The Vehicle Routing Problem with Stochastic Two-Dimensional Items

Jean-François Côté
Jean-Yves Potvin
Michel Gendreau

December 2013

CIRRELT-2013-84

The Vehicle Routing Problem with Stochastic Two-Dimensional Items

Jean-François Côté[1,2,*], Michel Gendreau[1,3], Jean-Yves Potvin[1,2]

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

[2] Department of Computer Science and Operations Research, Université de Montréal, P.O. Box 6128, Station Centre-Ville, Montréal, Canada H3C 3J7

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

Abstract. We consider a stochastic vehicle routing problem where a discrete probability distribution characterizes the two-dimensional size (height and width), as well as the weight of a subset of items to be delivered to customers. Although some item sizes and weights are not known with certainty when the routes are planned, they become known when it is time to load the vehicles, just before their departure. If it happens that not all items can be loaded in a vehicle, the items of one or more customers are put aside which lead to a penalty (or recourse cost). The objective is to minimize the sum of the routing and recourse costs. A fleet of *K* identical vehicles are available from the depot and each of them has a two-dimensional loading area and a maximal loading capacity. The routes that contains only deterministic customers have to be weight feasible and a feasible setup of the items into the loading area must exist. The expected weight and expected occupied area of the customers of the remaining routes have to be less or equal than the capacity and the loading area of the vehicles. It is also required that at least one scenario has a feasible setup of the items. The problem is modeled as a two-stage stochastic program and solved with the integer L-shaped method. Some new inequalities and lower bounds are proposed. Computational results are reported on test instances specifically generated for this problem, as well as classical instances for the deterministic case.

Keywords: Vehicle routing problem, stochastic two-dimensional items, loading constraints, L-shaped method.

Acknowledgements. Financial support for this work was provided by the Natural Sciences and Engineering Research Council of Canada (NSERC). This support is gratefully acknowledged.

---

* Corresponding author: Jean-Francois.Cote@cirrelt.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2013

1 Introduction

In the last decades, several variants of the classical Vehicle Routing Problem (VRP) have been introduced. In its simplest form, the VRP is aimed at at building routes, starting and ending at a central depot, to serve a set of customers with a fleet of identical vehicles. These routes must then satisfy various side constraints. Typically, each customer has a known demand (quantity of goods or number of items to be delivered or picked-up) and the total demand on a route should not exceed vehicle capacity.

Recently, mixed vehicle routing and loading problems have been studied [25]. In these problems, the packing of the items inside the loading area of the vehicle must be taken into account. In this work, a Two-Dimensional Orthogonal Packing Problem (2OPP) is considered where rectangular items must be delivered to customers. The items cannot be rotated and must fit in the rectangular loading area of each vehicle without overlap while satisfying unloading constraints. That is, at each delivery location, it should be possible to unload the items of the current customer by pulling them out of the vehicle without moving any item of other customers. Figure 1 shows an example for a route starting from the depot 0 and visiting the customers 1, 2, 3, 4 and 5, in this order. The figure shows two packings for this route: the first one satisfies the unloading constraints while the second one does not (note that the items are taken out from the top, which corresponds to the rear of the vehicle). In the second case, items of customers 2 and 5 must be moved to allow the items of customer 1 to be unloaded.

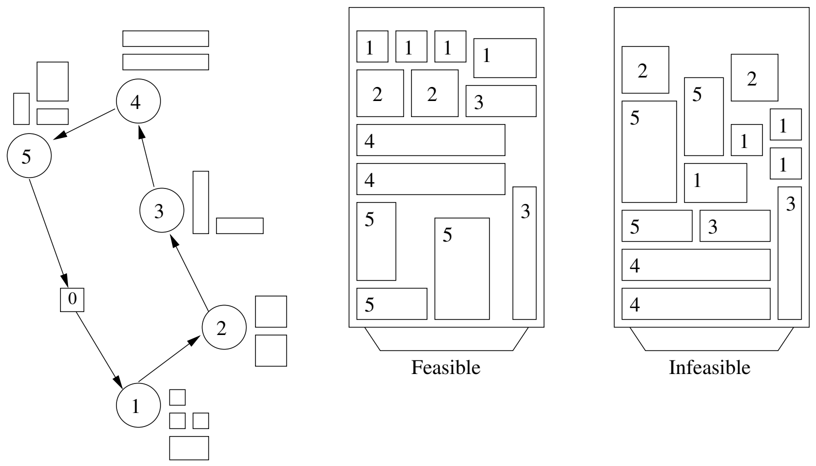

Figure 1: Packing examples

The 2OPP is often found as a subproblem of the Two-Dimensional Strip Packing Problem (2SPP) and the Two-Dimensional Bin Packing Problem (2BPP). Recent exact methods for the 2OPP can be found in [10, 14, 19, 34], while exact methods for the 2SPP are found in [2, 4, 9, 16, 33]. A recent work on the 2SPP with unloading constraints is reported in [37] where a GRASP heuristic, previously proposed in [1], and two approximation algorithms are used to solve the problem. The authors in [15] also report an exact method for the 2OPP with unloading constraints based on a Benders decomposition of the problem, which can solve instances with up to 52 items.

Mixed vehicle routing and loading problems are often addressed by local search heuristics which improve the current solution by applying different classes of modifications to the current routes (see, for example, [17, 20, 21, 31, 41]). Any modification to a vehicle route must lead to a feasible packing. To this end, simple packing heuristics like the Bottom-Left, Bottom-Left Fill and Touching Perimeter heuristics [41] are typically used. Only a few exact algorithms are reported in the literature for vehicle routing and loading problems due to the difficulty of the packing. In [26], a branch-and-cut algorithm is proposed where the classical Bottom-Left heuristic if first used to solve the packing and, if it fails, an exact branch-and-bound algorithm, based on the work in [33], is applied. The reader is referred to the surveys in [8, 25] for the three-dimensional case.

A collaboration with an industrial partner unveiled an important issue that arises in some practical applications. Quite often, the size of a few items might not be available when the delivery routes are planned, although the item type provides an indication about possible sizes and their corresponding probabilities. It means that a discrete probability distribution can be associated with these items, leading to a new class of stochastic VRPs, namely the Vehicle Routing with Stochastic Two-Dimensional Items (S2L-CVRP). When the sizes become known prior to vehicle loading and departure, it is possible that not all items assigned to a given vehicle in the planned solution will fit in the vehicle. In this case, the items of one or more customers are left on the dock (for delivery on some other day), leading to a penalty or recourse cost based on the number of unserved customers. The solution approach proposed here relies on the concept of a priori optimization [7]. That is, planned routes are built in a first-stage solution, without knowing the exact size of some items. Then, in a second-stage, the sizes become known and the recourse policy is applied when a failure occurs. The objective is to minimize the sum of the routing and (expected) recourse costs.

Different types of stochastic VRPs are reported in the literature, depending on the nature of the stochastic components. Of particular interest is the VRP with stochastic demands, where the (scalar) demand at some customers is not known with certainty [40]. These problems are typically solved with the integer L-shaped method [28], which is an extension of the method reported in [38] for continuous stochastic programs. The integer L-shaped method is basically a branch-and-cut algorithm where the expectation component of the objective is linearly bounded with optimality cuts. The method was later improved in [24, 27, 29] by introducing a weaker form of optimality cuts called Lower Bounding Functionals (LBFs) which are valid for a wider range of integer and fractional solutions. For example, VRPs with stochastic demands with up to 80 vertices (and 2 vehicles) have been solved in [27] using LBFs. However, the method becomes less effective when the number of vehicles increases because LBFs are derived from aggregation of partial routes. In [12], the authors partially address this problem by proposing a branch-and-price algorithm, but the latter is tailored for customer demands following a Poisson distribution and the size of the search space increases very quickly with the magnitude of the demand values. In our work, we go one step further along the LBF research avenue by proposing disaggregated LBFs. The idea comes from the disaggregated optimality cuts proposed in [36] where the recourse cost is distributed over a number of variables, not just a single variable.

The remainder of the paper is organized as follows. First, a formal definition and a mathematical model for our problem is presented in Section 2. Then, the integer L-shaped method is described in Section 3. Section 4 introduces our disaggregated LBFs, called L-cuts, which are based on several lower bounds on the recourse cost. These bounds are obtained by considering the set of customers in each route of the current solution. Another type of set-based inequalities is then presented in Section 5. The optimality cuts are introduced in Section 6. The approach for solving the packing problems that arise during the execution of our algorithm is described in Section 7. Computational results are finally reported in Section 8.

2 Model

The Vehicle Routing Problem with Stochastic Two-Dimensional Items or S2L-CVRP is defined as follows. We are given a complete undirected graph $G = (V, E)$ where $V = \{0, 1, 2, ..., n\}$ is the set of vertices of cardinality $n + 1$ and $E = \{(j, k) : j, k \in V : j < k\}$ is the set of edges with their associated cost c_{jk}, $(j, k) \in E$. Also, vertex 0 is the depot while $C = V \backslash \{0\}$ is the set of customers. We assume that K identical vehicles are available to execute delivery routes that start and end at the depot. The loading area of each vehicle has height H, width W and a maximum weight capacity Q. In this work, unloading constraints are also considered: at every service location, the items of the current customer can be unloaded by pulling them out of the vehicle without moving any item from other customers.

Each customer $j \in C$ has a demand for m_j two-dimensional items. Let I be the set of all items with cardinality $\sum_{j\in C} m_j = m$. For each item $i \in I$, there are d_i possible sizes in height, width and weight with an associated probability distribution ($d_i = 1$ for a deterministic item). That is, $\sum_{r=1}^{d_i} p_i^r = 1$ for every item $i \in I$, where p_i^r is the probability that item i has width w_i^r, height h_i^r and weight q_i^r, $w_i^r \leq W$, $h_i^r \leq H$, $q_i^r \leq Q$. Then,

$$\bar{a}_j = \sum_{i=1}^{m_j} \sum_{r=1}^{d_i} p_i^r h_i^r w_i^r$$

is the average area covered by the items of customer j and

$$\bar{q}_j = \sum_{i=1}^{m_j} \sum_{r=1}^{d_i} p_i^r q_i^r$$

is their average weight. In a feasible solution, the items delivered on each route must fit within the loading area of the vehicle, their total weight should not exceed capacity Q and the unloading constraints should be satisfied. When stochastic items are delivered on a given route, the average or expected area covered by these items plus the actual area covered by the deterministic items must be less than or equal to the loading area of the vehicle. Similarly, the expected weight of the stochastic items plus the actual weight of the deterministic items should be less than or equal to the vehicle capacity.

Although the actual sizes and weights of the stochastic items are unknown when the delivery routes are planned, they become known just prior to the loading and departure of the vehicles. If it happens that the items to be transported by a vehicle do not fit into the loading area, a recourse action must be considered. In our application, the items of one or more customers are left on the dock. In this case, these customers will be delivered later, which negatively impacts service quality. Accordingly, the recourse cost is based on the number of unserved customers.

The S2L-CVRP can be formulated as a stochastic VRP with additional constraints. In our case, we use the classical two-index formulation where x_{jk} is equal to 1 if edge $(j,k) \in E$ is used (with $j < k$), 0 otherwise . We also denote $F(x)$ the expected cost of the recourse of solution $x = (x_{jk})$. The formulation of the S2L-CVRP is then :

$$\min \sum_{j<k} c_{jk} x_{jk} + F(x) \tag{1}$$

$$\sum_{j \in C} x_{0j} = 2K \tag{2}$$

$$\sum_{h<j} x_{hj} + \sum_{k>j} x_{jk} = 2 \qquad j \in C \tag{3}$$

$$\sum_{j,k \in S} x_{jk} \leq |S| - \left\lceil \max \left\{ \frac{\sum_{j \in S} \bar{a}_j}{HW}, \frac{\sum_{j \in S} \bar{q}_j}{Q} \right\} \right\rceil \qquad S \subset C, 2 \leq |S| \leq n \tag{4}$$

$$\sum_{(j,k) \in R} x_{jk} \leq |R| - 1 \qquad R \in \mathcal{R}^{inf} \tag{5}$$

$$x_{jk} \in \{0,1\} \qquad 0 \leq j < k \leq n \tag{6}$$

The objective (1) is to minimize the sum of the routing and expected recourse costs. Constraints (2) force the fleet of K vehicles to be used. Constraints (3) state that exactly two edges must be used by a vehicle to visit a customer. Constraints (4) are the subtour-breaking and rounded-capacity constraints. Then, constraints (5) prohibit all routes that do not satisfy the loading requirements, including unloading constraints, which is denoted by the set $\mathcal{R}^{inf}$. This set contains infeasible routes with only deterministic customers, as well as routes with stochastic customers for which all possible realizations or scenarios are infeasible. In these constraints, route R is defined by the set of edges covered by the corresponding vehicle. Finally, the binary requirement on the decision variables is found in constraints (6). Note that by forcing the x_{jk} variables to be 0 or 1, back-and-forth routes to a single customer are forbidden (as it is typically done, see [26, 29]).

The recourse cost $F(x)$ is

$$F(x) = \sum_{R \in \mathcal{R}_x} F(R) \tag{7}$$

where $\mathcal{R}_x$ is the set of routes in solution x and $F(R)$ is the recourse cost of route R. Note that $F(R) = 0$ when route R has only deterministic items. Let Ω_R be the set of all possible realizations or scenarios for route R and p_{ω_R} the probability of scenario $\omega_R \in \Omega_R$. Then $F(R)$ is calculated as follows :

$$F(R) = c_f \cdot \sum_{\omega_R \in \Omega_R} p_{\omega_R} F(\omega_R) \tag{8}$$

where c_f is the cost associated with each unserved customer and $F(\omega_R)$ is the number of unserved customers under scenario ω_R. Thus, $F(R)$ is the expected recourse cost of route R over all scenarios.

3 The Integer L-Shaped Method

The integer L-shaped method is used to solve our problem. This method can be seen as a variant of the classical branch-and-cut algorithm for the deterministic VRP. First, the rounded-capacity (4) and infeasible path (5) constraints are relaxed and $F(x)$ in the objective is replaced by a lower bound θ to obtain the following initial model:

$$\min \sum_{j<k} c_{jk} x_{jk} + \theta \tag{9}$$

$$\sum_{j \in C} x_{0j} = 2K \tag{10}$$

$$\sum_{h<j} x_{hj} + \sum_{k>j} x_{jk} = 2 \qquad j \in C \tag{11}$$

$$x_{jk} \in \{0,1\} \qquad 0 \leq j < k \leq n \tag{12}$$

This problem is solved with CPLEX, the latter being in charge of computing solutions to the linear relaxations and branching on fractional variables. The inequalities are generated through methods that are called automatically by the CPLEX solver.

A pseudo-code for our L-Shaped method is shown in Algorithm 1. At each node of the branching tree, we first check for violated rounded capacity inequalities or RCIs (4). Then, two alternatives must be considered depending if the solution is fractional or not. If it is fractional, we look for violated L-cuts (15) and infeasible set inequalities (37). If there are none, then we branch on fractional variables. For an integer solution, the general idea is to go from the easiest to the hardest. First, the focus is on the non-ordered set of customers S associated with each route to generate L-cuts and infeasible set inequalities (37) which apply to a larger number of solutions (i.e., all routes where the customers in set S are visited consecutively, whatever the order of those visits). When no new inequalities of these types can be generated, we validate every route for infeasible path inequalities (5) and D-optimality cuts (40). These inequalities have a more restricted scope, as they apply to a single route, and are generated through more computationally expensive procedures. At the end, the best feasible integer solution found by the L-Shaped method is returned.

The rounded capacity inequalities (4) are classical inequalities that are generated using the package CVRPSEP from [32] and they will not be discussed anymore. The other inequalities and lower bounds mentioned in Algorithm 1 will be described in the following.

4 Lower Bounding Functionals

The L-cuts described in this section are in the class of Lower Bounding Functionals (LBFs). Although these cuts are used to bound θ in the objective, they are not optimality cuts because they come from a lower bound on the recourse (based on the number of unserved customers). They apply to non-ordered sets of customers visited consecutively in a route of the solution x^ν associated with the current node in the branching tree. Using a lower bound weakens the inequality but, on the other

Algorithm 1 L-Shaped Method

```
Require: a problem $N$
 1: Solve the linear relaxation of problem $N$ to obtain solution $x^\nu$
 2: if violated RCIs (4) are found then add them and go to Step 1
 3: if $x^\nu$ is fractional then
 4:    if violated $L$-cuts (15) and infeasible set inequalities (37) are found then add
       them and go to Step 1
 5:    else branch on fractional variables and call the L-Shaped Method recursively
       with each subproblem
 6: else
 7:    for each route $R$ in $x^\nu$ do
 8:       $S \leftarrow$ set of customers in route $R$
 9:       if $S$ contains only customers with deterministic items then
10:          Calculate a lower bound $LB(S)$ on the number of required vehicles
11:          if $LB(S) > 1$ then add an infeasible set inequality (37)
12:       else
13:          Calculate a lower bound on the recourse cost of route $R$ based on set $S$
             for each possible scenario
14:          if the lower bound on the recourse is positive over all scenarios then
             add infeasible set inequality (37) with $LB(S) = 2$
15:          else if the lower bound on the recourse is positive for at least one scenario
             then add $L$-cut (15)
16:       end if
17:    end for
18:    if inequalities were added then go to Step 1
19:    for each route $R$ in $x^\nu$ do
20:       if route $R$ contains only customers with deterministic items then
21:          if the packing problem is infeasible then add an infeasible path inequal-
             ity (5)
22:       else
23:          Calculate the exact recourse cost of route $R$ for each scenario by solving
             the corresponding packing problem
24:          if the recourse is positive over all scenarios then add an infeasible path
             inequality (5)
25:          else if the recourse is positive for least one scenario add a $D$-optimality
             cut
26:       end if
27:    end for
28:    if inequalities were added then go to Step 1
29:    else a new feasible integer solution has been found
30: end if
```

hand, its scope is larger than a single route, as it is valid for any route where the customers in a given set are visited consecutively, whatever their order.

4.1 L-cuts

Formally, let us consider the set of customers S in a route of solution x^ν with at least one stochastic item. For this set, we also assume that $\sum_{j \in S} \bar{a}_j \leq HW$ (i.e., an inequality of type (4) has been generated). Let us also denote $L(\omega_S)$ a lower bound on the number of unserved customers for a given realization or scenario $\omega_S \in \Omega_S$ with $L(S) = c_f \sum_{\omega_S \in \Omega_S} p_{\omega_S} L(\omega_S)$ (see Section 4.2 for a description of this lower bound).

Then, by arbitrarily selecting the customer j_S of minimum index among customers with stochastic items in set S, we have:

$$\theta_{j_S} \geq L(S) \cdot \left(\sum_{j \in S} x_{0j} + x(S) - |S| \right) \tag{13}$$

where $x(S) = \sum_{j,k \in S} x_{jk}$ sums the visited edges among all pairs of customers in set S. Given that S corresponds to the set of customers in a route of solution x^ν, two customers in set S are directly connected to the depot while $x(S)$ is equal to $|S|-1$. Accordingly, the right hand side of (13) reduces to $L(S)$ and defines a bound on θ_{j_S}.

This inequality can be extended by considering that the customers in set S can be visited (consecutively) before or after any other node, not only the depot. To this end, we propose the following approach. First, M variables θ_l, $l = 1, ..., M$, where M is some predefined number, are created. Our goal is to assign a set of variables θ_l, denoted Θ_S, with each subset of customers S for which $L(S) > 0$. Each time such a subset is found using the procedure described in Section 4.3, it is processed by Algorithm 2 and then added to an (initially empty) set $\bar{\mathcal{S}}$. As described in the pseudo-code, the subsets S' already in $\bar{\mathcal{S}}$ are processed one by one and each time $(S \cup S')$ is feasible for at least one scenario, $\Theta(S)$ is updated by setting it to the union of $\Theta(S)$ and $\Theta(S')$. If $\Theta(S)$ remains empty at the end of this loop, $\Theta(S)$ is assigned to some unused θ_l variable. If all variables are used, $\Theta(S)$ is assigned to $\Theta(S^*)$, where $S^* = \text{argmax}_{S' \in \bar{\mathcal{S}}}\{|S \cap S'|\}$.

We then have :

$$\theta \geq \sum_{l=1}^{M} \theta_l \tag{14}$$

Algorithm 2 Assignment of θ_l variables

```
Require: $S$ : subset of customers with $L(S) > 0$
Require: $\bar{\mathcal{S}}$ : set of previously generated subsets of customers with $L(S) > 0$
Require: $\bar{\Theta}$ : set of unused $\theta_l$ variables
 1: for $S' \in \bar{\mathcal{S}}$ do
 2:   if $S \cup S'$ is feasible then
 3:     $\Theta(S) \leftarrow \Theta(S) \cup \Theta(S')$
 4:   end if
 5: end for
 6: if $\Theta(S) = \emptyset$ then
 7:   if $\bar{\Theta} \neq \emptyset$ then
 8:     Let $\theta_{l'}$ be an unused variable in $\bar{\Theta}$
 9:     $\Theta(S) \leftarrow \{\theta_{l'}\}$
10:     $\bar{\Theta} \leftarrow \bar{\Theta} \backslash \{\theta_{l'}\}$
11:   else
12:     $S^* = \text{argmax}_{S' \in \bar{\mathcal{S}}}\{|S \cap S'|\}$
13:     $\Theta(S) \leftarrow \Theta(S^*)$
14:   end if
15: end if
```

$$\sum_{\theta_l \in \Theta(S)} \theta_l \geq L(S) \cdot (x(S) - |S| + 2) \qquad S \in \bar{\mathcal{S}} \tag{15}$$

Proposition. The inequalities (14) and (15) provide a lower bound θ on the recourse cost.

Proof. We first need to define the recourse cost for any fractional or integer solution. Let x^ν be a solution of the linear relaxation of model (9) - (12) with possibly some additional, previously generated, inequalities. We define $\mathcal{S}^\nu = \{S \subseteq C \mid x^\nu(S) > |S| - 2,\ L(S) > 0 \text{ and feasible}\}$ the set of all subsets of customers S with $L(S) > 0$ such that a path defined over S is feasible for at least one scenario.

Let also $\mathcal{X}$ be a subset of $\mathcal{S}^\nu$ such that :

(a) For each $S, S' \in \mathcal{X}, S \neq S'$, $S \cup S'$ is not feasible under any scenario.

It means that the subsets in $\mathcal{X}$ are maximal when considered pairwise. Now, let $\mathcal{P}^\nu$ be the set of all subsets $\mathcal{X}$ of $\mathcal{S}^\nu$ for which condition (a) is satisfied. Then, the recourse cost $\mathcal{L}(x^\nu)$ can be defined as :

$$\mathcal{L}(x^\nu) = \max_{\mathcal{X} \in \mathcal{P}^\nu} \left\{ \sum_{S \in \mathcal{X}} L(S) \cdot (x^\nu(S) - |S| + 2) \right\} \tag{16}$$

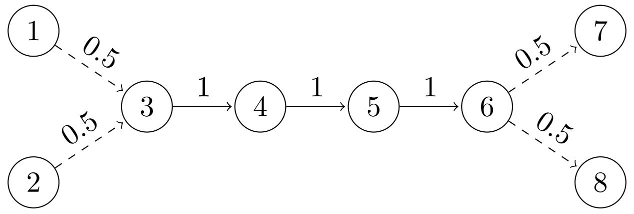

Figure 2: A fractional solution

Consider the example in Figure 2 where it is assumed that $\{3,4,5\}$, $\{4,5,6\}$ and $\{3,4,5,6\}$ are feasible and $L(\{3,4,5\})$, $L(\{4,5,6\})$, $L(\{3,4,5,6\})$ are all positive (i.e., they all cover at least one customer with stochastic items). Then, it is not possible for $\{3,4,5\}$ and $\{4,5,6\}$ to be together in some set $\mathcal{X}$ because these two sets can be combined to form the larger feasible set $\{3,4,5,6\}$. Accordingly, $L(\{3,4,5\})$ and $L(\{4,5,6\})$ will not be summed up in (16), which is fine. Otherwise, $L(\{4,5\})$ would be added twice.

For an integer solution, every set of customers S in $\mathcal{X}$ corresponds to a full route in x^ν and $\mathcal{L}(x^\nu)$ sums up the $L(S)$ values of the set of customers associated with each route. In the case of a fractional solution, $\mathcal{L}(x^\nu)$ might not be equal to the exact recourse cost. Consider the sets $S_1 = \{3,4,5,6\}$, $S_2 = \{1,3,4,5,6\}$, $S_3 = \{2,3,4,5,6\}$, $S_4 = \{3,4,5,6,7\}$ and $S_5 = \{3,4,5,6,8\}$ in Figure 2 with $L(S_q) > 0$, $q = 1,...,5$. All those sets are such that $x^\nu(S) > |S| - 2$. Then, the set of subsets $\mathcal{X}$ maximizing (16) is $\mathcal{X} = \{S_2, S_3, S_4, S_5\}$ for which $L(x^\nu) = \frac{1}{2}[L(S_2) + L(S_3) + L(S_4) + L(S_5)] \geq \frac{1}{2}[L(S_1) + L(S_1) + L(S_1) + L(S_1)] = 2L(S_1)$, because S_1 is included in S_2, S_3, S_4 and S_5. Thus, the contribution of S_1 is counted twice.

Now, let us consider $\bar{\mathcal{S}}$ the set of previously generated subsets of customers S with $L(S) > 0$ in Algorithm 2. Let $\bar{\mathcal{X}}$ be a subset of $\bar{\mathcal{S}}$ such that :

(b) For each $S' \neq S'' \in \bar{\mathcal{X}}$, $\theta(S') \cap \theta(S'') = \emptyset$.

Let $\bar{\mathcal{P}}$ be the set of all $\bar{\mathcal{X}}$ which satisfy condition (b). Then, from inequality (15) and condition (b), we have :

$$\sum_{\theta_l \in \bigcup_{S \in \bar{\mathcal{X}}} \Theta(S)} \theta_l \geq \sum_{S \in \bar{\mathcal{X}}} L(S) \cdot (x(S) - |S| + 2) \qquad \bar{\mathcal{X}} \in \bar{\mathcal{P}} \tag{17}$$

where $\bigcup_{S \in \bar{\mathcal{X}}} \theta(S)$ is the union of the sets of variables $\Theta(S)$ over all S in $\bar{\mathcal{X}}$. We also have a fortiori:

$$\sum_{l=1}^{M} \theta_l \geq \sum_{S \in \bar{\mathcal{X}}} L(S) \cdot (x(S) - |S| + 2) \qquad \bar{\mathcal{X}} \in \bar{\mathcal{P}} \tag{18}$$

and :

$$\sum_{l=1}^{M} \theta_l \geq \max_{\bar{\mathcal{X}} \in \bar{\mathcal{P}}} \left\{ \sum_{S \in \bar{\mathcal{X}}} L(S) \cdot (x(S) - |S| + 2) \right\} \tag{19}$$

Variable θ, as defined in (14) and (15), is a lower bound on the recourse cost if the following inequality is satisfied for any solution x^ν :

$$\mathcal{L}(x^\nu) \geq \sum_{l=1}^{M} \theta_l \tag{20}$$

Sets $S \in \bar{\mathcal{S}}$ with $x^\nu(S) \leq |S| - 2$ can be ignored when considering $\bar{\mathcal{P}}$ because $L(S) \cdot (x^\nu(S) - |S| + 2)$ is negative for these sets. Let $\bar{\mathcal{P}^\nu} = \{\bar{\mathcal{X}} \in \bar{\mathcal{P}} |\ x^\nu(S) > |S| - 2\ \forall S \in \bar{\mathcal{X}})$. Then, we have :

$$\sum_{l=1}^{M} \theta_l \geq \max_{\bar{\mathcal{X}} \in \bar{\mathcal{P}^\nu}} \left\{ \sum_{S \in \bar{\mathcal{X}}} L(S) \cdot (x^\nu(S) - |S| + 2) \right\} \tag{21}$$

Using inequalities (20) and (21) as well as the definition of $\mathcal{L}(x^\nu)$ in (16), we obtain :

$$\max_{\mathcal{X} \in \mathcal{P}^\nu} \left\{ \sum_{S \in \mathcal{X}} L(S) \cdot (x^\nu(S) - |S| + 2) \right\} \geq \max_{\bar{\mathcal{X}} \in \bar{\mathcal{P}^\nu}} \left\{ \sum_{S \in \bar{\mathcal{X}}} L(S) \cdot (x^\nu(S) - |S| + 2) \right\} \tag{22}$$

For inequality (22) to be true for any solution x^ν, we must show that $\bar{\mathcal{P}^\nu} \subseteq \mathcal{P}^\nu$. That is, if $\bar{\mathcal{X}} \in \bar{\mathcal{P}^\nu}$ then $\bar{\mathcal{X}} \in \mathcal{P}^\nu$. But we know from Algorithm 2 that $\Theta(S') \cap \Theta(S'') = \emptyset$ for S', $S'' \in \bar{\mathcal{X}}$, $S' \neq S''$ implies that $S' \cup S''$ is not feasible. Thus, the sets in $\bar{\mathcal{X}}$ satisfy condition (a). Given that these sets also satisfy $x^\nu(S) > |S| - 2$, $\bar{\mathcal{X}} \in \mathcal{P}^\nu$.

The benefits of this extension come from the fact that the subsets of customers S do not need to be connected to the depot and the recourse cost can be bounded as long as there is a path going through S. On the other hand, two sets S' and S'' with only a few customers (or even none) in common might be associated with the same θ_l variables, which induces a weaker bound on the recourse, see line 12 in Algorithm 2. But, initial experiments on difficult instances have shown that the subsets of customers S are often associated with a single variable. We will refer to inequalities (15) as L-cuts in the following.

4.2 Lower bounds

To obtain a lower bound on the recourse $L(S)$ for a set of customers S in a route of solution x^ν, we need to calculate a lower bound $L(\omega_S)$ for each scenario $\omega_S \in \Omega_S$. A tight lower bound can be obtained by solving a special knapsack problem with two-dimensional items where the knapsack stands for the loading area of the vehicle and where one unit of gain is achieved when all items of a given customer are in the loading area (note that the exact position of each item in the loading area does not need to be considered because S is not ordered). Given that solving this knapsack problem is computationally expensive, we rather consider three different relaxations and one feasibility test, leading to four lower bounds. These lower bounds are calculated in the order $L_1(\omega_S)$, $L_2(\omega_S)$, $L_3(\omega_S)$ and $L_4(\omega_S)$. As soon as one of these bounds is found to be strictly positive, the calculations stop and $L(\omega_S)$ is assigned to this positive value.

Bound L_1

Let h_i and w_i be the height and width of item i of customer j under scenario $\omega_S \in \Omega_S$ with $a_j = \sum_{i=1,\ldots,m_j} h_i w_i$. Also, let z_j be a binary variable which is equal to 1 when all items of customer j under scenario ω_S are in the loading area. Then $L_1(\omega_S)$ can be formulated as follow :

$$L_1(\omega_S) = |S| - \max\left\{\sum_{j\in S} z_j : \sum_{j\in S} a_j z_j \leq HW,\ z_j \in \{0,1\},\ j \in S\right\} \tag{23}$$

This bound can be easily obtained. We just need to sort the set of customers in non decreasing order of a_j and add them iteratively until the loading area HW is exceeded.

Bound L_2

The next lower bound is based on the solution of dual feasible functions [13]. More precisely, we consider L_{dff}^{BM} in [9] which returns a lower bound on the required height of the loading area to accommodate the items of all customers in set S. If this value is larger than the height H of the loading area, then at least one customer cannot be served.

$$L_2(\omega_S) = \begin{cases} 1 & \text{if } L_{dff}^{BM}(\omega_S) > H \\ 0 & \text{otherwise} \end{cases} \tag{24}$$

Bound L_3

The lower bound $L_3(\omega_S)$ is obtained by considering the Gilmore-Gomory formulation of the Cutting Stock Problem (CSP). Here, a pattern is defined through a subset of items I' taken from the set of items delivered to the customers in set S. A pattern is said to be H-feasible if $\sum_{i\in I'} h_i \leq H$ and W-feasible if $\sum_{i\in I'} w_i \leq W$. Let P^H and P^W be the sets of all such H-feasible and W-feasible patterns. We also have two different types of variables: z_j which is equal to 1 when all items of customer j are in the solution, 0 otherwise, and y_p which is the number of times pattern p is selected. Finally, we have $a_{ip} = 1$ if item i is in pattern p, 0 otherwise. Then, $L_3(\omega_S)$ is defined as follow:

$$L_3(\omega_S) = |S| - \max \sum_{j\in S} z_j \tag{25}$$

$$\sum_{p\in P^W} a_{ip} y_p = h_i z_j \qquad j \in S,\ i = 1 \text{ to } m_j \tag{26}$$

$$\sum_{p\in P^H} a_{ip} y_p = w_i z_j \qquad j \in S,\ i = 1 \text{ to } m_j \tag{27}$$

$$\sum_{p\in P^W} y_p \leq H \tag{28}$$

$$\sum_{p\in P^H} y_p \leq W \tag{29}$$

$$\sum_{j\in S} q_j z_j \leq Q \tag{30}$$

$$y_p \geq 0 \text{ and integer} \qquad p \in P^H \cup P^W \tag{31}$$

$$z_j \in \{0,1\} \qquad j \in S \tag{32}$$

Constraints (26) and (27) guarantee that h_i W-feasible and w_i H-feasible patterns are selected if item i is in the solution. Constraints (28), (29) and (30) relate to the size of the loading area and the maximum weight.

Solving the mathematical programming model (25) - (32) to obtain L_3 is computationally expensive and we rather solve, using column generation, a continuous relaxation where the y_p variables are continuous and $0 \leq z_j \leq 1$.

Bound L_4

The last lower bound $L_4(\omega_S)$ is based on the exact solution of the One-Dimensional Contiguous Bin Packing Problem (1CBP), a tight relaxation of the 2OPP, using the branch-and-bound algorithm in [16]. If the problem is feasible then all customers fit within the loading area, otherwise at least one customer must be removed.

$$L_4(\omega_S) = \begin{cases} 1 & \text{if 1CBP under scenario } \omega_S \text{ is infeasible} \\ 0 & \text{otherwise} \end{cases}$$

4.3 Separation procedure

This section describes the separation procedure aimed at identifying violated L-cuts when the current solution x^ν is fractional (see line 4 in Algorithm 1). The pseudo-code of this procedure is found in Algorithm 3 where:

$$a_j^{\max} = \sum_{i=1}^{m_j} \max_{r=1,\ldots,d_i} h_i^r w_i^r$$

$$x'(S', S'') = \sum_{j \in S'} \sum_{k \in S''} x_{jk}^\nu$$

.

Algorithm 3 Generation of L-cuts

```
Require: $C_s$ : set of customers with stochastic items
1: for $j \in C_s$ do
2:   $S \leftarrow \{j\}$
3:   repeat
4:     if $\sum_{k \in S} a_k^{\max} > HW$ then
5:       for $\omega_S \in \Omega_S$ do
6:         Calculate $L(\omega_S)$
7:       end for
8:       if at least one feasible scenario $\omega_S$ then
9:         Generate $L$-cut and go to Step 1.
10:      end if
11:    end if
12:    $j^* \leftarrow \text{argmin}_{k \in C \backslash S}\{x'(S \cup \{k\}, V \backslash (S \cup \{k\}))\}$
13:    $S \leftarrow S \cup \{j^*\}$
14:  until $\sum_{k \in S} \bar{a}_k > HW$ or $x'(S, V \backslash S) \geq 4$
15: end for
```

As indicated in this pseudo-code, each customer with stochastic items is considered in turn to initialize set S. At each step of the following iterative procedure, the customer j^* minimizing $x'(S \cup \{k\}, V \backslash (S \cup \{k\}))$ over $k \in C \backslash S$ is selected and added to S. Then, if (1) the mean area covered by the items of all customers in S is smaller than HW, (2) the maximum possible area covered by those same items is greater than HW and (3) the summation over the variables x_{jk}^ν with $j \in S$ and $k \in V \backslash S$ is less than 4, there is an opportunity to generate a new inequality. For quick filtering purposes, the packing problem associated with each scenario is first solved using the Bottom-Left heuristic. If it happens that all packing problems are feasible, no L-cut can be generated. Otherwise, the lower bound $L(S)$ is calculated by summing $L(\omega_S)$ over every scenario $\omega_S \in \Omega_S$. If $L(S) > 0$, a new L-cut (15) is added to the model. Note that when $x'(S, V \backslash S) \geq 4$, then $x(S) \leq |S| - 2$ and set S

covers at least two different routes in solution x^{ν}. So, there is no hope of generating a new L-cut. Note also that when there is no feasible scenario in line 8, then an infeasible set inequality (37) can be generated with $LB(S) = 2$.

5 Other set-based inequalities

When a route contains only deterministic items, a lower bound $LB(S)$ on the number of vehicles required to serve the set of customers S in the route can be calculated (lines 9-11 in Algorithm 1). If the lower bound indicates that more than one vehicle is needed, then an infeasible set inequality can be generated. The bound $LB(S)$ is derived from the bounds LB_1, LB_2, LB_3 and LB_4, which are described below. As soon as one of these bounds is greater than 1 (i.e., more than one vehicle is required to serve set S), the calculations stop and $LB(S)$ is set to this value. It should be noted that LB_2 and LB_3 come from a previous work on the Strip Packing Problem (SPP) where bounds are proposed on the required height of the loading area to accommodate all customers in set S [2, 16]. Dividing these values by the height H of the loading area provides a lower bound on the number of vehicles.

Bound LB_1

The first lower bound LB_1 is the classical *continuous* bound on the required area, where $a_j = \sum_{i=1}^{m_j} h_i w_i$.

$$LB_1(S) = \left\lceil \frac{\sum_{j \in S} a_j}{HW} \right\rceil \tag{33}$$

Bound LB_2

The lower bound LB_2 is obtained by taking the maximum value between L_{dff}^{BM} in [9] and L_3 in [2].

$$LB_2(S) = \left\lceil \frac{\max\{L_{dff}^{BM}(S), L_3(S)\}}{H} \right\rceil \tag{34}$$

Bound LB_3

The lower bound LB_4 comes from the work in [16].

$$LB_3(S) = \left\lceil \frac{L_4(S)}{H} \right\rceil \tag{35}$$

Bound LB_4

The lower bound LB_4 is obtained by solving the One-Dimensional Contiguous Bin Packing Problem (1CBP), a tight relaxation of the 2OPP, using the branch-and-bound algorithm in [16]. If the problem is infeasible, at least two vehicles are required to serve the set of customers S.

$$LB_4(S) = \begin{cases} 2 & \text{if 1CBP is infeasible for set } S \\ 1 & \text{otherwise} \end{cases} \tag{36}$$

Then, if $LB(S) > 1$, we can generate the following infeasible set inequality :

$$x(S) \leq |S| - LB(S) \tag{37}$$

6 Optimality cuts

The optimality cuts are considered at the end of Algorithm 1 (lines 23-25). At this point, the current solution x^ν is integer and no new inequalities have been generated in the previous steps. Assuming that x^ν is feasible, the following optimality cut can be generated to bound θ in the objective:

$$\theta \geq \sum_{R \in \mathcal{R}_{x^\nu}} F(R) \cdot \left(\sum_{R \in \mathcal{R}_{x^\nu}} \sum_{(j,k) \in R} x_{jk} - (n + K - 1) \right) \tag{38}$$

Given that the x_{jk} variables involved in the double summation are all equal to 1 in solution x^ν, this double summation equals $n + K$ for x^ν and the right hand side of equation (38) reduces to its recourse cost. For any other solution, the inequality is trivially satisfied.

It should be noted that this optimality cut is aggregated over all routes and applies only to solution x^ν, which is definitely a weakness. We thus propose disaggregated optimality cuts or D-optimality cuts which apply to individual routes. These cuts have been proposed in [36], but have never been implemented in practice. To generate them, the initial relaxed model (9) - (12) is extended by first defining a θ_j variable for each customer j with stochastic items and by adding inequality (39), where C_s stands for the set of customers with stochastic items.

During the execution of the L-shaped method, the customer with stochastic items of minimum index j_R is arbitrarily selected among each route with stochastic items in solution x^ν and a cut of type (40) is generated for every one of those routes.

$$\theta \geq \sum_{j \in C_s} \theta_j \tag{39}$$

$$\theta_{j_R} \geq F(R) \cdot \left(\sum_{(j,k) \in R} x_{jk} - |R| + 1 \right) \tag{40}$$

Like the aggregated cut (38), the right-hand side of (40) for route R reduces to its recourse cost. For every other route $R' \neq R$, the inequality is trivially satisfied.

To calculate the exact recourse $F(R)$ of route R, we need to account for the number of unserved customers under every possible scenario. To this end, we start with the lower bound on the number of unserved customers $L(\omega_S)$ in Section 4.2, where S is the set of customers in route R and $\omega_S \in \Omega_S$ is a possible realization or scenario for set S (or, equivalently, $\omega_R \in \Omega_R$ is a possible realization or scenario for route R). For any ω_S, $|S| - L(\omega_S)$ is an upper bound on the number of customers contained in the loading area. We thus enumerate all subsets of customers in route R with at most $|S| - L(\omega_S)$ customers and sort them from largest to smallest. Ties are broken by giving priority to subsets which cover a larger area. Then, we consider these subsets one by one and solve the corresponding packing problem (see Section 7) until a feasible solution is found with the corresponding number of unserved customers and recourse cost. The exact recourse cost $F(R)$ of route R is obtained at the end by summing these recourse costs over all possible scenarios, weighted by the corresponding scenario probability, see equation (8). Although this approach might appear computationally expensive, our tests have shown that only a few packing problems need to be solved.

7 Packing problems

This section discusses the methodology for solving the packing problems that are generated during the execution of Algorithm 1. For routes with only deterministic items, solving the packing problem is aimed at determining if the route is feasible or if a new infeasible path inequality (5) can be generated (lines 20-21). For routes with stochastic items, a packing problem is solved for every possible scenario to calculate the exact recourse cost of the route and generate a D-optimality cut (40) if the route is feasible or a new infeasible path inequality (5) otherwise (lines 23-25). The latter case occurs when the recourse cost is positive under every scenario (i.e., there is always at least one unserved customer).

Different approaches are used to quickly detect if a route is feasible or infeasible. As described in the following, the 2OPP and then the full 2OPP with unloading constraints (UL) are considered in this order. The 2OPP is considered first because it is a simpler problem than the 2OPPUL and it is easily obtained by relaxing the unloading constraints.

The various approaches listed below are called one by one until the infeasibility of the 2OPP (and thus, the 2OPPUL as well) is proven:

1. A simple lower bound is obtained by summing the areas of all items in the route. The latter is infeasible if this sum is larger than the loading area.
2. The more sophisticated lower bound L_{dff}^{BM} on the required height of the loading

area is then used [9]. If this bound is larger than the height H of the loading area, then the route is infeasible. It should be noted that only the first three dual feasible functions are used here. The fourth one, which proved to be time consuming and not really effective during preliminary tests, was disregarded.

3. Another lower bound on the required height of the loading area is obtained by invoking the alternating constructive procedure reported in [2].

4. Two additional lower bounds on the height and width of the loading area are based on the Gilmore-Gomory formulation of the Cutting Stock Problem. They correspond to L_3^H and L_3^W in [15]. If these bounds are larger than H and W, respectively, the route is infeasible.

5. The One-Dimensional Contiguous Bin Packing Problem (1CBP), a tight relaxation of the 2OPP, is finally solved with the branch-and-bound algorithm in [16]. If there is no feasible solution to the 1CBP, then the route is infeasible.

If the 2OPP has not been proven to be infeasible, we then consider the real problem, namely the 2OPPUL, and apply the following procedures in this order to determine its feasibility or infeasibility:

1. The problem is first solved with an approximate method, namely a variant of the heuristic reported in [30], originally developed for the Two-Dimensional Strip Packing Problem (2SPP). This is a two-phase heuristic, where a solution is first constructed and then improved with simulated annealing. Here, the original construction heuristic is replaced by the Bottom-Left and Max-Touching Parameter heuristics [41] to address the unloading constraints. If a feasible packing if found with this heuristic, the route is feasible.

2. The lower bound L_2 for the 2OPPUL, reported in [15], is then used to estimate the required area. If the value of L_2 is larger than the loading area HW, then the route is infeasible.

3. The branch-and-bound algorithm reported in [9], originally developed for the 2SPP, has been adapted to the 2OPPUL. It is applied with the following additional fathoming criterion : if an item does not fit at any position among a set of precalculated positions, then the current partial solution cannot lead to any feasible solution and the node can be fathomed. In practice, this algorithm can often find feasible solutions very quickly. We allow the generation of a maximum of 1,000,000 nodes in the branching tree before stopping the algorithm. If the algorithm returns a feasible packing, then the route is feasible. If the algorithm ends without finding any feasible packing, then the route is infeasible. Otherwise, we have to move to the next step.

4. The exact algorithm reported in [15] for solving the 2OPPUL is finally applied. It is based on a mathematical formulation of the 1CBP to which constraints

are added to satisfy the unloading requirements. In practice, this algorithm proved to be very good at detecting infeasibility in short computation times.

At the end, we know if the packing problem is feasible or infeasible and, if feasible, we have the corresponding solution.

8 Computational Results

In this section, we first compare our method over a set of existing instances proposed in [26] for the deterministic 2L-CVRP. Next, we explain how we generated new instances for the S2L-CVRP, before analyzing the contribution of the previously proposed inequalities on those instances. The final results are reported at the end.

Our L-Shaped Method was coded in C++ and called the CPLEX 12.5 solver. The tests were performed on a 3.07 Ghz Intel Xeon X5675 running under the Linux system. Note that the number of variables M used for generating L-cuts was set to the number of customers with stochastic items. Preliminary tests showed that larger values do not provide any benefit.

8.1 Comparison on the 2L-CVRP

By setting the number of possible values for the size and weight of each item to 1, deterministic instances are obtained. In this case, our L-Shaped Method reduces to a branch-and-cut algorithm which can be compared to the one reported by Iori et al. [26]. The algorithm of Iori et al. was coded in C and was run on a 3GHz Pentium IV with the CPLEX 9.0 solver. The packing problems were solved with a branch-and-bound algorithm based on the work in [33] for the Strip Packing Problem. This algorithm was run for 86,400 seconds on each instance, as compared to 7,200 seconds for ours.

To test their algorithm, the authors in [26] created five different types of instances from an original set of 36 instances, for a total of 180 instances. In all cases, the loading area of each vehicle has height $H = 40$ and width $W = 20$. In the first type of instances, every customer has a single item of width and height equal to 1. Since the packing is not constraining, these instances reduce to a classical vehicle routing problem with one-dimensional or scalar demand (weight). With regard to the other instances, each customer has 1 or 2 items in type 2, 1 to 3 items in type 3, 1 to 4 items in type 4 and 1 to 5 items in type 5. Furthermore, each item can have one of three different shapes, namely Vertical, Horizontal or Homogeneous. The exact number of items per customer and the shape of each item were randomly generated in the intervals shown in Table 1. The largest instances have up to 255 customers, 786 items and a fleet of 51 vehicles.

	# Items	Vertical		Horizontal		Homogeneous	
Type	per cust.	Height	Width	Height	Width	Height	Width
1	1	1	1	1	1	1	1
2	[1,2]	[.4H, .9H]	[.1W, .2W]	[.1H, .2H]	[.4W, .9W]	[.2H, .5H]	[.2W, .5W]
3	[1,3]	[.3H, .8H]	[.1W, .2W]	[.1H, .2H]	[.3W, .8W]	[.2H, .4H]	[.2W, .4W]
4	[1,4]	[.2H, .7H]	[.1W, .2W]	[.1H, .2H]	[.2W, .7W]	[.1H, .4H]	[.1W, .4W]
5	[1,5]	[.1H, .6H]	[.1W, .2W]	[.1H, .2H]	[.1W, .6W]	[.1H, .3H]	[.1W, .3W]

Table 1: Types of instances

	Iori et al.		*Our B&C*			
Type	Solved	Time	Solved	Time	New Solved	Time
1	12	4731.9	12	2.1	7	49.8
2	11	1123.6	11	9.8	2	2314.4
3	12	1332.0	12	6.4	3	385.2
4	10	1030.9	10	11.7	5	171.2
5	10	488.8	10	1.8	9	633.6
Avg.		1741.4		6.4		710.8
Sum	55		55		26	

Table 2: Comparison of two algorithms for the 2L-CVRP

The results are shown in Table 2 under the headings "*Iori et al.*" and "*Our B&C*". In each case, we indicate the number of instances of each type solved by both algorithms, as well as the average CPU time in seconds. The number of additional instances that were solved by our algorithm when compared to the algorithm of Iori et al., as well as the average CPU time in seconds for solving these instances, is also indicated. A total of 26 additional instances were solved by our algorithm

Overall, our algorithm was able to solve instances with up to 71 customers and 226 items while the algorithm of Iori et al. was limited to a maximum of 35 customers and 114 items. For the 55 instances solved by both algorithms, ours took only a few seconds as compared to hundreds or even thousands of seconds for the other algorithm. This is a very substantial improvement, even if we take into account the different specifications of the two machines used to run the algorithms. Note that this improvement is mostly explained by the use of sophisticated packing algorithms.

8.2 Stochastic instances

The same 2L-CVRP instances described in the previous section were used to generate our stochastic instances. However, given that the packing problems do not have any impact when solving the instances of type 1, they are not considered anymore in the following. We took all instances of types 2, 3, 4 and 5 with at most 71 customers (the largest number of customers that our algorithm can address), for a total of 20 instances of each type. From each one of these $4 \cdot 20 = 80$ instances, we generated six different stochastic instances, for a total of 480 instances, by varying the percentage of customers with stochastic items and the maximum size of the discrete domain for the height, width and weight of each stochastic item, as shown in Table 3. Note that when the domain can take up to x different values, each

% Stoch. cust.	Domain size
10	2
50	2
100	2
100	3
50	5
10	9

Table 3: S2L-CVRP instances

stochastic item has between 2 and x different height, width and weight values, with a given probability distribution defined over these values. The width and height values for each item were selected in the intervals $[\max\{1, h/2\}, \min\{h + h/2, H\}]$ and $[\max\{1, w/2\}, \min\{w+w/2, W\}]$, where h and w are the height and width of the item in the original deterministic instance. All real values were rounded to get only integers. The c_f parameter which is used to compute the recourse cost in equations (7) and (8) was set to 10.

We observed that our algorithm is very sensitive to the number of items per route, due to the difficulty of the packing problem. To get an increasing number of items per route from the instances of type 2 to the instances of type 5, without exceeding the computational limits of our algorithm, an average of 4 customers per route was allowed through the definition of appropriate weights (leading to an increasing average number of items per route from type 2 to type 5) . Basically, a weight was generated for each customer based on a normal law of mean $Q/4$, where Q is the vehicle capacity in the original 2L-CVRP instances. The weight obtained was then split randomly among the items of the corresponding customer. Some final adjustments were performed to guarantee that at least two customers could fit in a route.

The number of vehicles and an upper bound on the objective value were obtained with the adaptive large neighborhood search heuristic in [35]. The maximum number of iterations was set to 25,000 and a time limit of 1,000 seconds was imposed.

8.3 Impact of the various inequalities

We first report some results aimed at evaluating the effectiveness of the proposed inequalities. We created 5 different algorithmic variants for this purpose. The first setting “All cuts” correspond to the L-Shaped method described in Section 3, including D-optimality cuts for integer solutions and L-cuts for integer and fractional solutions. The application of L-cuts on fractional solutions was removed in the four other variants. So, “No Frac. L-cuts” is the original L-Shaped method minus L-cuts on fractional solutions, “No L-Cuts” removes all L-cuts on both fractional and integer solutions, “No D-cuts” uses L-cuts on integer solutions and replaces the D-optimality cuts by the global optimality cuts (38) and “No L-D-cuts” removes all L-cuts and replaces the D-optimality cuts by the global optimality cuts. In all cases, the resulting algorithm was run for a maximum of 1,200 seconds.

The results are summarized in Table 4 on the $6 \cdot 20 = 120$ stochastic instances of each type. The table shows the number of solved instances and the following averages: CPU time in seconds, CPU time in seconds for solving the packing problems, gap in percentage between the final solution and the heuristic solution, number of L-cuts, number of infeasible set constraints, number of infeasible path constraints, number of D-optimality cuts and number of VRP (integer) solutions. To allow a fair comparison, the averages for the various inequalities were calculated only over the instances solved by all variants. Similarly, the average gap was taken only over the instances that were not solved by any variant. The number of solved-by-all and not-solved-by-any instances is indicated in Table 4 for each type.

The first observation is about the L-cuts on fractional solutions, which do not appear to be useful when "All cuts" is compared with "No Frac. L-cuts". In particular, the number of solved instances slightly increases and the computation time decreases when they are removed. So, it appears that many of these cuts do not provide useful bounds on the objective.

The most useful inequalities are the L-cuts on integer solutions. When these cuts are present, the D-optimality cuts seem almost useless by comparing the results of "No Frac. L-cuts" and "No D-cuts". However, when the L-cuts are not present, the D-optimality cuts play a useful role, as indicated by the poor performance of "No L-D-cuts". Overall, "No Frac. L-cuts" is the best approach with regard to the number of solved instances and CPU time. Accordingly, this variant was used for the results reported in the next section.

8.4 Final results

The final results are reported in Tables 5 and 6 after running our algorithm for a maximum of 7,200 seconds (2 hours) on each instance. Table 5 summarizes the average results obtained over each set of 80 instances associated with a given percentage of customers with stochastic items and domain size. Otherwise, the format of this table is similar to Table 4. Table 5 shows in particular that a higher percentage of customers with stochastic items increases the complexity of the problem, as indicated by the number of solved instances and CPU time, in particular when going from 10% to 50%. Also, the total CPU time sharply increases when the domain of the probability distribution increases. For example, when the number of customers with stochastic items is low (10%), increasing the number of values in the domain of the probability distribution from 2 to 9 increases the total CPU time from 90.5 seconds to 337.4 seconds and the time for solving the packing problems from 22.4 seconds to 109.1 seconds. We also observed that the total CPU time on some of these instances was almost totally spent on the packing problems.

Table 6 shows another, more detailed, view of the results. Each identifier in this

Type 2 Variant	Solved	Total CPU	Packing CPU	Gap (%)	*L*-cuts	Inf. Set	Inf. Path	*D*-cuts	VRP
All Cuts	67	65.2	7.2	7.10%	269.3	300.8	12.4	21.0	27.9
No Frac L-cuts	70	43.7	8.2	7.22%	21.3	14.7	12.5	11.3	36.2
No L-Cuts	67	64.6	13.8	7.54%	0.0	10.4	29.1	24.6	37.4
No D-Cuts	69	61.4	7.2	7.24%	23.5	18.6	5.9	21.0	50.7
No L-D-Cuts	54	189.3	61.2	7.67%	0.0	12.4	36.4	631.7	651.8
Solved-by-all	54								
Not-solved-by-any	50								
Type 3									
All Cuts	76	69.8	15.8	7.95%	164.7	162.7	7.1	19.4	18.5
No Frac L-cuts	78	38.9	14.0	7.97%	17.0	2.9	7.8	10.0	25.4
No L-Cuts	75	66.0	30.3	8.35%	0.0	2.0	13.8	20.8	24.2
No D-Cuts	78	43.3	15.8	8.01%	17.9	3.1	4.1	19.4	36.1
No L-D-Cuts	65	139.9	79.4	8.50%	0.0	2.4	19.4	396.3	404.2
Solved-by-all	65								
Not-solved-by-any	41								
Type 4									
All Cuts	73	84.3	23.8	9.53%	118.8	55.4	4.0	12.6	13.2
No Frac L-cuts	77	46.6	21.0	9.46%	11.5	2.2	3.5	7.2	16.2
No L-Cuts	72	83.9	53.4	9.83%	0.0	1.8	6.0	15.3	15.9
No D-Cuts	75	46.6	23.8	9.48%	12.5	2.4	1.3	12.6	24.6
No L-D-Cuts	62	182.4	127.6	10.02%	0.0	2.4	6.9	291.9	298.0
Solved-by-all	62								
Not-solved-by-any	43								
Type 5									
All Cuts	80	25.2	18.4	5.50%	30.7	1.3	0.1	2.8	4.7
No Frac L-cuts	81	23.5	18.4	5.71%	6.3	0.1	0.2	2.5	6.7
No L-Cuts	65	73.6	68.8	6.17%	0.0	0.1	0.4	9.6	8.8
No D-Cuts	82	22.9	18.4	5.71%	6.4	0.1	0.0	2.8	7.9
No L-D-Cuts	54	156.8	146.8	6.54%	0.0	0.1	0.0	145.2	147.5
Solved-by-all	50								
Not-solved-by-any	36								
Overall									
All Cuts	296	63.0	16.5	7.58%	147.8	131.2	6.0	14.4	16.3
No Frac L-cuts	306	38.8	15.5	7.65%	14.2	4.9	6.1	7.9	21.4
No L-Cuts	279	72.1	41.0	8.02%	0.0	3.5	12.4	17.8	21.7
No D-Cuts	304	44.0	16.5	7.67%	15.3	5.9	2.9	14.4	30.3
No L-D-Cuts	235	166.5	102.6	8.23%	0.0	4.2	15.8	369.0	378.0
Solved-by-all	231								
Not-solved-by-any	170								

Table 4: Comparison of different variants

Domain size	% Stoch. cust.	Solved	Total CPU	Packing CPU	Gap (%)	L-cuts	Inf. Set	Inf. Path	D-cuts	VRP
2	10	59	90.5	22.4	6.4	6.9	9.8	9.8	3.5	22.6
2	50	53	254.1	32.6	7.8	32.4	4.6	7.7	14.6	34.8
2	100	55	564.1	340.8	7.1	41.4	0.3	1.6	18.7	30.9
3	100	50	491.2	311.2	6.8	41.2	0.0	0.7	17.6	27.7
5	50	53	385.7	90.2	8.6	39.5	4.6	6.4	18.5	37.8
9	10	62	337.4	109.1	7.6	17.0	10.9	12.4	8.4	33.0
Avg.			353.8	151.1	7.4	29.7	5.0	6.4	13.6	31.1
Sum		332				178.4	30.2	38.6	81.3	186.8

Table 5: Summary of final results

table is a 4-digit number: the first two digits identify the instance number from the 36 original instances in [26], while the last two digits identify the instance type. For example, identifiers 0202 to 0205 correspond to the deterministic instances of types 2 to 5 derived from the second original instance. As previously mentioned, six different stochastic instances were generated from the deterministic instance associated with the 4-digit identifier. In Table 6, heading "Ins" is the 4-digit identifier, n is the number of customers, "Solved" is the number of instances solved to optimality, "Total CPU" is the average computation time in seconds spent over the instances solved to optimality and "Packing CPU" is the average computation time spent on the packing problems over the instances solved to optimality. The two last headings are "Gap", which contains the average gap in percentage between the final solution and the heuristic solution over all instances that were not solved to optimality, and "Cuts" which is the average number of L-cuts, infeasible set inequalities, infeasible path inequalities and D-cuts that were added to the model over all instances solved to optimality.

These detailed results show in particular the limitations of our algorithm with regard to the problem size. Most instances from 15 to 32 customers can be solved within the time limit, but difficulties arise beyond 32 customers. In particular, the number of possible sets of customers S, when calculating $L(S)$, increases sharply. Overall, our method was able to solve 332 instances out of 480 using an average of 353.8 seconds of computation time.

9 Conclusion

This paper has introduced a stochastic variant of the 2L-CVRP where some item sizes are not known with certainty when the vehicle routes are planned. From a methodological standpoint, a new type of Lower Bounding Functionals, called L-cuts, has been introduced. The latter proved to be very effective when integrated within the reported L-Shaped Method. On the deterministic 2L-CVRP, the branch-and-cut algorithm derived from our L-Shaped Method also outperformed another state-of-the-art exact algorithm on a set of benchmark instances. Future work will consider different variants where, for example, it is possible to rotate items to better

Ins	n	Solved	Total CPU	Packing CPU	Gap (%)	Cuts	Ins	n	Solved	Total CPU	Packing CPU	Gap (%)	Cuts
0102	15	6	0.5	0.5	-	2.5	1102	29	6	996.3	28.8	-	195.5
0103	15	6	2.6	2.6	-	10.8	1103	29	4	888.1	112.3	6.3	150.8
0104	15	6	3.0	2.9	-	9.5	1104	29	5	362.6	164.6	4.5	166.2
0105	15	6	11.1	11.1	-	2.3	1105	29	5	242.3	160.1	1.9	33.8
0202	15	6	1.4	1.3	-	19.3	1202	30	4	881.5	41.5	9.7	270.0
0203	15	6	4.0	4.0	-	27.2	1203	30	6	110.7	27.8	-	83.0
0204	15	6	3.2	3.2	-	3.0	1204	30	5	1262.7	341.8	9.3	92.0
0205	15	6	6.6	6.6	-	3.2	1205	30	6	1846.6	1821.6	-	46.0
0302	20	5	22.6	9.6	2.8	92.4	1302	32	4	429.2	9.1	1.6	70.8
0303	20	6	22.6	22.1	-	47.2	1303	32	6	62.0	15.3	-	72.7
0304	20	6	27.0	23.6	-	27.0	1304	32	6	138.3	39.5	-	41.0
0305	20	6	8.6	7.6	-	11.3	1305	32	6	861.0	855.4	-	19.8
0402	20	6	75.0	13.1	-	80.7	1402	32	1	4864.8	4.6	1.9	240.0
0403	20	6	10.8	8.0	-	22.2	1403	32	4	845.2	7.6	1.0	67.3
0404	20	6	35.7	35.6	-	23.3	1404	32	5	2112.3	50.5	0.5	98.2
0405	20	5	1156.1	1154.3	3.2	11.6	1405	32	4	398.4	35.2	1.0	58.8
0502	21	6	17.1	8.0	-	30.3	1502	32	1	277.8	1.4	1.7	49.0
0503	21	6	9.2	4.6	-	18.7	1503	32	0	-	-	2.9	-
0504	21	6	17.8	13.8	-	28.8	1504	32	0	-	-	3.8	-
0505	21	6	21.0	20.5	-	15.0	1505	32	1	6350.3	5222.1	5.1	27.0
0602	21	6	93.6	15.8	-	76.2	1602	35	1	1683.0	39.4	5.3	258.0
0603	21	6	78.0	28.5	-	43.7	1603	35	1	4380.6	79.2	6.9	210.0
0604	21	6	119.0	117.6	-	32.2	1604	35	2	656.1	36.9	9.0	124.0
0605	21	6	14.7	13.9	-	15.7	1605	35	6	266.4	28.6	-	40.2
0702	22	6	202.5	34.1	-	85.5	1702	40	0	-	-	7.9	-
0703	22	6	47.3	43.0	-	38.3	1703	40	2	736.4	1.7	6.7	27.5
0704	22	6	32.6	32.2	-	11.5	1704	40	0	-	-	7.9	-
0705	22	6	31.5	31.0	-	12.5	1705	40	3	649.4	77.3	4.8	25.3
0802	22	6	4.6	4.3	-	22.0	1802	44	0	-	-	4.1	-
0803	22	6	31.4	30.4	-	34.3	1803	44	0	-	-	4.8	-
0804	22	6	21.4	21.2	-	12.7	1804	44	0	-	-	8.4	-
0805	22	6	46.3	45.6	-	13.3	1805	44	2	43.2	21.0	2.1	23.0
0902	25	5	106.6	20.9	11.7	187.2	1902	50	0	-	-	11.2	-
0903	25	6	352.7	307.8	-	69.5	1903	50	0	-	-	11.9	-
0904	25	6	189.1	151.8	-	63.7	1904	50	0	-	-	14.6	-
0905	25	5	1041.6	1041.0	5.0	10.2	1905	50	0	-	-	5.2	-
1002	29	6	127.8	32.3	-	190.0	2002	71	0	-	-	14.2	-
1003	29	6	100.4	11.0	-	65.0	2003	71	0	-	-	15.0	-
1004	29	6	1434.3	178.1	-	101.2	2004	71	0	-	-	15.3	-
1005	29	6	526.4	508.3	-	38.8	2005	71	0	-	-	10.4	-

Table 6: A more detailed view of final results

fill the loading area. Also, we want to address an extension with both pickups and deliveries along the routes. In practice, this type of problem occurs when an item must be exchanged for another at a customer location (due to some defect).

Acknowledgments. Financial support for this work was provided by the Canadian Natural Sciences and Engineering Research Council (NSERC). This support is gratefully acknowledged.

References

[1] Alvarez-Valdes R., Parreño F., Tamarit J.M., A GRASP algorithm for constrained two-dimensional non guillotine cutting problems. Journal of the Operational Research Society, 414-425, 2005.

[2] Alvarez-Valdes R., Parreño F., Tamarit J.M., A branch and bound algorithm for the strip packing problem. OR Spectrum 31, 431-459, 2009.

[3] Amaral A., Letchford A., An improved upper bound for the two-dimensional non-guillotine cutting problem. Technical Report, Lancaster University, UK, 2003.

[4] Arahori Y., Imamichi T., Nagamochi H., An exact strip packing algorithm based on canonical forms. Computers & Operation Research 39, 2991-3011, 2012.

[5] Baldacci R., Boschetti M.A., A cutting-plane approach for the two-dimensional orthogonal non-guillotine cutting problem. European Journal of Operational Research 183, 1136-1149, 2007.

[6] Belov G., Rohling H., LP bounds in an interval-graph algorithm for orthogonal-packing feasibility. Operations Research 61, 483-497, 2013.

[7] Bertsimas D. J., Jaillet P., Odoni A.R., A priori optimization. Operations Research 38, 1019-1033, 1990.

[8] Bortfeldt A., Wäscher G., Container loading problems - A state-of-the-art review. FEMM Working Paper No. 7/2012, Otto-von-Guericke University Magdeburg, 2012.

[9] Boschetti M. A., Montaletti L., An exact algorithm for the two-dimensional strip-packing problem. Operations Research 58, 1774-1791, 2010.

[10] Caprara A., Monaci M., Bidimensional packing by bilinear programming. Mathematical Programming 118, 75-108, 2009.

[11] Caprara A., Monaci M., On the two-dimensional knapsack problem. Operations Research Letters 32, 5-14, 2004.

[12] Christiansen C.H., Lysgaard J., A branch-and-price algorithm for the capacitated vehicle routing problem with stochastic demands. Operations Research Letters 35, 773-781, 2007.

[13] Clautiaux F., Alves C., de Carvalho J.V., A survey of dual-feasible and superadditive functions. Annals of Operation Research 179, 317-342, 2010.

[14] Clautiaux F., Carlier J., Moukrim A., A new exact method for the two-dimensional orthogonal packing problem. European Journal of Operational Research 183, 1196-1211, 2007.

[15] Côté J.-F., Gendreau M., Potvin J.-Y., An exact algorithm for the two-dimensional orthogonal packing problem with unloading constraints. Technical Report CIRRELT-2013-26, Montreal, 2013.

[16] Côté J.-F., Iori M., Dell'Amico M., Combinatorial Benders' cuts for the strip packing problem. Forthcoming in Operations Research, 2013.

[17] Duhamel C., Lacomme P., Quilliot A., Toussaint H., A multi-start evolutionary local search for the two-dimensional loading capacitated vehicle routing problem. Computers & Operation Research 38, 617-640, 2011.

[18] Fekete S., Schepers J., A combinatorial characterization of higher-dimensional orthogonal packing. Mathematics of Operations Research 29, 353-368, 2004.

[19] Fekete S., Schepers J., van der Veen J.C., An exact algorithm for higher-dimensional orthogonal packing. Operations Research 55, 569-587, 2007.

[20] Fuellerer G., Doerner K.F., Hartl R.F., Iori M., Ant colony optimization for the two-dimensional loading vehicle routing problem. Computers & Operations Research 36, 655-673, 2009.

[21] Gendreau M., Iori M., Laporte G., Martello S., A tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints. Networks 51, 4-18, 2008.

[22] Gendreau M., Laporte G., Séguin R., An exact algorithm for the vehicle routing problem with stochastic demands and customers. Transportation Science 29, 143-155, 1995.

[23] Herz J.C., Recursive computational procedure for the two dimensional stock cutting. IBM J. Res. Development 16, 462-469, 1972.

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

[25] Iori M., Martello S., Routing problems with loading constraints. TOP 18, 4-27, 2010.

[26] Iori M., Salazar-González J.-J., Vigo D., An exact approach for the vehicle routing problem with two-dimensional loading constraints. Transportation Science 41, 253-264, 2007.

[27] Jabali O., Rei W., Gendreau M., Laporte G., New valid inequalities for the multi-vehicle routing problem with stochastic demands. Technical Report CIRRELT-2012-58, Montreal, 2012.

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

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

[30] Leung C.H.S., Zhang D., Kwang M.S., A two-stage intelligent search algorithm for the two-dimensional strip packing problem. European Journal of Operational Research 215, 57-69, 2011.

[31] Leung S.C.H., Zhang Z., Zhang D., Hua X., Lim M.K., A meta-heuristic algorithm for heterogeneous fleet vehicle routing problems with two-dimensional loading constraints. European Journal of Operational Research 225, 199-210, 2013.

[32] Lysgaard J., Letchford A.N., Eglese R.W., A new branch-and-cut algorithm for the capacitated vehicle routing problem. Mathematical Programming 100, 423-445, 2004.

[33] Martello S., Monaci M., Vigo D., An exact approach to the strip-packing problem. INFORMS Journal on Computing 15, 310-319, 2003.

[34] Mesyagutov M., Scheithauer G., Belov G., LP bounds in various constraint programming approaches for orthogonal packing. Computers & Operation Research 39, 2425-2438, 2012.

[35] Ropke S., Pisinger D., An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transportation Science 40, 455-472, 2006.

[36] Séguin R., Problèmes stochastiques de tournées de véhicules, Thèse de doctorat, Université de Montréal, Canada, 1994.

[37] da Silveira J.L.M., Miyazawa F.K., Xavider E.C., Heuristics for the strip packing problem with unloading constraints. Computers & Operations Research 40, 991-1003, 2013.

[38] Van Slyke R.M., Wets R., L-shaped linear programs with applications to optimal control and stochastic programming. SIAM Journal on Applied Mathematics 17, 638-663, 1969.

[39] Wäscher G., Haussner H., Schumann H., An improved typology of cutting and packing problems. European Journal of Operational Research 183, 1109-1130, 2007.

[40] Weyland, D., Stochastic vehicle routing - From theory to practice, Doctoral dissertation, Faculty of Informatics, Universit della Svizzera Italiana, Switzerland, 2013.

[41] Zachariadis E.E., Tarantilis C.D., Kiranoudis C.T., A guided tabu search for the vehicle routing problem with two-dimensional loading constraints. European Journal of Operational Research 3, 729-743, 2009.