \\ v_i \neq v_1}} 3x_{ij},$$

which is equal to 0 if $|S_h^r| = 1$. If $|S_h^1| \geq 2$ then the edge connected to the depot will have a coefficient of 1, this is expressed by $\sum_{(v_1,v_j)\in S_h^1} x_{1j}$. Similarly, if $|S_h^b| \geq 2$ the term $\sum_{(v_1,v_j)\in S_h^b} x_{1j}$ attributes a coefficient of 1 to the edge connected to the depot in S_h^b. As expressed by

$$\sum_{r=1}^{b-1} \sum_{v_i,v_j\in U_h^r} 3x_{ij},$$

a coefficient of 3 is attributed to all possible edges variables between the vertices of a UVS. Because not all vertices of the UVS are necessarily connected in h, the above summation implies that the functional $W_h(x)$ may contain variables associated to edges that do not appear in the partial route h.

The functional contains a coefficient of 3 for edge variable between articulation vertices which are not the depot, and each vertex in its corresponding UVS. This is expressed by

$$\sum_{r=1}^{b-1} \sum_{\substack{v_j\in U_h^r \\ v_{l^{rh}} \neq v_1}} 3x_{l^{rh},j} + \sum_{r=2}^{b} \sum_{\substack{v_j\in U_h^{r-1} \\ v_{1^{rh}} \neq v_1}} 3x_{1^{rh},j}.$$

Again the above summation implies that the functional $W_h(x)$ may contain variables associated to edges that do not appear in the partial route h.

If the articulation vertex of the first chain is the depot, i.e., $v_{l^{1h}} = v_1$, then all edges between the depot and each vertex in the first UVS have a coefficient of 1. Thus,

$$\sum_{\substack{v_j\in U_h^1 \\ v_{l^{1h}} = v_1}} x_{l^{1h},j}.$$

For $v_{l^{b-1,h}} = v_1$, if the partial route consists of a single UVS and two single vertex chains, i.e., $v_{1^{bh}} = v_1$ and $v_{l^{b-1,h}} = v_1$, then to avoid double counting, all edges between the depot and each vertex in the last UVS will have a coefficient of 0. Otherwise, i.e., if $|S_h^1| + |S_h^b| > 2$ or $b > 2$, all edges between the depot and each vertex in the last UVS will have a coefficient of 1. This is expressed by

$$\sum_{\substack{v_j \in U_h^{b-1} \\ v_{1^{bh}} = v_1 \\ v_{l^{b-1,h}} \neq v_1}} x_{1^{bh},j}.$$

In order for $W_h(x)$ to equal 1 for a given h, we subtract the sum of variables associated to edges included in h multiplied by their corresponding coefficients in $W_h(x)$ and add 1. All variables that appear in h and are associated to edges connected to the depot have a coefficient of 1 in $W_h(x)$, while all others have a coefficient of 3. Therefore, we subtract the constant $3(|R_h| - 2)$ accounting for each edge not connected to the depot and subtract 2 to account for the edges connected to the depot. Therefore, the sum of the edge variables is expressed as

$$3(|R_h| - 2) + 2 = 3|R_h| - 4.$$

Thus, considering h, the functional $W_h(x)$ equals 1 when $3|R_h| - 5$ is subtracted from the sum of variables associated to edges included in h, multiplied by their corresponding coefficients in $W_h(x)$.

4.2.2 Projection of $W_h(x)$ onto the solution space

In what follows we show that $W_h(x)$ takes a value of 1 not only on partial route h, but on a broader set of routes. We also show that for a solution x' in which the vertices of h appear in more than one route, $W_h(x')$ is at most 0. For this purpose we will introduce the notion of *ordered* vertices.

The functional $W_h(x)$ may contain variables associated to edges that do not appear in the partial h. Therefore, for $i < j$ we define the vector $\hat{x}_{ij} = \{\zeta_{12}, \ldots, \zeta_{1n}, \zeta_{23}, \ldots, \zeta_{2n}, \ldots, \zeta_{n-1,n}\}$, where $\zeta_{ij} = 1$ and all other components are equal to 0. Let $G_h(x_{ij}) = W_h(\hat{x}_{ij}) + 3|R_h| - 5$.

Definition 4.1. *Let $\bar{x}$ be a solution satisfying constraints (2), (3), (5) and (6) and let x' be a feasible solution to the VRPSD. Let h be a partial route of $\bar{x}$ and $J = \{v_{k_1}, \ldots, v_{k_q}\}$ such that for all $1 \leq i < q$ the following relations hold: 1) $v_{k_i} \neq v_1$, 2) $G_h(\bar{x}_{k_i,k_{i+1}}) > 0$, and 3) $x'_{k_i,k_{i+1}} > 0$. Then J is said to be* ordered *in the feasible solution x' as in h.*

Definition 4.1 implies that if v_{k_i} and $v_{k_{i+1}}$ are part of a chain in h, then $x'_{i,i+1} > 0$. If v_{k_i} is an articulation vertex, then $x'_{i,i+1} > 0$ where $v_{k_{i+1}}$ is one of the vertices associated with the corresponding UVS. If $v_{k_{i+1}}$ is an articulation vertex, then $x'_{i,i+1} > 0$ where v_{k_i} is one of the vertices associated with the corresponding UVS. Otherwise v_{k_i} and $v_{k_{i+1}}$ belong to a UVS of h.

Definition 4.2. *Consider a solution $\bar{x}$ satisfying constraints (2), (3), (5) and (6) containing a partial route h, and another feasible solution x' containing route h'. Partial route h is said to be* compatible *with h' if $R_h = R_{h'}$ and all customer vertices are ordered in h' as in h. Furthermore, $\bar{x}$ is said to be* compatible *with x' if for each partial route in $\bar{x}$ there exists a compatible route in x'.*

Next we show that $W_h(x) = 1$ when route compatibility is attained and that $W_h(x) \leq 0$ otherwise. Finally we construct the valid inequality showing that $\bar{x}$ is compatible with x'.

Consider a solution $\bar{x}$ satisfying constraints (2), (3), (5) and (6) containing a partial route h, and another feasible solution x' containing route h'. We start by showing that if $R_{h'} = \{R_h \setminus J\}$, where $|J| \geq 1$, then $W_h(x') \leq 0$. Given $W_h(x)$ defined by partial route h, we define the functional

$$\begin{aligned} W_h(J|x) \;=\; & \sum_{r=1}^{b} \sum_{\substack{(v_i,v_j)\in S_h^r\cap J \\ v_i\neq v_1}} 3x_{ij} + \sum_{(v_1,v_j)\in S_h^1\cap J} x_{1j} + \sum_{(v_1,v_j)\in S_h^b\cap J} x_{1j} + \sum_{r=1}^{b-1} \sum_{v_i,v_j\in U_h^r\cap J} 3x_{ij} \\ & + \sum_{r=1}^{b-1} \sum_{\substack{v_j\in U_h^r\cap J \\ v_{l^{rh}}\neq v_1}} 3x_{l^{rh},j} + \sum_{r=2}^{b} \sum_{\substack{v_j\in U_h^{r-1}\cap J \\ v_{1^{rh}}\neq v_1}} 3x_{1^{rh},j} + \sum_{\substack{v_j\in U_h^1 \\ v_{l^{1h}}=v_1}} x_{l^{1h},j} \\ & + \sum_{\substack{v_j\in U_h^{b-1}\cap J \\ v_{1^{bh}}=v_1 \\ v_{l^{b-1,h}}\neq v_1}} x_{1^{bh},j}. \qquad (15) \end{aligned}$$

Let $W_h(J|x')$ be the value taken by functional (15) evaluated at x'. The functional $W_h(J|x')$ is maximized when the vertices included in J are ordered in h' as in h. In Lemma 4.3, we introduce an upper bound on $W_h(J|x')$. This result is then used in Proposition 4.4 to show that $W_h(x') \leq 0$ when $R_{h'} = \{R_h \setminus J\}$, where $J = \{v_{k_1}, \ldots, v_{k_q}\}, 1 \leq q \leq |R_h| - 2$, and $v_{k_i} \neq v_1$ for all $1 \leq i < q$. Subsequently, in Proposition 4.5 we show that if h is compatible with h', then $W_h(x') = 1$.

Lemma 4.3. *Let $\bar{x}$ be a solution satisfying constraints (2), (3), (5) and (6) and let x' be a feasible solution to the VRPSD. Let h be a partial route of $\bar{x}$ and let $J = \{v_{k_1}, \ldots, v_{k_q}\}$, where $1 \leq q \leq |R_h| - 2$, and $v_{k_1} \neq v_1$ for all $1 \leq i < q$. Define $W_h(J|x')$ as the value obtained by functional (15) evaluated at x'. Then $W_h(J|x') \leq 3(|J| - 1) + 2$.*

Proof. Let $J_1 = \{v^1_{k_1}, \dots, v^1_{k_q}\}$, where $J_1 \subset J$, and let h'_1 be a route of x' such that $J_1 \subset R_{h'_1}$. Then, considering partial route h, we observe that

$$\sum_{r=1}^{b} \sum_{\substack{(v_i,v_j)\in S_h^r \cap J_1 \\ v_i \neq v_1}} 3{x'}_{ij} \sum_{r=1}^{b-1} \sum_{v_i,v_j \in U_h^r \cap J_1} 3{x'}_{ij} + \sum_{r=1}^{b-1} \sum_{\substack{v_j \in U_h^r \cap J_1 \\ v_{l^{rh}} \neq v_1}} 3{x'}_{l^{rh},j} + \sum_{r=2}^{b} \sum_{\substack{v_j \in U_h^{r-1} \cap J_1 \\ v_{1^{rh}} \neq v_1}} 3{x'}_{1^{rh},j}$$
$$\leq \ 3(|J_1| - 1),$$

where the equality holds if J_1 is ordered in h'_1 ordered as is in h. Furthermore,

$$\sum_{(v_1,v_j)\in S_h^1 \cap J_1} {x'}_{1j} + \sum_{(v_1,v_j)\in S_h^b \cap J_1} {x'}_{1j} + \sum_{\substack{v_j \in U_h^1 \cap J_1 \\ v_{l^{1h}} = v_1}} {x'}_{l^{1h},j} + \sum_{\substack{v_j \in U_h^{b-1} \\ v_{1^{bh}} = v_1 \\ v_{l^{b-1,h}} \neq v_1}} x'_{1^{bh},j} \ \leq \ 2,$$

where the equality holds if $G_h(\bar{x}_{1,k^1_1}) > 0$ and $G_h(\bar{x}_{1,k^1_q}) > 0$. Therefore,

$$W_h(J_1|x') \ \leq \ 3(|J_1| - 1) + 2. \tag{16}$$

Let $\{J_1, \dots, J_w\}$ define a partition of J such that $J_i = \{v^i_{k_1}, \dots, v^i_{k_q}\}$ for all $1 \leq i \leq w$. Let h'_i be a route of x' such that $J_i \in R_{h'_i}$. From inequality (16) we infer that

$$W_h(J_i|x') \leq 3(|J_i| - 1) + 2 \qquad (1 \leq i \leq w).$$

Therefore,

$$W_h(J|x') = \sum_{i=1}^{w} W_h(J_i|\bar{x}) \ \leq \ \sum_{i=1}^{w} [3(|J_i| - 1) + 2]$$
$$\leq \ \sum_{i=1}^{w} 3(|J_i|) - 3w,$$

which is maximized for $w = 1$. We therefore conclude that $W_h(J|\bar{x}) \leq 3(|J| - 1) + 2$.

□

Lemma 4.3 implies that $W_h(J|x')$ is maximized when the vertices of J appear in a single route and are ordered as in h.

Proposition 4.4. *Let $\bar{x}$ be a solution satisfying constraints (2), (3), (5) and (6) and let x' be a feasible solution to the VRPSD. Let h be a partial route of $\bar{x}$ and let h' be a route of x'. Let $R_{h'} = \{R_h \setminus J\}$, where $J = \{v_{k_1}, \ldots, v_{k_q}\}, 1 \leq q \leq |R_h| - 2$, and $v_1 \neq v_{k_i}$ for all $1 \leq i < q$. Then $W_h(x') \leq 0$.*

Proof. Let $\{R'_h \setminus v_1\} = \{v_{k'_1}, \ldots, v_{k'_l}\}$. We recall that $W_h(R_h|\bar{x}) = 3(|R_h| - 2) + 2$. As in Lemma 4.3, the value $W_h(R_{h'}|x')$ is maximized if all vertices in $R_{h'} \setminus \{v_1\}$ are ordered in h' as in h. Therefore,

$$W_h(R_{h'}|x') \leq 3(|R_{h'}| - 2) + 2,$$

where the equality holds if $R'_h \setminus \{v_1\}$ is ordered in x' ordered as in h, $G_h(\bar{x}_{1,k_1^1}) > 0$ and $G_h(\bar{x}_{1,k_q^1}) > 0$. Since J and $R_{h'}$ define a partition of R_h, then

$$W_h(R_{h'} \cup J|x') = W_h(R_{h'}|x') + W_h(J|x').$$

In Lemma 4.3 we have shown that $W_h(J|x') \leq 3(|J| - 1) + 2$. Therefore,

$$\begin{aligned} W_h(R_{h'}|x') + W_h(J|x') &\leq 3(|R_{h'}| - 2) + 2 + 3(|J| - 1) + 2 \\ &\leq 3(|R_h| - 3) + 4 \\ &\leq 3(|R_h| - 2) + 1. \end{aligned}$$

Since, $W_h(R_h|\bar{x}) - W_h(R_{h'} \cup J|x') \geq 1$, we conclude that $W_h(x') \leq 0$.

□

We now proceed to show that $W_h(x') = 1$ if h' contains exactly the same vertices as h, and these are ordered in h' as h.

Proposition 4.5. *Let $\bar{x}$ be a solution satisfying constraints (2), (3), (5) and (6) and let x' be a feasible solution to the VRPSD. Let h be a partial route of $\bar{x}$ and let h' be a route of x'. Then $W_h(x') = 1$ if $R_{h'} = R_h$ and the vertices in h' are ordered as in h.*

Proof. Proposition 4.4 states that $W_h(x') \leq 0$ for $R_{h'} = \{R_h \setminus J\}$ where $1 \leq |J| \leq |R_h| - 2$. Next we show that $W_h(x') \leq 0$ for $R_{h'} = \{R_h \cup J\}$. Let $J = \{v_{k_1}\}$. We distinguish between the following two cases:

1) $x'_{1,k_1} = 1$, then

$$\begin{aligned} &\sum_{r=1}^{b} \sum_{\substack{(v_i,v_j) \in S_h^r \cap R_{h'} \\ v_i \neq v_1}} 3{x'}_{ij} + \sum_{r=1}^{b-1} \sum_{v_i,v_j \in U_h^r \cap R_{h'}} 3{x'}_{ij} + \sum_{r=1}^{b-1} \sum_{\substack{v_j \in U_h^r \cap R_{h'} \\ v_{l^{rh}} \neq v_1}} 3{x'}_{l^{rh},j} + \sum_{r=2}^{b} \sum_{\substack{v_j \in U_h^{r-1} \cap R_{h'} \\ v_{1^{rh}} \neq v_1}} 3{x'}_{1^{rh},j} \\ \leq\ & 3(|R_h| - 2), \end{aligned}$$

where the equality holds if all vertices in R_h are ordered in x' as in h. Furthermore,

$$\sum_{(v_1,v_j) \in S_h^1 \cap R_{h'}} {x'}_{1j} + \sum_{(v_1,v_j) \in S_h^b \cap R_{h'}} {x'}_{1j} + \sum_{\substack{v_j \in U_h^1 \cap R_{h'} \\ v_{l^{1h}} = v_1}} {x'}_{l^{1h},j} + \sum_{\substack{v_j \in U_h^{b-1} \cap R_{h'} \\ v_{1^{bh}} = v_1 \\ v_{l^{b-1},h} \neq v_1}} {x'}_{1^{bh},j} \ \leq\ 1.$$

This inequality stems from the fact that $x'_{1,i_k} = 1$.

2) $x'_{i,k_1} = 1$, where $v_i \in \{R_h \setminus v_1\}$. This entails that the vertices in R_h are not ordered in h' as in h and therefore,

$$\begin{aligned} &\sum_{r=1}^{b} \sum_{\substack{(v_i,v_j) \in S_h^r \cap R_{h'} \\ v_i \neq v_1}} 3{x'}_{ij} + \sum_{r=1}^{b-1} \sum_{v_i,v_j \in U_h^r \cap R_{h'}} 3{x'}_{ij} + \sum_{r=1}^{b-1} \sum_{\substack{v_j \in U_h^r \cap R_{h'} \\ v_{l^{rh}} \neq v_1}} 3{x'}_{l^{rh},j} + \sum_{r=2}^{b} \sum_{\substack{v_j \in U_h^{r-1} \cap R_{h'} \\ v_{1^{rh}} \neq v_1}} 3{x'}_{1^{rh},j} \\ \leq\ & 3(|R_h| - 3) \end{aligned}$$

and

$$\sum_{(v_1,v_j) \in S_h^1 \cap R_{h'}} {x'}_{1j} + \sum_{(v_1,v_j) \in S_h^b \cap R_{h'}} {x'}_{1j} + \sum_{\substack{v_j \in U_h^1 \cap R_{h'} \\ v_{l^{1h}} = v_1}} {x'}_{l^{1h},j} + \sum_{\substack{v_j \in U_h^{b-1} \cap R_{h'} \\ v_{1^{bh}} = v_1 \\ v_{l^{b-1},h} \neq v_1}} {x'}_{1^{bh},j} \ \leq\ 2.$$

We conclude that for $J = \{v_{k_1}\}$,

$$W_h(R_{h'}|x') \ \leq\ 3(|R_{h'}| - 2) + 1.$$

When $J = \{v_{k_1}, \ldots, v_{k_q}\}$ and $q \geq 2$, $W_h(R_{h'}|x')$ is also bounded above by $3(|R_h| - 2) + 1$. This bound is reached when $x'_{1,k_q} = 1$, $x'_{k_i,k_{i+1}} = 1$ for all $1 \leq i < q$, and all vertices in R_h are ordered in h' as in h. Thus, we infer that $W_h(R_{h'}|x') \leq 3(|R_h| - 2) + 1$. Since, $W_h(R_h|\bar{x}) - W_h(R_{h'}|x') \geq 1$, we conclude that $W_h(x') \leq 0$ for $R_h = \{R_{h'} \setminus J\}$ and $q \geq 1$. Any case for which $|J_1| \geq 1$, $|J_2| \geq 1$ and $R_{h'} = \{R_h \setminus J_1\} \cup J_2$, where $J_1 \subset R_h$ and $\{J_2 \cap R_h\} = \emptyset$, will be a combination of the cases presented above and of the ones presented in Proposition 4.4, yielding $W_h(x') \leq 0$. Finally it is straightforward to show that $W_h(x') = 1$ when $R_{h'} = R_h$ and all customer vertices are ordered in h' as in h.

□

4.2.3 Valid inequality

We recall that Θ is a lower bound on the cost of recourse, L is the lower bound on Θ defined in Step 0 of the integer L-shaped algorithm, and P is defined by Equation (13).

Proposition 4.6. *Let $\bar{x}$ be a solution satisfying constraints (2), (3), (5) and (6) and let x' be a feasible solution to the VRPSD. The constraint*

$$\Theta \geq L + (P - L)\left(\sum_{h=1}^{r} W_h(\bar{x}) - r + 1\right) \tag{17}$$

is a valid inequality for the (VRPSD).

Proof. It follows from Proposition 4.5 that $W_h(x') = 1$ if partial route h is compatible with a route h' in x', otherwise $W_h(x) \leq 0$. Therefore, $\sum_{h=1}^{r} W_h(x') = r$ if $\bar{x}$ is compatible with x'. If $\bar{x}$ is not compatible with x', then $\sum_{h=1}^{r} W_h(x) \leq r - 1$. We conclude that $\sum_{h=1}^{f} W_h(x') - r + 1 = 1$ if $\bar{x}$ is compatible with x', and $\sum_{h=1}^{f} W_h(x') - r + 1 \leq 0$ otherwise. Therefore, inequality (17) reduces to $\Theta \geq P$ for $\sum_{h=1}^{f} W_h(x') - r + 1 = 1$, and is redundant otherwise. □

4.3 Three special cases of the lower bounding functional

Various LBFs can be obtained by applying different aggregation strategies on the connected components along the partial route. In this paper we are interested in the three particular structures illustrated in Figure 2. In the following three special cases, we define different structures yet the lower bound P and the lower bounding functional defined earlier remain unchanged.

4.3.1 α-routes

The *partial routes* originally proposed by Hjorring and Holt [14] correspond to our α-routes. A standard partial route is made up of three components: two chains ($b = 2$) connected by one unstructured component, which may or may not include chains (Figure 2a). The two CVSs are denoted by S_h^1 and S_h^2; in addition the UVS is equal to U_h^1. In this case $W_h(x)$ is defined by (14).

4.3.2 β-routes

The β-route depicted in Figure 2b is an alternation of b chains and $b-1$ UVSs, where $b \geq 2$. The functional $W_h^\beta(x)$ applied in this case is identical to $W_h(x)$. Defining P_h^β and P_h^α as P_h adapted to α-routes and β-routes, respectively, the relationship between the proposed partial routes is $P_h^\beta \geq P_h^\alpha$. This stems from the fact that when compared to P_h^α, the lower bound P_h^β better exploits the specificity of h because it is computed by making use of all sequences present in that partial route. However, the LBF computed with $W_h^\beta(x)$ will be active on a smaller solution space, when compared to $W_h^\alpha(x)$.

4.3.3 γ-routes

In order to define γ-routes, we first consider the same structure as for β-routes, i.e., an alternating sequence of b chains and $b-1$ unstructured components. We then consider each chain as an unstructured component to yield a sequence of $2b-1$ UVSs but no chains (Figure 2c). The articulation points remain unchanged.

Defining the P_h^γ as the lower bound P_h adapted to γ-routes and using the same argument as in Section 4.3.2, one observes that $P_h^\beta \geq P_h^\alpha \geq P_h^\gamma$. Again, the LBF computed with $W_h^\gamma(x)$ will be active on a larger solution space, when compared to $W_h^\alpha(x)$ and $W_h^\beta(x)$.

5. Exact separation procedure for the lower bounding functional

The exact separation procedure identifies partial routes for a given solution and generates their corresponding bounds $P^\lambda = \sum_{h=1}^{f+1} P_h^\lambda$, where λ stands for α, β or γ. This procedure explores the induced graph associated with the current solution in order to detect partial routes, if they exist. For each identified partial route, the procedure provides its associated chains and UVSs. All vertices not contained in the identified partial routes are grouped in a set used to compute P_{r+1}. Next we provide a description of this separation procedure. The two pseudo-codes detailing this procedure are presented in the appendix as Algorithm 1 and Algorithm 2.

Recall that Laporte et al. [20] proposed a heuristic separation procedure for constraints (17). This procedure determines α-routes by first initializing sets S_h^1 and S_h^2 with two vertices strongly connected to the depot in the current solution and then by expanding these sets using a greedy strategy. However, it should be noted that the procedure proposed in Laporte et al. [20] only applies to α-routes. In order to separate constraints associated with β- and γ-routes described in Section 4, it is necessary to first identify all chains and unstructured components of the current solution, and then combine them into partial routes that will become candidates for LBFs.

The pseudo-code of the exact separation procedure we have developed to construct the general partial routes is summarized in Algorithm 1. This algorithm is called once the separation procedure associated with constraints (17) has been applied to no avail at the solution of the current node. Algorithm 2 identifies partial routes by first expanding unstructured vertex sets as much as possible. Once these sets are constructed, then all other vertices assigned to a partial route are part of chains.

The procedure operates on two types of stacks, one for unstructured vertex sets (U stack), and another one for the chains (C stack). These stacks serve as temporary buffers that hold vertices until a complete component is identified, i.e., a complete unstructured vertex set or chain. Once identified, the components are added to a *components list* and are then used to construct partial routes if they exist. The vertices that are not contained in the partial routes are stored into set B, which is used to compute the general lower bound L in constraint (17).

6. Implementation and computational results

The integer L-shaped algorithm was coded in a C++ environment with CPLEX 12.3. All experiments were conducted on a cluster of 15 computers, each having two 2.2-GHz Dual-Core AMD Opteron processor 275 with 7.7 GB of RAM, and operating under Linux SuSe 11.3. The branching procedure was implemented by using the OOBB package developed at the CIRRELT. The separation procedure of the subtour elimination and capacity constraints (4) was performed using the CVRPSEP package of Lysgaard et al. [23]. These cuts were added as long as violated constrains were found. The instances were generated based on the same principles as in Laporte et al. [20]. Namely, n vertices were generated in $[0,100]^2$ following a continuous uniform distribution. Also, five rectangular obstacles in $[20,80]^2$ were generated, each having a base of 4 and height of 25, covering 5% of the entire area.

In our instances customer demands ξ_i follow a normal distribution $\mathcal{N}(\mu_i,\sigma_i)$ truncated at zero, and all demands are independently distributed. The coefficient of variation of the demand distribution was set equal to 30%. Let $\overline{f}=\sum_{i=2}^{n}\mu_i/mD$ be the average vehicle filling coefficient. Table 1 summarizes the parameter combinations used in our experiments. In each case, 10 instances were generated for a total of 270 instances. The computation time limit for any given instance was set to 10 hours.

Table 1: Parameter combinations used in the experiments

m	n	$\overline{f}$
2	$60,70,80$	$90\%,92.5\%,95\%$
3	$50,60,70$	$85\%,87.5\%,90\%$
4	$40,50,60$	$80\%,82.5\%,85\%$

We have performed extensive numerical analyses to assess the performance of the algorithm. Section 6.1 assesses our separation procedure, whereas Section 6.2, compares several LBF schemes.

6.1 Heuristic vs. exact separation procedure

In order to evaluate the added value of incorporating the exact separation procedure, we have compared the performance of the heuristic separation algorithm proposed by Laporte et al. [20] with the exact separation procedure of Section 5. The results of the former algorithm are denoted by LBF_0 while those of the latter are denoted by LBF_α.

Out of the 270 test instances, LBF_0 solved 77 to optimality, while LBF_α solved five additional instances. Table 2 summarizes the experimental results for these instances. The last four columns report average values over the ten instances of each type. The reported averages at the bottom of the table are weighted averages. The number of instances solved decreases with the number of vehicles, for example, 40 out 90 instances with two vehicles were solved to optimality while only nine instances with four vehicles were solved to optimality. Similarly, the number of solved instances decreases with the average filling coefficient. The runtime for the instances solved by both LBF_0 and LBF_α, is significantly lower for LBF_α when compared to LBF_0. The last two columns of Table 2 report the gaps for instances unsolved by both LBF_0 and LBF_α. For these cases the average gap of LBF_α is 1.07% while that of LBF_0 is 1.15%.

Table 2: Results for solved instances by both LBF_0 and LBF_α

n	m	$\overline{f}$	Number solved	Runtime (min)	Runtime (min)	Gap LBF_0	Gap LFB_α
			solved	LBF_0	LBF_α	(unsolved by both)	(unsolved by both)
60	2	0.90	6	40.7	31.9	0.3%	0.3%
60	2	0.93	9	25	18.9	0.65%	0.63%
60	2	0.95	2	25.8	26.1	0.88%	0.85%
70	2	0.90	7	25	22.6	0.86%	0.78%
70	2	0.93	4	73.3	72.1	0.99%	0.87%
70	2	0.95	1	44	23.7	1.35%	1.14%
80	2	0.90	9	49.7	49.1	0.24%	0.24%
80	2	0.93	2	8.4	8.5	0.6%	0.62%
80	2	0.95	0			1.13%	1.14%
50	3	0.85	3	69.4	60	1.09%	0.96%
50	3	0.88	4	152.8	114	1.32%	0.69%
50	3	0.90	2	56.3	41.3	2.53%	2.6%
60	3	0.85	6	65	50.1	1.52%	1.49%
60	3	0.88	2	215.1	184.4	1.16%	1.07%
60	3	0.90	1	359.4	318.7	2.08%	2%
70	3	0.85	8	75.5	62.4	1.28%	1.31%
70	3	0.88	2	253.5	184.9	1.13%	1.09%
70	3	0.90	0			2.85%	2.87%
40	4	0.80	1	3.5	3.2	2.76%	2.63%
40	4	0.82	1	26.7	24.2	2.75%	2.25%
40	4	0.85	1	47.1	48.9	4.09%	4.18%
50	4	0.80	2	297.7	212.9	2.36%	2.32%
50	4	0.82	1	117.5	91.9	3.23%	3.14%
50	4	0.85	0			4.63%	4.54%
60	4	0.80	1	425.9	394.6	3.07%	2.16%
60	4	0.82	2	191.3	192.8	2.21%	2.21%
60	4	0.85	0		0	3.05%	3.32%
Average				82.1	68.7	1.15%	1.07%
Total			77				

Table 3: Solved and unsolved instances by LBF_0 and LBF_α

n	m	Number	LBF_α solved, LBF_0 unsolved	
			Runtime (min) LBF_α	Gap 0
60	2	1	510.7	0.12%
70	2	1	446.0	0.02%
80	2	1	569.4	0.03%
50	3	0		
60	3	0		
70	3	0		
40	4	1	252.3	0.13%
50	4	1	103.8	0.29%
60	4	0		

A total of five instances were solved by LBF_α only. Table 3 summarizes the results for these cases. The gaps obtained by LBF_0 on these instances are reported in the last column of the table.

6.2 Comparative assessment of the lower bounding functionals

We now assess and compare the performance of the three families of LBFs using the exact separation procedure. We present six experimental settings. The first three correspond to α-routes, β-routes, and γ-routes. In the fourth set, α-route and β-route LBFs are added simultaneously: these are denoted by $\mathrm{LBF}_{\alpha\beta}$. Similarly, in the fifth set we present experiments for $\mathrm{LBF}_{\beta\gamma}$. Finally, we combine the three options in the experimental set $\mathrm{LBF}_{\alpha\beta\gamma}$.

Table 4 summarizes the number of instances solved for each of the three routes types, and three of their combinations. When using a single type of partial route, the γ-routes solve the largest number of instances when compared to α- and β-routes. Although LBF_{γ} yields a weak lower bound, the corresponding cuts are active on a larger solution space, which leads to overall better results in terms of the number of solved instances. In contrast, LBF_{β} solves the least number of instances to optimality, and LBF_{β} yields the strongest lower bound. Yet their corresponding cuts are active on a smaller subset of the solution space, which translates into the exact solution of fewer instances. However, using LBF_{β} in conjunction with LBF_{γ} (combination $\mathrm{LBF}_{\beta\gamma}$) yields results that are superior to those achieved by LBF_{β} alone in terms of the number of instances solved to optimality. Table 5 summarizes the runtimes for the solved instances for all experimental settings. On average LBF_{β} has relatively large runtimes.

Table 6 presents the average gaps for the unsolved instances. All algorithms achieved an average gap of less than 2.6%. The table shows that the gaps increase with the number of vehicles. A similar observation can be made by considering the total number of instances solved in Table 4. Although LBF_{β} solved the least number of instances to optimality, it yielded a relatively low average gap. This is partly due to the fact that LBF_{β} yields low gaps on instances solved to optimality by other combinations.

We have computed in Table 7 the cumulative percentage of instances solved for several ranges of gaps. For example, 83.7% of the 270 instances solved with LBF_{β} had a gap of at most 3%. Similarly we observe that LBF_{0} solves more than 94% of the instances with a gap under 5%, indicating that while the exact separation procedure helped solve a larger number of instances to optimality, the heuristic separation procedure yielded a relatively low variability in the distribution of the resulting gaps.

Finally, in Table 8 we compare several solution characteristics for instances solved by both LBF_{0} and LBF_{γ}, the latter representing the best solution strategy in terms of the number of solved instances. In total 72 instances were solved to optimality by both LBF_{0} and LBF_{γ}, and an additional 15 instances were solved to optimality only by LBF_{γ}. The runtime of the

Table 4: Number of solved instances for families of partial routes

m	n	$\overline{f}$	LBF$_\alpha$	LBF$_\beta$	LBF$_\gamma$	LBF$_{\alpha\beta}$	LBF$_{\beta\gamma}$	LBF$_{\alpha\beta\gamma}$
60	2	0.90	7	6	8	7	8	8
60	2	0.93	9	9	9	9	9	9
60	2	0.95	2	2	2	2	2	3
70	2	0.90	7	7	7	7	7	7
70	2	0.93	4	4	5	4	6	6
70	2	0.95	2	1	2	1	2	2
80	2	0.90	9	8	10	9	10	10
80	2	0.93	3	2	4	2	4	4
80	2	0.95	0	0	0	0	0	0
50	3	0.85	3	3	5	3	3	3
50	3	0.88	4	3	3	4	3	2
50	3	0.90	2	2	1	2	1	1
60	3	0.85	6	6	6	6	6	6
60	3	0.88	2	2	2	1	1	2
60	3	0.90	1	1	0	1	0	0
70	3	0.85	8	8	9	8	7	6
70	3	0.88	2	2	2	2	2	2
70	3	0.90	0	0	0	0	0	0
40	4	0.80	1	1	1	1	1	1
40	4	0.82	2	1	3	2	2	2
40	4	0.85	1	1	1	1	1	1
50	4	0.80	3	3	3	3	3	3
50	4	0.82	1	1	1	1	1	1
50	4	0.85	0	0	0	0	0	0
60	4	0.80	1	1	1	1	1	1
60	4	0.82	2	2	2	1	2	2
60	4	0.85	0	0	0	0	0	0
Total			82	76	87	78	82	82

instances solved by LBF$_\gamma$ is 14.8 minutes shorter than that of LBF$_0$, and in 54 out of the 72 instances the runtime for LBF$_0$ was longer than that of LBF$_\gamma$. The instances solved by LBF$_0$ required on average 345.7 fewer subtour elimination and capacity constraints. The third line of Table 7 compares the total number of LBF cuts produced by LBF$_\gamma$ and LBF$_0$. We observe that on average 1244.7 additional LBF cuts were added per instance in LBF$_0$. Furthermore, the number of optimality cuts added by LBF$_\gamma$ is on average only 299 more than the corresponding number for LBF$_0$.

Table 5: Runtimes in minutes for the solved instances

m	n	$\overline{f}$	LBF$_\alpha$	LBF$_\beta$	LBF$_\gamma$	LBF$_{\alpha\beta}$	LBF$_{\beta\gamma}$	LBF$_{\alpha\beta\gamma}$
60	2	0.90	1911.9	7006.3	708.4	9317.4	787.2	943
60	2	0.93	1132.4	2409.3	703.6	2035.4	456.6	887.8
60	2	0.95	1567.8	3583.8	551.9	2272.2	1057	10753.5
70	2	0.90	1354.6	4964.6	3296.6	1796.7	1084.9	1187.8
70	2	0.93	4325.5	8709.2	8102	5128	12455.7	14551.5
70	2	0.95	1424.5	5539.1	14773.4	2150	6188.4	7773.1
80	2	0.90	2947.7	3998	5116.7	3422.1	5137.4	5524.6
80	2	0.93	508.6	807.1	8092.5	613.7	7185	6897.9
80	2	0.95						
50	3	0.85	3599.9	2901.9	10356.3	3786	751	916.7
50	3	0.88	6837.4	3417.5	2046.7	11156.9	2839.7	1250.9
50	3	0.90	2477.2	3661	9979.1	8086.6	23453.8	30122.4
60	3	0.85	3005.1	4425.8	3634.1	4684.4	2688.8	3202.5
60	3	0.88	11062.3	20868.4	18523.6	1964.4	10666.7	22203.6
60	3	0.90	19122.3	19794.8		28082.4		
70	3	0.85	3741.6	5096.7	6774.1	5663.8	3131.6	2474.3
70	3	0.88	11095.5	13528.6	4734.7	11560.1	15413.9	17117.8
70	3	0.90						
40	4	0.80	191.3	189	95.7	163.5	140.3	141.4
40	4	0.82	1451.7	24556.4	8202.3	19216.7	11621.5	10669.2
40	4	0.85	2931.5	3883.4	8784.6	3136.4	3775.3	4950.4
50	4	0.80	12775	14283.1	4667.9	9214.2	4732.8	5845.3
50	4	0.82	5511.4	5366.7	1557.1	7127.8	1821.8	1692.7
50	4	0.85						
60	4	0.80	23676	22045.2	9163.7	23240.7	12689.6	16564.4
60	4	0.82	11567.2	16238.4	5943.8	2486.8	13353.2	10704.6
60	4	0.85						
Average			4092.4	6489.2	5253.6	5850.3	4679.1	5774.5

Table 6: Optimality gaps for the unsolved instances

m	n	$\overline{f}$	LBF$_\alpha$	LBF$_\beta$	LBF$_\gamma$	LBF$_{\alpha\beta}$	LBF$_{\beta\gamma}$	LBF$_{\alpha\beta\gamma}$
60	2	0.90	0.3%	0.3%	0.1%	0.3%	0.1%	0.1%
60	2	0.93	0.6%	1.2%	0.4%	1.1%	0.5%	0.5%
60	2	0.95	0.9%	1%	0.6%	1%	0.7%	0.8%
70	2	0.90	0.8%	0.9%	0.3%	0.8%	0.3%	0.3%
70	2	0.93	0.9%	1.2%	0.8%	1%	1%	1%
70	2	0.95	1.1%	1.1%	0.9%	1.1%	0.9%	0.9%
80	2	0.90	0.2%	0.2%	0%	0.3%	0%	0%
80	2	0.93	0.6%	0.6%	0.6%	0.6%	0.6%	0.6%
80	2	0.95	1.1%	1.3%	1.1%	1.2%	1%	1.1%
50	3	0.85	1%	1.2%	0.5%	1%	0.7%	0.7%
50	3	0.88	0.7%	1.1%	0.6%	1.2%	1.1%	1.1%
50	3	0.90	2.6%	2.5%	2%	2.7%	2.1%	2.1%
60	3	0.85	1.5%	1.6%	1.2%	1.6%	1.9%	2.3%
60	3	0.88	1.1%	1.2%	1.5%	1.1%	1.4%	1.6%
60	3	0.90	2.0%	2.0%	1.5%	2.0%	1.9%	2.1%
70	3	0.85	1.3%	1.3%	2.8%	1.4%	1%	1%
70	3	0.88	1.1%	1.2%	1.5%	1.2%	1.7%	1.7%
70	3	0.90	2.9%	2.9%	3.2%	3.2%	3.1%	3.3%
40	4	0.80	2.6%	2.8%	2.7%	3.4%	3.3%	3.7%
40	4	0.82	2.2%	2.1%	1.8%	2.3%	4.6%	2.9%
40	4	0.85	4.2%	4.2%	4.2%	4.2%	5.6%	6.1%
50	4	0.80	2.3%	2.3%	2.5%	2.3%	2.4%	2.8%
50	4	0.82	3.1%	3.2%	3.1%	3.2%	3.4%	3.3%
50	4	0.85	4.5%	4.9%	7.1%	4.6%	7.1%	7.3%
60	4	0.80	2.2%	2.9%	3.2%	2.9%	3.1%	3.1%
60	4	0.82	2.2%	2.3%	2.4%	1.9%	2.5%	2.5%
60	4	0.85	3.3%	3.5%	4%	3.2%	4.6%	4.5%
Average			2.0%	2.1%	2.3%	2.1%	2.6%	2.6%

Table 7: Cumulative percentage of instances solved for several ranges of gaps

Range	LBF_0	LBF_α	LBF_β	LBF_γ	$\mathrm{LBF}_{\alpha\beta}$	$\mathrm{LBF}_{\beta\gamma}$	$\mathrm{LBF}_{\alpha\beta\gamma}$
=0%	28.5%	30.4%	28.1%	32.2%	28.9%	30.4%	30.4%
$\leq$ 1%	57.0%	59.3%	54.4%	62.2%	57.0%	58.9%	59.3%
$\leq$ 3%	81.9%	83.7%	83.0%	82.6%	81.9%	80.0%	77.8%
$\leq$ 5%	94.1%	94.8%	93.7%	91.9%	94.1%	88.9%	89.3%
$\leq$ 7%	97.8%	97.8%	97.4%	95.6%	97.0%	94.4%	94.8%
$\leq$ 9%	98.5%	98.9%	98.5%	96.7%	98.5%	95.9%	95.6%

Table 8: Results for the instances solved by both LBF_0 and LBF_γ

	Average			% of instances
	LBF_0	LBF_γ	$\mathrm{LBF}_0 - \mathrm{LBF}_\gamma$	$\mathrm{LBF}_0 > \mathrm{LBF}_\gamma$
Runtime (min)	64.1	49.3	14.8	75.0%
Subtour elimination and capacity constraints (4)	1329.4	1675.1	-345.7	25.0%
Total LBF cuts	1385.7	141.0	1244.7	98.6%
Optimality cuts (10)	237.1	536.1	-299	44.4%

7. Conclusions

We have developed an exact algorithm for the vehicle routing problem with stochastic demands. It extends and improves the integer L-shaped algorithm of Laporte et al. [20] by generalizing the lower bounding functional introduced by these authors and by separating them exactly. The proposed LBFs stem from a generalization of the concept of a partial route originally proposed by Hjorring and Holt [14]. Extensive computational experiments have shown that some combinations of the proposed LBFs outperform the classical version. As a result, on a set of 270 benchmark instances the number of optimally solved instances increases from 77 to 87, which is significant in the context of stochastic vehicle routing.

Our experiments have shown that the exact separation procedure solved a larger number of instances to optimality compared with the heuristic version of Laporte et al. [20]. Using our algorithm the largest instances solved with normally distributed demands contain 60 vertices and four vehicles, or 80 vertices and two vehicles. Our success can be attributed to the use of new LBFs which substantially reduce the number of cuts added to the relaxed problem. Our results also indicate that the overall performance of LFB$_\gamma$ is better than that of LFB$_\beta$, implying that in our instances a weaker lower bound active on a larger solution space outperforms a stronger bound restricted to a smaller space.

This study can be extended in a number of ways. First, the generalization of a partial route may lead to the development of other families of LBFs by using different aggregation mechanisms for chains and unstructured components. Second, the LBFs we have developed can potentially be incorporated within exact algorithms applicable to the solution of other types of stochastic routing problems involving, for example, different recourse functions or problems with stochastic customers.

Appendix: Pseudo-codes for the separation procedure

Algorithm 1 Partial route separation procedure

```
1: consider all edges of the current solution
2: repeat
3:    if there exists an integer edge from the depot, not already visited in the $C$ stack then
4:       initialize $C$ stack with depot
5:    else
6:       initialize $U$ stack with depot
7:    end if
8:    repeat
9:       if $U$ stack is not empty then
10:          generate an unstructured vertex set from the first vertex of the $U$ stack.
11:          remove vertex from $U$ stack iteratively while adding sequentially all adjacent edges with fractional values linked
             to it. If integer edges are encountered fill $C$ stack with the connected vertices
12:          if there is only one vertex in the $U$ stack that is not the depot then
13:             then transfer vertex to the $C$ stack
14:          end if
15:          if two chains or more are coming out of the $U$ stack then
16:             expand unstructured vertex set by Algorithm 2
17:          end if
18:       end if
19:       repeat
20:          if $C$ stack is not empty then
21:             generate chain from solution by sequentially adding and iteratively removing vertices that are connected by
                integer edges and putting fractional edges in $U$ stack.
22:             insert chain into component list and remove from $C$ stack
23:          end if
24:       until $C$ stack is empty
25:       if current $U$ stack contains the depot or has at least two elements then
26:          insert unstructured vertex set into component list and remove from $U$ stack
27:       end if
28:    until both stacks are empty
29:    if the number of vehicles involved equals one and the $U$ stack and $C$ stack are empty then
30:       current sequence of components induces a partial route
31:    end if
32:    if number of vehicles involved is greater than one then
33:       the identified structure is not a partial route
34:       merge all components and place all vertices into the $L$ component
35:    end if
36: until all edges have been processed
```

Algorithm 2 Expanding unstructured component

```
1: repeat
2:     repeat
3:         generate chain from solution by sequentially adding and iteratively removing vertices that are connected by integer
           edges and putting fractional edges in $U$ stack.
4:         if segment contains depot then
5:             insert segment into component list
6:         else
7:             insert segment into expanded unstructured vertex set
8:         end if
9:     until $C$ stack is empty
10:    repeat
11:        generate an unstructured vertex set from the first vertex of the $U$ stack.
12:        remove vertex from $U$ stack iteratively while adding sequentially all adjacent edges with fractional values linked to
           it. If integer edges are encountered fill $C$ stack with the connected vertices
13:        if current unstructured vertex set contains the depot or has at least two elements then
14:            insert current unstructured vertex into the expanded unstructured vertex set
15:        end if
16:    until $U$ stack is empty
17: until one chain coming out of the $U$ expanded unstructured vertex set
```

Acknowledgments

Thanks are due to Serge Bisaillon for his outstanding assistance with the implementation of the algorithm and Marco Veneroni for his valuable comments. This research was partly supported by the Canadian Natural Sciences and Engineering Research Council under grants 338816-10, 39682-10, and 227837-04. This support is gratefully acknowledged.

References

[1] A. Ak and A. Erera. A paired-vehicle recourse strategy for the vehicle-routing problem with stochastic demands. *Transportation Science*, 41(2):222–237, 2007.

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

[3] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.

[4] D. J. Bertsimas. A vehicle routing problem with stochastic demand. *Operations Research*, 40(3):574–585, 1992.

[5] D. J. Bertsimas, P. Jaillet, and A. R. Odoni. A priori optimization. *Operations Research*, 38(6):1019–1033, 1999.

[6] D.J. Bertsimas. *Probabilistic Combinatorial Optimization Problems*. PhD thesis, Operations Research Center, Massachusetts Institute of Technology, 1988.

[7] K. Chepuri and T. Homem de Mello. Solving the vehicle routing problem with stochastic demands using the cross entropy method. *Annals of Operations Research*, 134:153–181, 2005.

[8] C.H. Christiansen and J. Lysgaard. A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research Letters*, 35(6):773–781, 2007.

[9] J.-F. Cordeau, G. Laporte, J.-Y. Potvin, and M.W.P Savelsbergh. *Transportation*, pages 367–428. Handbooks in Operations Research and Management Science. Elsevier, Amsterdam, 2007.

[10] M. Gendreau, G. Laporte, and R. Séguin. An exact algorithm for the vehicle routing problem with stochastic demands and customers. *Transportation Science*, 29(2):143–155, 1995.

[11] M. Gendreau, G. Laporte, and R. Séguin. A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. *Operations Research*, 44(3):469–477, 1996.

[12] B.L. Golden, S. Raghavan, and E.A. Wasil. *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer, New York, 2008.

[13] J. C. Goodson, J. W. Ohlmann, and B. W. Thomas. Cyclic-order neighborhoods with application to the vehicle routing problem with stochastic demand. *European Journal of Operational Research*, 217(2):312–323, 2012.

[14] C. Hjorring and J. Holt. New optimality cuts for a single-vehicle stochastic routing problem. *Annals of Operations Research*, 86:569–584, 1999.

[15] P. Jaillet. A priori solution of a traveling salesman problem in which a random subset of the customers are visited. *Operations Research*, 36(6):929–936, 1978.

[16] A. Juan, J. Faulin, S. Grasman, D. Riera, J. Marull, and C. Mendez. Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands. *Transportation Research Part C: Emerging Technologies*, 19(5):751–765, 2011.

[17] V. Lambert, G. Laporte, and F.V. Louveaux. Designing collection routes through bank branches. *Computers & Operations Research*, 20(7):783–791, 1993.

[18] G. Laporte. Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416, 2009.

[19] G. Laporte and F. V. Louveaux. The integer L-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13(3):133–142, 1993.

[20] G. Laporte, F. V. Louveaux, and L. Van hamme. An integer L-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. *Operations Research*, 50(3):415–423, 2002.

[21] H. Lei, G. Laporte, and B. Guo. The capacitated vehicle routing problem with stochastic demands and time windows. *Computers & Operations Research*, 38(12):1775–1783, 2011.

[22] F.V. Louveaux. An introduction to stochastic transportation models. In M. Labbé, G. Laporte, K. Tanczos, and P. Toint, editors, *Operations Research and Decision Aid Methodologies in Traffic and Transportation Management*, pages 244–263. NATO ASI Series, Series F: Computer and Systems Sciences, Springer-Verlag, Berlin and Heidelberg, 1998.

[23] J. Lysgaard, A. N. Letchford, and R. W. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100(2):423–445, 2004.

[24] W. Rei, M. Gendreau, and P. Soriano. A hybrid monte carlo local branching algorithm for the single vehicle routing problem with stochastic demands. *Transportation Science*, 44(1):136–146, 2010.

[25] N. Secomandi and F. Margot. Reoptimization approaches for the vehicle-routing problem with stochastic demands. *Operations Research*, 57(1):214–230, 2009.

[26] R. M. Van Slyke and R. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17(4):638–663, 1969.

[27] I. Sungur, F. Ordóñez, and M. Dessouky. A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty. *IIE Transactions*, 40:509–523, 2008.

[28] P. Toth and D. Vigo, editors. *The Vehicle Routing Problem.* SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, 2002.

[29] W.-H Yang, K. Mathur, and R.H. Ballou. Stochastic vehicle routing problem with restocking. *Transportation Science*, 34(3):99–112, 2000.