Partial-route inequalities for the multi-vehicle routing problem with stochastic demands

Ola Jabali [a,*], Walter Rei [b], Michel Gendreau [c], Gilbert Laporte [a]

[a] CIRRELT and HEC Montréal 3000, chemin de la Côte-Sainte-Catherine, Montréal, Canada H3C 3J7

[b] CIRRELT and Department of Management and Technology, Université du Québec à Montréal, 315 rue Sainte-Catherine est, Montréal, Canada H2X 3X2

[c] CIRRELT and Department of Mathematics and Industrial Engineering, École Polytechnique de Montréal, C.P. 6128, succursale Centre-ville, Montréal, Canada H3C 3J7

Article history:
Received 29 November 2012
Received in revised form 20 February 2014
Accepted 22 May 2014
Available online 14 June 2014

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 [17]). A number of variants of the VRP have been proposed (see, e.g., Toth and Vigo [27] and Golden et al. [11]) 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

* Corresponding author. Tel.: +1 5143406154; fax: +1 5143406834.
E-mail addresses: ola.jabali@hec.ca, Ola.Jabali@cirrelt.ca (O. Jabali), Walter.Rei@cirrelt.ca (W. Rei), Michel.Gendreau@cirrelt.ca (M. Gendreau), Gilbert.Laporte@cirrelt.ca (G. Laporte).

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 [21]). As a result, several stochastic versions of the VRP have been studied in recent years (Cordeau et al. [8]).

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. [16]), in home oil delivery (Chepuri and Homem de Mello [23]), 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 [5], Jaillet [14] and Bertsimas et al. [6]. 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 [7], Gendreau et al. [9,10], Goodson et al. [12], Hjorring and Holt [13], Laporte et al. [19], Lei et al. [20] 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. [15]). Note that the algorithms described in Ak and Erera [1], Hjorring and Holt [13], Rei et al. [24] 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 study by Sungur et al. [26] defines it in the context of robust optimization which yields routes that minimize transportation costs while satisfying all the 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. [19]).

As in Laporte et al. [19], 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 [18], an extension of the *L*-shaped method of Van Slyke and Wets [28] 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 [13] and Laporte et al. [19], 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 [13] for the single vehicle case, and by Laporte et al. [19] 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 solving to optimality 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. [19], 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 3–5.

2.1. The VRPSD model

The VRPSD is defined on a complete undirected graph $G = (V, E)$, where $V = \{v_1, \ldots, v_n\}$ 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. [19] 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

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

subject to

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

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

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

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

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

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

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. [19], the computation of $\mathcal{Q}^{k,1}$ for route k defined by $(v_{i_1} = v_1, v_{i_2}, \ldots, v_{i_{t+1}} = v_1)$ is given by

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

The first factor in the double summation is the probability of incurring the lth 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. [19]. 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,

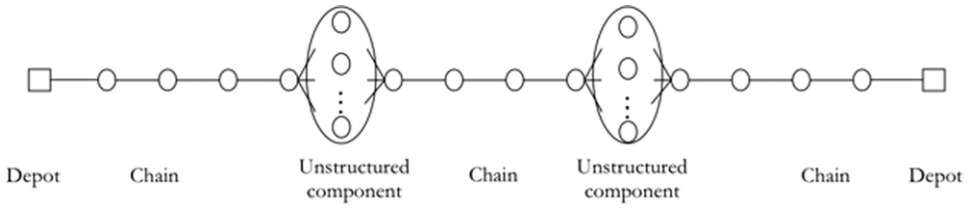


Fig. 1. Example of a general partial route.

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^ν) 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 $cx^\nu + \Theta^\nu \geq \bar{z}$, 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 $\Theta^\nu \geq Q(x^\nu)$, then fathom the current node and return to Step 1. Otherwise add an optimality cut defined as

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

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 [13] and extended to the multi-vehicle case by Laporte et al. [19]. In this section we introduce a generalized definition of partial routes. It is broader than the one proposed by Hjorring and Holt [13] 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. [19] that a rather large number of the Hjorring and Holt [13] LBFs have to be added to the master problem in order to solve the VRPSD.

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 Fig. 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 rth 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 kth 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,

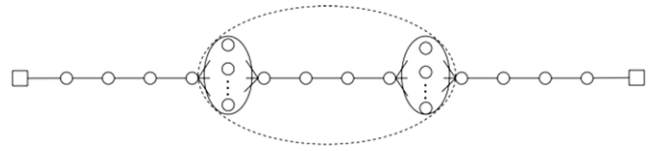
$$\sum_{(v_i,v_j)\in S_h^r} x_{ij} = |S_h^r| - 1.$$

Let U_h^r denote the rth UVS in h. Then

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

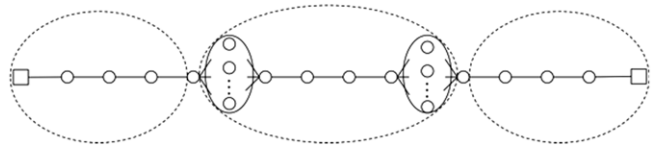
For each $1 \leq r \leq b-1$,

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



(a) α-route.

(b) β-route.



(c) γ-route.

Fig. 2. Examples of general partial routes.

Similarly, for each $2 \leq r \leq b$,

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

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 Fig. 1 can be viewed differently depending on how chains and vertices are interpreted. For example, in Fig. 2(a) only the first and last chains are viewed as such, whereas the intermediate part of the solution is viewed as a UVS. In Fig. 2(b), the original alternation of chains and UVSs is maintained. In Fig. 2(c), 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.

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, $r = 1, \ldots, b-1$, 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 $(v_0 = v_{1^{1h}}, \ldots, v_{l^{rh}}, v^1, v_{1^{2h}}, \ldots, v_{l^{2h}}, v^2, \ldots, v^{b-1}, v_{1^{bh}}, \ldots, v_{l^{bh}})$ and the value of $\mathcal{Q}^{k,\delta}$ for this route k is computed as in Eq. (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 taking the value $W_h(x)$ for each partial route h in x. The sum of the functional values $W_h(x)$ 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 extend W_h 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 value taken by the functional W_h for a partial route h of solution x:

$$W_h(x) = \sum_{r=1}^{b} \sum_{\substack{(v_i,v_j)\in S_h^r \\ v_i \neq v_1}} 3x_{ij} + \sum_{(v_1,v_j)\in S_h^1} x_{1j} + \sum_{(v_1,v_j)\in S_h^b} x_{1j} + \sum_{r=1}^{b-1} \sum_{v_i,v_j \in U_h^r} 3x_{ij}$$
$$+ \sum_{r=1}^{b-1} \sum_{\substack{v_j \in U_h^r \\ v_{lrh} \neq v_1}} 3x_{l^{rh}j} + \sum_{r=2}^{b} \sum_{\substack{v_j \in U_h^{r-1} \\ v_{1rh} \neq v_1}} 3x_{1^{rh}j} + \sum_{\substack{v_j \in U_h^1 \\ v_{l1h} = v_1}} x_{l^{1h}j} + \sum_{\substack{v_j \in U_h^{b-1} \\ v_{1bh} = v_1 \\ v_{lb-1,h} \neq v_1}} x_{1^{bh}j} - (3|R_h| - 5). \tag{14}$$

In Eq. (14), the coefficients of all edges containing the depot are equal to 1 while all other edges have a coefficient of 3. The definition of $W_h(x)$ 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. [19], 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 $W_h(x)$ 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

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

Again the above summation implies that $W_h(x)$ 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. This is represented by the term

$$\sum_{\substack{v_j \in U_h^1 \\ v_{l1h} = v_1}} x_{l^{1h}j}.$$

If the partial route consists of a single UVS and two single vertex chains, i.e., $v_{1^{bh}} = v_1$ and $v_{lb-1,h} = v_1$, 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

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

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

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

This may also be expressed as

$$3(|R_h \setminus \{v_1\}| - 1) + 2.$$

Thus, considering h, the value of $W_h(x)$ 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. Extension 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.

In (14), $W_h(x)$ 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) $v_1 \notin J$, (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

$$W_h(J|x) = \sum_{r=1}^{b} \sum_{\substack{(v_i,v_j) \in S_h^r \cap J \\ v_i \neq v_1}} 3x_{ij} + \sum_{(v_1,v_j) \in S_h^1 \cap J \cup \{v_1\}} x_{1j} + \sum_{(v_1,v_j) \in S_h^b \cap J \cup \{v_1\}} x_{1j} + \sum_{r=1}^{b-1} \sum_{v_i,v_j \in U_h^r \cap J} 3x_{ij}$$
$$+ \sum_{r=1}^{b-1} \sum_{\substack{v_j \in U_h^r \cap J \\ v_{lrh} \neq v_1}} 3x_{l^{rh},j} + \sum_{r=2}^{b} \sum_{\substack{v_j \in U_h^{r-1} \cap J \\ v_{1^{rh}} \neq v_1}} 3x_{1^{rh},j} + \sum_{\substack{v_j \in U_h^1 \cap J \\ v_{l1h} = v_1}} x_{l^{1h},j} + \sum_{\substack{v_j \in U_h^{b-1} \cap J \\ v_{1bh} = v_1 \\ v_{lb-1,h} \neq v_1}} x_{1^{bh},j}.$$

The value of $W_h(J|x')$ 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 $1 \leq i < q$. Then $W_h(J|x') \leq 3(|J| - 1) + 2$.*

Proof. We observe that

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

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

$$\sum_{(v_1,v_j)\in S_h^1\cap J\cup\{v_1\}} x'_{1j}+\sum_{(v_1,v_j)\in S_h^b\cap J\cup\{v_1\}} x'_{1j}+\sum_{\substack{v_j\in U_h^1\cap J\\ v_{l1h}=v_1}} x'_{l^{1h},j}+\sum_{\substack{v_j\in U_h^{b-1}\cap J\\ v_{1bh}=v_1\\ v_{lb-1,h}\neq v_1}} x'_{1^{bh},j}\leq 2,$$

where the equality holds if $G_h(\bar{x}_{1,k_1^1})>0$ and $G_h(\bar{x}_{1,k_q^1})>0$. Therefore,

$$W_h(J|x')\leq 3(|J|-1)+2.\quad\square$$

We note that for any partition $J=\{J_1,J_2,\ldots\}$ we have $W_h(J|x')\geq\sum_i W_h(J_i|x')$. Therefore, 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 in x' there is no edge connecting a customer vertex of $R_{h'}$ to J, 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 $W_h(x')\leq 0$. $\square$

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

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

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

$$\sum_{(v_1,v_j)\in S_h^1\cap R_{h'}} x'_{1j}+\sum_{(v_1,v_j)\in S_h^b\cap R_{h'}} x'_{1j}+\sum_{\substack{v_j\in U_h^1\cap R_{h'}\\ v_{l1h}=v_1}} x'_{l^{1h},j}+\sum_{\substack{v_j\in U_h^{b-1}\cap R_{h'}\\ v_{1bh}=v_1\\ v_{lb-1,h}\neq v_1}} x'_{1^{bh},j}\leq 1.$$

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,

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

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 Eq. (13).

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

$$\Theta \geq L + (P - L)\left(\sum_{h=1}^{r} W_h(\bar{x}) - r + 1\right) \tag{15}$$

is a valid inequality for the (VRPSD).

Proof. Let x' be a feasible solution to the VRPSD. 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 (15) 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 Fig. 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 [13] 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 (Fig. 2(a)). 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 Fig. 2(b) is an alternation of b chains and $b - 1$ UVSs, where $b \geq 2$. The functional W_h^β applied in this case is identical to W_h. 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^β will be active on a smaller solution space, when compared to W_h^α.

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 (Fig. 2(c)). 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^γ will be active on a larger solution space, when compared to W_h^α and W_h^β.

Table 1
Parameter combinations used in the experiments.

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

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. The complexity of Algorithm 2 is $O(|E|)$ since it scans a stack of edges. The complexity of Algorithm 1 is therefore $O(|E|^2)$ since it calls Algorithm 2 $O(|E|)$ times.

Recall that Laporte et al. [19] proposed a heuristic separation procedure for constraints (15). 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. [19] 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 (15) has been applied to no avail on the solution of the current subproblem. 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 these 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 (15).

6. Implementation and computational results

The integer L-shaped algorithm was coded in a C++ environment with CPLEX 12.3. All experiments are conducted on an Intel(R) Xeon(R) CPU X5675 with 12-Core 3.07 GHz and 96 GB of RAM (by using a single thread). 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. [22]. These cuts were added as long as violated constraints were found. The instances were generated based on the same principles as in Laporte et al. [19]. 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 $\bar{f} = \sum_{i=2}^{n} \mu_i / mD$ 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 5 h.

We have performed extensive numerical analyses to assess the performance of the algorithm. Section 6.2 assesses our separation procedure, whereas Section 6.3, compares several LBF schemes.

6.1. No LBF vs. with LBF

In order to evaluate the added value of introducing LBF cuts, we have compared the algorithm without LBF cuts with the basic LBF implementation using the heuristic separation algorithm proposed by Laporte et al. [19]. The results of the former algorithm are denoted by no LBF while those of the latter are denoted by LBF_0.

Out of the 270 test instances, no LBF solved 56 instances to optimality, while LBF_0 solved 99 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. All instances solved with no LBF were also solved with LBF_0. Considering these 56 instances the average runtimes for LBF_0 are less than half the average runtime for no LBF. We report average values for the instances solved to optimality. If the algorithm did not solve any of the ten instances to optimality, then we do not report a value.

Table 2
Results for the instances solved to optimality by both no LBF and LBF_0.

n	m	$\bar{f}$	Number of solved instances	Runtime (min) no LBF	Runtime (min) LBF_0	Gap no LBF (unsolved by both) (%)	Gap LFB_0 (unsolved by both) (%)
60	2	0.90	6	2.5	10.2	0.3	0.1
60	2	0.93	8	31.9	3.8	0.3	0.2
60	2	0.95	4	59.5	19.8	0.7	0.4
70	2	0.90	6	29.3	6.9	0	0
70	2	0.93	3	3.6	7.2	0.4	0.4
70	2	0.95	1	19.1	7.1	0.9	0.7
80	2	0.90	6	2.8	3.5	0.1	0.1
80	2	0.93	4	69.8	65.9	0.8	0.7
80	2	0.95				0.9	0.7
50	3	0.85	3	7.1	0.7	1.1	0.2
50	3	0.88	1	0.4	0.5	1.7	0.6
50	3	0.90				2	1.2
60	3	0.85	3	11.4	2.2	1.1	0.5
60	3	0.88				1.4	0.8
60	3	0.90				1.5	1.2
70	3	0.85	2	5.4	1.4	1.2	1.2
70	3	0.88	1	115.8	33.7	1.4	1.1
70	3	0.90				2.1	2.1
40	4	0.80	3	33.5	2.7	2.2	1.3
40	4	0.82	2	8.6	0.6	2	1.9
40	4	0.85				2.7	2
50	4	0.80	2	69.3	26.8	2.1	1.8
50	4	0.82				1.5	1.5
50	4	0.85				2.3	2.5
60	4	0.80				2.6	1.4
60	4	0.82	1	34.2	94.2	1.3	1.2
60	4	0.85				2.6	2.6
Average				26.5	13	1.6	1.3
Total			56				

6.2. 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. [19] 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 and LBF_α solved 99 to optimality. Table 3 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, 49 out 90 instances with two vehicles were solved to optimality while only 15 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 lower for LBF_α when compared to LBF_0. The last two columns of Table 3 report the gaps for instances unsolved by both LBF_0 and LBF_α. For these cases the average gap of LBF_α is 1.28% while that of LBF_0 is 1.29%.

6.3. 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 $LBF_{\alpha\beta}$. Similarly, in the fifth set we present experiments for $LBF_{\beta\gamma}$. Finally, we combine the three options in the experimental set $LBF_{\alpha\beta\gamma}$.

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 $LBF_{\beta\gamma}$) 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 $LBF_{\alpha\beta\gamma}$ has relatively large runtimes.

Table 6 presents the average gaps for the unsolved instances. All algorithms achieved an average gap of less than 1.5%. The table shows that the gaps increase with the number of vehicles. A similar observation can be made by considering the

Table 3
Results for the instances solved to optimality by both LBF_0 and LBF_α.

n	m	$\bar{f}$	Number of solved instances	Runtime (min) LBF_0	Runtime (min) LBF_α	Gap LBF_0 (unsolved by both) (%)	Gap LFB_α (unsolved by both) (%)
60	2	0.90	8	39.5	39.3	0.1	0.1
60	2	0.93	9	4.3	4.4	0.2	0.2
60	2	0.95	4	19.8	21.9	0.4	0.4
70	2	0.90	10	11.9	11.6		
70	2	0.93	3	7.2	7	0.4	0.4
70	2	0.95	2	103.1	100.8	0.7	0.7
80	2	0.90	8	23.6	22.8	0.1	0.1
80	2	0.93	5	104	101.7	0.7	0.7
80	2	0.95				0.7	0.7
50	3	0.85	7	57.3	57.1	0.2	0.3
50	3	0.88	6	103.4	101.9	0.6	0.6
50	3	0.90	2	23.4	23.9	1.2	1.2
60	3	0.85	5	23.1	22.4	0.5	0.5
60	3	0.88	3	68.5	66.5	0.8	0.8
60	3	0.90	2	113.2	110.9	1.2	1.2
70	3	0.85	8	9.6	9.5	1.2	1.2
70	3	0.88	2	24.3	26.6	1.1	1.1
70	3	0.90				2.1	2.1
40	4	0.80	3	2.7	2.7	1.3	1.3
40	4	0.82	4	15	14.3	1.9	1.9
40	4	0.85	1	12.6	12.8	2.0	2.0
50	4	0.80	4	46.4	49.2	1.8	1.8
50	4	0.82	1	56.6	60.7	1.5	1.5
50	4	0.85				2.5	2.3
60	4	0.80	1	174	158.3	1.4	1.4
60	4	0.82	1	94.2	102.1	1.2	1.2
60	4	0.85				2.6	2.5
Average				38.6	38.3	1.3	1.3
Total			99				

Table 4
Number of solved instances to optimality for several families of partial routes.

n	m	$\bar{f}$	LBF_α	LBF_β	LBF_γ	$LBF_{\alpha\beta}$	$LBF_{\beta\gamma}$	$LBF_{\alpha\beta\gamma}$
60	2	0.90	8	6	10	7	10	10
60	2	0.93	9	9	9	9	9	9
60	2	0.95	4	4	5	4	6	6
70	2	0.90	10	10	10	10	10	10
70	2	0.93	3	3	5	3	5	5
70	2	0.95	2	2	2	1	2	2
80	2	0.90	8	6	9	8	9	9
80	2	0.93	5	3	4	3	4	4
80	2	0.95	0	0	0	0	0	0
50	3	0.85	7	7	8	6	6	7
50	3	0.88	6	6	6	4	5	4
50	3	0.90	2	2	2	2	1	1
60	3	0.85	5	5	5	5	6	5
60	3	0.88	3	3	1	3	1	1
60	3	0.90	2	2	0	1	1	0
70	3	0.85	8	8	7	8	7	7
70	3	0.88	2	2	2	2	2	2
70	3	0.90	0	0	0	0	0	0
40	4	0.80	3	4	4	4	4	4
40	4	0.82	4	4	4	4	5	5
40	4	0.85	1	1	1	1	1	1
50	4	0.80	4	4	4	4	4	4
50	4	0.82	1	1	1	1	0	1
50	4	0.85	0	0	0	0	0	0
60	4	0.80	1	1	1	0	1	1
60	4	0.82	1	1	2	1	1	1
60	4	0.85	0	0	0	0	0	0
Total			99	94	102	91	100	99

Table 5
Runtimes in minutes of the solved instances to optimality for several families of partial routes.

n	m	$\bar{f}$	LBF_{α}	LBF_{β}	LBF_{γ}	$LBF_{\alpha\beta}$	$LBF_{\beta\gamma}$	$LBF_{\alpha\beta\gamma}$
60	2	0.90	39.3	32	42.1	43.0	44.1	48.2
60	2	0.93	4.4	7.8	4.3	8.3	3.2	3.5
60	2	0.95	21.9	29.5	19.5	29.3	74.7	56.1
70	2	0.90	11.6	31.4	18.9	19.7	7.4	6.3
70	2	0.93	7.0	12.9	84.5	13.3	68	83
70	2	0.95	100.8	121.7	59.4	10.0	25.3	44.6
80	2	0.90	22.8	3.8	34.0	21.8	41.5	45.3
80	2	0.93	101.7	11.4	21.5	8.1	23.4	24.8
80	2	0.95						
50	3	0.85	57.1	72.4	103.5	43.6	78.1	85.0
50	3	0.88	101.9	107	110.6	51.5	82.4	73.9
50	3	0.90	23.9	95.3	176.6	29.1	255.5	291.2
60	3	0.85	22.4	33.6	39.5	39.9	46.8	89.7
60	3	0.88	66.5	80.3	79.1	116.4	256.6	171
60	3	0.90	110.9	95.8		23.9	275.1	
70	3	0.85	9.5	23.5	25.6	17.2	42.2	38.1
70	3	0.88	26.6	29.3	41.9	30.8	70.7	75.9
70	3	0.90						
40	4	0.80	2.7	63.5	21.4	69.0	24.3	30.9
40	4	0.82	14.3	14.8	11.3	22.5	52.1	57.3
40	4	0.85	12.8	12.7	55.3	15.1	21.6	29.7
50	4	0.80	49.2	61.9	25	64.5	21.4	20.9
50	4	0.82	60.7	47.0	13.4	65.9		16.7
50	4	0.85						
60	4	0.80	158.3	174.3	94.5		133.7	139.8
60	4	0.82	102.1	91.5	75.8	37.2	67.3	63.8
60	4	0.85						
Average			38.3	43.7	45.2	32.7	49	49.4

Table 6
Optimality gaps of the unsolved instances for several families of partial routes.

m	n	$\bar{f}$	LBF_{α} (%)	LBF_{β} (%)	LBF_{γ} (%)	$LBF_{\alpha\beta}$ (%)	$LBF_{\beta\gamma}$ (%)	$LBF_{\alpha\beta\gamma}$ (%)
60	2	0.90	0.1	0.2		0.2		
60	2	0.93	0.2	0.3	0.1	0.3	0.1	0.2
60	2	0.95	0.4	0.5	0.4	0.5	0.5	0.5
70	2	0.90						
70	2	0.93	0.4	0.5	0.2	0.4	0.3	0.3
70	2	0.95	0.7	0.8	0.7	0.7	0.7	0.7
80	2	0.90	0.1	0.2	0.1	0.1	0.1	0.1
80	2	0.93	0.7	0.6	0.5	0.6	0.5	0.5
80	2	0.95	0.7	0.8	0.6	0.8	0.6	0.6
50	3	0.85	0.3	0.5	0.2	0.3	0.4	0.3
50	3	0.88	0.6	0.7	0.3	0.4	0.8	0.7
50	3	0.90	1.2	1.1	1	1.2	1.3	1.3
60	3	0.85	0.5	0.7	0.5	0.6	0.8	0.6
60	3	0.88	0.8	0.8	0.7	0.8	0.6	0.7
60	3	0.90	1.2	1.2	0.8	1.1	0.9	0.8
70	3	0.85	1.2	0.9	0.5	0.9	0.6	0.6
70	3	0.88	1.1	1	1.1	1.1	1.4	1.4
70	3	0.90	2.1	2	2.2	2	2.4	2.4
40	4	0.80	1.3	1.5	1.3	1.6	1.7	1.7
40	4	0.82	1.9	1.7	0.7	1.9	1.1	1.4
40	4	0.85	2	2.2	2.1	2.1	2.9	2.6
50	4	0.80	1.8	1.8	1.7	1.9	1.7	1.7
50	4	0.82	1.5	1.6	1.4	1.6	2	2
50	4	0.85	2.3	2.6	2.4	2.8	2.5	2.5
60	4	0.80	1.4	1.7	1.4	1.5	1.4	1.2
60	4	0.82	1.2	1.2	1.9	1.3	1.8	1.8
60	4	0.85	2.5	2.3	2.6	2.4	3.2	3
Average			1.3	1.3	1.2	1.3	1.4	1.4

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

Range	LBF_0 (%)	LBF_α (%)	LBF_β (%)	LBF_γ (%)	$LBF_{\alpha\beta}$ (%)	$LBF_{\beta\gamma}$ (%)	$LBF_{\alpha\beta\gamma}$ (%)
$= 0\%$	36.7	36.7	34.8	38.1	33.7	37	37
$\leq 1\%$	70.4	70.4	67.8	76.7	68.5	70.7	71.9
$\leq 3\%$	93.7	93.7	94.4	93	93.3	90.4	90.7
$\leq 5\%$	99.6	99.6	99.3	98.1	99.3	97	97
$\leq 7\%$	100	100	100	100	100	100	100

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

	Average			% of instances
	LBF_0	LBF_γ	$LBF_0 - LBF_\gamma$	$LBF_0 > LBF_\gamma$
Runtime (min)	29.5	31.1	−1.5	64.4%
Subtour elimination and capacity constraints (4)	1285.0	1933.7	−648.7	15.6%
Total LBF cuts	1125.1	126.5	998.6	90.0%
Optimality cuts (10)	283.2	507.2	−224.0	30.0%

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, 94.4% of the 270 instances solved with LBF_β had a gap of at most 3%. Similarly we observe that LBF_0 solves more than 99.6% of the instances with a gap under 5%, indicating that while the exact separation procedure helped to 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 90 instances were solved to optimality by both LBF_0 and LBF_γ, and an additional 12 instances were solved to optimality only by LBF_γ. The runtime of the instances solved by LBF_γ is 1.5 min longer than that of LBF_0, yet in 58 out of the 90 instances the runtime for LBF_0 was longer than that of LBF_γ. The instances solved by LBF_0 required on average 648.7 fewer subtour elimination and capacity constraints. The third line of Table 7 compares the total number of LBF cuts produced by LBF_γ and LBF_0. We observe that on average 998.6 additional LBF cuts were added per instance in LBF_0. Furthermore, the number of optimality cuts added by LBF_γ is on average only 224 more than the corresponding number for LBF_0.

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. [19] 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 [13]. 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 solved instances to optimality increases from 99 to 102.

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. [19]. 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_γ is better than that of LFB_β, 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 other stochastic features.

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 436014-2013, 338816-10, 39682-10, and 227837-04. This support is gratefully acknowledged.

Appendix. Pseudo-codes for the separation procedure

Algorithm 1 Partial route separation procedure

```
consider all edges of the current solution
repeat
    if there exists an integer edge from the depot, not already visited in the C stack then
        initialize C stack with depot
    else
        initialize U stack with depot
    end if
    repeat
        if U stack is not empty then
            generate an unstructured vertex set from the first vertex of the U stack.
            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
            if there is only one vertex in the U stack that is not the depot then
                then transfer vertex to the C stack
            end if
            if two chains or more are coming out of the U stack then
                expand unstructured vertex set by Algorithm 2
            end if
        end if
        repeat
            if C stack is not empty then
                generate chain from solution by sequentially adding and iteratively removing vertices that are connected by integer edges and putting fractional edges in U stack.
                insert chain into component list and remove from C stack
            end if
        until C stack is empty
        if current U stack contains the depot or has at least two elements then
            insert unstructured vertex set into component list and remove from U stack
        end if
    until both stacks are empty
    if the number of vehicles involved equals one and the U stack and C stack are empty then
        current sequence of components induces a partial route
    end if
    if number of vehicles involved is greater than one then
        the identified structure is not a partial route
        merge all components and place all vertices into the L component
    end if
until all edges have been processed
```

Algorithm 2 Expanding unstructured component

```
repeat
    repeat
        generate chain from solution by sequentially adding and iteratively removing vertices that are connected by integer edges and putting fractional edges in U stack.
        if segment contains depot then
            insert segment into component list
        else
            insert segment into expanded unstructured vertex set
        end if
    until C stack is empty
    repeat
        generate an unstructured vertex set from the first vertex of the U stack.
        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
        if current unstructured vertex set contains the depot or has at least two elements then
            insert current unstructured vertex into the expanded unstructured vertex set
        end if
    until U stack is empty
until one chain coming out of the U expanded unstructured vertex set
```

References

[1] A. Ak, A. Erera, A paired-vehicle recourse strategy for the vehicle-routing problem with stochastic demands, Transp. Sci. 41 (2) (2007) 222–237.
[2] R. Baldacci, A. Mingozzi, R. Roberti, New route relaxation and pricing strategies for the vehicle routing problem, Oper. Res. 59 (5) (2011) 1269–1283.
[3] J.F. Benders, Partitioning procedures for solving mixed-variables programming problems, Numer. Math. 4 (1962) 238–252.

[4] D.J. Bertsimas, A vehicle routing problem with stochastic demand, Oper. Res. 40 (3) (1992) 574–585.
[5] D.J. Bertsimas, Probabilistic combinatorial optimization problems (Ph.D. thesis), Operations Research Center, Massachusetts Institute of Technology, 1988.
[6] D.J. Bertsimas, P. Jaillet, A.R. Odoni, A priori optimization, Oper. Res. 38 (6) (1999) 1019–1033.
[7] C.H. Christiansen, J. Lysgaard, A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands, Oper. Res. Lett. 35 (6) (2007) 773–781.
[8] J.-F. Cordeau, G. Laporte, J.-Y. Potvin, M.W.P Savelsbergh, Transportation, in: Handbooks in Operations Research and Management Science, Elsevier, Amsterdam, 2007, pp. 367–428.
[9] M. Gendreau, G. Laporte, R. Séguin, An exact algorithm for the vehicle routing problem with stochastic demands and customers, Transp. Sci. 29 (2) (1995) 143–155.
[10] M. Gendreau, G. Laporte, R. Séguin, A tabu search heuristic for the vehicle routing problem with stochastic demands and customers, Oper. Res. 44 (3) (1996) 469–477.
[11] B.L. Golden, S. Raghavan, E.A. Wasil, The Vehicle Routing Problem: Latest Advances and New Challenges, Springer, New York, 2008.
[12] J.C. Goodson, J.W. Ohlmann, B.W. Thomas, Cyclic-order neighborhoods with application to the vehicle routing problem with stochastic demand, European J. Oper. Res. 217 (2) (2012) 312–323.
[13] C. Hjorring, J. Holt, New optimality cuts for a single-vehicle stochastic routing problem, Ann. Oper. Res. 86 (1999) 569–584.
[14] P. Jaillet, A priori solution of a traveling salesman problem in which a random subset of the customers are visited, Oper. Res. 36 (6) (1978) 929–936.
[15] A. Juan, J. Faulin, S. Grasman, D. Riera, J. Marull, C. Mendez, Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands, Transp. Res. C 19 (5) (2011) 751–765.
[16] V. Lambert, G. Laporte, F.V. Louveaux, Designing collection routes through bank branches, Comput. Oper. Res. 20 (7) (1993) 783–791.
[17] G. Laporte, Fifty years of vehicle routing, Transp. Sci. 43 (4) (2009) 408–416.
[18] G. Laporte, F.V. Louveaux, The integer *L*-shaped method for stochastic integer programs with complete recourse, Oper. Res. Lett. 13 (3) (1993) 133–142.
[19] G. Laporte, F.V. Louveaux, L. Van hamme, An integer *L*-shaped algorithm for the capacitated vehicle routing problem with stochastic demands, Oper. Res. 50 (3) (2002) 415–423.
[20] H. Lei, G. Laporte, B. Guo, The capacitated vehicle routing problem with stochastic demands and time windows, Comput. Oper. Res. 38 (12) (2011) 1775–1783.
[21] F.V. Louveaux, An introduction to stochastic transportation models, in: M. Labbé, G. Laporte, K. Tanczos, P. Toint (Eds.), Operations Research and Decision Aid Methodologies in Traffic and Transportation Management, in: NATO ASI Series, Series F: Computer and Systems Sciences, Springer-Verlag, Berlin and Heidelberg, 1998, pp. 244–263.
[22] J. Lysgaard, A.N. Letchford, R.W. Eglese, A new branch-and-cut algorithm for the capacitated vehicle routing problem, Math. Program. 100 (2) (2004) 423–445.
[23] K. Chepuri, T. Homem de Mello, Solving the vehicle routing problem with stochastic demands using the cross entropy method, Ann. Oper. Res. 134 (2005) 153–181.
[24] W. Rei, M. Gendreau, P. Soriano, A hybrid monte carlo local branching algorithm for the single vehicle routing problem with stochastic demands, Transp. Sci. 44 (1) (2010) 136–146.
[25] N. Secomandi, F. Margot, Reoptimization approaches for the vehicle-routing problem with stochastic demands, Oper. Res. 57 (1) (2009) 214–230.
[26] I. Sungur, F. Ordóñez, M. Dessouky, A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty, IIE Trans. 40 (2008) 509–523.
[27] P. Toth, D. Vigo (Eds.), The vehicle routing problem, SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, 2002.
[28] R.M. Van Slyke, R. Wets, *L*-shaped linear programs with applications to optimal control and stochastic programming, SIAM J. Appl. Math. 17 (4) (1969) 638–663.
[29] W.-H Yang, K. Mathur, R.H. Ballou, Stochastic vehicle routing problem with restocking, Transp. Sci. 34 (3) (2000) 99–112.